

LDAP для учёных-ракетчиков

Данное открытое руководство о LDAP, OpenLDAP 2.x и ApacheDS на Linux и BSD (FreeBSD, OpenBSD и NetBSD). Оно предназначено для новичков, тех, кто собирается стать учёными-ракетчиками, и всех тех, кто находится между ними.

LDAP - сложный предмет. Данное руководство родилось из наших жалких попыток понять LDAP, поскольку он обещал настоящую нирвану - общий источник информации, неограниченную масштабируемость путём использования модели репликации, врождённую устойчивость, высокоскоростное получение данных, детальный контроль над тем, кто, что и с какими данными может сделать - этот список можно продолжать. Чудесная вещь!

На этом хорошие новости заканчиваются.

Плохая новость заключается в том, что, по нашему скромному мнению, никогда не было написано так много непонятого об одном предмете, за исключением, пожалуй, BIND и ... и ещё ... Есть бесчисленные отличные HOWTO, разбросанные по Интернету, которые прекрасны, если Вам нужно тактическое решение частной проблемы, и Вы готовы смириться со смутным неприятным чувством, что полностью зависите от чего-то такого, чего на самом деле не понимаете. Нам не нужно тактическое решение, нам было нужно стратегическое решение целого комплекса проблем, каждая из которых кажется идеально предназначенной для LDAP, но нам нужно было понять предмет ... нам нужно было WHYTO. Эта и есть наша, - возможно, жалкая, - попытка создать его.

Давным-давно OpenLDAP был единственной звездой на небосклоне Open Source LDAP. Он до сих пор считается эталонной реализацией LDAP и остается отличной системой с большим количеством реализованных возможностей, активно развивающейся и необыкновенно комплексной, что позволяет использовать его для решения нетривиальных задач. Однако, это уже не единственная звезда на небосклоне. Теперь есть [389 Directory Server](#) (бывший Fedora Directory Server), другая производная от проекта Мичиганского Университета, [OpenDS](#) - оригинальная реализация на Java от Sun (сейчас под заботливым присмотром Oracle), и проект [ApacheDS \(Apache Directory\)](#). Всё это чудесные проекты, и, вместе с OpenLDAP, они ошеломляют обилием продвинутых возможностей и функциональности в пространстве Open Source LDAP. Некоторые заметки об этих проектах и наши о них рассуждения Вы найдете [здесь](#), если, конечно, Вас интересуют подобного рода вещи.

Во все будущие версии данного руководства будет постепенно добавляться материал, описывающий использование ApacheDS, при этом документирование OpenLDAP будет продолжено.

<Предупреждение> Эта работа ещё далека от завершения. Если Вы нашли ошибку - не ворчите, сообщите нам. Посмотрите также наш [список доработок](#), и если Вы хотите внести вклад - милости просим. И за этот тяжкий труд мы можем пообещать Вам лишь тёплые чувства нашего доброго отношения и признание Ваших заслуг в лицензии. **</Предупреждение>**

Содержание

Что нового в Руководстве версии [0.1.15](#)

1. Соглашения и терминология

- [1.1 Цели и сфера применения](#)
- [1.2 Как читать данное руководство](#)
- [1.3 Используемая терминология и условные обозначения](#)
- [1.4 Благодарности](#)
- [1.5 Авторские права и лицензия](#)

Раздел 1 - Обзор и концепции

2. Обзор LDAP

- 2.1 Краткая история LDAP
- 2.2 Обзор LDAP
- 2.3 LDAP и базы данных
 - 2.3.1 Использование LDAP - резюме
- 2.4 Модель данных (объектная модель) LDAP
 - 2.4.1 Структура дерева объектов
 - 2.4.2 Атрибуты
 - 2.4.3 Объектные классы
 - 2.4.4 Описание дерева и добавление данных
 - 2.4.5 Навигация по дереву (DN и RDN)
- 2.5 Отсылки и репликация LDAP
 - 2.5.1 Отсылки
 - 2.5.2 Репликация

3. Схема данных, объектные классы и атрибуты LDAP

- 3.1 Обзор элементов LDAP
- 3.2 Наборы схемы данных
- 3.3 Объектные классы
- 3.4 Атрибуты
- 3.5 Правила соответствия
- 3.6 Операционные атрибуты и объекты LDAP

Раздел 2 - Поехали!

4. Установка LDAP

- 4.1 Установка LDAP
- 4.2 OpenLDAP на *NIX и Windows
- 4.3 ApacheDS на *NIX и Windows

5. Примеры конфигураций OpenLDAP

- 5.1 Простой каталог
 - 5.1.1 Проектирование DIT
 - 5.1.2 Выбор структурного объектного класса
 - 5.1.3 Файл slapd.conf
 - 5.1.4 Файл LDIF
 - 5.1.5 Загрузка LDIF-файла
 - 5.1.6 Добавление записей с помощью LDIF
 - 5.1.7 Изменение записей с помощью LDIF
 - 5.1.8 Напоследок просто подурчимся
- 5.2 Защита каталога
 - 5.2.1 Политика безопасности
 - 5.2.2 Добавление групп
 - 5.2.3 ACL - директивы access файла slapd.conf
 - 5.2.4 Тестирование ACL
- 5.3 Расширение иерархии
 - 5.3.1 Расширение требований
 - 5.3.2 Реализация расширения
 - 5.3.3 LDIF для реализации расширения
 - 5.3.4 ACL - директивы access файла slapd.conf
 - 5.3.5 Тестирование ACL
- 5.4 Создание и добавление объектов
 - 5.4.1 Требования
 - 5.4.2 Реализация

- 5.4.3 Определения атрибутов
- 5.4.4 Определение объектного класса и набора схемы
- 5.4.5 ACL - директивы access файла slapd.conf
- 5.4.6 LDIF
- 5.4.7 Тестирование изменений
- 5.5 Технология единого входа (Single Sign On)
- 5.6 Отсылки и репликация

6. Конфигурационные файлы

- 6.1 Обзор slapd.conf
 - 6.1.1 Использование OLC (cn=config)
 - 6.1.1.1 Обзор OLC (cn=config)
 - 6.1.1.2 Переконвертация slapd.conf в OLC (cn=config)
 - 6.1.1.3 Макет OLC (cn=config)
 - 6.1.1.4 Использование OLC (cn=config) (чтение, модификация)
 - 6.1.1.4.1 Общие замечания по конфигурации OLC (cn=config)
 - 6.1.1.4.2 Добавление/удаление наборов схемы данных при использовании OLC (cn=config)
 - 6.1.1.4.3 Добавление/удаление ACP/ACL при использовании OLC (cn=config)
 - 6.1.1.4.4 Добавление/удаление модулей при использовании OLC (cn=config)
 - 6.1.1.4.5 Добавление/удаление баз данных при использовании OLC (cn=config)
 - 6.1.1.4.6 Добавление/удаление наложений при использовании OLC (cn=config)
- 6.2 Список директив (OLC (cn=config) и slapd.conf)
- 6.3 Директивы глобального раздела (OLC (cn=config) и slapd.conf)
 - 6.3.1 Директивы TLS (OLC (cn=config) и slapd.conf)
- 6.4 Директивы раздела backend (OLC (cn=config) и slapd.conf)
- 6.5 Директивы раздела database (OLC (cn=config) и slapd.conf)
 - 6.5.1 Директивы overlay (OLC (cn=config) и slapd.conf)
- 6.6 Директивы ldap.conf
- 6.7 Конфигурация ApacheDS

7. Репликация и отсылки

- 7.1 Обзор репликации и отсылок
- 7.2 Репликация
 - 7.2.1 Репликация OpenLDAP
 - 7.2.1.1 Репликация OpenLDAP в стиле slurpd
 - 7.2.1.1.1 Ошибки при репликации OpenLDAP в стиле slurpd
 - 7.2.1.2 Репликация OpenLDAP в стиле syncrepl
 - 7.2.1.2.1 Тип RefreshOnly репликации syncrepl в OpenLDAP
 - 7.2.1.2.2 Тип RefreshAndPersist репликации syncrepl в OpenLDAP
 - 7.2.1.2.3 Репликация syncrepl с несколькими главными серверами (Multi-Master) в OpenLDAP
 - 7.2.1.2.4 Журналы доступа и delta-синхронизация при репликации syncrepl в OpenLDAP
 - 7.2.2 Репликация ApacheDS
- 7.3 Синхронизация DIT перед запуском репликации в стиле slurpd
- 7.3 Синхронизация DIT перед запуском репликации в стиле syncrepl
- 7.4 Отсылки
 - 7.4.1 Сцепление при отсылках

Раздел 3 - Справочная

8. LDIF и DSML

- 8.1 Обзор LDIF
- 8.2 Формат и директивы LDIF
 - 8.2.1 Формат файла LDIF
 - 8.2.1.1 Терминология LDIF и типы строк
 - 8.2.1.2 Пример LDIF

8.2.2 Директивы LDIF

- 8.2.2.1 Директива add
- 8.2.2.2 Директива attributename
- 8.2.2.3 Директива changetype
- 8.2.2.4 Директива control
- 8.2.2.5 Директива delete
- 8.2.2.6 Директива deleteoldrdn
- 8.2.2.7 Директива dn
- 8.2.2.8 Директива newrdn
- 8.2.2.9 Директива newsuperior
- 8.2.2.10 Директива objectclass
- 8.2.2.11 Директива replace
- 8.2.2.12 Директива version

8.3 Обработка двоичных данных (включая пароли) в LDIF

8.4 Импорт файлов LDIF

8.5 Примеры LDIF

8.6 DSML

9. Функциональная модель LDAP

9.4 LDAP URL

10. LDAP API

Раздел 4 - Операции OpenLDAP

11. OpenLDAP HowTo

- Конфигурирование нескольких DIT в OpenLDAP
- Настройка отсылок в OpenLDAP
- Настройка сцепления при отсылках в OpenLDAP
- Настройка репликации в стиле slurpd в OpenLDAP
- Настройка репликации в стиле syncrepl в OpenLDAP
- Настройка delta-синхронизации (syncrepl) в OpenLDAP
- Настройка и использование cn=config в OpenLDAP
- Замечания по запуску/установке OpenLDAP
- Замечания о наложениях в OpenLDAP (или как заставить это работать)
- Переход к OLC (cn=config) в OpenLDAP
- Использование OLC (cn=config)
- Конфигурирование групп пользователей в OpenLDAP

12. Ошибки и устранение неисправностей OpenLDAP

13. Производительность OpenLDAP

14. Инструменты LDAP

Инструменты OpenLDAP

- [ldapadd](#) - добавление LDIF-записи в каталог LDAP
- [ldapauth](#) - добавление LDIF-записи в каталог LDAP
- [ldapdelete](#) - удаление LDAP-записи
- [ldapmodify](#) - модификация существующих записей LDAP
- [ldapmodrdn](#) - модификация DN записи LDAP
- [ldappasswd](#) - модификация пароля записи
- [ldapsearch](#) - поиск записей LDAP
- [ldapwhoami](#) - выполнение LDAP-операции Who Am I на сервере

[slapacl](#) - проверка доступа к атрибутам путём изучения конфигурации DIT
[slapadd](#) - добавление записи LDAP в базу данных
[slapauth](#) - проверка данных SASL на соответствие с DIT
[slapcat](#) - экспорт LDIF из базы данных LDAP
[slapdn](#) - проверка DN на соответствие конфигурации DIT
[slapindex](#) - переиндексация базы данных LDAP
[slappasswd](#) - генерация пароля
[slaptest](#) - проверка файла slapd.conf или директории cn=config (slapd.d)

LDAP-браузеры

[LDAPBrowser/Editor](#) - некоторые заметки по использованию

Инструменты ApacheDS

Инструменты [ApacheDS](#) - инструменты и утилиты

Раздел 5 - Безопасность LDAP

15. Безопасность LDAP

[15.1 Обзор безопасности OpenLDAP](#)
[15.4 Настройка TLS/SSL в OpenLDAP](#)

Приложения: Ресурсы

[Приложение A: Заметки и пояснения к LDAP](#)
[Приложение B: Ресурсы LDAP](#)
[Приложение C: LDAP RFC и документация](#)
[Приложение D: Глоссарий LDAP](#)
[Приложение E: Наборы схемы данных, объектные классы и атрибуты LDAP](#)

Технологическая информация о книге

[Список доработок - что ещё нужно сделать](#)

[Журнал изменений](#)

Глава 1. Соглашения и терминология

[1.1 Цели и сфера применения](#)
[1.2 Как читать данное руководство](#)
[1.3 Используемые терминология и условные обозначения](#)
[1.4 Благодарности](#)
[1.5 Авторские права и лицензия](#)
[1.6 Отказ от ответственности](#)

1.1 Цели и сфера применения

Данное открытое руководство предназначено для того, чтобы:

1. Разъяснить основные принципы LDAP и различных моделей (именования, функциональной, безопасности и информационной).
2. Предоставить все материалы, необходимые для понимания и развертывания подходящего решения на базе LDAP.
3. Предоставить информацию по проектированию LDAP-систем - выбору схем данных, объектных классов и атрибутов.
4. Выступать в качестве источника справочной информации по LDAP со ссылками на первоисточники.
5. Разъяснить наборы схем LDAP - что имеется и что делать, если чего-то не достаёт.

6. Предоставить руководство по быстрому запуску OpenLDAP в различных конфигурациях.
7. Предоставить HOWTO-руководства для часто требуемых конфигураций.
8. Разъяснить, как обеспечить безопасность реализации LDAP - механизм списков контроля доступа (Access Control List, ACL).
9. Разъяснить применение инструментов и утилит LDAP.
10. Разъяснить LDAP API, функциональную модель и LDAP URL.
11. Разъяснить механизм загрузки и сохранения данных - LDIF-файлы.
12. Предоставить ссылки и ресурсы LDAP, включая альтернативное Open Source программное обеспечение и инструменты LDAP.
13. Быть доступным On-line (HTML) и в пригодных для печати форматах (OpenOffice, DocBook и PDF).

В текущей версии этого документа реализовано несколько из вышеперечисленных задач.

1.2 Как читать данное руководство

Данное руководство - это не книга, его не нужно читать от корки до корки. Приведённая ниже обобщённая информация по разделам поможет разобраться кому, что и когда следует читать:

1. [Раздел 1](#): Обзор: материал общего назначения и уровня концепций. Прочитайте, если Вы хотите разобраться с концепциями LDAP, объектной моделью, иерархией, уникальными именами (Distinguished Names, DN), схемой данных, объектными классами и другой жаргонной терминологией. Если Вы планируете делать что-то большее, чем просто следовать HOWTO, Вам нужно понимать эти вещи.
2. [Раздел 2](#): Развёртывание и настройка OpenLDAP - некоторые распространенные конфигурации, такие как простой каталог, база данных учётных данных пользователей (для входа в систему) и пример с электронной почтой, используются для иллюстрации применения LDAP. Если Вам нужно быстро развернуть OpenLDAP, не сильно углубляясь в LDAP, используйте этот раздел. Материалы этого раздела, в основном, из серии 'делаем строго по инструкции'. Вы можете что-то действительно узнать здесь лишь по чистой случайности.
3. [Раздел 3](#): Справочники и сложные вопросы. Формат LDIF, LDAP URL, функциональная модель LDAP (чтение, модификация, запросы и т.п.) и LDAP C API (абстракции PHP и Ruby). Если Вы настроены серьезно, рано или поздно Вам понадобятся все эти вещи.
4. [Раздел 4](#): Работа с LDAP. HOWTO, возникающие проблемы и сообщения об ошибках, инструменты командной строки. Вопросы производительности.
5. [Раздел 5](#): Безопасность LDAP. Контроль доступа от грубого до тонкого, пароли и иерархический контроль.
6. [Приложение A](#): Если Вам нужны разъяснения, советы или некоторые справочные материалы по OpenLDAP и LDAP, используйте это приложение.
7. [Приложение B](#): Если Вы хотите найти альтернативное (не OpenLDAP) Open Source программное обеспечение LDAP, инструменты или ссылки, используйте это приложение.
8. [Приложение C](#): Если Вам нужны реальные вещи - в частности, все номера соответствующих RFC и спецификаций X.500 - используйте это приложение.
9. [Приложение D](#): Глоссарий ужасной терминологии, используемой в LDAP и X.500.
10. [Приложение E](#): Распространённые объектные классы и атрибуты - с большим количеством гиперссылок для упрощения просмотра. Ссылки на входящие в дистрибутив OpenLDAP стандартные наборы схемы данных и на некоторые другие, которые используем мы.

1.3 Используемая терминология и условные обозначения

В LDAP используется много терминов. Однако, на наш взгляд, одна из наиболее важных причин того, что мир LDAP с самого начала столь запутан - то, что терминология в найденных нами статьях и учебниках по LDAP непоследовательна сама по себе, да ещё и применяется непоследовательно. **Первосвященники** мира LDAP чувствуют себя комфортно в этих условиях, мы - нет. Кроме того, мы утверждаем, что причина такого большого количества повторяющихся вопросов в группах новостей или большого количества неоптимальных реализаций не в том, что пользователи глупы от природы, а в том, что они

попросту теряются.

Все термины, в том числе стандартные LDAP-термины, наши собственные термины и наше понимание других часто используемых в литературе терминов, мы объединили в [гlossарий](#). Там могут быть какие-то новые термины, изобретённые нами при попытке прояснить некоторые вопросы и концепции. Они могут оказаться неверными. Если так случилось, вежливо укажите нам на ошибки в рассуждениях, ТОЛЬКО не забудьте предоставить альтернативу. Мы всегда будем отвечать в том же духе, что и Вы нам напишете.

Условные обозначения

Условное обозначение	Описание и использование
....	Точки, появляющиеся во фрагментах наборов схемы и других файлах, показывают, что дополнительные строки, которые могут присутствовать (или не присутствовать), для краткости опущены. В реальном файле точек быть не должно.
a.k.a.	Также известен как (Also Known As). Указывает, что у термина есть альтернативная форма, возможно более широко используемая или используемая в литературе.

1.4 Благодарности

Благодарности в этом руководстве адресованы программистам, разработчикам документации, тестерам и администраторам, которые много и тяжело трудятся, чтобы претворить дальновидные концепции в жизнеспособную реальность, называемую Open Source.

Персоналии

Следующие люди сделали существенные комментарии к данному руководству - большое спасибо за то, что нашли время.

- Quanah Gibson-Mount

1.5 Авторские права и лицензия



Данная работа распространяется на условиях [Creative Commons License](#).

1.6 Отказ от ответственности

Автор и издатель приложили все усилия при подготовке данного руководства, чтобы обеспечить достоверность информации. Однако, информация, содержащаяся в данном руководстве, предоставляется без гарантий, явных или подразумеваемых. Ни автор, ни издатель не несут ответственности за любые убытки, вызванные или предположительно вызванные, прямо или косвенно, этим руководством.

Логотипы, торговые марки и символы, используемые в этом документе, являются собственностью соответствующих владельцев.

2. Понятия и обзор LDAP

Если Вы уже понимаете, что из себя представляет LDAP, для каких целей он подходит, что такое [схемы данных](#), [объектные классы](#), [атрибуты](#), [правила соответствия](#), [операционные объекты](#) и вся эта кухня — можете пропустить этот раздел. Однако, если Вы собираетесь делать что-то большее, чем следовать рецептам HOWTO, Вы должны понимать все эти вещи.

LDAP и X.500 имеют много общих терминов, некоторые из них важны, некоторые — просто ерунда. Для Вашего удобства мы создали [гlossарий](#), термины в который включены либо потому, что они важны, либо потому, что они часто используются в литературе.

- 2.1 Краткая история LDAP
- 2.2 Обзор LDAP
- 2.3 LDAP и базы данных
 - 2.3.1 Использование LDAP — резюме
- 2.4 Модель данных (объектная модель) LDAP
 - 2.4.1 Структура дерева объектов
 - 2.4.2 Атрибуты
 - 2.4.3 Объектные классы
 - 2.4.4 Описание дерева и добавление данных
 - 2.4.5 Навигация по дереву
- 2.5 Отсылки и репликация LDAP
 - 2.5.1 Отсылки
 - 2.5.2 Репликация

Небольшое замечание о чувствительности к регистру: здесь есть некоторая путаница, по крайней мере, мы считаем это путаницей. Единственные чувствительные к регистру вещи в LDAP — это пароли и содержимое **отдельных** (очень малоизвестных) атрибутов, в зависимости от их [правил соответствия](#). Всё. В этой и других документациях Вы будете встречать и objectclass, и objectClass, и даже ObjectClass. Все эти формы работают. Точка. После первых шести лет изучения LDAP (шутка, конечно, хватит и четырёх) Вы уже не будете хвататься за сердце всякий раз, когда Вам доведётся опечататься в каком-нибудь имени.

2.1 Краткая история LDAP

Когда-то в тёмном и далёком прошлом (конец 70-х — начало 80-х) [ITU \(International Telecommunication Union\)](#) начал работу над почтовыми стандартами серии X.400. Для этого стандарта требовался каталог имён (и другой информации), который мог бы быть доступен по сети в иерархической манере, схожей с DNS (для тех из Вас, кто знаком с её архитектурой).

Эта потребность в глобальном сетевом каталоге сподвигла ITU к разработке стандартов серии X.500 и, в частности X.519, которые определяют [DAP](#) (Directory Access Protocol), протокол для доступа к сетевой службе каталогов.

Серии стандартов X.400 и X.500 разрабатывались как составная часть полного стека OSI и были большими, громоздкими и потребляли много ресурсов. Фактически, стандартная ситуация для ITU.

Перенесёмся в начало 90-х. IETF осознала необходимость доступа к глобальным службам каталогов (первоначально, во многом, по тем же самым причинам хранения адресов электронной почты, что и ITU), но не поднимая при этом всех этих ужасных перенагруженных протоколов (OSI), и начала работу над Lightweight Directory Access Protocol (LDAP). LDAP разрабатывался так, чтобы обеспечить почти столько же функциональности, что и оригинальный стандарт X.519, но с использованием стека протоколов TCP/IP, при этом оставляя возможность взаимодействия с каталогами, основанными на X.500. Действительно, взаимодействие с X.500 (DAP) и его отображение всё ещё является частью серии RFC о LDAP от IETF.

Большинство вопросов в спецификациях LDAP, вызывающих серьёзную головную боль, как раз связаны с обратной совместимостью с X.500 и концепцией глобальной службы каталогов. Самый показательный из них — соглашение об [именовании корневой записи](#).

В широком смысле, LDAP отличается от DAP в следующих аспектах:

1. В LDAP используется TCP/IP — DAP использует OSI в качестве транспортного/ сетевого слоёв.
2. Некоторое сокращение функциональности — неясные, дублирующие и редко используемые функции X.519 (конёк ITU) тихо и благополучно отброшены.
3. Замена в LDAP **некоторых** из ASN.1-нотаций (X.519) текстовым представлением (LDAP URL и поисковые фильтры). В этом вопросе IETF не снискала нашей безграничной благодарности: значительное большинство ASN.1-нотаций всё ещё остаются в прежнем виде.



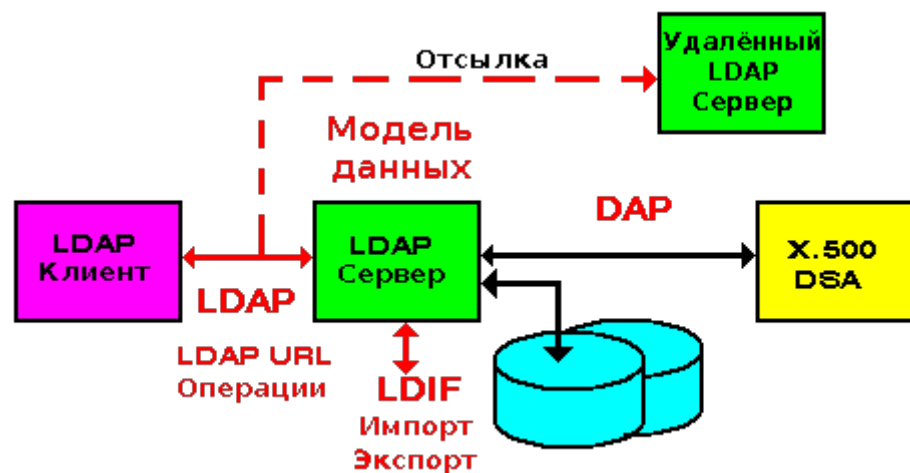
2.2 Обзор LDAP

Технически, LDAP — это всего лишь протокол, определяющий методы, посредством которых осуществляется доступ к данным каталога. Он также определяет и описывает, как данные **представлены** в службе каталогов (**Модель данных**). Наконец, он определяет, каким образом данные загружаются (импортируются) и выгружаются (экспортируются) из службы каталогов (с использованием **LDIF**). LDAP не определяет, **как** происходит хранение и манипулирование данными. С точки зрения стандарта хранилище данных и методы доступа к нему — это "чёрный ящик", за который, как правило, отвечают модули **back-end** (механизмы манипуляции данными) какой-либо конкретной реализации LDAP.

LDAP определяет четыре **модели**, которые мы перечислим и кратко обсудим, а затем Вы можете спокойно забыть о них, поскольку для понимания LDAP они мало что дают.

1. **Информационная модель:** мы склонны использовать термин **модель данных**, на наш взгляд он более интуитивный и понятный. Модель данных (или информационная модель) определяет, каким образом информация или данные **представлены** в системе LDAP. Это может совпадать или не совпадать с фактическим методом представления данных в хранилище. Как упоминалось выше, вопрос хранилищ данных лежит за пределами стандартов LDAP.
2. **Модель именования:** определяет все вещи наподобие 'dc=example,dc=com', с которыми Вы сталкиваетесь в системах LDAP. Здесь мы максимально придерживаемся спецификации, поскольку эти термины используются очень широко.
3. **Функциональная модель:** при чтении, поиске, записи или модификации LDAP Вы используете функциональную модель — кто бы мог подумать!
4. **Модель безопасности:** Вы можете контролировать, причём весьма детально, кто, что и с какими именно данными может сделать. Это сложная, но мощная штука. Мы постепенно внедрялись в данную концепцию и посвятили ей отдельную [главу](#). На начальном этапе можно забыть о безопасности. Вы всегда можете вернуться и модернизировать безопасность в LDAP. Если модернизация впоследствии будет невозможна, мы будем описывать реализацию безопасности по тексту.

Сфера стандартов LDAP показана на диаграмме ниже. Обозначенные красным вещи определены в протоколе (различных RFC, определяющих LDAP). Происходящее же в "чёрных ящиках" (или, в данном случае, в зелёных, жёлтых и малиновых ящиках) выходит за рамки стандартов.



Каждый компонент кратко описан здесь, немного подробнее ниже и в мельчайших деталях в последующих главах. Но для начала четыре важных момента:

1. LDAP не определяет, каким образом данные хранятся, только каким образом к ним осуществляется доступ; НО большинство реализаций LDAP используют стандартные базы данных в качестве механизмов манипуляции данными, и лишь OpenLDAP предлагает выбор поддерживаемых механизмов манипуляции данными.
2. Когда Вы общаетесь с сервером LDAP, Вы понятия не имеете, откуда поступают данные: на самом деле одна из ключевых задач стандарта — скрывать такой уровень детализации. Теоретически, данные могут поступать из одной ИЛИ НЕКОЛЬКИХ локальных баз данных, либо одной или нескольких служб X.500. Откуда и каким образом Вы будете получать данные — это детали реализации, они важны только на этапе определения конфигурации Вашего LDAP-сервера

(серверов).

3. Две концепции, — доступ к службе LDAP и эксплуатация службы LDAP, — должны быть чётко разделены в Вашем сознании. Когда Вы проектируете каталог, сконцентрируйтесь на том, чего Вы хотите от него добиться, для чего он будет предназначен (схема и организация данных) и не думайте о реализации. Затем, во второй фазе, во время конфигурации LDAP, сконцентрируйтесь на том, где данные будут располагаться, какие будут применяться системы хранения.
4. Ряд коммерческих продуктов баз данных предоставляет LDAP-представление (LDAP-обёртку или LDAP-абстракцию) реляционной базы данных или базы данных другого типа.



2.3 LDAP и базы данных

LDAP характеризуется как сервис "один раз записал — много раз прочитал". Другими словами, от данных, обычно хранящихся в каталоге LDAP, не ожидается, чтобы они менялись при каждом доступе. Для иллюстрации: LDAP НЕ подходит для ведения учета банковских операций, так как они, по своей природе, изменяются при каждом доступе (сделке). С другой стороны, LDAP как нельзя лучше подходит для учета различных аспектов деятельности банка, меняющихся значительно реже: отделений, часов работы, сотрудников и т.д.

Оптимизация на чтение

Во фразе "один раз записал — много раз прочитал" до конца не ясно, насколько много это **много**?

Где проходит грань разумного использования между LDAP и классическими, ориентированными на транзакции реляционными базами данных, к примеру, MySQL, PostgreSQL? Если обновление происходит при каждом втором доступе, будет ли разумно использовать в таком приложении LDAP, или нужно, чтобы обновления происходили раз в тысячу или в миллион обращений?

Литература по этой теме крайне редка и имеет тенденцию придерживаться таких бесприкрытых LDAP-приложений, как адресные книги, которые изменяются, возможно, раз в столетие.

Простого ответа на этот вопрос нет, однако следующие замечания могут оказаться полезными:

1. Увеличение нагрузки при операциях записи происходит из-за обновления индексов. Чем больше индексов (для ускорения поиска), тем, по возможности, реже должны выполняться обновления каталога. Соотношение чтение:запись в хорошо оптимизированных на чтение каталогах должно быть 1000:1 и даже больше.
2. Репликация LDAP генерирует несколько транзакций для каждого обновления, таким образом, желательно снизить практическую нагрузку, связанную с обновлениями (1000:1 или больше).
3. Если объём данных велик (скажем, больше 10 000 записей), то даже при небольшом количестве индексов время обновления может быть значительным. Поэтому желательно снизить количество обновлений до минимально возможного (10 000:1).
4. Если объём данных относительно невелик (скажем, меньше 1000 записей), индексов немного и отсутствует репликация, мы не видим рационального объяснения, почему Вы не можете использовать LDAP в форме, приближенной к транзакционной системе, то есть каждые 5-10 доступов на чтение чередуются циклом записи (**модификации** на жаргоне LDAP).
5. Мы полагаем, что настоящий ответ на этот вопрос (с уважением к памяти ушедшего от нас [Douglas Noel Adams](#)): соотношение количества чтений к количеству записей составляет 42!

Представление организации данных

Применяемые для доступа к каталогам LDAP примитивы (элементы протокола LDAP) используют модель данных, которая (возможно) абстрагирована от её физической организации. Эти примитивы обеспечивают **представление** данных в виде объектной модели и не заботятся об актуальной структуре данных. В действительности, относительная простота LDAP происходит именно от этой характеристики. Конкретная

реализация будет выполнять отображение примитивов LDAP в физическую организацию данных, используя свой **механизм манипуляции данными** как "чёрный ящик" в чистом виде.

В этом резкий контраст со, скажем, SQL, поскольку в SQL-запросах, используемых для получения данных, присутствует полное и детальное представление структур данных, их организация в таблицы, объединения и т.д.

Синхронизация и репликация данных

Реляционные и транзакционные базы данных идут на крайние меры для обеспечения целостности данных во время циклов записи/обновления, используя такие техники, как транзакции, блокировки, откаты и другие методы. Для такого типа баз данных это жизненно важные и необходимые требования. Данная экстремальная форма синхронизации данных также действует при репликации данных на несколько хостов или серверов. Все представления данных будут соответствовать друг другу.

Главный LDAP-сервер и подчинённые ему серверы (или равноправные ему серверы в среде с несколькими главными серверами (multi-master)) используют простой асинхронный процесс репликации данных. В связи с этим во время цикла репликации возможна рассинхронизация данных в главной и подчинённой (или равноправной) системах. Во время этого (обычно короткого) периода времени запрос к главному и подчинённому серверам **может** дать разные результаты. Если вследствие такого расхождения мир с треском расколется пополам, значит для подобного приложения LDAP не подойдёт. Если же Bob Smith на несколько секунд (или даже меньше) будет числиться в бухгалтерии на одном LDAP-сервере, а на другом — в отделе продаж, вряд ли это кого-то сильно огорчит. В эту категорию попадает удивительно большое количество приложений.

Примечание: Современные реализации LDAP, особенно те, которые поддерживают конфигурации с несколькими главными серверами (Multi-master), становятся всё более изощренными в репликации обновлений. Кроме того, высокоскоростные сети связи позволяют значительно быстрее выполнять операции репликации. Однако, подобные решения всего лишь уменьшают промежуток времени, в течение которого две какие-либо системы будут рассинхронизированы, они не устраняют саму природную особенность LDAP, заключающуюся в возникновении рассинхронизации, даже если в большинстве современных реализаций таковая длится всего лишь доли секунды.

2.3.1 Использование LDAP — резюме

Так в чём же преимущества LDAP (каталогов), и почему каждый здравомыслящий человек будет их использовать?

Прежде чем попытаться ответить на этот вопрос, давайте абстрагируемся от тактических соображений производительности. В целом, реляционные СУБД всё ещё значительно быстрее реализаций LDAP. По мере разработки служб каталогов второго поколения это положение меняется, и, хотя реляционные СУБД всегда будут оставаться быстрее LDAP, разрыв значительно сократился вплоть до точки, в которой различия становятся уже практически несущественными, если, конечно, Вы сравниваете подобное с подобным (единичные сетевые транзакции, а не обновление высокоиндексированного атрибута при каждой операции — в этом случае, не обессудьте, Вы получите (или не получите) ровно столько, сколько заслужили).

Так почему же нужно использовать LDAP? Вот наш список ключевых характеристик, которые делают высокий (всё ещё) уровень боли терпимым.

1. LDAP предоставляет стандартизированные как удалённый, так и локальный методы доступа к данным. Таким образом, вполне реально заменить одну реализацию LDAP на другую, совершенно не влияя на внешний интерфейс доступа к данным. Реляционные СУБД предоставляют стандарты локального доступа, такие как SQL, но удалённые интерфейсы всегда остаются проприетарными.
2. Поскольку в LDAP используются стандартизированные методы доступа к данным, клиенты и серверы могут разрабатываться отдельно или быть получены из разных источников. В продолжении этой темы, LDAP может быть использован для абстрактного представления данных, содержащихся в ориентированных на транзакции базах данных, скажем, для выполнения пользовательских запросов, позволяя пользователям прозрачно (для LDAP-запросов) обращаться к разным поставщикам транзакционных баз данных.

3. LDAP предоставляет метод, посредством которого данные могут быть перемещены (делегированы) в несколько мест, не затрагивая при этом внешнего доступа к этим данным. Используя метод отсылок LDAP, данные могут быть перемещены на альтернативные LDAP-серверы путём только изменения операционных параметров.
4. Системы LDAP могут быть настроены на репликацию данных одному или нескольким приложениям только за счет операционных параметров, без необходимости добавления дополнительного кода или изменения внешнего доступа к данным.

Приведённые выше постулаты концентрируют внимание исключительно на стандартной природе доступа к данным LDAP, не учитывая отношения количества чтений к количеству записей, которое, как отмечалось выше, зависит от количества обслуживаемых операционных индексов. В данном случае неявно игнорируется использование каталогов для обработки транзакций, хотя и здесь есть признаки того, что реализации LDAP учитывают такие возможности.



2.4 Модель данных (объектная модель) LDAP

Каталоги LDAP используют модель данных, которая **считает** или **представляет** данные как иерархию объектов. Это не означает, что LDAP является объектно-ориентированной базой данных. Как уже отмечалось выше, LDAP сам по себе является протоколом, позволяющим получить доступ к службам LDAP, и не определяющий, каким образом данные хранятся, — но операционные примитивы (чтение, удаление, модификация) работают с моделью (описанием) данных, имеющих (в основном) объектоподобные характеристики.

2.4.1 Структура дерева объектов

В этом разделе определяется сущность LDAP. Если Вы поймёте этот раздел и различные термины и отношения, с ним связанные, Вы поймёте LDAP.

В LDAP-каталоге данные представлены как иерархия объектов, каждый из которых называется **записью**. Полученная в результате древовидная структура называется **Информационным деревом каталога (Data Information Tree, DIT)**. Верхнюю часть данного дерева обычно называют **корнем (root)**, (а также **базой (base)** или **суффиксом (suffix)**).

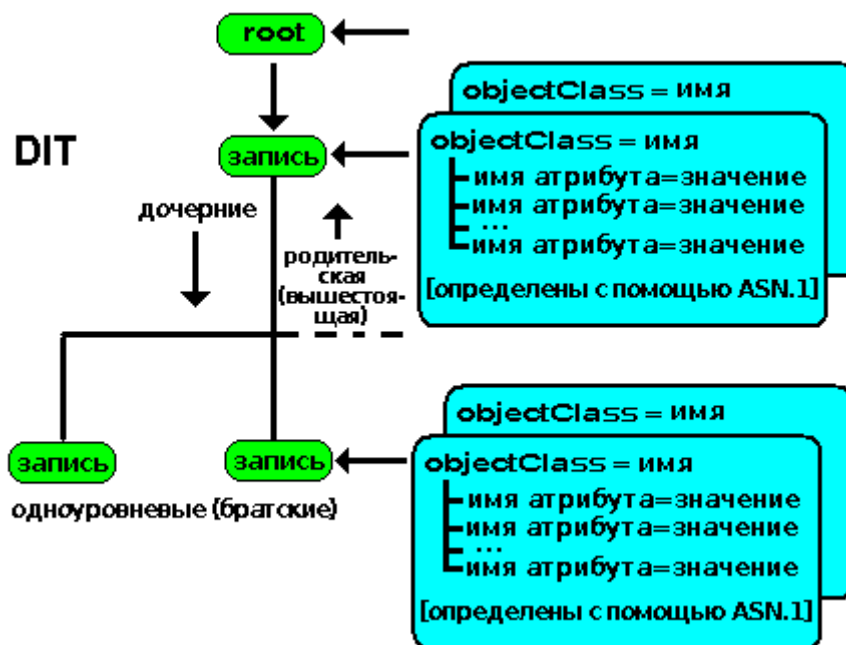
У каждой **записи** есть одна родительская запись (объект) и ноль или более дочерних записей (объектов). Каждая дочерняя запись (объект) является одноуровневой (братской) по отношению к другим дочерним записям своей родительской записи.

Каждая запись состоит из (является экземпляром) одного или нескольких **объектных классов (objectClass)**. **Объектные классы** содержат ноль или более **атрибутов (attribute)**. Атрибуты имеют имена (и, иногда, аббревиатуры или псевдонимы) и обычно содержат данные (наконец-то!).

Характеристики (свойства) **объектных классов** и их **атрибутов** описываются определениями **ASN.1**.

Уф! Теперь Вы знаете всё, что только можно знать о LDAP. Остальное — детали. Да, этих деталей, пожалуй, очень много. Но суть LDAP именно в этом.

Представленная ниже диаграмма иллюстрирует эти отношения:



Подытожим:

1. Каждая **запись** состоит из одного или нескольких **объектных классов**.
2. У каждого **объектного класса** есть имя.
3. У каждого **атрибута** есть имя, он является членом **объектного класса** и обычно содержит данные.

Теперь можно смело брать выходной на остаток дня и хорошенько отпраздновать!



2.4.2 Атрибуты

Каждый **атрибут** имеет имя и обычно содержит данные. **Атрибуты** всегда связаны (являются членами) с одним или несколькими **объектными классами**. У **атрибутов** есть ряд интересных особенностей:

1. Все **атрибуты** являются членами одного или нескольких **объектных классов**.
2. Каждый **атрибут** определяет **тип данных**, которые он может содержать.
3. **Атрибуты** могут быть необязательными (ключевое слово MAY) или обязательными (ключевое слово MUST), согласно определениям ASN.1 того **объектного класса**, членами которого они являются. **Атрибут** может быть необязательным в одном **объектном классе** и обязательным в другом. Это свойство определяется в рамках **объектного класса**.

В разных местах этой документации в записях-примерах подобраны совершенно различные атрибуты, что может показаться путаницей. На самом же деле так происходит из-за необязательного характера большинства атрибутов. Это позволяет при составлении записей использовать подход "выбирай и соединяй": находим нужный нам **атрибут**, подключаем к записи его **объектный класс**, и надеемся, что все остальные **атрибуты** из данного **объектного класса**, которые мы не хотим использовать, окажутся необязательными! Чтобы яснее это себе представить, попробуйте посмотреть [здесь](#).

4. У **атрибутов** может быть single (одно) или multi (несколько) значений (как описано в их определениях ASN.1). Single означает, что только одно значение данных может быть задано для этого **атрибута**. Multi означает, что для данного **атрибута** может быть задано одно или несколько значений данных. Если **атрибут** описывает, скажем, адрес электронной почты, у него может быть одно, два или 500 значений — это один из ряда методов работы с почтовыми псевдонимами, применяемых при построении каталога. Значением по умолчанию для атрибута является multi (что позволяет иметь несколько значений).
5. У **атрибутов** есть имена и, иногда, псевдонимы или аббревиатуры (как описано в их определениях ASN.1), например, **commonName** является членом **объектного класса**, называемого **person** (и многих других), и имеет сокращённое имя **cn**. Для ссылки на этот **атрибут** может использоваться

как **commonName**, так и **cn**.

6. На всех уровнях иерархии данные, содержащиеся в каком-то из **атрибутов**, могут использоваться для однозначной идентификации **записи**. Это может быть любой **атрибут** в **записи**. Это может быть даже комбинация двух или более атрибутов.

Предположим, у Вас есть классический каталог "белые страницы", содержащий имена, номера телефонов, адреса, любимые напитки (да-да, существует стандартный атрибут **favoritedrink**), и т.д. Чтобы однозначно идентифицировать конкретную запись, Вы можете выбрать имя человека (атрибут **commonName** или **cn**). Если имя, например "Bob Smith", не является уникальным в данном каталоге LDAP, то при поиске "Bob Smith" будут возвращены все записи в каталоге, содержащие имя "Bob Smith", и пользователю нужно будет выбирать, какая ему больше нравится. При чтении и поиске это может оказаться приемлемо или даже желательно, поскольку люди часто используют такую широко известную информацию, как имя человека.

При сохранении или обновлении записи недопустимо, чтобы "Bob Smith" был не абсолютно уникальным — какую из возвращённых записей нам следует обновить? В таком случае может потребоваться выбрать другой атрибут, являющийся абсолютно уникальным. Вполне допустимо использовать более одного атрибута для доступа к данным в зависимости от контекста, например имя человека при чтении и поиске, а при сохранении — телефонный номер. Кроме того, как упоминалось выше, допустимо использовать более одного атрибута для определения уникальности записи (**cn+favoritedrink!**)

Атрибут (атрибуты), выбранные для хранения данных, составляющих уникальность записи, называются **атрибутом (атрибутами) именованного** или **относительным уникальным именем (Relative Distinguished Name, RDN)** — но подробнее об этом позже.

Взгляните на некоторые распространённые **объектные классы и атрибуты**. На данном этапе это выглядит очень страшно. Просто забудьте о том, что Вы только что видели, и продолжим чтение.

И ещё кое-что об атрибутах — но только, если Вы чувствуете себя хорошо и спокойно, в противном случае продолжайте читать здесь.



2.4.3 Объектные классы

Объектные классы являются, по существу, пакетами **атрибутов**. Существует огромное число предопределённых **объектных классов**, в каждом из которых полным-полно **атрибутов** для почти всех возможных приложений. Но того, который НУЖЕН Вам, конечно же ещё не определили! У **объектных классов** есть ещё две характеристики:

1. **Объектный класс** определяет, **должен (MUST)** ли входящий в него атрибут присутствовать в записи (**обязательный атрибут**), или он **может (MAY)** присутствовать (**необязательный атрибут**).
2. **Объектный класс** может быть частью иерархии, в этом случае он **наследует** все характеристики своих родительских **объектных классов**.

Вот неполный список наиболее часто используемых **объектных классов** и их **атрибутов**.

Ещё кое-что об объектных классах — но только, если Вы чувствуете себя хорошо и спокойно, в противном случае продолжайте читать здесь.



2.4.4 Описание дерева и добавление данных

В конце концов мы хотим поместить немного данных в наш каталог и использовать-таки эту замороченную штуку.

Описание древовидной структуры и первоначальное наполнение данными осуществляется путем добавления записей (с ассоциированными с ними объектными классами и атрибутами), начиная от корневой записи DIT и двигаясь вниз по иерархии. Таким образом, родительская запись всегда должна

быть добавлена перед тем, как пытаться добавить дочерние записи. Добавление записей может происходить разными путями, один из которых — использовать файлы формата **LDAP Data Interchange Files (LDIF)**, который полностью описан в [одной из последующих глав](#). Файлы LDIF — это текстовые файлы, описывающие древовидную иерархию — [информационное дерево каталога \(Directory Information Tree, DIT\)](#), и данные, добавляемые в каждый атрибут. Ниже приведён простой пример LDIF-файла, описывающий корневое DN (dc=example,dc=com) и добавляющий одну дочернюю запись в ветку **people**.

Примечания:

1. На данном этапе не так важно досконально разбираться во всех значениях этого LDIF-файла. В примерах [главы 5](#) охвачены подробности настройки LDIF-файлов, а в [главе 8](#) LDIF-файлы разъясняются в красочных деталях. На данном этапе достаточно знать, что LDIF-файлы могут использоваться для создания DIT и выглядят примерно так, как приведено ниже.
2. Добавление LDAP-записей может быть также выполнено с помощью LDAP-клиентов, таких как [LDAP-браузеры](#) общего назначения (смотрите [LDAPBrowser/Editor](#)) или специализированные приложения.

```
version: 1

## указание версии не является строго необходимым (и некоторые реализации его отклоняют),
## но, в общем случае, это хорошая практика

## Определяем DIT ROOT/BASE/SUFFIX #####
## используется формат RFC 2377 (доменные имена)

## dcObject - это ВСПОМОГАТЕЛЬНЫЙ объектный класс и, кроме него, запись
## ДОЛЖНА иметь СТРУКТУРНЫЙ объектный класс (в данном случае, organization)
# это последовательность ЗАПИСИ и ей предшествует ПУСТАЯ СТРОКА

dn: dc=example,dc=com
dc: example
description: Лучшая компания в целом свете
objectClass: dcObject
objectClass: organization
o: Example, Inc.

## ПЕРВЫЙ уровень иерархии - люди (people)
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой

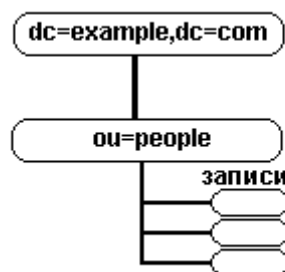
dn: ou=people, dc=example,dc=com
ou: people
description: Все люди в организации
objectClass: organizationalUnit

## ВТОРОЙ уровень иерархии - записи людей
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой

dn: cn=Joe Schmo,ou=people,dc=example,dc=com
objectclass: inetOrgPerson
cn: Joe Schmo
sn: Schmo
uid: jschmo
mail: joe@example.com
mail: j.schmo@example.com
ou: sales
```

Важное замечание: Строки в приведённом выше LDIF-файле, начинающиеся с 'dn:', по существу сообщают LDAP-серверу, каким образом структурировать или располагать запись в DIT. В общем случае не важно, значение какого атрибута используется для этой цели, при условии уникальности 'dn:'. В последней записи приведённого примера для этой цели было выбрано cn=Joe Schmo. Точно также могло быть выбрано, скажем, uid=jschmo. При LDAP-поиске может применяться любая комбинация атрибутов и запись будет найдена независимо от того, какое значение было использовано в 'dn:' при её создании. Однако, если запись планируется использовать для аутентификации пользователя, скажем, для входа в систему или организации технологии единого входа (Single Sign-On), значение 'dn:' становится чрезвычайно важным и определяет идентификатор входа в систему (или, на жаргоне, DN подключения). Иногда (особенно в контексте LDAP, используемого в Microsoft AD) он называется DN принципа (Principal DN), хотя этот термин не используется в определениях стандартов LDAP. [Дополнительная информация по этой теме](#).

Мы расскажем о LDIF-файлах позже, поскольку они нам ещё понадобятся, а приведённый выше LDIF определяет следующую структуру:



После того, как DIT определён и запущен в работу, в дальнейшем информацию в него можно добавлять с помощью LDIF, LDAP-браузера, web-интерфейса или другого программного интерфейса.

Используя LDIF-файлы, данные также можно экспортировать (сохранить) в качестве резервной копии или для других целей.



2.4.5 Навигация по дереву

После того, как мы поместили данные в наше дерево (DIT), обычно самое время начать с ними работать!

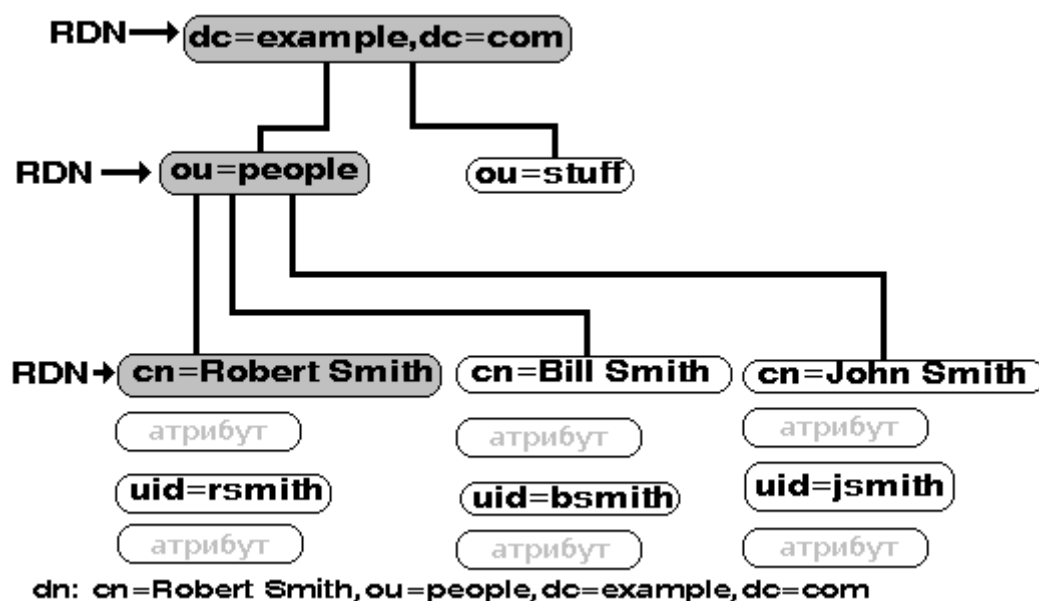
Для этого мы должны посылать команды (на чтение, поиск, модификацию и т.д.) нашему LDAP-серверу, а чтобы сделать это, мы должны быть в состоянии сообщить LDAP-серверу, где находятся данные (для записи) или хотя бы где они примерно находятся (для поиска).

Короче говоря, мы должны осуществлять навигацию (ориентироваться) в каталоге.

Ранее мы определили, что у **атрибутов** есть имена, и что на каждом уровне иерархии один или несколько **атрибутов** должны содержать данные, каким-то образом уникально идентифицирующие каждую запись.

Построив пути из этих атрибутов именования и содержащихся в них данных, мы можем добраться до нужной нам записи или точки начала поиска.

Такие пути причудливо называются уникальными или отличительными именами (Distinguished Names, DN). Каждый атрибут с уникальными данными, являющийся частью этого DN, называется относительным уникальным именем (Relative Distinguished Name, RDN). Можете с нами не согласиться, но способность групп стандартизации LDAP и X.500 генерировать оригинальную терминологию не подлежит сомнению. Следующая диаграмма иллюстрирует DN и RDN.



Дальнейшее объяснение с некоторыми рабочими примерами.

2.5 Отсылки и репликация LDAP

Одним из наиболее мощных аспектов LDAP (и X.500) является вложенная в них способность делегировать ответственность за поддержание части каталога другому серверу, сохраняя при этом общую картину каталога как единого целого. Таким образом, можно создать в каталоге делегирование ответственности (**отсылку (referral)** в терминологии LDAP) той части всего каталога, в которой описан конкретный отдел, LDAP-серверу этого отдела. В этом аспекте LDAP почти полностью отражает концепцию делегирования DNS.

В отличие от системы DNS, в стандартах не предусмотрена возможность сообщить LDAP-серверу проследовать по отсылке (разрешить отсылку), LDAP-клиенту оставлена возможность напрямую обращаться к новому серверу, используя возвращённую отсылку. Равным образом, поскольку в стандарте не определена организация данных LDAP, возможность перехода LDAP-сервера по ссылке (разрешение ссылки) не противоречит стандартам и некоторые LDAP-серверы выполняют эту функцию автоматически, используя процесс, обычно называемый **сцеплением (chaining)**.

OpenLDAP буквально следует стандартам и по умолчанию не выполняет **сцепление**, а всегда возвращает отсылку. Однако OpenLDAP может быть настроен на выполнение **сцепления** с помощью директивы [overlay chain](#).

Встроенная функция репликации LDAP позволяет создать одну или несколько подчинённых копий каталога (DIT) от одной главной (и даже, в некоторых реализациях, распространять изменения между несколькими главными копиями DIT), таким образом, по сути, создаётся устойчивая структура.

Важно, однако, подчеркнуть разницу между LDAP и транзакционными базами данных. При выполнении обновления в главном каталоге LDAP, для обновления подчинённых копий каталога (или других главных копий в конфигурации с несколькими главными серверами) может потребоваться некоторое время (в компьютерной терминологии) — главный и подчинённый каталоги (или разные главные каталоги в конфигурации с несколькими главными серверами) могут быть рассинхронизированы в течение какого-то периода времени.

В контексте LDAP, временное отсутствие синхронизации DIT считается несущественным. В случае же транзакционной базы данных, даже временное отсутствие синхронизации считается катастрофическим. Это подчёркивает различия в характеристиках данных, которые должны храниться в каталоге LDAP и в транзакционной базе данных.

Конфигурация репликации и отсылок [рассматривается далее](#), а также в [приводимых примерах](#).

2.5.1 Отсылки LDAP

Рисунок 2.5-1 демонстрирует поисковый запрос с базовым DN `dn:cn=thingie,o=widgets,dc=example,dc=com` к LDAP-системе с отсылками, который полностью удовлетворяется первым LDAP-сервером (LDAP1):

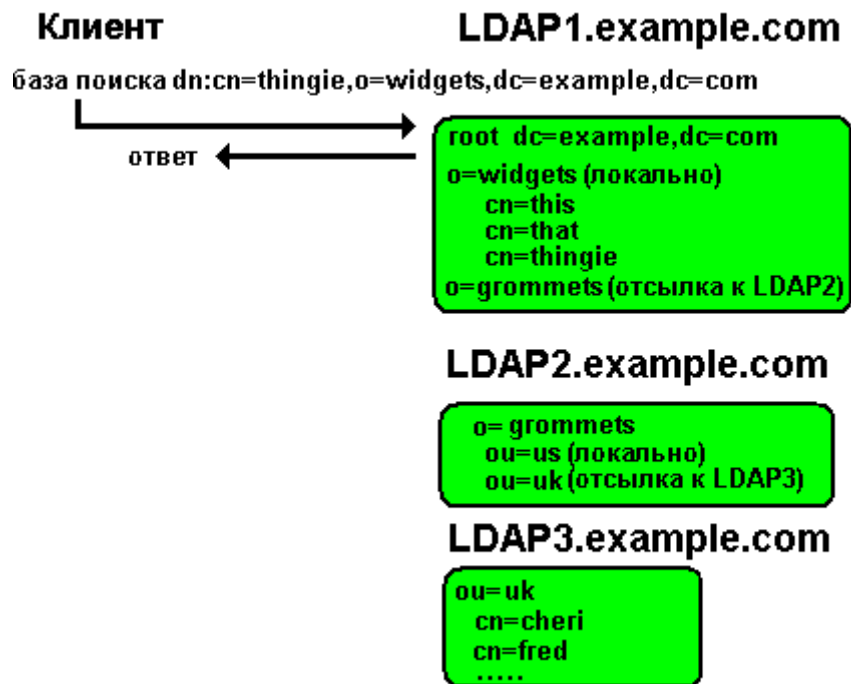


Рисунок 2.5-1 — Запрос, удовлетворяемый только от LDAP1

Рисунок 2.5-2 демонстрирует поисковый запрос с базовым DN dn:cn=cheri,ou=uk,o=grommets,dc=example,dc=com к LDAP-системе с отсылками, ответ на который возвращается в результате серии отсылок к серверам LDAP2 и LDAP3:

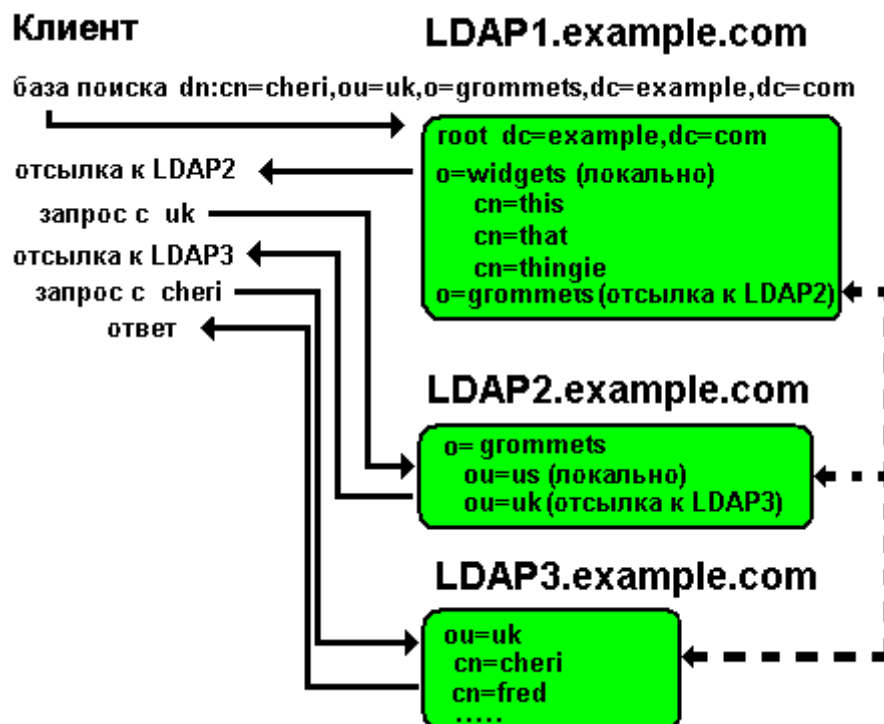


Рисунок 2.5-2 — Запрос, генерирующий отсылки к LDAP2 и LDAP3

Примечания:

1. Все клиентские запросы начинаются с обращения к глобальному каталогу LDAP1.
2. На сервере LDAP1 запросы любых данных, содержащие **widgets** в качестве **RDN** в **DN**, удовлетворяются непосредственно из LDAP1, например:

```
dn: cn=thingie,o=widget,dc=example,dc=com
```

3. На сервере LDAP1 запросы любых данных, содержащие **grommets** в качестве **RDN** в **DN**, генерируют отсылку на сервер LDAP2, например:

```
dn: uid=cheri,ou=uk,o=grommets,dc=example,dc=com
```

4. На сервере LDAP2 запросы любых данных, содержащие **uk** в качестве **RDN** в **DN**, генерируют отсылку на сервер LDAP3, например:

```
dn: uid=cheri,ou=uk,o=grommets,dc=example,dc=com
```

5. Если LDAP-сервер сконфигурирован на выполнение сцепления (то есть на следование отсылкам, как показано альтернативными пунктирными линиями), то LDAP-клиенту будет отправлен один единственный ответ. Сцепление контролируется конфигурацией LDAP-сервера и значениями в поисковых запросах. Информация по сцеплению [здесь](#).
6. На рисунках показано явное сцепление с использованием объектного класса **referral**. Серверы OpenLDAP могут быть [настроены так](#), чтобы возвращать общую отсылку в случае, если запрашиваемый DN не был найден во время операции поиска.

2.5.2 Репликация LDAP

Функции репликации позволяют копировать обновления LDAP DIT на одну или несколько LDAP-систем в целях резервирования и/или повышения производительности. В этом контексте стоит подчеркнуть, что репликация работает на уровне DIT, а не на уровне LDAP-сервера, поскольку на одном LDAP-сервере может обслуживаться несколько DIT. Репликация происходит периодически, в течение промежутка времени, известного как **время цикла репликации** (по сути, это время, необходимое для отправки обновленных данных на сервер-реплику и получения подтверждения об успешном завершении операции). В общем случае, существуют методы уменьшения времени цикла репликации с помощью настроек, но обычно они приводят к снижению производительности или увеличению нагрузки на сеть. Исторически для выполнения репликации в OpenLDAP использовался отдельный демон (slurpd), но, начиная с версии 2.3, стратегия репликации коренным образом изменилась, были достигнуты серьезные улучшения в гибкости и возможностях настройки времени цикла репликации. Существует два возможных типа конфигураций репликации, у каждого из которых есть несколько вариантов.

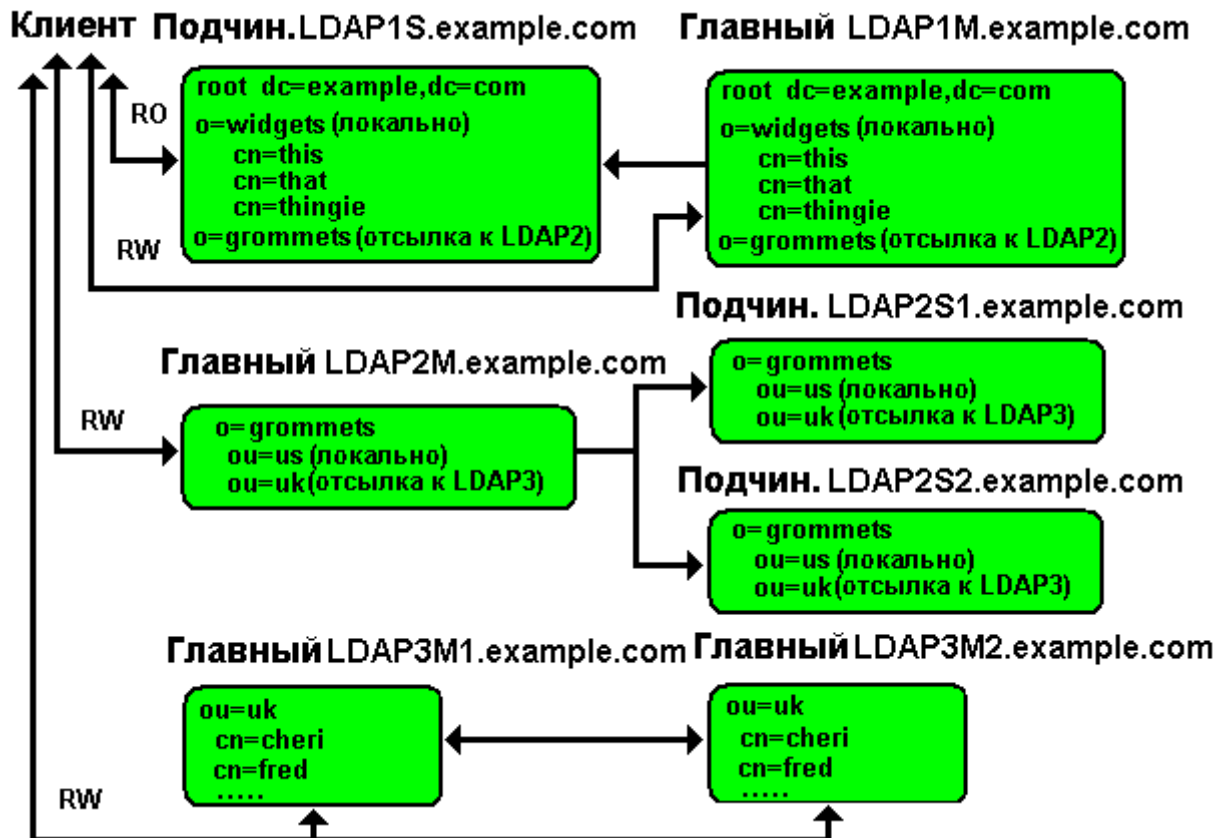
1. **Главный-подчиненный (Master-Slave):** В конфигурации главный-подчиненный обновляется одно единственное DIT (на жаргоне OpenLDAP оно называется главным (master) или поставщиком репликации (provider)), и эти обновления реплицируются или копируются на один или несколько указанных LDAP-серверов, на которых запущены подчиненные DIT (на жаргоне OpenLDAP они называются потребителями репликации (consumer)). Подчиненные серверы оперируют с доступной только для чтения копией главного DIT. Пользователи, которые выполняют только чтение, будут превосходно себя чувствовать, работая с серверами, содержащими подчиненные DIT, а пользователи, которым нужно вносить изменения в каталог, должны обращаться к серверу, содержащему главное DIT. При определенных условиях конфигурация главный-подчиненный позволяет существенно сбалансировать нагрузку. Однако, у неё есть два очевидных недостатка:
 - Если всем (большинству пользователей) дана возможность (требуется) вносить изменения в DIT, то либо им потребуется получать доступ к одному серверу (с подчиненным DIT) для осуществления чтения и к другому серверу (с главным DIT) для выполнения обновлений, либо они всегда будут обращаться к серверу, на котором запущено главное DIT. В последнем случае репликация выполняет только функцию резервного копирования.
 - Поскольку существует только один сервер, содержащий главное DIT, он представляет собой единую точку отказа для операций записи, которая может повлиять на работоспособность всей системы (хотя, в случае серьезного сбоя, подчиненное DIT может быть переконфигурировано для работы в качестве главного).
2. **Несколько главных (Multi-Master):** В конфигурации с несколькими главными серверами могут обновляться несколько главных DIT, запущенных на одном или нескольких серверах, и результаты обновлений распространяются на другие главные серверы.

Исторически OpenLDAP довольно долго не поддерживал операции с несколькими главными серверами, но в версии 2.4 окончательно введены такие возможности. В этом контексте, наверное, стоит отметить две вариации общей проблемы **конкуренции обновлений**, специфичной для всех конфигураций с несколькими главными серверами. Эта проблема выявлена проектом OpenLDAP для

своей конфигурации с несколькими главными серверами, но она относится ко всем системам LDAP:

1. **Конкуренция значений.** Если выполняется два обновления одного и того же атрибута в одно и то же время (в пределах времени цикла репликации) и при этом атрибуту назначаются разные значения, то, в зависимости от типа атрибута (SINGLE или MULTI-VALUED), результирующая запись может быть в неправильном или непригодном для использования состоянии.
2. **Конкуренция удаления.** Если один пользователь добавляет дочернюю запись в то же самое время (в пределах времени цикла репликации), когда другой пользователь удаляет оригинальную запись, то удалённая запись вновь появится.

Рисунок 2.5-3 показывает несколько возможных конфигураций репликации.



Примечания:

1. **RO** = только чтение, **RW** = чтение-запись
2. В случае LDAP1 клиент взаимодействует с подчинённой системой, доступной только для чтения. Для выполнения операций модификации (записи) клиенты должны обращаться к главной системе.
3. В случае LDAP2 клиент взаимодействует с главной системой, которая реплицируется на две подчинённые.
4. LDAP3 является системой с несколькими главными серверами и для выполнения операций чтения и записи клиенты могут обращаться к любому серверу. В данной конфигурации каждый главный сервер может, в свою очередь, иметь одно или несколько подчинённых DIT.

Глава 3 — Схема данных, объектные классы и атрибуты LDAP

Глава 3. Схема данных, объектные классы и атрибуты LDAP

Эта глава не для слабонервных. Вы можете либо прочитать её сейчас, либо перейти к разделу [примеров](#), где много обратных ссылок на эту главу для подробного объяснения элементов.

LDAP и X.500 имеют много общих терминов, некоторые из них важны, некоторые - просто ерунда.

Для Вашего удобства мы создали [гlossарий](#), термины в который включены либо потому, что они важны, либо потому, что они часто используются в литературе. Иногда мы используем термины, которые, как нам кажется, более понятны, но за ними мы обычно приводим стандартные термины LDAP.

Поскольку наборы схемы данных, объектные классы и атрибуты сильно взаимосвязаны, для их коллективного описания мы используем сугубо технический термин **элементы**. Вы можете заменить его на более подходящий термин *фенечки*, *прибамбасы* или то, что Вам ближе.

Когда Вы создаёте [запись](#) в [DIT](#), составляющие её данные содержатся в **атрибутах**, которые сгруппированы в **объектные классы**, которые в свою очередь упакованы в **наборы схемы данных**.

Мощь и запутанность LDAP происходят от того, что есть огромное количество атрибутов и огромное количество объектных классов, беспорядочно рассеянных по очевидно случайно (и неизменно бесполезно) названным наборам схемы данных. Вам придётся либо придерживаться некоторых хорошо известных приложений, тогда Вы можете использовать хорошо известные объектные классы и атрибуты, либо потратить некоторое время на изучение их языка, чтобы выяснить, какие объектные классы и атрибуты на самом деле будут оптимальны для Вашего приложения, или даже создать свои собственные.

Медленно но верно мы делаем стандартные наборы схемы данных [доступными для просмотра](#). Если бы мы были умнее и у нас было побольше времени, мы бы переделали это с использованием **htags** и **gtags**.

Содержание

- [3.1 Обзор элементов LDAP](#)
- [3.2 Наборы схемы данных](#)
- [3.3 Объектные классы](#)
- [3.4 Атрибуты](#)
- [3.5 Правила соответствия](#)
- [3.6 Операционные атрибуты и объекты LDAP](#)

3.1 Обзор элементов LDAP

Всё в LDAP иерархично, в том числе **объектные классы** и **атрибуты**. **Наборы схемы** важны, но не особо интересны, они представляют собой упаковку, в которой **грубо** сгруппированы между собой **объектные классы** и **атрибуты**.

Ниже определены важные правила, относящиеся к каждой из этих 'фенечек':

1. **Наборы схемы** - это просто упаковка:
 - Все **объектные классы** и **атрибуты** определяются внутри набора схемы (существует некоторое количество так называемых [операционных](#) объектных классов и атрибутов, которые встроены в программное обеспечение LDAP-сервера и не требуют определения, но пока мы их просто проигнорируем).
 - Все наборы схемы, включающие используемые в какой-либо реализации LDAP **объектные классы** и **атрибуты**, должны быть известны LDAP-серверу (в OpenLDAP это определяется с помощью директивы [include](#) конфигурационного файла slapd.conf).
 - **Атрибут**, определённый в одном **наборе схемы**, может использоваться в **объектном классе**, определённом в другом **наборе схемы**.
2. В **объектных классах** сгруппированы наборы **атрибутов**:
 - **Объектные классы** определяются внутри **наборов схемы**.
 - **Объектные классы** могут быть организованы в иерархию, в этом случае они наследуют все свойства своих родительских или вышестоящих (SUP) объектных классов.

- **Объектные классы** могут быть структурными (STRUCTURAL), в этом случае они используются для создания **записей** (объектов данных), вспомогательными (AUXILIARY), в этом случае они могут быть добавлены в любую подходящую **запись**, или абстрактными (ABSTRACT) - несуществующая 'фенечка'. Чаще всего встречается абстрактный объектный класс **top**, формирующий самый верхний уровень любой иерархии объектных классов и ограничивающий любую иерархию.
- Если **объектный класс** является частью иерархии, он должен быть того же типа (структурный, вспомогательный), что и любой из вышестоящих **объектных классов**. Исключение из этого правила - случай, когда вышестоящий объектный класс абстрактного типа.
- **Объектные классы** предназначены для подключения атрибутов (на жаргоне это называется контейнерами атрибутов).
- **Объектные классы** определяют, является ли атрибут обязательным (MUST), то есть должен присутствовать в записи, либо необязательным (MAY), то есть может присутствовать в записи с данным объектным классом.
- **Объектные классы** определяются с помощью нотации **ASN.1**.

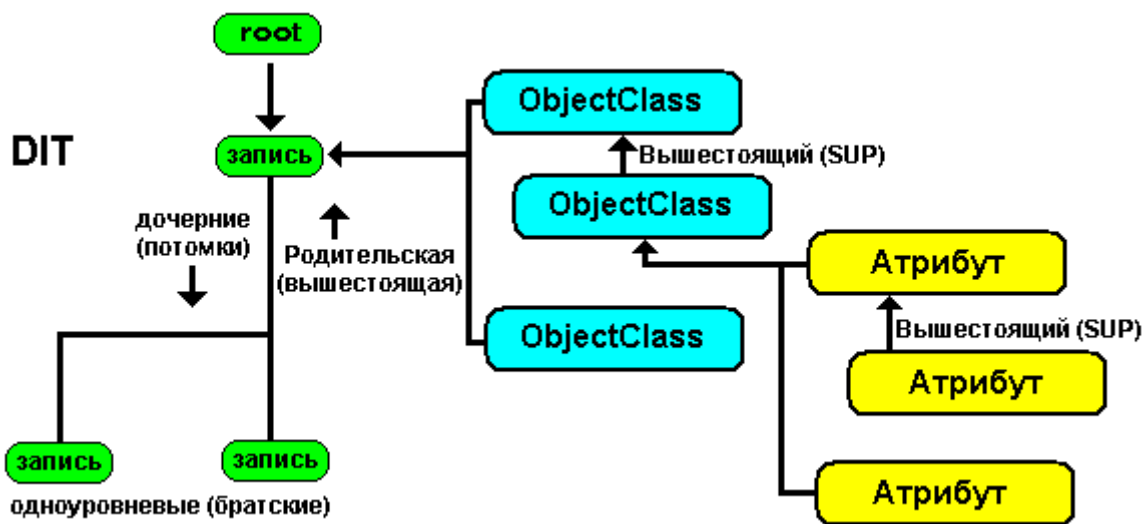
3. Атрибуты обычно содержат данные:

- Каждый **атрибут** входит в один или несколько **объектных классов**.
- Чтобы использовать **атрибут** в **записи**, в определение этой записи должен быть включен его **объектный класс** и этот **объектный класс** должен быть включен в **набор схемы данных**. В свою очередь, этот набор схемы должен быть известен LDAP-серверу.
- Характеристики **атрибута** определяются с помощью нотации **ASN.1**.
- **Атрибут** может появляться лишь однажды в любом экземпляре содержащего его **объектного класса** (SINGLE-VALUE), либо может появляться более одного раза в любом экземпляре содержащего его **объектного класса** (MULTI-VALUE). Значение по умолчанию - MULTI-VALUE.
- Определение **атрибута** может быть частью иерархии, в этом случае оно наследует все свойства своих родителей. Например, commonName (cn), givenName (gn), surname (sn) являются потомками атрибута **name**.
- Определение **атрибута** включает его тип или синтаксис (SYNTAX), например, строка или число, а также как он будет вести себя в определённых условиях, например, будет ли при операциях сравнения учитываться регистр символов или нет.

4. В **записях** в пределах DIT сгруппированы наборы **объектных классов**:

- **Записи** должны содержать один и только один структурный **объектный класс**. У структурного **объектного класса** может быть вышестоящий объектный класс, тоже являющийся структурным. То есть структурный объектный класс может быть частью иерархии и такая иерархия может быть представлена как единый структурный **объектный класс**.
- **Записи** могут содержать любое количество вспомогательных **объектных классов**.
- **Записи** могут содержать только один абстрактный **объектный класс**.
- У **записи** могут быть **дочерние** записи, располагающиеся ниже неё в иерархии адресов.
- У **записи** могут быть **родительские** записи, располагающиеся выше неё в иерархии адресов.
- У **записи** могут быть **одноуровневые** записи, располагающиеся на одном с ней уровне в иерархии адресов. У **одноуровневых** записей одна и та же общая **родительская** запись.

Следующая диаграмма иллюстрирует некоторые из этих отношений:



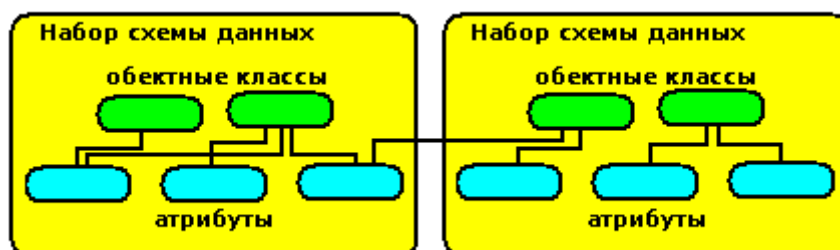
3.2 Наборы схемы данных LDAP

Наборы схемы данных LDAP - не более чем удобная упаковка для содержания во многом схожих [объектных классов](#) и [атрибутов](#).

Возможно, когда-то существовал единственный набор схемы данных, содержащий всё необходимое для LDAP-реализаций (как схема данных реляционной базы данных), но теперь это не так. Вы найдёте полезные атрибуты и объектные классы разбросанными по разным местам - возможно, мощь LDAP как раз и происходит от подобной лёгкости создания и использования этой кажущейся анархии.

Основное правило: Каждый используемый в реализации LDAP атрибут или объектный класс, включая их вышестоящие объектные классы или атрибуты, должен быть определен в **наборе схемы**, и этот набор схемы должен быть **известен** серверу LDAP. В OpenLDAP серверу сообщают о наборах схемы с помощью директивы [include](#) в конфигурационном файле [slapd.conf](#).

Следующая диаграмма иллюстрирует использование наборов схемы как упаковочных единиц:



3.3 Объектные классы LDAP

Объектный класс - это коллекция атрибутов (или контейнер атрибутов). У него следующие характеристики:

1. **Объектный класс** определяется внутри **набора схемы**.
2. **Объектный класс** может быть частью иерархии объектных классов, в этом случае он наследует все свойства своих родительских объектных классов. Например, [inetOrgPerson](#) является потомком [organizationalPerson](#), который в свою очередь является потомком [person](#), который является потомком [top](#) (абстрактный объектный класс, ограничивающий все иерархии объектных классов).
3. **Объектный класс** имеет глобальное уникальное имя или идентификатор.
4. **Объектный класс**, поскольку является контейнером атрибутов, также сам по себе атрибут и может фигурировать в операциях поиска.

5. **Объектный класс** определяет составляющие его атрибуты, а также то, должны ли они присутствовать (обязательные атрибуты), либо могут присутствовать в записи (необязательные атрибуты).
6. **Объектный класс** может быть частью иерархии, в этом случае он наследует все свойства своего родительского объектного класса.
7. В **записи** LDAP должны присутствовать один или несколько **объектных классов**.
8. В **записи** LDAP должен присутствовать один и только один структурный (STRUCTURAL) **объектный класс**.
9. Все **объектные классы**, поддерживаемые сервером LDAP, формируют **коллекцию** под названием **objectclasses**, просмотреть которую можно через **subschema**.

Определение объектного класса

Формальное определение объектного класса дано в [разделе 4.4 RFC 2252](#) и выглядит так:

```
ObjectClassDescription = "(" whsp
    numericoid whsp          ; идентификатор объектного класса
    [ "NAME" qdescr ]
    [ "DESC" qdstring ]
    [ "OBSOLETE" whsp ]
    [ "SUP" oids ]           ; вышестоящий объектный класс
    [ ( "ABSTRACT" / "STRUCTURAL" / "AUXILIARY" ) whsp ]
    ; по умолчанию структурный
    [ "MUST" oids ]          ; типы атрибутов
    [ "MAY" oids ]          ; типы атрибутов
    whsp ")"
```

Вот это да! **whsp** означает пробельный символ, и когда говорят, что он там должен быть, поверьте на слово. Вместо того, чтобы объяснять каждый параметр отдельно, давайте начнём с нескольких примеров.

Объектный класс определяется с помощью нотации [ASN.1](#). Далее рассмотрим определение простого стандартного объектного класса [country](#), взятого из набора схемы [core.schema](#), поставляемого с дистрибутивами OpenLDAP.

```
objectclass ( 2.5.6.2 NAME 'country' SUP top STRUCTURAL
    MUST c
    MAY ( searchGuide $ description ) )
```

Давайте разберём это определение:

Ключевое слово **objectclass** указывает, что это определение объектного класса. Как видите, всё не так уж сложно!

2.5.6.2 NAME 'country' определяет **глобально уникальное** имя данного объектного класса, состоящее из двух частей: **NAME 'country'** просто позволяет обращаться к этому объектному классу с помощью некоего полупонятного текста - в данном случае английское слово **country**. **Глобально уникальная** часть определяется с помощью **2.5.6.2**, так называемого **OID (ObjectIdentifier, идентификатора объекта)**. OID 2.5.6.2, похоже, третий из объектных классов, когда-либо определённых X.500 (это следует из того факта, что 2.5.6 определяет совместные объектные классы itu-iso x.500, а последняя 2 - порядковый номер в пределах этого семейства OID). Не важно, какая организация присваивает этот номер, но он должен быть **УНИКАЛЬНЫМ**. Получение корпоративного OID (формально Private Enterprise Number (PEN)), позволяющего определять свои собственные **атрибуты и объектные классы**, - простая и бесплатная процедура, которую предоставляет [IANA](#). Повторно использовать существующие OID - очень скверная штука (VERY BAD THING™).

SUP 'top' указывает, что у этого объектного класса есть **РОДИТЕЛЬСКИЙ** (или **ВЫШЕСТОЯЩИЙ**) объектный класс, то есть он является частью иерархии. В данном случае родительским объектным классом будет **top** - специальный класс, который ограничивает все объектные классы (является самым высоким уровнем). Объектный класс может иметь один или несколько родительских объектных классов.

STRUCTURAL (структурный) указывает, что этот объектный класс содержит атрибуты и может формировать **запись** в DIT. В записи может быть только один структурный объектный класс, но структурный объектный класс может быть частью иерархии (то есть иметь другой структурный объектный класс в качестве

вышестоящего). ([Дополнительная информация о структурных иерархиях и наследовании](#)). **Объектные классы** также могут быть абстрактными (ABSTRACT), что указывает на несуществующий объектный класс, используемый для удобства. Наиболее часто используемый абстрактный объектный класс - это **top**, просто ограничивающий иерархию объектных классов. Наконец, **объектный класс** может быть вспомогательным (AUXILIARY), что говорит о том, что он содержит атрибуты и может использоваться для формирования записи с любым структурным объектным классом, но самостоятельно формировать запись в DIT не может.

DESC 'описание'. Ну ладно, мы выбрали неудачный пример, где нет оператора DESC - зато он короткий. DESC - это необязательное значение, предоставляющее короткое текстовое описание содержимого объектного класса. Оно предназначено для чтения людьми и другого применения не имеет. Вот на что стало бы похоже **country**, если бы в него включили осмысленный оператор DESC:

```
objectclass ( 2.5.6.2 NAME 'country' SUP top STRUCTURAL
  DESC '2 character iso assigned country code'
  MUST c
  MAY ( searchGuide $ description ) )
```

Встречаются самые разные значения DESC: от несущественных до трогательных, иногда полезных, а иногда вводящих в заблуждение и способных запутать кого угодно.

MUST c. MUST указывает, что атрибуты из следующего за этим оператором списка являются обязательными. В данном случае атрибут **c** (**c** или **countryName**) должен присутствовать, иначе не создастся экземпляр объектного класса и, если это структурный (STRUCTURAL) объектный класс, запись не будет создана. Единичные значения записываются как было показано, несколько атрибутов заключаются в круглые скобки и разделяются знаком \$ (доллар), вот так: **(attr1 \$ attr2 \$ attrn)**. Если у объектного класса нет обязательных атрибутов, эта секция не включается.

MAY (searchGuide \$ description) MAY указывает, что атрибуты из следующего за этим оператором списка являются необязательными и для создания экземпляра объектного класса их присутствие не требуется. Несколько значений записываются как было показано, а для единичного атрибута круглые скобки не нужны (смотрите выше). Если у объектного класса нет необязательных атрибутов, эта секция не включается.

Ещё несколько примеров объектных классов

Вот как определяется объектный класс **top**:

```
objectclass ( 2.5.6.0 NAME 'top' ABSTRACT
  MUST objectClass )
```

Здесь демонстрируется использование оператора ABSTRACT в объектном классе. Поскольку **top** всегда находится в вершине иерархии, очевидно, что в его определении не может быть оператора **SUP**. OID также из присвоенных группе стандартизации X.500.

Авторы в многих документах настаивают, чтобы объектный класс **top** включался в файлы LDIF - [на самом деле это не всегда необходимо](#).

Вот как определяется объектный класс **dcObject**:

```
objectclass ( 1.3.6.1.4.1.1466.344 NAME 'dcObject'
  DESC 'RFC2247: domain component object'
  SUP top AUXILIARY MUST dc )
```

Данный пример демонстрирует использование оператора AUXILIARY. Невозможно создать запись только на основе вспомогательного объектного класса. В этом примере OID показывает использование [private enterprise OID \(OID из пространства частного предприятия\)](#). Следующий фрагмент демонстрирует довольно типичное определение базового DN с использованием **dcObject**:

```
dn: dc=example,dc=com
dc: example.com
objectclass: dcObject
objectclass: organization
o: Example, Inc.
```

Данная запись создаётся объектным классом **objectclass: organization**. **dcObject** идёт прицепом к этому объектному классу.

В следующем примере определяется объектный класс **pilotOrganization** и демонстрируется, что у объектного класса может быть один или несколько вышестоящих (родительских) объектных классов (в данном примере родительскими являются **organization** и **organizationalUnit**), в этом случае потомки наследуют свойства BCEX родителей (чем-то похоже на людей, не правда ли?):

```
objectClasses: ( 0.9.2342.19200300.100.4.20 NAME 'pilotOrganization'
  SUP ( organization $ organizationalUnit ) STRUCTURAL
  MAY buildingName )
```

[Дополнительная информация по наследованию.](#)

Мы пропустили объяснение оператора OBSOLETE (устаревший). Если он присутствует, это означает, что объектный класс использовать не нужно (хм!).



3.4 Атрибуты LDAP

Атрибуты обычно содержат данные и имеют следующие характеристики:

1. Каждый **атрибут** входит в один или несколько **объектных классов**.
2. **objectclass** также является **атрибутом** и может использоваться в операциях поиска.
3. Чтобы использовать **атрибут** в **записи**, в определение этой записи должен быть включен его **объектный класс** и этот **объектный класс** должен быть включен в **набор схемы данных**. В свою очередь, этот набор схемы должен быть известен LDAP-серверу.
4. Характеристики **атрибута** определяются с помощью нотации [ASN.1](#).
5. **Атрибут** может появляться лишь однажды в любом экземпляре содержащего его **объектного класса** (SINGLE-VALUE), либо может появляться более одного раза в любом экземпляре содержащего его **объектного класса** (MULTI-VALUE). Значение по умолчанию - MULTI-VALUE.
6. Определение **атрибута** может быть частью иерархии, в этом случае оно наследует все свойства своих родителей. Например, commonName (cn), givenName (gn), surname (sn) являются потомками атрибута **name**.
7. Определение **атрибута** включает его тип или синтаксис (SYNTAX), например, строка или число, а также как он будет вести себя в определённых условиях; например, будет ли при операциях сравнения учитываться регистр символов или нет.
8. **Атрибуты**, поддерживаемые сервером LDAP, формируют **коллекцию** под названием **attributetypes**, просмотреть которую можно через [subschema](#).

Определение атрибута

Формальное определение атрибута дано в [разделе 4.4 RFC 2252](#) и выглядит так:

```
AttributeTypeDescription = "(" whsp
  numericoid whsp          ; идентификатор типа атрибута
  [ "NAME" qdscrs ]         ; имя, используемое в типе атрибута
  [ "DESC" qdstring ]       ; описание
  [ "OBSOLETE" whsp ]
  [ "SUP" woid ]             ; происходит от этого
                             ; типа атрибута
  [ "EQUALITY" woid         ; имя правила соответствия
  [ "ORDERING" woid         ; имя правила соответствия
  [ "SUBSTR" woid ]         ; имя правила соответствия
  [ "SYNTAX" whsp noidlen whsp ] ; OID синтаксиса
  [ "SINGLE-VALUE" whsp ]    ; по умолчанию multi-valued
  [ "COLLECTIVE" whsp ]     ; по умолчанию not collective
  [ "NO-USER-MODIFICATION" whsp ] ; по умолчанию user modifiable
  [ X-ORDERED whsp type ]   ; non-standard - default not X-ORDERED
  [ "USAGE" whsp AttributeUsage ] ; по умолчанию userApplications
  whsp ")"
```

whsp означает пробельный символ, и он должен присутствовать. Вместо того, чтобы разбирать каждое слово

из этой тарабарщины, давайте снова начнём с нескольких примеров.

Атрибуты определяются с помощью нотаций [ASN.1](#). Далее рассмотрим определение простого стандартного атрибута **commonName (cn)**, взятого из набора схемы [core.schema](#), поставляемого с дистрибутивами OpenLDAP.

```
attributetype ( 2.5.4.3 NAME ( 'cn' 'commonName' ) SUP name )
```

Давайте разберём это определение:

attributetype указывает, что это определение атрибута - кто бы мог подумать!

2.5.4.3 NAME ('cn' 'commonName') определяет **глобально уникальное** имя для этого атрибута и состоит из двух значений в скобках: **NAME ('cn' 'commonName')**. Это позволяет Вам обращаться к этому атрибуту с помощью некоего полупонятного текста - в данном случае либо английское слово **commonName**, либо сокращение (псевдоним) **cn**. В принципе, нет ограничений на количество сокращений или псевдонимов, которые Вы можете иметь, лишь бы они были уникальными. В данной форме записи с несколькими именами эти имена заключаются в круглые скобки и разделяются пробелом. Так как **cn** появляется первым, он называется **первичным (primary)** именем, что очень важно, когда речь заходит об **индексировании** записей.

Глобально уникальная часть определяется с помощью **2.5.4.3**, так называемого **OID (ObjectIdentifier, идентификатора объекта)**. OID 2.5.4.3 возможно, четвёртый из атрибутов, когда-либо определённых X.500 (2.5.4 означает, что это совместные типы атрибутов itu-iso x.500, последняя 3 - порядковый номер в пределах этого семейства OID). Не важно, какая организация присваивает этот номер, но он должен быть **УНИКАЛЬНЫМ**. Получение организацией OID, позволяющего определять свои собственные **атрибуты** и **объектные классы**, - простая процедура, которую предоставляет [IANA](#). Повторно использовать существующие OID - очень скверная штука (VERY BAD THING™).

SUP 'name' указывает, что у этого атрибута есть **РОДИТЕЛЬСКИЙ** (или **ВЫШЕСТОЯЩИЙ**) атрибут, то есть он является частью иерархии. В данном случае родительским атрибутом будет **name**, который мы сейчас рассмотрим подробно, поскольку, если Вы помните, атрибут-потомок всегда наследует свойства родительского (или вышестоящего) атрибута (и может иметь свои дополнительные свойства). В операторе SUP может использоваться либо 'name', либо OID. Определения SUP 'name' и SUP 2.5.4.1 означают одно и то же - но не для бедняги читателя!

Вот определение атрибута **name**, это гораздо более серьёзное определение атрибута, являющегося вышестоящим (родительским) по отношению к приведённому выше атрибуту **cn**:

```
attributetype ( 2.5.4.41 NAME 'name'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )
```

А вот и более серьёзный разбор этого определения:

attributetype указывает, что это определение атрибута - то же, что и раньше.

2.5.4.41 NAME 'name' определяет **глобально уникальное** имя для данного атрибута и, как и прежде, состоит из двух частей: **NAME 'name'** просто позволяет обращаться к этому атрибуту с помощью некоего полупонятного текста, и OID **2.5.4.41** указывает, что данный атрибут определён группой стандартизации X.500. Данный формат используется, поскольку здесь только одно значение имени, его не требуется заключать в круглые скобки как в приведённом выше примере с **commonName**.

EQUALITY caseIgnoreMatch указывает, как данный атрибут (и все его потомки) будут себя вести при использовании в **поисковом фильтре**, таком как **(cn=jimbob)** (**cn** потомок **name**), где не используется **поисковых шаблонов**. В данном случае определяется, что соответствие будет искаться без учёта регистра символов. **caseIgnoreMatch** - это **правило соответствия**, которое определяется в **subschema**.

SUBSTR caseIgnoreSubstringsMatch указывает, как данный атрибут (и все его потомки) будут себя вести при использовании в **поисковом фильтре**, использующем подстроки, например **(cn=jim*)** (**cn** потомок **name**) и содержащем один или несколько **поисковых шаблонов**. В данном случае определяется, что соответствие будет искаться без учёта регистра символов. **caseIgnoreSubstringMatch** - это снова **правило соответствия**, которое определяется в **subschema**.

SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} - это **OID**, определяющий тип данных, а также какие

правила (проверки данных) применяются к этим данным. Полный список приведён в [разделе 4.3.2 RFC 2252](#). В данном случае OID указывает, что это будет тип Directory String, определённый в [разделе 6.10 RFC 2252](#), и данные будут из набора символов ISO 10646 в форме UTF-8. Значение **{32768}** указывает максимальную длину строки и не является обязательным. [Дополнительная информация по типам данных LDAP](#).

Другие характеристики

SINGLE-VALUE. Пропуск этого оператора означает, что данный атрибут multi-value, то есть он может появляться более одного раза в **объектном классе** или записи. Если какой-то атрибут может принимать только одно значение, это должно быть явно указано, как в определении атрибута **dc** ниже:

```
attributetype ( 0.9.2342.19200300.100.1.25
  NAME ( 'dc' 'domainComponent' )
  DESC 'RFC1274/2247: domain component'
  EQUALITY caseIgnoreIA5Match
  SUBSTR caseIgnoreIA5SubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
  SINGLE-VALUE )
```

USAGE 'AttributeUsage'. По умолчанию - userApplication (или пользовательский атрибут), будет возвращён, если при поиске в списке атрибутов есть значение *. Если же значение данного оператора - dSAOperation, тогда этот атрибут будет операционным (возвращается, если при поиске в списке атрибутов есть значение +).

ORDERING 'matchingrule' указывается редко и используется для определения сопоставлений - лексикографического порядка сортировки (позволяет выполнять поиск с использованием <= и >=).

```
attributetype ( 2.5.4.46 NAME 'dnQualifier'
  EQUALITY caseIgnoreMatch
  ORDERING caseIgnoreOrderingMatch
  SUBSTR caseIgnoreSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.44 )
```

Дополнительные элементы определения:

Большинство читателей могут, к счастью, пропустить данные замечания. Они приводятся здесь только для тех, кто копается в недрах OpenLDAP или пытается разобраться, что означают все эти {}, то и дело встречающиеся при работе с системой конфигурации времени исполнения OpenLDAP ([OLC cn=config](#)).

1. 'X-ORDERED type' - нестандартный элемент (в настоящее время определённый только в draft-chu-ldap-xordered-00, да и то не полностью), широко применяемый в системе [OLC \(cn=config\)](#) OpenLDAP. Наличие данного элемента в определении атрибута позволяет пользоваться упорядоченными множествами. **type** может быть либо 'VALUES', либо 'SIBLINGS'. 'VALUES' используется только с атрибутами MULTI-VALUE и указывает на то, что каждый атрибут будет иметь порядковый элемент в форме {x}, добавляемый в начало значения данного атрибута (и становящийся частью этого значения), где x - число, означающее порядковый номер и начинающееся с 0. Это позволяет при обращении к атрибутам с несколькими значениями явно указывать то значение, к которому мы хотим обратиться (для операций модификации или удаления), а при добавлении новых значений атрибута, для которых важен порядок их указания (например, при составлении ACL/ACP с помощью атрибута **olcAccess**), явно указать место его нахождения в упорядоченном множестве. 'SIBLINGS' используется только с атрибутами SINGLE-VALUE. Если атрибут с таким значением присутствует в записи, это говорит о том, что её непосредственные дочерние записи (только на один уровень ниже) должны иметь порядковое значение {x} в своих DN. Данный префикс с порядковым значением может быть указан явно при добавлении дочерней записи, либо, если таковой отсутствует, в качестве него будет назначено значение следующего порядкового номера, начиная с 0.

3.5 Правила соответствия

Правила соответствия - часть так называемых [операционных](#) характеристик сервера LDAP.

Правила соответствия определяют методы сравнения, доступные в сервере LDAP:

1. **Правила соответствия** обычно встроены в сервер LDAP и не требуют явного указания.
2. **Правила соответствия** формируют **коллекцию** под названием **matchingrules**, просмотреть которую можно через [subschemata](#).
3. **Правило соответствия** определяется для каждого **атрибута** с помощью операторов [EQUALITY](#), [SUBSTR](#) и [ORDERING](#) по мере необходимости - то есть определяются только те свойства, которые требуются. Если при поиске не могут использоваться **поисковые шаблоны**, свойство SUBSTR определять не нужно.

3.5.1 Определение правила соответствия

Большинство **правил соответствия** встроены, и Вам почти никогда не придётся определять их, но, как и у всего в LDAP, у них есть синтаксис определения. Вот пример определения правила соответствия **caseIgnoreMatch**:

```
matchingRule ( 2.5.13.2 NAME 'caseIgnoreMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

В результате разбора увидим следующее:

matchingrule указывает начало определения правила соответствия.

2.5.13.2 NAME 'caseIgnoreMatch' указывает **глобально уникальное** имя для данного правила соответствия и всегда состоит из двух частей: **NAME 'caseIgnoreMatch'** позволяет обращаться к этому правилу соответствия с помощью некоего полупонятного текста, и **OID 2.5.13.2** указывает, что данное правило соответствия определено группой стандартизации X.500. Описание правила:

"Правило Case Ignore Match сравнивает совпадение предоставленной строки со значением атрибута типов PrintableString, NumericString, TeletexString, BMPString, UniversalString или DirectoryString без учета регистра (верхнего или нижнего) символов строки (например, "Dundee" и "DUNDEE" совпадают).

Правило возвращает TRUE, если строки имеют одинаковую длину и соответствующие символы идентичны, кроме возможного расхождения регистра символов.

SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 указывает, что данное правило соответствия применяется к определённым типам, в данном случае к DirectoryString (строка в формате UTF-8).

Правила соответствия, встроенные в OpenLDAP

Следующий ниже список может быть найден в OpenLDAP путём опроса **subschemata** с помощью следующей команды:

```
ldapsearch -H ldap://ldap.example.com -x -s base -b "cn=subschema"
"(objectclass=*)" matchingrules
# matchingrules можно заменить на
# attributetypes, objectclasses и другое.
```

Приведённая команда должна записываться в одну строку - она разделена только по соображениям форматирования HTML. Замените ldap.example.com на имя хоста Вашего LDAP-сервера. Если Вы выполняете команду на том же сервере, где запущен LDAP-сервер, можно опустить аргумент -H.

В качестве альтернативы можно использовать любой хороший LDAP-браузер, указывая в качестве базового DN "cn=subschema".

Приведенная выше команда возвращает следующий список (OpenLDAP 2.1.12 на FreeBSD):

```
# Subschema
dn: cn=Subschema
matchingRules: ( 2.5.13.0 NAME 'objectIdentifierMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
matchingRules: ( 2.5.13.1 NAME 'distinguishedNameMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )
matchingRules: ( 2.5.13.2 NAME 'caseIgnoreMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

```

matchingRules: ( 2.5.13.3 NAME 'caseIgnoreOrderingMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
matchingRules: ( 2.5.13.4 NAME 'caseIgnoreSubstringsMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )
matchingRules: ( 2.5.13.5 NAME 'caseExactMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
matchingRules: ( 2.5.13.6 NAME 'caseExactOrderingMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
matchingRules: ( 2.5.13.7 NAME 'caseExactSubstringsMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )
matchingRules: ( 2.5.13.8 NAME 'numericStringMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.36 )
matchingRules: ( 2.5.13.10 NAME 'numericStringSubstringsMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )
matchingRules: ( 2.5.13.13 NAME 'booleanMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 )
matchingRules: ( 2.5.13.14 NAME 'integerMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )
matchingRules: ( 2.5.13.15 NAME 'integerOrderingMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )
matchingRules: ( 2.5.13.16 NAME 'bitStringMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.6 )
matchingRules: ( 2.5.13.17 NAME 'octetStringMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 )
matchingRules: ( 2.5.13.18 NAME 'octetStringOrderingMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 )
matchingRules: ( 2.5.13.20 NAME 'telephoneNumberMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.50 )
matchingRules: ( 2.5.13.21 NAME 'telephoneNumberSubstringsMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.58 )
matchingRules: ( 2.5.13.23 NAME 'uniqueMemberMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.34 )
matchingRules: ( 2.5.13.27 NAME 'generalizedTimeMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 )
matchingRules: ( 2.5.13.28 NAME 'generalizedTimeOrderingMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 )
matchingRules: ( 2.5.13.29 NAME 'integerFirstComponentMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )
matchingRules: ( 2.5.13.30 NAME 'objectIdentifierFirstComponentMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
matchingRules: ( 2.5.13.34 NAME 'certificateExactMatch'
  SYNTAX 1.2.826.0.1.3344810.7.1 )
matchingRules: ( 1.3.6.1.4.1.1466.109.114.1 NAME 'caseExactIA5Match'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
matchingRules: ( 1.3.6.1.4.1.1466.109.114.2 NAME 'caseIgnoreIA5Match'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
matchingRules: ( 1.3.6.1.4.1.1466.109.114.3 NAME 'caseIgnoreIA5SubstringsMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
matchingRules: ( 1.3.6.1.4.1.4203.1.2.1 NAME 'caseExactIA5SubstringsMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
matchingRules: ( 1.2.840.113556.1.4.803 NAME 'integerBitAndMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )
matchingRules: ( 1.2.840.113556.1.4.804 NAME 'integerBitOrMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

```

Вы можете найти, что означают эти **OID**, и, следовательно, точное описание правил соответствия на английском языке с помощью этого [замечательного сайта](#).

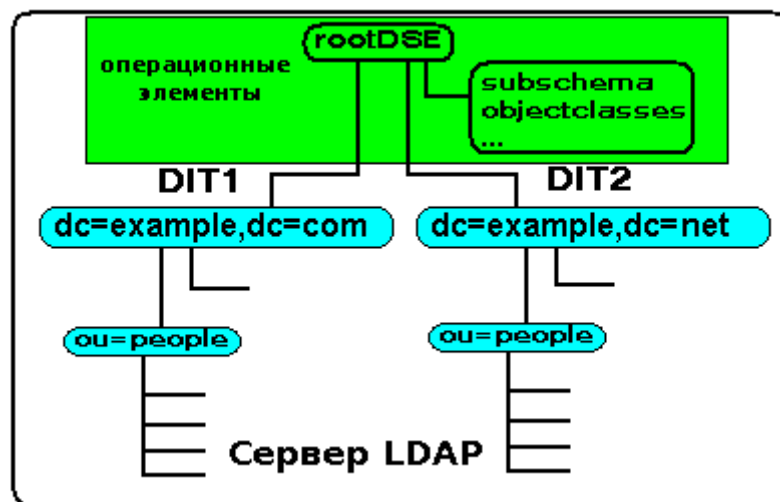


3.6 Операционные атрибуты и объекты LDAP

Есть куча встроенных в LDAP-сервер атрибутов и объектных классов, управляющих тем, как этот сервер функционирует. Такие атрибуты и объектные классы обычно называются **операционными**.

Эти **операционные элементы** располагаются в [rootDSE](#) и при обычных операциях не видны.

Здесь показаны отношения между DIT и входящими в них записями, а также RootDSE и входящими в него объектами:



rootDSE можно просмотреть с помощью подходящего LDAP-браузера (вот инструкции для [LDAPBrowser/Editor](#)), указав пустой базовый DN, либо с помощью следующей команды:

```
ldapsearch -H ldap://ldap.mydomain.com -x -s base -b "" +
# обратите внимание, что + указывает вернуть операционные атрибуты
```

Вывод команды будет примерно таким, как показан ниже (в OpenLDAP 2.4.8), значения в скобках - это добавленные нами объяснения, они не возвращаются сервером:

```
dn:
structuralObjectClass: OpenLDAPProotDSE
configContext: cn=config
namingContexts: dc=example,dc=com
namingContexts: dc=example,dc=net
monitorContext: cn=Monitor
supportedControl: 1.3.6.1.4.1.4203.1.9.1.1 (Contentsync RFC 4530)
supportedControl: 2.16.840.1.113730.3.4.18 (ProxiedAuthv2 RFC 4370)
supportedControl: 2.16.840.1.113730.3.4.2 (ManagedSAIT RFC3377)
supportedControl: 1.3.6.1.4.1.4203.1.10.1 (SubEntries RFC3673)
supportedControl: 1.2.840.113556.1.4.319 (pagedResults RFC2696)
supportedControl: 1.2.826.0.1.3344810.2.3 (MatchedValues RFC3876)
supportedControl: 1.3.6.1.1.13.2 (Post Read RFC4527)
supportedControl: 1.3.6.1.1.13.1 (Pre-Read RFC4527)
supportedControl: 1.3.6.1.1.12 (Assertion RFC4528)
supportedExtension: 1.3.6.1.4.1.4203.1.11.1 (ModifyPassword RFC3088)
supportedExtension: 1.3.6.1.4.1.4203.1.11.3 (WhoAmI RFC4532)
supportedExtension: 1.3.6.1.1.8 (Cancel RFC3909)
supportedFeatures: 1.3.6.1.1.14 (Modify-Increment RFC4525)
supportedFeatures: 1.3.6.1.4.1.4203.1.5.1 (OperationalAttrs RFC3674)
supportedFeatures: 1.3.6.1.4.1.4203.1.5.2 (ObjectClassAttrs RFC4529)
supportedFeatures: 1.3.6.1.4.1.4203.1.5.3 (TrueFalse RFC4526)
supportedFeatures: 1.3.6.1.4.1.4203.1.5.4 (LanguageTag RFC3866)
supportedFeatures: 1.3.6.1.4.1.4203.1.5.5 (LanguageRange RFC3866)
supportedLDAPVersion: 3
supportedSASLMechanisms: NTLM
supportedSASLMechanisms: GSSAPI
supportedSASLMechanisms: DIGEST-MD5
supportedSASLMechanisms: CRAM-MD5
entryDN:
subschemaSubentry: cn=Subschema
```

Объяснение каждого **supportedExtension** можно найти с помощью этого [замечательного сайта](#). В этом листинге показано, что данный LDAP-сервер поддерживает два **DIT** - указаны как **namingContexts**, это было сконфигурировано с помощью описанной [здесь](#) процедуры.

Существует возможность добавить расширения, используя директиву [rootDSE](#) в OpenLDAP slapd.conf.



Объекты subschemasubentry

subschemasubentry - интересный **элемент**, который можно исследовать с помощью подходящего LDAP-браузера (вот инструкции для [LDAPBrowser/Editor](#)), указав в качестве базового DN "cn=subschema", либо с помощью следующей команды:

```
ldapsearch -H ldap://ldap.mydomain.com -x -s base -b "cn=subschema" objectclasses
# Список атрибутов, которые могут быть перечислены:
# matchingruleuse ldapsyntaxes matchingrules attributetypes
# (всё это - коллекции), а также
# createtimestamp modifytimestamp
# Если Вы используете единственный знак +, Вы получите огромный
# список всего, что знает LDAP-сервер.
```

Эта команда сгенерирует следующий вывод:

Примечание: в данный LDAP-сервер включены cosine.schema, core.schema, nis.schema, inetorgperson.schema.

```
# Subschema
dn: cn=Subschema
objectClasses: ( 2.5.6.0 NAME 'top'
  DESC 'top of the superclass chain'
  ABSTRACT
  MUST objectClass )
objectClasses: ( 1.3.6.1.4.1.1466.101.120.111 NAME 'extensibleObject'
  DESC 'RFC2252: extensible object' SUP top AUXILIARY )
objectClasses: ( 2.5.6.1 NAME 'alias'
  DESC 'RFC2256: an alias' SUP top STRUCTURAL
  MUST aliasedObjectName )
objectClasses: ( 2.16.840.1.113730.3.2.6 NAME 'referral'
  DESC 'namedref: named subordinate referral' SUP top STRUCTURAL
  MUST ref )
objectClasses: ( 1.3.6.1.4.1.4203.1.4.1 NAME ( 'OpenLDAProotDSE' 'LDAProotDSE' )
  DESC 'OpenLDAP Root DSE object' SUP top STRUCTURAL
  MAY cn )
objectClasses: ( 2.5.17.0 NAME 'subentry' SUP top STRUCTURAL
  MUST ( cn $ subtreeSpecification ) )
objectClasses: ( 2.5.20.1 NAME 'subschema'
  DESC 'RFC2252: controlling subschema (sub)entry' AUXILIARY
  MAY ( dITStructureRules $ nameForms $ ditContentRules
    $ objectClasses $ attributeTypes $ matchingRules $ matchingRuleUse ) )
objectClasses: ( 1.3.6.1.4.1.4203.666.3.2 NAME 'monitor'
  DESC 'OpenLDAP system monitoring' STRUCTURAL MUST cn )
objectClasses: ( 2.5.6.2 NAME 'country' DESC 'RFC2256: a country'
  SUP top STRUCTURAL MUST c MAY ( searchGuide $ description ) )
objectClasses: ( 2.5.6.3 NAME 'locality'
  DESC 'RFC2256: a locality' SUP top STRUCTURAL
  MAY ( street $ seeAlso $ searchGuide $ st $ l $ description ) )
objectClasses: ( 2.5.6.4 NAME 'organization'
  DESC 'RFC2256: an organization' SUP top STRUCTURAL
  MUST o
  MAY ( userPassword $ searchGuide $ seeAlso $ businessCategory
    $ x121Address $ registeredAddress $ destinationIndicator
    $ preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier
    $ telephoneNumber $ internationalISDNNumber $ facsimileTelephoneNumber
    $ street $ postOfficeBox $ postalCode $ postalAddress
    $ physicalDeliveryOfficeName $ st $ l $ description ) )
objectClasses: ( 2.5.6.5 NAME 'organizationalUnit'
  DESC 'RFC2256: an organizational unit' SUP top STRUCTURAL
  MUST ou
  MAY ( userPassword $ searchGuide $ seeAlso $ businessCategory
    $ x121Address $ registeredAddress $ destinationIndicator
    $ preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier
    $ telephoneNumber $ internationalISDNNumber $ facsimileTelephoneNumber
    $ street $ postOfficeBox $ postalCode $ postalAddress
    $ physicalDeliveryOfficeName $ st $ l $ description ) )
objectClasses: ( 2.5.6.6 NAME 'person'
  DESC 'RFC2256: a person' SUP top STRUCTURAL
  MUST ( sn $ cn )
  MAY ( userPassword $ telephoneNumber $ seeAlso $ description ) )
objectClasses: ( 2.5.6.7 NAME 'organizationalPerson'
  DESC 'RFC2256: an organizational person' SUP person STRUCTURAL
  MAY ( title $ x121Address $ registeredAddress $ destinationIndicator
    $ preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier
    $ telephoneNumber $ internationalISDNNumber
    $ facsimileTelephoneNumber $ street $ postOfficeBox $ postalCode
    $ postalAddress $ physicalDeliveryOfficeName $ ou $ st $ l ) )
objectClasses: ( 2.5.6.8 NAME 'organizationalRole'
  DESC 'RFC2256: an organizational role' SUP top STRUCTURAL
  MUST cn
  MAY ( x121Address $ registeredAddress $ destinationIndicator
    $ preferredDeliveryMethod $ telexNumber
    $ teletexTerminalIdentifier $ telephoneNumber
    $ internationalISDNNumber $ facsimileTelephoneNumber
    $ seeAlso $ roleOccupant $ preferredDeliveryMethod $ street
    $ postOfficeBox $ postalCode $ postalAddress
    $ physicalDeliveryOfficeName $ ou $ st $ l $ description ) )
objectClasses: ( 2.5.6.9 NAME 'groupOfNames'
```

```

DESC 'RFC2256: a group of names (DNs)' SUP top STRUCTURAL
MUST ( member $ cn )
MAY ( businessCategory $ seeAlso $ owner $ ou $ o $ description ) )
objectClasses: ( 2.5.6.10 NAME 'residentialPerson'
DESC 'RFC2256: an residential person' SUP person STRUCTURAL
MUST l
MAY ( businessCategory $ x121Address $ registeredAddress
$ destinationIndicator $ preferredDeliveryMethod $ telexNumber
$ teletexTerminalIdentifier $ telephoneNumber
$ internationalISDNNumber $ facsimileTelephoneNumber
$ preferredDeliveryMethod $ street $ postOfficeBox
$ postalCode $ postalAddress $ physicalDeliveryOfficeName $ st $ l ) )
objectClasses: ( 2.5.6.11 NAME 'applicationProcess'
DESC 'RFC2256: an application process' SUP top STRUCTURAL
MUST cn
MAY ( seeAlso $ ou $ l $ description ) )
objectClasses: ( 2.5.6.12 NAME 'applicationEntity'
DESC 'RFC2256: an application entity' SUP top STRUCTURAL
MUST ( presentationAddress $ cn )
MAY ( supportedApplicationContext $ seeAlso $ ou $ o $ l $ description ) )
objectClasses: ( 2.5.6.13 NAME 'dSA'
DESC 'RFC2256: a directory system agent (a server)'
SUP applicationEntity STRUCTURAL
MAY knowledgeInformation )
objectClasses: ( 2.5.6.14 NAME 'device'
DESC 'RFC2256: a device' SUP top STRUCTURAL
MUST cn
MAY ( serialNumber $ seeAlso $ owner $ ou $ o $ l $ description ) )
objectClasses: ( 2.5.6.15 NAME 'strongAuthenticationUser'
DESC 'RFC2256: a strong authentication user' SUP top AUXILIARY
MUST userCertificate )
objectClasses: ( 2.5.6.16 NAME 'certificationAuthority'
DESC 'RFC2256: a certificate authority' SUP top AUXILIARY
MUST ( authorityRevocationList $ certificateRevocationList
$ cACertificate ) MAY crossCertificatePair )
objectClasses: ( 2.5.6.17 NAME 'groupOfUniqueNames'
DESC 'RFC2256: a group of unique names (DN and Unique Identifier)'
SUP top STRUCTURAL
MUST ( uniqueMember $ cn )
MAY ( businessCategory $ seeAlso $ owner $ ou $ o $ description ) )
objectClasses: ( 2.5.6.18 NAME 'userSecurityInformation'
DESC 'RFC2256: a user security information' SUP top AUXILIARY
MAY supportedAlgorithms )
objectClasses: ( 2.5.6.16.2 NAME 'certificationAuthority-V2'
SUP certificationAuthority AUXILIARY
MAY deltaRevocationList )
objectClasses: ( 2.5.6.19 NAME 'cRLDistributionPoint'
SUP top STRUCTURAL
MUST cn
MAY ( certificateRevocationList $ authorityRevocationList
$ deltaRevocationList ) )
objectClasses: ( 2.5.6.20 NAME 'dmd' SUP top STRUCTURAL
MUST dmdName
MAY ( userPassword $ searchGuide $ seeAlso $ businessCategory
$ x121Address $ registeredAddress $ destinationIndicator
$ preferredDeliveryMethod $ telexNumber
$ teletexTerminalIdentifier $ telephoneNumber
$ internationalISDNNumber $ facsimileTelephoneNumber $ street
$ postOfficeBox $ postalCode $ postalAddress
$ physicalDeliveryOfficeName $ st $ l $ description ) )
objectClasses: ( 2.5.6.21 NAME 'pkiUser'
DESC 'RFC2587: a PKI user' SUP top AUXILIARY
MAY userCertificate )
objectClasses: ( 2.5.6.22 NAME 'pkiCA'
DESC 'RFC2587: PKI certificate authority' SUP top AUXILIARY
MAY ( authorityRevocationList $ certificateRevocationList
$ cACertificate $ crossCertificatePair ) )
objectClasses: ( 2.5.6.23 NAME 'deltaCRL'
DESC 'RFC2587: PKI user' SUP top AUXILIARY
MAY deltaRevocationList )
objectClasses: ( 1.3.6.1.4.1.250.3.15 NAME 'labeledURIObject'
DESC 'RFC2079: object that contains the URI attribute type'
SUP top AUXILIARY MAY labeledURI )
objectClasses: ( 0.9.2342.19200300.100.4.19 NAME 'simpleSecurityObject'
DESC 'RFC1274: simple security object' SUP top AUXILIARY
MUST userPassword )
objectClasses: ( 1.3.6.1.4.1.1466.344 NAME 'dcObject'
DESC 'RFC2247: domain component object' SUP top AUXILIARY
MUST dc )
objectClasses: ( 1.3.6.1.1.3.1 NAME 'uidObject'
DESC 'RFC2377: uid object' SUP top AUXILIARY
MUST uid )
objectClasses: ( 0.9.2342.19200300.100.4.4 NAME ( 'pilotPerson' 'newPilotPerson' )
SUP person STRUCTURAL
MAY ( userid $ textEncodedORAddress
$ rfc822Mailbox $ favouriteDrink $ roomNumber $ userClass
$ homeTelephoneNumber $ homePostalAddress $ secretary

```



```

    $ personalTitle $ preferredDeliveryMethod $ businessCategory
    $ janetMailbox $ otherMailbox $ mobileTelephoneNumber
    $ pagerTelephoneNumber $ organizationalStatus $ mailPreferenceOption
    $ personalSignature ) )
objectClasses: ( 0.9.2342.19200300.100.4.5 NAME 'account'
    SUP top STRUCTURAL
    MUST userid
    MAY ( description $ seeAlso $ localityName $ organizationName
        $ organizationalUnitName $ host ) )
objectClasses: ( 0.9.2342.19200300.100.4.6 NAME 'document'
    SUP top STRUCTURAL
    MUST documentIdentifier
    MAY ( commonName $ description $ seeAlso $ localityName
        $ organizationName $ organizationalUnitName $ documentTitle
        $ documentVersion $ documentAuthor $ documentLocation
        $ documentPublisher ) )
objectClasses: ( 0.9.2342.19200300.100.4.7 NAME 'room'
    SUP top STRUCTURAL
    MUST commonName
    MAY ( roomNumber $ description $ seeAlso $ telephoneNumber ) )
objectClasses: ( 0.9.2342.19200300.100.4.9 NAME 'documentSeries'
    SUP top STRUCTURAL
    MUST commonName
    MAY ( description $ seeAlso $ telephonenumber $ localityName
        $ organizationName $ organizationalUnitName ) )
objectClasses: ( 0.9.2342.19200300.100.4.13 NAME 'domain'
    SUP top STRUCTURAL
    MUST domainComponent
    MAY ( associatedName $ organizationName $ description
        $ businessCategory $ seeAlso $ searchGuide $ userPassword
        $ localityName $ stateOrProvinceName $ streetAddress
        $ physicalDeliveryOfficeName $ postalAddress $ postalCode
        $ postOfficeBox $ streetAddress $ facsimileTelephoneNumber
        $ internationalISDNNumber $ telephoneNumber
        $ teletexTerminalIdentifier $ telexNumber
        $ preferredDeliveryMethod $ destinationIndicator
        $ registeredAddress $ x121Address ) )
objectClasses: ( 0.9.2342.19200300.100.4.14 NAME 'RFC822localPart'
    SUP domain STRUCTURAL
    MAY ( commonName $ surname $ description $ seeAlso
        $ telephoneNumber $ physicalDeliveryOfficeName
        $ postalAddress $ postalCode $ postOfficeBox $ streetAddress
        $ facsimileTelephoneNumber $ internationalISDNNumber
        $ telephoneNumber $ teletexTerminalIdentifier
        $ telexNumber $ preferredDeliveryMethod
        $ destinationIndicator $ registeredAddress $ x121Address ) )
objectClasses: ( 0.9.2342.19200300.100.4.15 NAME 'dnsDomain'
    SUP domain STRUCTURAL
    MAY ( ARecord $ MRecord $ MXRecord $ NSRecord $ SOARecord
        $ CNAMERecord ) )
objectClasses: ( 0.9.2342.19200300.100.4.17 NAME 'domainRelatedObject'
    DESC 'RFC1274: an object related to an domain'
    SUP top AUXILIARY
    MUST associatedDomain )
objectClasses: ( 0.9.2342.19200300.100.4.18 NAME 'friendlyCountry'
    SUP country STRUCTURAL
    MUST friendlyCountryName )
objectClasses: ( 0.9.2342.19200300.100.4.20 NAME 'pilotOrganization'
    SUP ( organization $ organizationalUnit ) STRUCTURAL
    MAY buildingName )
objectClasses: ( 0.9.2342.19200300.100.4.21 NAME 'pilotDSA'
    SUP dsa STRUCTURAL
    MAY dsaQuality )
objectClasses: ( 0.9.2342.19200300.100.4.22 NAME 'qualityLabelledData'
    SUP top AUXILIARY
    MUST dsaQuality MAY ( subtreeMinimumQuality $ subtreeMaximumQuality ) )
objectClasses: ( 2.16.840.1.113730.3.2.2 NAME 'inetOrgPerson'
    DESC 'RFC2798: Internet Organizational Person'
    SUP organizationalPerson STRUCTURAL
    MAY ( audio $ businessCategory $ carLicense $ departmentNumber
        $ displayName $ employeeNumber $ employeeType $ givenName
        $ homePhone $ homePostalAddress $ initials $ jpegPhoto
        $ labeledURI $ mail $ manager $ mobile $ o $ pager
        $ photo $ roomNumber $ secretary $ uid $ userCertificate
        $ x500uniqueIdentifier $ preferredLanguage $ userSMIMECertificate
        $ userPKCS12 ) )
objectClasses: ( 1.3.6.1.1.1.2.0 NAME 'posixAccount'
    DESC 'Abstraction of an account with POSIX attributes'
    SUP top AUXILIARY
    MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
    MAY ( userPassword $ loginShell $ gecos $ description ) )
objectClasses: ( 1.3.6.1.1.1.2.1 NAME 'shadowAccount'
    DESC 'Additional attributes for shadow passwords'
    SUP top AUXILIARY MUST uid MAY ( userPassword $ shadowLastChange
        $ shadowMin $ shadowMax $ shadowWarning $ shadowInactive
        $ shadowExpire $ shadowFlag $ description ) )
objectClasses: ( 1.3.6.1.1.1.2.2 NAME 'posixGroup'

```



```

DESC 'Abstraction of a group of accounts'
SUP top STRUCTURAL
MUST ( cn $ gidNumber )
MAY ( userPassword $ memberUid $ description ) )
objectClasses: ( 1.3.6.1.1.1.2.3 NAME 'ipService'
DESC 'Abstraction an Internet Protocol service'
SUP top STRUCTURAL
MUST ( cn $ ipServicePort $ ipServiceProtocol )
MAY description )
objectClasses: ( 1.3.6.1.1.1.2.4 NAME 'ipProtocol'
DESC 'Abstraction of an IP protocol' SUP top STRUCTURAL
MUST ( cn $ ipProtocolNumber $ description )
MAY description )
objectClasses: ( 1.3.6.1.1.1.2.5 NAME 'oncrpc'
DESC 'Abstraction of an ONC/RPC binding'
SUP top STRUCTURAL
MUST ( cn $ oncrpcNumber $ description )
MAY description )
objectClasses: ( 1.3.6.1.1.1.2.6 NAME 'ipHost'
DESC 'Abstraction of a host, an IP device'
SUP top AUXILIARY
MUST ( cn $ ipHostNumber )
MAY ( l $ description $ manager ) )
objectClasses: ( 1.3.6.1.1.1.2.7 NAME 'ipNetwork'
DESC 'Abstraction of an IP network'
SUP top STRUCTURAL
MUST ( cn $ ipNetworkNumber )
MAY ( ipNetmaskNumber $ l $ description $ manager ) )
objectClasses: ( 1.3.6.1.1.1.2.8 NAME 'nisNetgroup'
DESC 'Abstraction of a netgroup'
SUP top STRUCTURAL
MUST cn
MAY ( nisNetgroupTriple $ memberNisNetgroup $ description ) )
objectClasses: ( 1.3.6.1.1.1.2.9 NAME 'nisMap'
DESC 'A generic abstraction of a NIS map'
SUP top STRUCTURAL
MUST nisMapName
MAY description )
objectClasses: ( 1.3.6.1.1.1.2.10 NAME 'nisObject'
DESC 'An entry in a NIS map' SUP top STRUCTURAL
MUST ( cn $ nisMapEntry $ nisMapName )
MAY description )
objectClasses: ( 1.3.6.1.1.1.2.11 NAME 'ieee802Device'
DESC 'A device with a MAC address' SUP top AUXILIARY
MAY macAddress )
objectClasses: ( 1.3.6.1.1.1.2.12 NAME 'bootableDevice'
DESC 'A device with boot parameters' SUP top AUXILIARY
MAY ( bootFile $ bootParameter ) )

```

Здесь перечислены все известные данному серверу **объектные классы**. Вы можете исследовать, что означают все эти OID, с помощью этого [замечательного сайта](#).

Глава 4. Установка LDAP

В рамках нашей новой политики в отношении данного руководства, мы предоставляем информацию и по OpenLDAP, и по ApacheDS. Порядок, в котором они представлены ниже, просто алфавитный. Какую из этих служб каталогов будете использовать Вы? [Здесь](#) Вы найдёте некоторые наши замечания и наблюдения, которые могут помочь (или не помочь) в Вашем выборе.

Apache DS	Установка в Fedora, FreeBSD и Windows
OpenLDAP	Установка в Fedora, FreeBSD и Windows



Глава 4. Установка ApacheDS

[Установка в FreeBSD](#)

[Установка в Fedora Core](#)

[Установка в Windows \(Windows 2000 Server и Windows 7\)](#)

Установка в FreeBSD

Уже совсем скоро (One day real soon now ™).



Установка в Fedora Core

Уже совсем скоро (One day real soon now ™).



Установка ApacheDS в Win2k

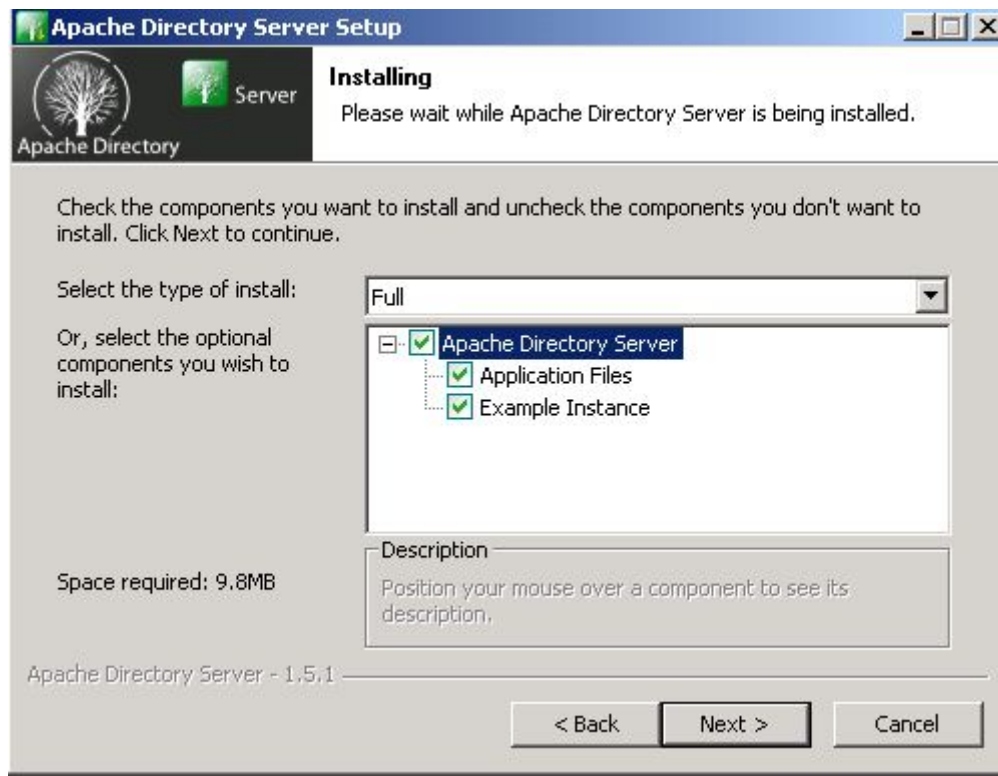
В этом разделе описывается установка ApacheDS на Windows 2000 Server (то же самое применимо и для Windows XP и Server 2003).

ApacheDS - относительно новая (в 2007 году) Open Source реализация LDAPv3, разрабатываемая под эгидой Apache project. ApacheDS - быстрая (тесты показывают, что при выполнении некоторых операций она работает в 10 раз быстрее OpenLDAP), основанная на Java реализация, в состав которой входит интегрированный движок базы данных. Установка ApacheDS включает встроенный движок базы данных и использует уникальное для этого класса систем множество функций транзакционных баз данных, таких как хранимые процедуры и триггеры, разработанные с целью оптимизации производительности. Поскольку ApacheDS написан на Java, ему по природе свойственна кроссплатформенность. Дополнительную информацию по ApacheDS можно получить [на его официальном сайте](#).

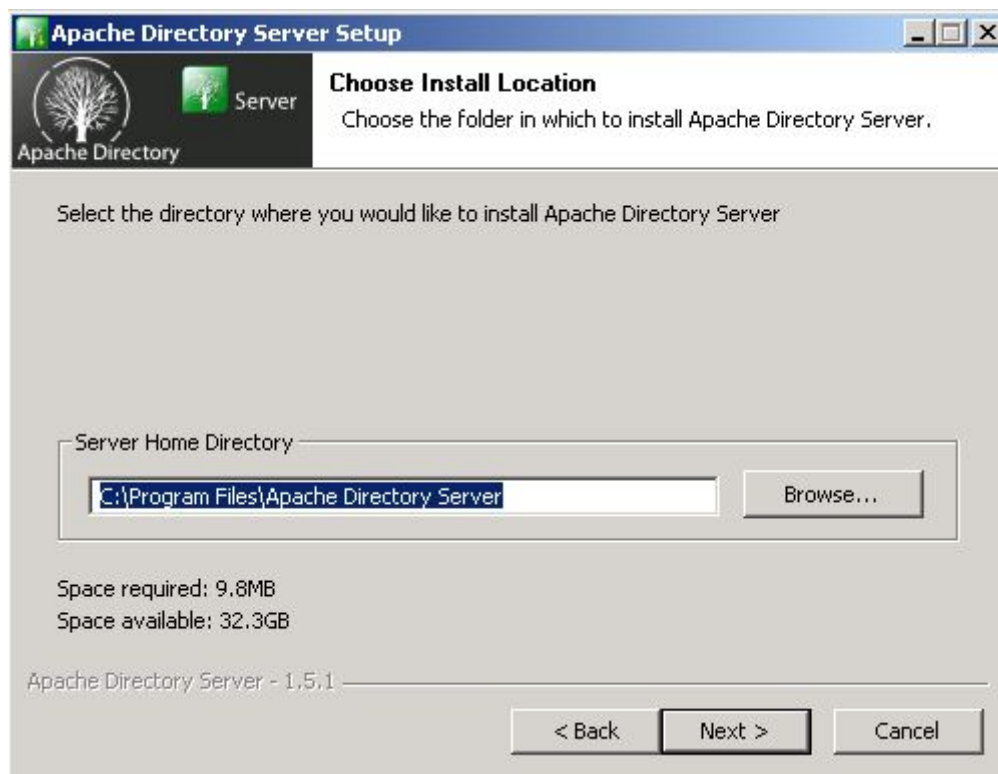
Примечание: ApacheDS разработан на [фреймворке Eclipse](#) и, если Вы намерены серьезно углубиться в изучение данной службы каталогов, то ряд дополнительных инструментов, таких как Directory Studio, потребует установки фреймворка Eclipse. Установка Eclipse - серьезная задача, на решение которой может быть затрачено значительное время. Чтобы использовать только ApacheDS Directory Server, наличия Eclipse не требуется, но требуется наличие Java Run Time Environment (JRE), которое можно получить на [java.com](#). В этом документе освещаются только вопросы установки самого ApacheDS, установка JRE не рассматривается.

Установка в Windows

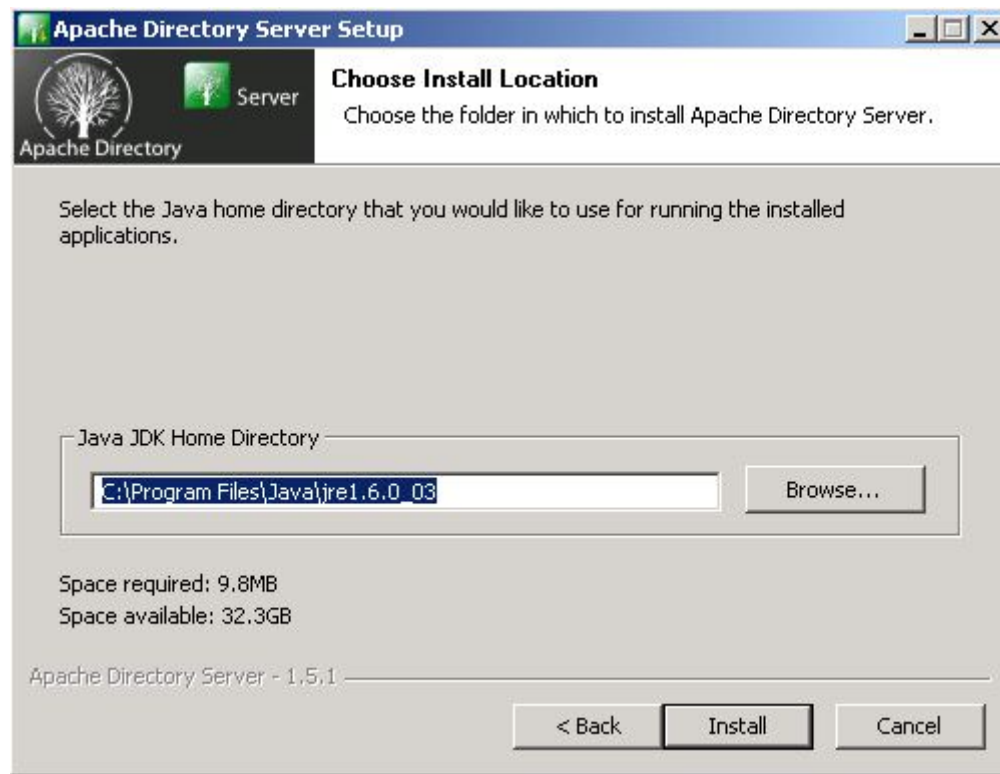
1. Скачайте инсталлятор ApacheDS для Windows [отсюда](#) (на момент создания страницы была версия 1.5.5, на апрель 2012 года - 1.5.7). Выберите и скачайте версию, подходящую для Вашей операционной системы - пользователи Windows должны выбрать 'Download Windows Installer' (скачивается файл apacheds-x.x.x-setup.exe, где x.x.x - требуемая версия), а пользователи *nix должны выбрать соответствующую архитектуру и формат сжатия. Скачайте файл в удобную Вам директорию. **Примечание:** Приведённый ниже процесс инсталляции производился на Windows 2000 Server, но недавно мы устанавливали ApacheDS на Windows 7 и не обнаружили в процедуре установки существенных различий.
2. Дважды щёлкните мышкой по скаченному файлу apacheds-server-x.x.x-setup.exe, чтобы запустить процесс инсталляции. В последующих инструкциях подразумевается использование пути по умолчанию для установки (в нашем случае c:\Program Files\Apache Directory Server).
3. Процесс инсталляции начинается с обычного экрана принятия условий лицензии и выбора устанавливаемых компонентов - по умолчанию будут установлены все компоненты:



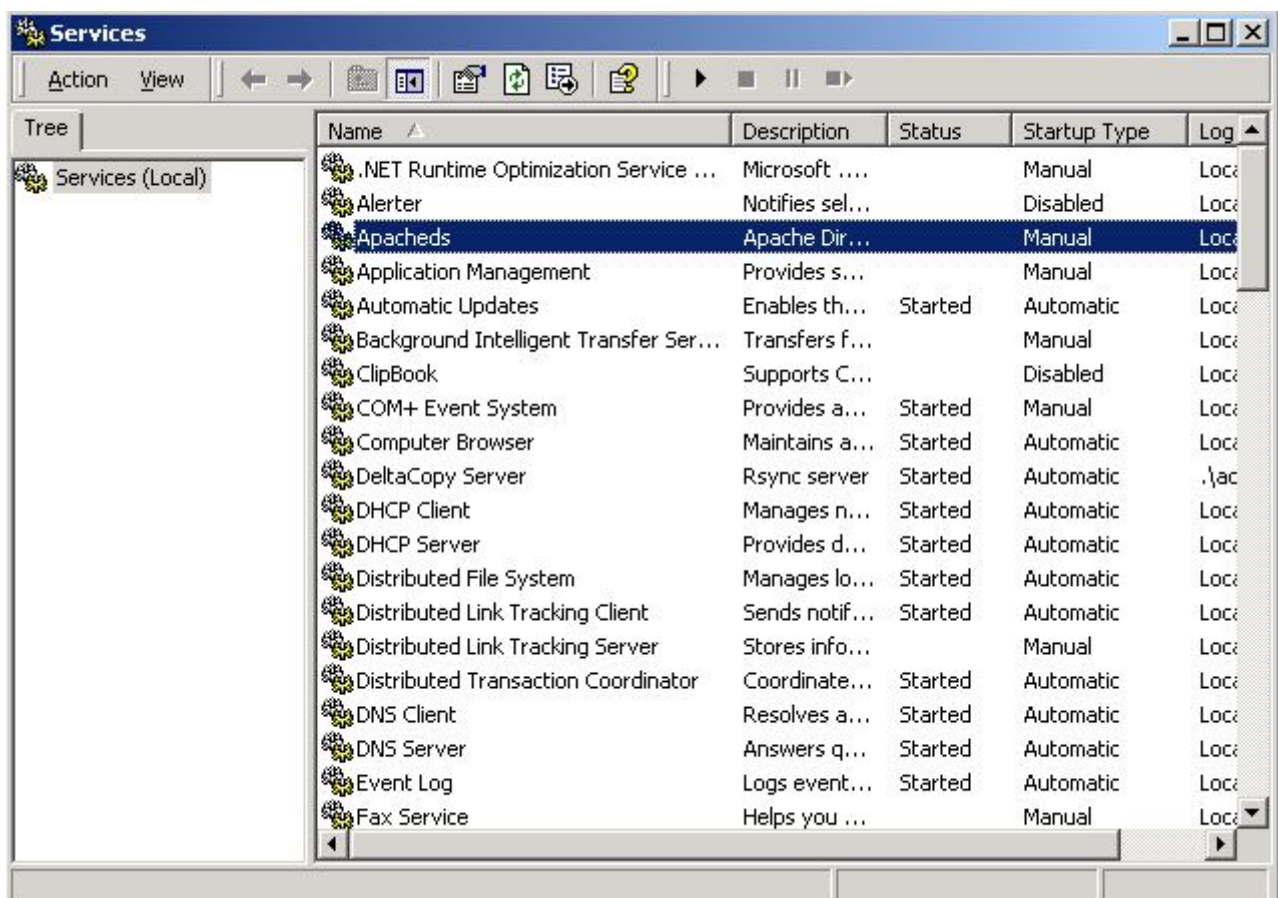
4. На следующем экране вводится путь установки. Мы оставили значение по умолчанию. Пользователи, у которых отторжение имён каталогов с пробелами происходит на глубинном уровне, могут изменить его по своему вкусу.



5. На следующем экране запрашивается расположение Java Run Time Environment (не обращайтесь на путанный текст в заголовке экрана). В нашем случае мы заранее установили последнюю версию (на момент написания статьи JRE 1.6.0_03), полученную с [официального сайта](#). Те пользователи, у которых не установлено JRE, должны отменить инсталляцию ApacheDS (нажав кнопку "Cancel"). Нужно установить JRE и повторить процесс установки ApacheDS. Мы согласились с предложенным расположением по умолчанию.

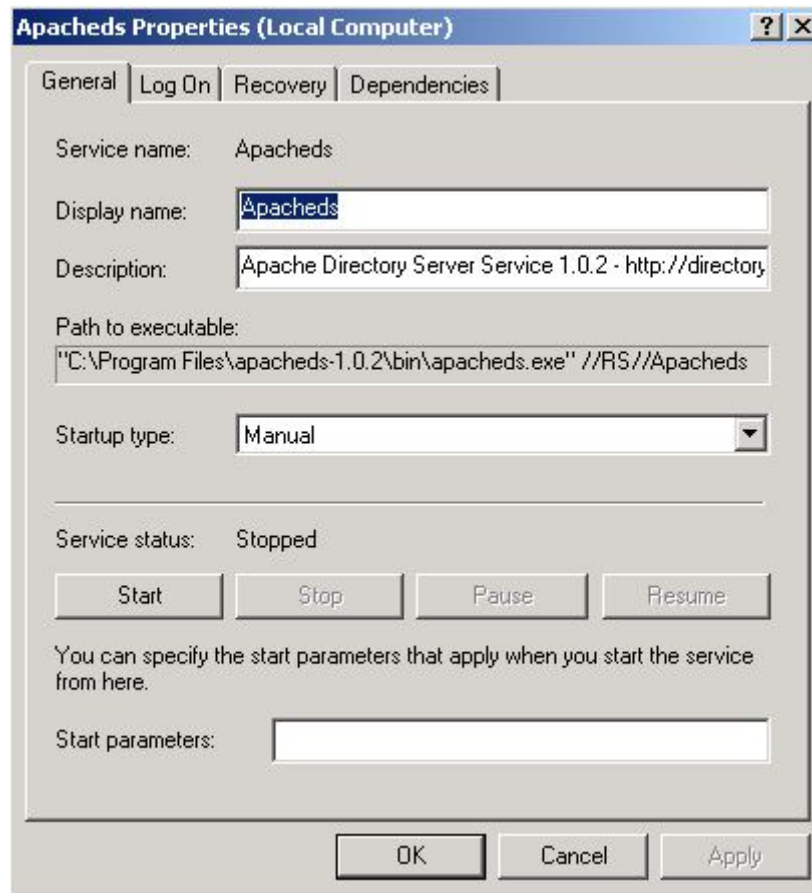


6. Итак, процесс инсталляции завершён. ApacheDS установлен в виде службы Windows NT, которая будет запущена автоматически после перезагрузки. Чтобы изменить это поведение или запустить эту службу прямо сейчас вручную, используйте Пуск->Панель управления->Администрирование->Службы (Start->Control Panel->Administrative Tools->Services), а затем найдите службу Apacheds:



7. Дважды щёлкните мышкой по этой записи и появится следующее окошко, на котором можно изменить *Тип запуска (Startup Type)*: Авто (Automatic) - значение по умолчанию, Вручную (Manual) или Отключено (Disabled). Поскольку эта служба будет использоваться эпизодически, мы установим тип запуска в значение "Вручную". Чтобы запустить службу, нажмите кнопку "Пуск" ("Start"). Сообщения об ошибках будут помещаться в журнал событий (Event Log) (Пуск->Панель управления->Администрирование->Просмотр событий (Start->Programs->Administrative Tools->Event Viewer)).

>Event Viewer)).



8. Конфигурационный файл ApacheDS называется `server.xml` и, если Вы разбираетесь в XML и Java, он Вас ничуть не удивит, в противном случае - он выглядит как стихийное бедствие. Существуют две копии этого файла: `%ApplicationRoot%\conf\server.xml` и `%ApplicationRoot%\instances\default\server.xml`.

По умолчанию ApacheDS использует для службы ldap порт 10389 (это означает, что привилегии администратора/root для выделения порта не требуются). Если Вы хотите, чтобы использовался более привычный порт 389, найдите и отредактируйте эту настройку в файле `server.xml` (обычно этого достаточно).

Глава 4.1. Установка OpenLDAP

4.1.1 Установка в FreeBSD

4.1.2 Установка в Fedora Core

4.1.3 Установка в Windows (XP, Vista, Windows 7)

4.1.1 Установка в FreeBSD

В этом разделе описывается установка OpenLDAP в FreeBSD 8.x.

Место зарезервировано.

4.1.2 Установка в Fedora Core

Уже совсем скоро (One day real soon now TM).



Глава 4.1.3 Установка OpenLDAP в Windows

4.1.1 Установка в FreeBSD

4.1.2 Установка в Fedora Core

4.1.3 Установка в Windows

4.1.3 Установка в Windows

Если Вы хотите развернуть LDAPv3-совместимый Open Source сервер в Windows (XP, Windows 7 или даже, по недоразумению, в Windows Vista), у Вас есть три варианта:

1. OpenLDAP под [cygwin](#).

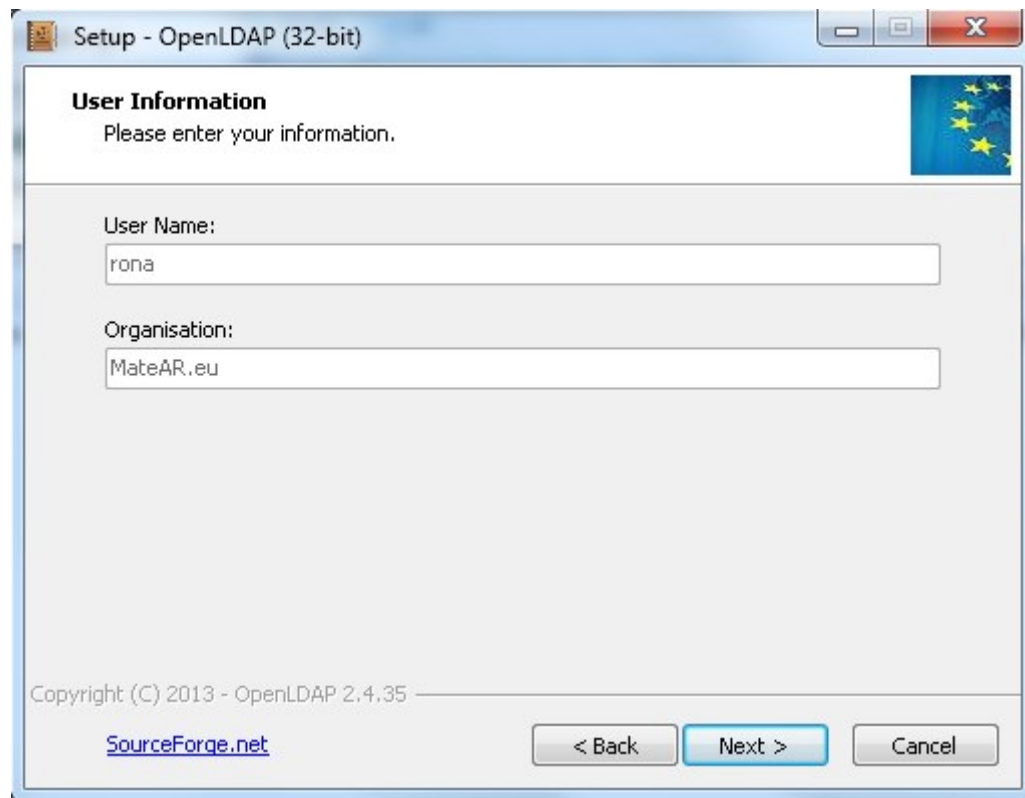
Разработчики установщика cygwin проделали большую работу, чтобы установка была хоть и многословным, но очень простым процессом (полная установка может занять более 30 минут). И спрятали они OpenLDAP так, чтобы никто не догадался (он в категории Libs установщика, но мы Вам этого не говорили). Главный недостаток - то, что версия OpenLDAP обновляется нерегулярно. Если Вы собираетесь заниматься разработкой, либо запускать другие *nix-пакеты под Windows - выбор очевиден.

2. [ApacheDS](#). Работает под Java и включает прекрасный LDAP-клиент/систему разработки, называемую [Apache Directory Studio](#). Этот превосходный инструмент можно использовать в качестве клиента к любой системе, в том числе OpenLDAP. Возможно, процесс установки покажется Вам немного сложным, поскольку система встроена в среду разработки Eclipse (которая любит всё всегда усложнять), но усилия стоят того.
3. Если же Вы хотите простую инсталляцию текущей версии OpenLDAP на Windows в пару кликов, то нет ничего лучше сборки [OpenLDAP для Windows](#). Она обновляется достаточно часто (по состоянию на июль 2013 года версия OpenLDAP 2.4.35). Она устанавливает Berkley DB (базы данных bdb и hdb OpenLDAP), OpenSSL (обеспечивает поддержку TLS в OpenLDAP) и даже Cyrus SASL (обеспечивает поддержку Kerberos). Процесс установки мы опишем ниже. OpenLDAP выполняется не как задача Windows, а как dos-приложение.

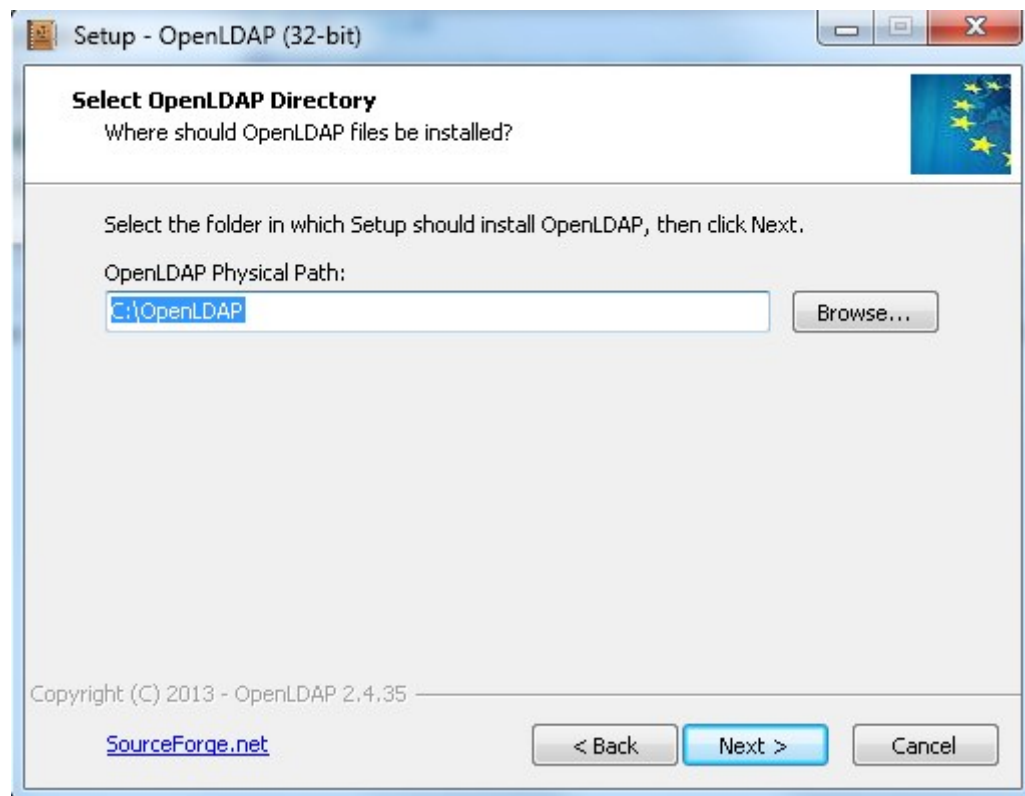
Установка OpenLDAP для Windows

Далее следуют некоторые заметки об установке и использовании OpenLDAP для Windows. При прочтении документации к пакету складывается впечатление, что его возможности значительно больше, чем предоставление основных сервисов OpenLDAP, в частности, там обсуждается использование Microsoft-SQL. Мы проигнорировали все эти новомодные штучки (поскольку не являемся пользователями MS-SQL) и всё равно получили отличную, высоко функциональную инсталляцию OpenLDAP. Порядок действий:

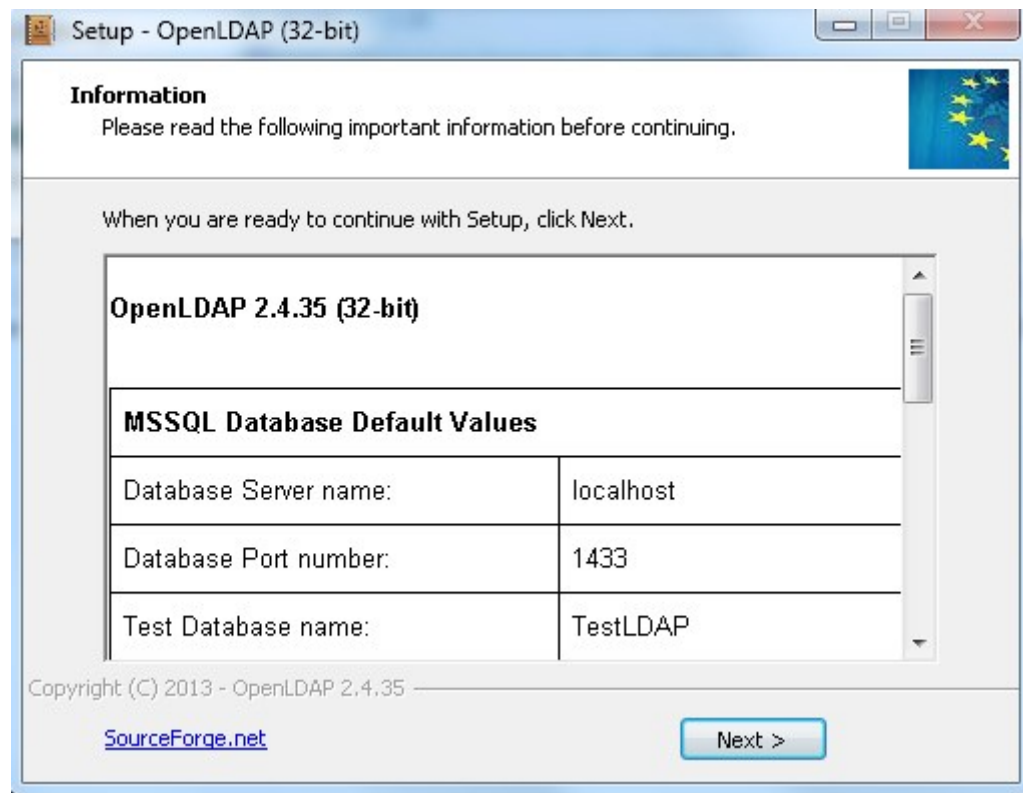
1. Скачайте программное обеспечение [отсюда](#) в подходящую директорию файловой системы.
2. Распакуйте архив и дважды щёлкните для запуска OpenLDAP-2.y.xx-x86.exe (y - старший номер версии, а xx - младший номер версии). Следуйте запросам мастера установки. Установка может быть запущена с правами обычного пользователя (административных привилегий не требуется). Далее показаны экраны с немного путанным содержимым, для которых приводятся дополнительные разъяснения.
3. На этом экране Вас просят ввести Ваши данные, но ввод данных не позволяет. Попробуй разберись! Пройгнорируйте его и нажмите "Next". Ничего страшного не случится.



4. На этом экране показана директория по умолчанию, куда будет произведена установка. Измените её в соответствии с Вашими потребностями или просто нажмите "Next".



5. После установки файлов этот экран показывает некоторую общую информацию о конфигурации сервера. Большая часть её пригодна лишь в том случае, если Вы собираетесь использовать конфигурацию по умолчанию.



На следующем за этим экране запрашивается, хотите ли Вы прочитать документ readme.pdf. Наш совет - не стоит. Снимите галочку и читайте дальше эту инструкцию.

6. Когда последний экран мастера установки закроется и станет историей, у Вас будет следующая конфигурация (подразумевается, что Вы произвели инсталляцию в директорию по умолчанию C:\OpenLDAP, либо, если Вы один из тех, кого раздражает всё, что по умолчанию, скорректируйте значения на свои):

1. Система настроена на использование файла slapd.conf в директории \etc\openldap (директория slapd.d отсутствует - смотрите заметки по olc/cn=config [здесь](#)). Этот slapd.conf полностью работоспособен, в него стоит заглянуть хотя бы потому, что в нём содержатся относительные пути, указывающие на поддиректории и файлы внутри установочной директории. Обслуживание осуществляется на стандартных номерах портов LDAP (389 и 636 для ldaps). Если Вы собираетесь использовать свой собственный файл slapd.conf, обратите внимание на стандартные расположения, на которые указывают директивы pidfile, argsfile (по умолчанию \var\run) и logfile (по умолчанию \var\log) стандартного файла и для упрощения процесса подкорректируйте свои значения, аналогично проверьте расположение файлов наборов схемы данных (\etc\openldap\schema) и директивы directory (\var\db\openldap-data) в Вашем разделе (разделах) database (используйте указанную и при необходимости создайте новые директории).
2. Один из самых запутанных аспектов этой инсталляции OpenLDAP - собраны ли модули статически или динамически. В OpenLDAP для Windows они собраны статически (правильный выбор), это означает, что в директивах loadmodule или loadpath нет необходимости.

В назидание потомкам мы приводим файл slapd.conf по умолчанию пакета OpenLDAP для Windows:

```
#
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#
include          /schema/core.schema

# Define global ACLs to disable default read access.

# Do not enable referrals until AFTER you have a working directory
# service AND an understanding of referrals.
#referral        ldap://root.openldap.org

pidfile          /run/slapd.pid
argsfile         /run/slapd.args
```

```

# Load dynamic backend modules:
# modulepath
# moduleload      back_bdb.la
# moduleload      back_hdb.la
# moduleload      back_ldap.la

# Sample security restrictions
#       Require integrity protection (prevent hijacking)
#       Require 112-bit (3DES or better) encryption for updates
#       Require 63-bit encryption for simple bind
# security ssf=1 update_ssf=112 simple_bind=64

# Sample access control policy:
#       Root DSE: allow anyone to read it
#       Subschema (sub)entry DSE: allow anyone to read it
#       Other DSEs:
#           Allow self write access
#           Allow authenticated users read access
#           Allow anonymous users to authenticate
#       Directives needed to implement policy:
# access to dn.base="" by * read
# access to dn.base="cn=Subschema" by * read
# access to *
#       by self write
#       by users read
#       by anonymous auth
#
# if no access controls are present, the default policy
# allows anyone and everyone to read anything but restricts
# updates to rootdn. (e.g., "access to * by * read")
#
# rootdn can always read and write EVERYTHING!

#####
# BDB database definitions
#####

database      bdb
suffix        "dc=my-domain,dc=com"
rootdn        "cn=Manager,dc=my-domain,dc=com"
# Cleartext passwords, especially for the rootdn, should
# be avoid. See slapdpasswd(8) and slapd.conf(5) for details.
# Use of strong authentication encouraged.
rootpw        secret
# The database directory MUST exist prior to running slapd AND
# should only be accessible by the slapd and slap tools.
# Mode 700 recommended.
directory     /openldap-data
# Indices to maintain
index          objectClass      eq

```

3. Для запуска сервера выполните "Пуск" (Start) -> "Все программы" (All Programs) -> OpenLDAP -> Start LDAP Server:



Примечание: OpenLDAP для Windows использует инсталляционный файл типа .exe, а не .msi, и потому до появления значков в меню "Все программы" (All Programs) может пройти довольно много времени (до 30 минут).

Если Вы нетерпеливы, перейдите в директорию libexec и дважды щёлкните на StartLDAP.cmd. Это приведёт к немедленному запуску LDAP-сервера.

При старте сервера откроется окно сессии dos, куда будет выведена тонна информации, после чего окно останется открытым (Вы должны явно завершить OpenLDAP, нажав CTL-C в этом окне dos). Если что-то пошло не так, окно немедленно закроется. Если Вы используете директиву logfile (поставляемый по умолчанию файл slapd.conf использует \var\log\openldap.log), то ищите ошибки в файле журнала. Если Вы не используете директиву logfile, значит Вам не повезло.

Большой объем информации, выводимой в окне dos, может серьезно замедлить работу сервера. Чтобы его уменьшить или ликвидировать, просто откройте (с помощью notepad) файл \libexec\StartLDAP.cmd и из последней строки этого файла либо вообще удалите аргумент **-d -1** (для отмены вывода, за исключением катастрофических ошибок), либо измените -1 на какое-либо другое значение (подробнее [здесь](#)). Например, если значение равно 8, будет выводиться только информация о соединениях. Вот этот файл во всей красе, чтобы Вы могли увидеть, что представляет из себя последняя строка:

```
@echo off
verify on
Rem SET HOME=
SET ODBCINI=..\etc\odbc.ini
SET ODBCYSINI=..\etc
SET FREETDS=..\etc\freetds.conf
SET TDSVER=8.0
SET TDS_DUMP=..\var\log\freetds.log
SET RANDFILE=..\bin\rfile.rnd
SET LDAPCONF=..\etc\openldap\ldap.conf
SET LDAPRC=..\bin\ldaprc

Rem Adjust accordingly
Rem SET KRB5_CONFIG=C:\Heimdal\etc\krb5-pkinit.conf
Rem SET KRB5_KTNAME=C:\Heimdal\etc\krb5.keytab
Rem SET KRB5CCNAME=FILE:C:/Heimdal/tmp/krb5cc_500
SET FQDN=localhost

slapd.exe -d -1 -h "ldap://%FQDN/ ldaps://%FQDN/" -f ..\etc\openldap\slapd.conf
```

4. Стандартные ldap-утилиты OpenLDAP (ldapsearch и другие) располагаются в директории bin. В OpenLDAP для Windows есть удобное окно командной строки, предварительно сконфигурированное на эту директорию:



Вы также можете открыть любое окно с сессией dos и перейти в c:\openldap\bin, либо поместите этот путь в свою переменную path ("Пуск" (Start) -> "Панель управления" (Control panel) -> "Система" (System) -> в открывшемся окне "Свойства системы" (Advanced system settings) выберите вкладку "Дополнительно" (Advanced) -> нажмите кнопку "Переменные среды" (Environmental settings) -> в открывшемся окне "Переменные среды" в нижней части найдите переменную path и добавьте **;c:\openldap\bin**). Откройте любое окно с сессией dos ("Пуск" (Start)->"Выполнить" (Run)->cmd) и можете запускать утилиты ldap. **Примечание:** slap-утилиты (slapadd и другие) находятся в директории sbin, возможно, Вы захотите добавить в переменную path также и путь **;c:\openldap\sbin**.

7. Как мы уже говорили, для остановки сервера OpenLDAP перейти в окно с сессией dos, в котором он выполняется, и нажать CTRL-C. Сервер остановится, и Вам будет выведен запрос **"Terminate Batch Job?"**; если в ответ на него вы нажмете "y", окно закроется.

Если эта процедура не была соблюдена (например, Вы завершили работу компьютера, не остановив LDAP-сервер), сервер, вероятнее всего, откажется стартовать при следующем запуске. В

этом случае перейдите в директорию `c:\openldap\var\run` и удалите все файлы в этой директории (`slapd.args` и `slapd.pid`). Теперь сервер должен запуститься. Если нет - обратитесь к файлу журнала (по умолчанию в `\var\log`). Ведь Вы же установили директиву `logfile`, правда?

Глава 5. Примеры OpenLDAP

В этом разделе приводится набор решений для каталога в стиле "делаем строго по инструкции" со ссылками на разъясняющую информацию более высокого уровня. Переходя по ссылкам, Вы можете нечаянно что-то узнать — не говорите потом, что Вас не предупреждали.

В этом разделе мы возьмём за основу реализацию отдельно стоящего каталога LDAP и будем последовательно, шаг за шагом, её совершенствовать.

Перед тем, как начать — если Вы ещё не обзавелись LDAP-браузером, сделайте это сейчас. Open Source предоставляет [таковых](#) с избытком. LDAP-браузер — это специализированный [LDAP-клиент](#), позволяющий, в общем случае, получать доступ и исследовать LDAP-каталог. Это бесценный инструмент. Без него Вы слепы. Как минимум, LDAP-браузер должен быть способен производить подключения — либо анонимно, либо используя указанный DN, экспортировать LDIF-файлы и выполнять поиск. Если же он может ещё и добавлять записи и атрибуты, а также импортировать данные из LDIF-файлов — это вообще прекрасно. Если Вы мазохист, можете, конечно, использовать утилиты командной строки, такие как [ldapadd](#) и [ldapsearch](#). Но ведь можно найти и более действенные способы удовлетворения своей страсти — побиться головой о стену, например. В любом случае, выбор полностью за Вами.

Содержание

5.1 Простой каталог

- 5.1.1 Проектирование DIT
- 5.1.2 Выбор структурного объектного класса
- 5.1.3 Файл `slapd.conf`
- 5.1.4 Файл LDIF
- 5.1.5 Загрузка LDIF-файла
- 5.1.6 Добавление записей с помощью LDIF
- 5.1.7 Изменение записей с помощью LDIF
- 5.1.8 Напоследок просто подурчимся

5.2 Защита каталога

- 5.2.1 Политика безопасности
- 5.2.2 Добавление групп
- 5.2.3 ACL — директивы `access` файла `slapd.conf`
- 5.2.4 Тестирование ACL

5.3 Расширение иерархии

- 5.3.1 Расширение требований
- 5.3.2 Реализация расширения
- 5.3.3 LDIF для реализации расширения
- 5.3.4 ACL — директивы `access` файла `slapd.conf`
- 5.3.5 Тестирование ACL

5.4 Создание и добавление объектов

- 5.4.1 Требования
- 5.4.2 Реализация
- 5.4.3 Определения атрибутов
- 5.4.4 Определение объектного класса и набора схемы
- 5.4.5 ACL — директивы `access` файла `slapd.conf`
- 5.4.6 LDIF
- 5.4.7 Тестирование изменений

5.5 Технология единого входа (Single Sign On)

5.6 Отсылки и репликация

В данных примерах описано последовательное усовершенствование каталога от простейшей отправной точки в следующем порядке:

1. **Шаг 1: обыкновенный (заурядный)** каталог имён и адресов без системы безопасности (анонимный доступ на чтение и ограниченный доступ на запись).
2. **Шаг 2:** добавление политики безопасности — анонимный доступ на чтение определённых полей, защищённый доступ к другим полям и ограниченный защищённый доступ на запись.
3. **Шаг 3:** добавление двух дополнительных веток к иерархии — общедоступной, но защищённой ветки `customer` и ветки `equipment`.
4. **Шаг 4:** добавление новых объектных классов и некоторых атрибутов.
5. **Шаг 5:** усовершенствование до технологии единого входа (`single sign on`, SSO) для поддержки Samba, FTP, Apache и email (в данном случае `courier`).
6. **Шаг 6:** распределение контроля (отсылки), внедрение одного **скрытого** главного сервера (репликация).

Примечание: OpenLDAP находится в процессе перехода от текстового конфигурационного файла (`slapd.conf`) к конфигурации времени исполнения (`on-line configuration`, OLC или `cn=config`). Поскольку OpenLDAP серии версий 2.4 всё ещё поддерживает `slapd.conf`, большинство дистрибутивов продолжают плыть по воле волн, используя эти файлы. Другие решили перейти к использованию OLC (`cn=config`), некоторые даже утверждают, что они это сделали и предоставляют специальные инструменты и скрипты чтобы "облегчить Вашу жизнь". Если Ваша инсталляция относится к этой последней категории, Вам нужно сделать три вещи: прочитать документацию, поставляемую с реализацией OpenLDAP Вашего дистрибутива; хорошенько познакомиться с [OLC \(cn=config\)](#); немного погоревать — изначальная кривая Вашего обучения задралась несколько круче, чем предполагалось. Тем не менее, Вы всегда можете вернуться к классической конфигурации `slapd.conf`, и мигрировать на OLC лишь когда Вы сами посчитаете это нужным (для понимания процесса перехода внимательно прочитайте [этот раздел](#)). Разделы примеров содержат замечания и пояснения, охватывающие конфигурацию `slapd.conf` и, возможно, OLC (`cn=config`).

5.1 Простой каталог

Начнём с простого каталога имён и адресов без системы безопасности. Когда Вы начинаете работу с таким сложным приложением, как LDAP, есть около 6-ти миллионов вещей, которые могут пойти не так. Мы пропускаем систему безопасности, чтобы уменьшить количество ошибок примерно до 3-х миллионов. Не помещайте в этот каталог какую-либо конфиденциальную информацию. Безопасность, — даже тривиальная система безопасности, — неотъемлемая часть любого каталога. Безопасность, однако, поддаётся наращиванию; она не влияет на хранящиеся в каталоге данные, и может быть [модернизирована](#) в любое время.

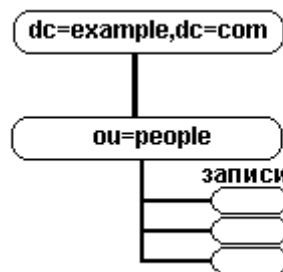


5.1.1 Проектирование DIT

На самом деле, это очень важный этап, и Вы можете провести всю оставшуюся жизнь, проектируя своё DIT. Люди даже пишут книги по этой теме. Существует несколько [документированных](#) нами **общепринятых практик**, но в Вашем конкретном случае они могут не заработать. Примите их за то, чем они, по сути, и являются — отправной точкой для Ваших размышлений по этому предмету.

Мы решили следовать классическому, [описанному в RFC 2377 формату](#), и использовали формат **dc** для [корневой записи](#) DIT.

Также мы организовали иерархию адресной книги в разделе **people** и использовали для этого раздела атрибут **ou**. Итак, мы остановились на вот такой структуре DIT:



5.1.2 Выбор структурного объектного класса

Вселенский плач и скрежет зубов — обычный аккомпанемент для выбора первичного **объектного класса**. И всё же это необходимый ритуал LDAP, так давайте займёмся этим прямо сейчас!

В общем случае, на наш взгляд, наиболее полезный для **обыкновенных (заурядных)** каталогов **объектный класс** — это [inetorgperson](#), поскольку он является частью БОЛЬШОЙ иерархии с большим количеством атрибутов (убедитесь сами, следуя по ссылкам SUP — вышестоящей иерархии). Если нам не хватит какого-нибудь атрибута — можно добавить его позже, в [шаге 4](#) как раз это и делается. Решение принято; дело закрыто.



5.1.3 Файл slapd.conf

Вот пример slapd.conf, который позволит нам начать работу. В нём используется механизм манипуляции данными Oracle Berkeley DataBase (BDB) (в настоящее время предпочитаемая и рекомендуемая OpenLDAP база данных):

```
#
##### ПРИМЕР 1 — ПРОСТОЙ КАТАЛОГ #####
#
# Примечание: inetorgperson использует атрибуты и
#            объектные классы из всех трёх наборов схемы
#
# NB: В RH Linux наборы схемы располагаются в /etc/openldap
#
include      /usr/local/etc/openldap/schema/core.schema
include      /usr/local/etc/openldap/schema/cosine.schema
include      /usr/local/etc/openldap/schema/inetorgperson.schema

# НЕТ СИСТЕМЫ БЕЗОПАСНОСТИ — условия безопасности пропущены
# по умолчанию анонимный доступ на чтение
# только rootdn может осуществлять запись

# НЕТ ОТСЫЛОК

# НЕ заботимся о файле ARGS, пока не обретём уверенности в своих силах
# stop-скрипту slapd следующая строка необходима для работы:
pidfile /var/run/slapd.pid

# Включаем большое количество вывода в журнал — нам это может понадобится
# но при этом генерируются журналы огромных размеров
loglevel     -1

# Определения ЗАГРУЗКИ МОДУЛЕЙ
# до версии 2.3 не требуется (закомментируйте)
moduleload   back_bdb.la

# НЕТ соединений TLS

# Определение backend не требуется

#####
# Определение базы данных bdb
#
# Замените ниже example и com на подходящее имя домена
#
# Если у Вас нет домена, то, поскольку example.com зарезервирован для
```

```
# экспериментов, можете оставить его, либо поменять на my.inc
#
#####

database bdb
suffix "dc=example, dc=com"

# root или администратор
rootdn "cn=jimbob, dc=example, dc=com"
rootpw dirtysecret
# Директория базы данных ДОЛЖНА существовать до того, как Вы запустите slapd
# Не забудьте поменять путь на подходящий Вам
directory /var/db/openldap/example-com

# Устанавливаемые для каталога индексы
# Только полное соответствие для unique id
index uid eq
# Стандартный поиск для commonname, givenname и email
index cn,gn,mail eq,sub
# Несколько вариантов для поиска surname
index sn eq,sub
# В индексах выше sub включает в себя subinitial, subany, subfinal
# Оптимизируем поиск по подразделениям
index ou eq
# Если при поиске будет встречаться objectClass, раскомментируйте следующую строку
# index objectClass eq
# Продемонстрировано использование параметра индексирования default
index default eq,sub
# Настройка индексирования пропущена, по умолчанию используется eq,sub
index telephoneNumber

# Другие параметры базы данных
# Дополнительная информация в разделе справочника по slapd.conf
cachesize 10000
checkpoint 128 15
```

Чтобы получить данный пример в виде отдельного текстового файла, используйте [эту ссылку](#) и функцию **сохранить как** Вашего браузера.

Примечания:

1. Вы можете настроить [специфичные для баз данных BDB параметры](#). Приведённые в примере значения по умолчанию для cachesize и checkpoint представляют собой разумные настройки, способные предотвратить большинство распространённых сбоев при записи в базу данных. Если Вы хотя бы немного заботитесь о производительности, Вы должны хорошо понимать настройки BDB и экспериментировать с ними.
2. Настройки безопасности определяются с помощью директив [access](#). В данном случае их отсутствие позволяет осуществлять анонимный доступ на чтение (аутентификация не требуется), доступ на запись не даётся. Поскольку в примере есть директивы [rootdn](#) и [rootpw](#), можно осуществлять запись в каталог от имени этого [dn](#), а также экспериментировать с [подключениями](#) при создании записей.
3. В примере отсутствует директива [backend](#), строгой необходимости в которой нет, даже несмотря на то, что в некоторых документациях утверждается обратное. Если Вам нравится печатать, добавьте её.
4. Выбранные нами [индексы](#) призваны оптимизировать некоторые формы доступа. При поиске можно использовать и те атрибуты, которые не были проиндексированы — просто это будет дольше.



5.1.4 Файл LDIF

Представленный ниже LDIF создаёт структуру DIT и добавляет единственную запись в раздел **people**, позднее с помощью LDIF мы добавим другие две записи. Для добавления других записей, поиска и всех остальных забавных вещей можно также воспользоваться LDAP-браузером.

Данный LDIF будет добавлен с помощью утилиты [ldapadd](#) при запущенном OpenLDAP (slapd). Если у Вас много данных, которые Вы готовы поместить в каталог, Вам придётся вновь и вновь создавать LDIF и помещать в каталог столько данных, сколько Вам надо. Если у Вас действительно **много** данных (больше 1 000 записей), возможно Вы захотите воспользоваться методом загрузки записей [slapadd](#) из соображений экономии ресурсов.

Перед тем, как создавать LDIF-файл, мы должны выяснить, какие данные ДОЛЖНЫ присутствовать — в нашем случае это также позволит минимизировать записи, чтобы примеры оставались короткими. Беглый просмотр иерархии объектных классов [inetorgperson](#) позволяет найти все те атрибуты, которые должны присутствовать. Наша инспекция показала, что всего два атрибута, **cn** и **sn**, являются обязательными. Следующий LDIF устанавливает нашу первоначальную иерархию, а также добавляет одну запись в раздел **people**:

```
## Определение DIT ROOT/BASE/SUFFIX ####
## используется формат RFC 2377
## замените встречающиеся ниже example и com на требуемые
## или, для экспериментов, оставьте как есть

## dcObject — это ВСПОМОГАТЕЛЬНЫЙ объектный класс и, кроме него, запись
## ДОЛЖНА иметь СТРУКТУРНЫЙ объектный класс (в данном случае, organization)
# это последовательность ЗАПИСИ и ей предшествует ПУСТАЯ СТРОКА

dn: dc=example,dc=com
dc: example
description: Моя замечательная компания. В эту строку можно поместить столько текста,
сколько хотите
в этой строке продолжение информации из предыдущей строки вплоть до 32Kb
строка оканчивается либо на <CR>, либо на <CR><LF>, то есть отрабатывается ENTER
с систем как Windows, так и *nix — новая строка ДОЛЖНА начинаться с ОДНОГО ПРОБЕЛА
objectClass: dcObject
objectClass: organization
o: Example, Inc.

## ПЕРВЫЙ уровень иерархии — люди (people)
## для объектных классов используется смешанная форма записи в верхнем и нижнем регистре
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой

dn: ou=people, dc=example,dc=com
ou: people
description: All people in organisation
objectclass: organizationalunit

## ВТОРОЙ уровень иерархии
## ДОБАВЛЯЕМ одну запись в ПЕРВЫЙ уровень (people)
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой
# ou: Human Resources — это название подразделения

dn: cn=Robert Smith,ou=people,dc=example,dc=com
objectclass: inetOrgPerson
cn: Robert Smith
cn: Robert J Smith
cn: bob smith
sn: smith
uid: rjsmith
userpassword: rJsmith
carlicense: HISCAR 123
homephone: 555-111-2222
mail: r.smith@example.com
mail: rsmith@example.com
mail: bob.smith@example.com
description: swell guy
ou: Human Resources
```

Чтобы получить данный пример в виде отдельного текстового файла, используйте [эту ссылку](#) и функцию **сохранить как** Вашего браузера.

Примечания:

1. **<предупреждение>** В версиях данного руководства до 0.1.2 в приведённом выше примере мы некорректно определяли **dn: dc=example,dc=com** как **dn: dc=example.com**. Это успешно загружалось в OpenLDAP 2.0 и 2.1, но стало отклоняться в OpenLDAP 2.2 (ошибка 64). **</предупреждение>**
2. Записи при добавлении начинаются со строки, первыми символами которой являются 'dn:'. В общем случае, для этих целей может использоваться любой атрибут при условии соблюдения уникальности 'dn:', и, во избежание излишней нагрузки при поиске, обычно им становится наиболее часто используемый при доступе к записи DN. Поэтому в последней записи в приведённом выше LDIF используется значение **cn=Robert Smith,ou=people,dc=example,dc=com**, подразумевая, что cn= будет наиболее часто используемым DN при доступе к записям. Однако, если запись будет использоваться для аутентификации, то данный DN ДОЛЖЕН быть таким, чтобы его можно было использовать во время операций подсоединения. В этом случае более подходящими могут быть DN типа **uid=rjsmith,ou=people,dc=example,dc=com**. Поскольку для данного начального DN (или DN

'создания') нет специального термина в стандартах LDAP, то, при использовании в целях идентификации, он иногда называется [DN принципа \(Principal DN\)](#). [Дополнительная информация по этой теме](#).

3. Для упрощения описания некоторых концепций LDIF мы используем некоторые введённые нами [термины](#).
4. Как упоминалось в комментариях, директива **version**: не является строго обязательной. Если она присутствует, то (в настоящий момент) она должна быть установлена в 1 для указания версии 1 формата LDIF. Данная директива включена просто потому, что делать это полезно (Good Thing™ to do). Будущие версии могут быть несовместимы, либо могут налагать более строгие требования к корректности — лучше занять хорошие привычки сейчас. Во время нашего тестирования мы заметили, что некоторые версии OpenLDAP могут приводиться в ступор директивами **version**, выдавая сообщения об ошибках с предположениями об отсутствии DN. Если это произошло, удалите строку version и все соответствующие комментарии.
5. Комментарии обозначаются # только в первом символе строки. В следующей строке # интерпретируется как содержимое:

```
cn: my name #this is my name
```

В результате атрибут cn будет содержать 'my name #this is my name'.

6. Между записями должна быть КАК МИНИМУМ одна [ПУСТАЯ строка](#) (перед строками, начинающимися с **dn:**). Это ОЧЕНЬ важно — в противном случае могут возникать странные ошибки.
7. Предполагается, что строка является [строкой ПРОДОЛЖЕНИЯ](#), если предыдущая строка оканчивается разделителем строк, а текущая начинается РОВНО С ОДНОГО ПРОБЕЛА.
8. В именах атрибутов в приведённом выше файле непоследовательно используются символы в верхнем и нижнем регистрах — в частности, эта ужасная псевдовенгерская нотация **objectClass** или все буквы в нижнем регистре **objectclass**. Работает и то, и другое. Следуйте любому стилю на Ваш выбор.
9. Строка **cn: bob smith** содержит несколько **кажущихся** ошибок. Здесь есть два пробела между 'bob' и 'smith' и оба они в нижнем регистре. Ни одна из этих **кажущихся** ошибок не имеет никакого эффекта в работе каталога, поскольку атрибут **cn** (дочерний от атрибута **name**) использует нечувствительное к регистру [правило соответствия](#) и LDAP при поиске выполняет некоторые [интересные вещи](#).
10. В большом количестве документации Вы встретите **objectclass: top** и определение всех **объектных классов** в иерархии. Начиная с OpenLDAP 2.0, [это не является обязательным](#). Принимайте решение, будете ли Вы это делать, исходя из требований Вашей системы.
11. Пробел, следующий за : в каждой строке, **ЖИЗНЕННО НЕОБХОДИМ**.
12. Во многих документациях Вы найдёте определённую для **rootdn (администратора)** запись в файле LDIF (в данном примере slapd.conf это **rootdn "cn=jimbob,dc=example,dc=com"**). Так как **rootpw** определён в файле slapd.conf, наличие данной записи в каталоге не является обязательным и может быть потенциально опасным, поскольку таким образом данная запись предоставляется для внешнего доступа. Мы категорически не советуем Вам так поступать и приводим подробное обсуждение этой темы [здесь](#).
13. Мы использовали несколько значений **cn**, чтобы увеличить шансы найти человека по имени. Мы использовали параметры **eq,sub** (в файле slapd.conf) при индексировании атрибута **cn**. Поступите Вы так или нет — вопрос политики и предпочтений. Можно предположить, что **sn** будет основным параметром при поиске человека, в этом случае, возможно, Вы захотите использовать параметр индексирования **approx** для поиска по **созвучным** значениям. Выбор значения **Robert Smith** для помещения в строку **dn: (dn: cn=Robert Smith,dc=example,dc=com)** абсолютно произволен, его можно заменить на любое другое из значений **cn**.
14. В каждой из этих записей неявно используется директива **changetype: add**, которая должна следовать сразу за строкой **dn:** и которая является значением по умолчанию. Мы рассмотрим использование LDIF-директивы **changetype** позже.

15. Некоторые из добавленных нами атрибутов могут показаться немного странными, просто они будут использоваться в дальнейших примерах для иллюстрации некоторых моментов. Вы можете добавить в LDIF любые другие атрибуты из выбранной Вами иерархии объектных классов.

Большинство систем каталогов может быть **построено** с помощью приведённого выше подмножества [полного набора директив LDIF-файлов](#).



5.1.5 Загрузка LDIF-файла

Чтобы загрузить LDIF-файл, нам понадобится работающий сервер OpenLDAP (slapd), поэтому, если он еще не запущен, мы должны запустить его сейчас, с помощью команды, подобной следующей:

```
[redhat] /etc/rc.d/init.d/ldap start
[bsd] /usr/local/etc/openldap/slapd.sh start

# убеждаемся, что slapd запущен
ps ax | grep slapd
# (если slapd успешно запустился, Вы увидите запись его процесса)
```

Примечание: аргументы командной строки демона slapd можно найти [здесь](#).

Предположим, что мы сохранили приведённый выше LDIF как createdit.ldif в директорию /tmp. Тогда мы будем загружать LDIF-файл с помощью примерно такого вызова [ldapadd](#) (приведённая ниже команда разбита на две строки только из соображений форматирования HTML и должна записываться в одну строку):

```
ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"
-f /tmp/createdit.ldif -w dirtysecret
```

Замените **ldaphost.example.com** на имя того хоста, на котором располагается Ваш каталог LDAP. Если мы вызываем команду на том же хосте, на котором работает LDAP-сервер, параметр -H может быть опущен. Параметр -x указывает, что мы не будем использовать SASL-аутентификацию, он требуется только начиная с OpenLDAP версий 2.x. В параметре -D указан DN, от имени которого мы производим подключение, он необходим, поскольку мы определили [rootdn](#) и не определили других установок **access**; — администратору приходится выполнять всю работу. Указывать пароль вслед за параметром -w, как мы это сделали в примере, мягко скажем, небезопасно, впрочем, как небезопасен на текущий момент и весь наш каталог!



5.1.6 Добавление записей с помощью LDIF

Следующий LDIF демонстрирует, как добавить дополнительные записи с помощью LDIF.

```
version: 1

## Добавление одной записи в раздел people

dn: cn=John Smith,ou=people,dc=example,dc=com
objectclass: inetOrgPerson
cn: John Smith
cn: John J Smith
sn: Smith
uid: jsmith
userpassword: jSmith
carlicense: HISCAR 124
homephone: 555-111-2223
mail: j.smith@example.com
mail: jsmith@example.com
mail: john.smith@example.com
ou: Sales

## Добавление другой записи в раздел people

dn: cn=Sheri Smith,ou=people,dc=example,dc=com
objectclass: inetOrgPerson
cn: Sheri Smith
```

```
sn: smith
uid: ssmith
userpassword: sSmith
carlicense: HERCAR 125
homephone: 555-111-2225
mail: s.smith@example.com
mail: ssmith@example.com
mail: sheri.smith@example.com
ou: IT
```

Чтобы получить данный пример в виде отдельного текстового файла, используйте [эту ссылку](#) и функцию **сохранить как** Вашего браузера.

Предположим, что мы сохранили приведённый выше LDIF как addentry.ldif в директории /tmp. Тогда мы будем загружать LDIF-файл с помощью примерно такого вызова `ldapadd` (приведённая ниже команда разбита на две строки только из соображений форматирования HTML и должна записываться в одну строку):

```
ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"
-f /tmp/addentry.ldif -w dirtysecret
```



5.1.7 Изменение записей

Следующий LDIF демонстрирует, как изменить (модифицировать) записи с помощью LDIF. Обычно быстрее и проще воспользоваться для этих целей Вашим LDAP-браузером, однако, если Ваши изменения носят массовый характер, метод LDIF будет быстрее.

```
version: 1

## МОДИФИЦИРУЕМ запись Robert Smith

dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: 555-555-1212
telephonenumber: 212
-
replace: uid
uid: rjosmith
-
replace: mail
mail: robert.smith@example.com
mail: bob.smith@example.com
-
# добавление атрибута с использованием формата URL
add: jpegphoto
jpegphoto:< file://path/to/jpeg/file.jpg
-
delete: description
```

Чтобы получить данный пример в виде отдельного текстового файла, используйте [эту ссылку](#) и функцию **сохранить как** Вашего браузера.

Примечания:

1. Директива `changetype: modify` сообщает OpenLDAP, что мы собираемся вносить изменения в данную запись. Для удаления записи целиком можно использовать директиву `changetype: delete`.
2. Директива `add: атрибут` сообщает OpenLDAP, что мы собираемся добавить **атрибут**, определение которого должно следовать сразу за этой директивой. В данном случае мы добавляем к записи атрибуты **title** и **telephonenumber** (являющиеся атрибутами объектного класса `organizationalPerson`).
3. Директива `replace: атрибут` сообщает OpenLDAP, что мы собираемся заменить **атрибут**, определение которого должно следовать сразу за этой директивой. В данном случае мы заменяем атрибуты **uid** и **mail**. Для атрибута **mail** мы [первоначально добавили три значения](#), при выполнении `replace` все они будут удалены и добавлены новые из данного LDIF. После завершения операции у записи **Robert Smith** будет только **два** атрибута **mail** — те, которые были взяты из

данного LDIF.

4. Строка **jpegphoto:< file:///path/to/jpeg/file.jpg** сообщает OpenLDAP, что требуется прочитать содержимое файла, указанного в полном описании пути. В этом случае знак **<** ДОЛЖЕН следовать сразу за :, ПРОБЕЛ ставится после него. Если у Вас нет под рукой подходящего JPEG-файла (он может быть любым), прокомментируйте эти две строки.
5. Строка **delete: description** сообщает OpenLDAP, что требуется удалить атрибут **description** — мы решили, что Mr. Robert Smith не такой уж замечательный парень (swell guy).

Предположим, что мы сохранили приведённый выше LDIF как `modentry.ldif` в директории `/tmp`. Тогда мы будем загружать LDIF-файл с помощью примерно такого вызова `ldapadd` (приведённая ниже команда разбита на две строки только из соображений форматирования HTML и должна записываться в одну строку):

```
ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"
-f /tmp/modentry.ldif -w dirtysecret
```

Замените **ldaphost.example.com** на имя того хоста, на котором располагается Ваш каталог LDAP. Файл не обязательно должен иметь расширение `.ldif`, это просто соглашение, которое мы решили использовать; его с таким же успехом можно было бы назвать `modentry.file` или вообще как угодно.



5.1.8 Напоследок просто подурчимся

Теперь можно поэкспериментировать с базовой структурой LDAP. Мы советуем Вам воспользоваться этой возможностью, чтобы набраться опыта и приобрести уверенность в выполнении операций LDAP перед тем, как двигаться дальше.

1. С помощью LDAP-браузера или LDIF-файлов добавьте дополнительные записи или атрибуты к существующим записям. Для записи в каталог Вы должны подключаться от имени **cn=jimbob,dc=example,dc=com** (`rootdn` или администратор) и использовать ассоциированный с ним `rootpw`.
2. Используйте `ldapsearch` или Ваш LDAP-браузер для поиска по различным критериям.
3. Наконец, либо в семействе браузеров Mozilla (например, Firefox или K-meleon), либо MS Explorer (5+) попробуйте напечатать в адресной строке и перейти по такому LDAP URL:

```
ldap://ldaphost.example.com/ou=people,dc=example,dc=com??one?(objectclass=*)
```

Замените **ldaphost.example.com** на имя того хоста, на котором располагается Ваш каталог LDAP. Также замените все ссылки на **example** и **com**, чтобы они соответствовали Вашей реализации.



5.2 Защита каталога

5.2.1 Политика безопасности

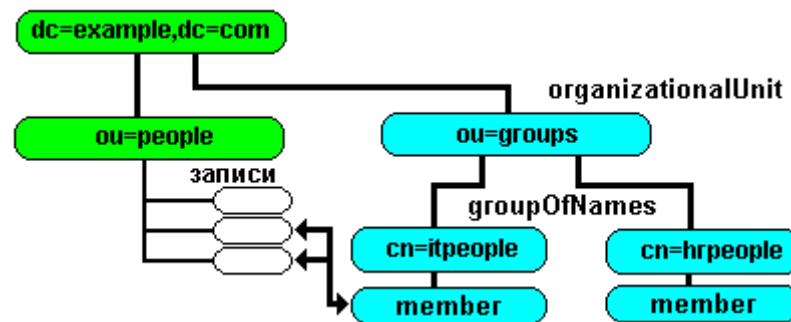
А теперь с помощью директив `access` в `slapd.conf` добавим к нашему каталогу несколько правил безопасности.

Мы собираемся строить политику контроля доступа (Access Control Policy, ACP, известную также как ACL) на основании корпоративной политики безопасности (круто!), которая гласит:

1. Владелец записи каталога имеет право просматривать ВСЕ атрибуты своей записи, включая пароли.
2. Сотрудники отдела Human Resources должны иметь право обновлять ЛЮБУЮ запись, но не должны иметь прав на чтение или запись паролей пользователей.

3. В записях каталога атрибуты **carlicence**, **homepostaddress** и **homephone** не должны быть доступны для чтения кому бы то ни было, за исключением сотрудников отдела Human Resources, а также владельца записи каталога.
4. Все пользователи должны проходить аутентификацию (анонимный доступ не разрешается).
5. Сотрудники отдела IT должны иметь право обновлять или изменять пароль записи во всех записях каталога.

Что бы Вы ни думали о такой политике, мы собираемся разрабатывать контроль доступа, воплощающий её в жизнь. Первое, что мы сделаем — это создадим две группы для **hrpeople** и **itpeople**, что позволит нам назначать **групповые** привилегии. Мы разместим эти группы в ветке **groups**, которая будет находиться в первом уровне иерархии сразу за корневой записью DIT. Приведённая ниже диаграмма показывает нашу новую структуру.



5.2.2 Добавление групп

Следующий LDIF демонстрирует, как мы добавили группы с помощью LDIF.

```

version: 1

# Создаём ветку groups ПЕРВОГО уровня иерархии
dn: ou=groups,dc=example,dc=com
objectclass: organizationalunit
ou: groups
description: generic groups branch

# Создаём запись itpeople в groups
dn: cn=itpeople,ou=groups,dc=example,dc=com
objectclass: groupofnames
cn: itpeople
description: IT security group
member: cn=William Smith,ou=people,dc=example,dc=com

# Создаём запись hrpeople в groups
dn: cn=hrpeople,ou=groups,dc=example,dc=com
objectclass: groupofnames
cn: hrpeople
description: Human Resources group
member: cn=Robert Smith,ou=people,dc=example,dc=com
  
```

Чтобы получить данный пример в виде отдельного текстового файла, используйте [эту ссылку](#) и функцию **сохранить как** Вашего браузера.

Примечания:

1. Для определения группы мы использовали объектный класс **groupOfNames**.
2. **member: dn** определяет члена (членов) группы по их DN.
3. Наблюдательные (или до сих пор не заснувшие) читатели отметили, что в настоящий момент в нашем DIT отсутствует запись для **member: cn=William Smith,ou=people,dc=example,dc=com**. Это вполне допустимо. При добавлении атрибута **member** никаких проверок не производится. В данном случае единственным следствием этого будет лишь то, что в текущий момент в нашем DIT

нет ни одной записи, которая была бы членом группы **itpeople**. Возможно, мы забыли добавить William Smith, возможно, мы добавим эту запись позже. А может быть мы просто ошиблись!

Предположим, что мы сохранили приведённый выше LDIF как `addgroups.ldif` в директории `/tmp`. Тогда мы будем загружать LDIF-файл с помощью примерно такого вызова `ldapadd` (приведённая ниже команда разбита на две строки только из соображений форматирования HTML и должна записываться в одну строку):

```
ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"
-f /tmp/addgroups.ldif -w dirtysecret
```

Замените **ldaphost.example.com** на имя того хоста, на котором располагается Ваш каталог LDAP.



5.2.3 ACL — директивы access файла slapd.conf

Теперь нам нужно перевести нашу политику безопасности в списки контроля доступа (Access Control List, ACL), используя директивы `access to` файла `slapd.conf` OpenLDAP.

[Здесь](#) описывается, как мы реализовали данную политику с подробными примечаниями, а также показано, как это выглядит в формате OLC (`cn=config`).

В листинге ниже приводится наш оригинальный пример `slapd.conf` с изменениями, включающими нашу политику контроля доступа.

```
#
##### ПРИМЕР 2 — КАТАЛОГ С ACL #####
#
# Примечание: inetorgperson использует атрибуты и
#             объектные классы из всех трёх наборов схемы
#
# NB: В RH Linux наборы схемы располагаются в /etc/openldap
#
include          /usr/local/etc/openldap/schema/core.schema
include          /usr/local/etc/openldap/schema/cosine.schema
include          /usr/local/etc/openldap/schema/inetorgperson.schema

# НЕТ ОТСЫЛОК

# НЕ заботимся о файле ARGS, пока не обретём уверенность в своих силах
# stop-скрипту slapd следующая строка необходима для работы:
pidfile /var/run/slapd.pid

# Включаем большое количество вывода в журнал — нам это может понадобится
loglevel         -1

# НЕТ динамических модулей backend

# НЕТ соединений TLS

#####
# Определение базы данных bdb
#
# Замените ниже example и com на подходящее имя домена
#
# Если у Вас нет домена, то, поскольку example.com зарезервирован для
# экспериментов, можете оставить его, либо поменять на my.inc
#####

database bdb
suffix "dc=example, dc=com"
# ACL1
access to attrs=userpassword
      by self write
      by anonymous auth
      by group.exact="cn=itpeople,ou=groups,dc=example,dc=com"
      by * write
      by * none
# ACL2
access to attrs=carlicense,homepostaladdress,homephone
      by self write
      by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com"
      by * write
      by * none
# ACL3
```

```

access to *
    by self write
    by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com"
        write
    by users read
    by * none

# root или администратор
rootdn "cn=jimbob, dc=example, dc=com"
rootpw dirtysecret
# Директория базы данных ДОЛЖНА существовать до того, как Вы запустите slapd
# Не забудьте поменять путь на подходящий Вам
directory /var/db/openldap/example-com

# Устанавливаемые для каталога индексы
# Требуется, если будет использоваться поиск
# Только полное соответствие для unique id
index uid eq
# Стандартный поиск для commonname, givenname и email
index cn,gn,mail eq,sub
# Несколько вариантов для поиска surname
index sn eq,sub
# В индексах выше sub включает в себя subinitial,subany,subfinal
# Оптимизируем поиск по подразделениям
index ou eq
# Если при поиске будет встречаться objectClass, раскомментируйте следующую строку
# index objectClass eq
# Продемонстрировано использование параметра индексирования default
index default eq,sub
# Настройка индексирования пропущена, по умолчанию используется eq,sub
index telephoneNumber

# Другие параметры базы данных
# Дополнительная информация в разделе справочника по slapd.conf
cachesize 10000
checkpoint 128 15

```

Примечание: Атрибуты **carlicense** и **hometelephone** присутствуют не в каждой записи созданного в настоящий момент DIT, а атрибута **homePostalAddress** вообще нет ни в одной записи. В связи с этим можно выделить два момента. Во-первых, данные ACL являются выражением нашей политики безопасности и не связаны с текущим содержимым DIT. Во-вторых, поскольку все эти атрибуты являются частью иерархии объектных классов **inetOrgPerson** (->organizationalPerson->Person), они могут быть в будущем в любой момент добавлены в любую запись, таким образом эти ACL необходимы для определения нашей полной политики контроля доступа с самого начала создания нашего DIT.

Чтобы получить данный пример в виде отдельного текстового файла, используйте [эту ссылку](#) и функцию **сохранить как** Вашего браузера.

Теперь нам нужно остановить и снова запустить OpenLDAP, чтобы он принял новый файл slapd.conf. Это делается с помощью следующих команд:

```

# остановка и запуск OpenLDAP (slapd)
# Linux/Redhat
/etc/rc.d/init.d/ldap restart

# BSD
[bsd] /usr/local/etc/rc.d/slapd.sh stop
# затем
[bsd] /usr/local/etc/rc.d/slapd.sh start

# убеждаемся, что slapd запущен
ps ax | grep slapd

```



5.2.4 Тестирование ACL

Теперь нам нужно проверить вновь установленную нами политику. Для проверки ACL используйте Ваш LDAP-браузер и следующие тесты:

1. Настройте Ваш LDAP-браузер на подсоединение с аутентификацией от имени dn: cn=Robert Smith, ou=people, dc=example, dc=com с паролем (**userpassword**) rJsmithH (регистр символов имеет значение). Поскольку у данной записи есть привилегии **hrpeople**, её владелец сможет

просматривать и изменять все записи, в том числе атрибуты **carlicense**, **homepostaladdress** и **homephone**, но не атрибут **userpassword** (за исключением своей собственной записи).

2. Настройте Ваш LDAP-браузер на подключение с аутентификацией от имени `dn: cn=Sheri Smith, ou=people, dc=example, dc=com` с паролем (**userpassword**) `sSmithH` (регистр символов имеет значение). Поскольку у данной записи есть привилегии **itpeople**, её владелица сможет просматривать и изменять атрибут **userpassword** всех записей, но не сможет просматривать атрибуты **carlicense**, **homepostaladdress** и **homephone** какой-либо записи, за исключением своей собственной.
3. Настройте Ваш LDAP-браузер на подключение с аутентификацией от имени `dn: cn=John Smith, ou=people, dc=example, dc=com` с паролем (**userpassword**) `jSmithH` (регистр символов имеет значение). Поскольку у данной записи нет привилегий, её владелец не сможет просматривать атрибуты **carlicense**, **homepostaladdress**, **homephone** и **userpassword** какой-либо записи, за исключением своей собственной (которую, также, сможет и модифицировать).
4. Настройте Ваш LDAP-браузер на анонимное подключение и убедитесь, что доступ запрещён.
5. Наконец, пройдите аутентификацию от имени настроенного нами **rootdn** или администратора (определён в `slapd.conf` как `cn=jimbob,dc=example,dc=com`, пароль `dirtysecret`) и убедитесь, что в этом случае игнорируются все наши привилегии и можно просматривать и изменять всё что угодно.

Примечание: Во всех приведённых выше тестах Вы должны быть в состоянии с помощью Вашего LDAP-браузера просматривать ветку **groups** и записи **hrpeople** и **itpeople**. Если этого не происходит, то, возможно, Вы установили значение **Base DN** (или **Root DN**) Вашего LDAP-браузера в **ou=people,dc=example,dc=com**. Установите его в **dc=example,dc=com**, и тогда Вы сможете просматривать (но не редактировать) ветку **groups** и её записи.



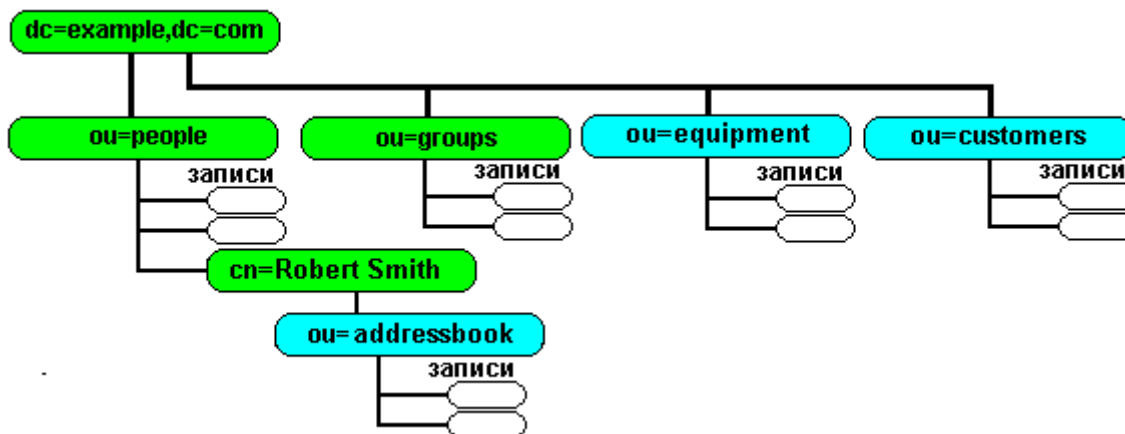
5.3 Расширение иерархии

5.3.1 Расширение требований

Наша текущая реализация оказалась настолько чудесной, что руководители корпорации решили сделать ещё четыре улучшения:

1. Добавить общую адресную книгу клиентов и контактной информации, которая будет доступна для чтения всем прошедшим аутентификацию пользователям. Добавлять же в неё записи смогут только сотрудники отдела продаж (**sales**).
2. Предусмотреть возможность хранения сведений о всех IT-ресурсах компании (компьютерах, принтерах и т.д.) в каталоге. Эти сведения могут обновляться только членами группы **itpeople**, но будут доступны для чтения всем.
3. Добавить в каталог персональные адресные книги в виде дочерней к записи пользователя ветки **addressbook**. Записи в адресной книге могут быть созданы, прочитаны и изменены только самим пользователем — владельцем записи. Но сам пользователь не сможет удалить ветку **addressbook** своей записи (это смогут сделать только члены группы **itpeople**, которым не будет разрешено просматривать записи адресной книги).
4. Группа **hrpeople** будет отвечать за добавление и удаление записей во всех группах.

Пересмотренное DIT выглядит так:



5.3.2 Реализация расширения

После того, как нас обескуражили этими требованиями, нужно подумать о том, как их реализовать. Необходимо сделать следующее:

1. Добавить новую группу **salespeople** в нашу существующую ветку group.
2. Добавить новую ветку **equipment** в наше DIT. Записи в ней будут использовать объектный класс [device](#).
3. Добавить новую ветку **customers** в наше DIT. Записи в ней будут использовать стандартный объектный класс [inetorgperson](#).
4. Добавить новую ветку **addressbook** в каждую запись **people**. Записи в ней будут использовать стандартный объектный класс [inetorgperson](#).

5.3.3 LDIF для реализации расширения

Следующий LDIF показывает, как мы изменили DIT согласно указанным выше требованиям:

```
version: 1

# Создаём ветку customers ПЕРВОГО уровня иерархии
dn: ou=customers,dc=example,dc=com
objectclass: organizationalunit
ou: customers
description: customer address book branch

# Создаём ветку equipment ПЕРВОГО уровня иерархии
dn: ou=equipment,dc=example,dc=com
objectclass: organizationalunit
ou: equipment
description: IT assets branch

# создаём запись в equipment
dn: cn=LP1,ou=equipment,dc=example,dc=com
objectclass: device
cn: LP1
description: Some brand of printer
serialnumber: 1-77-23-15
l: Room 17
owner: cn=John Smith,ou=people,dc=example,dc=com
ou: printers

# создаём запись salespeople в groups
dn: cn=salespeople,ou=groups,dc=example,dc=com
objectclass: groupofnames
cn: salespeople
description: Sales group
```



```
member: cn=John Smith,ou=people,dc=example,dc=com
# создаём запись addressbook для каждого человека
dn: ou=addressbook,cn=John Smith,ou=people,dc=example,dc=com
objectclass: organizationalunit
ou: addressbook
description: Personal Address Book
dn: ou=addressbook,cn=Sheri Smith,ou=people,dc=example,dc=com
objectclass: organizationalunit
ou: addressbook
description: Personal Address Book
dn: ou=addressbook,cn=Robert Smith,ou=people,dc=example,dc=com
objectclass: organizationalunit
ou: addressbook
description: Personal Address Book
```

Чтобы получить данный пример в виде отдельного текстового файла, используйте [эту ссылку](#) и функцию **сохранить как** Вашего браузера.

Примечание:

1. Мы создали одну запись в **equipment**.

Предположим, что мы сохранили приведённый выше LDIF как enhance.ldif в директории /tmp. Тогда мы будем загружать LDIF-файл с помощью примерно такого вызова `ldapadd` (приведённая ниже команда разбита на две строки только из соображений форматирования HTML и должна записываться в одну строку):

```
ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"
-f /tmp/enhance.ldif -w dirtysecret
```

Замените **ldaphost.example.com** на имя того хоста, на котором располагается Ваш каталог LDAP.



5.3.4 ACL — директивы access файла slapd.conf

Теперь нам нужно перевести наши обновления политики безопасности в списки контроля доступа (Access Control List, ACL), используя директивы [access to](#) файла slapd.conf OpenLDAP.

Управление доступом к персональной **адресной книге** — очень сложное обновление, которое мы описали в упомогаемых подробностях [здесь](#). В данном примере используются те же списки ACL, что описаны нами на предыдущем шаге, за исключением того, что мы добавили ACL8A для контроля доступа к разделу equipment, разобраться в котором несложно после чтения комментариев к предыдущим ACL.

Примечание: Значимые изменения были внесены в этот файл после повторного запуска этих тестов на OpenLDAP версии 2.4.16+. Обратите внимание.

В листинге ниже приводится наш оригинальный пример slapd.conf с изменениями, включающими нашу политику контроля доступа.

```
#
##### ПРИМЕР 2 — КАТАЛОГ С УЛУЧШЕННЫМИ ACL #####
#
# Примечание: inetorgperson использует атрибуты и
#             объектные классы из всех трёх наборов схемы
#             объектный класс device определён в core.schema
#
# NB: В RH Linux наборы схемы располагаются в /etc/openldap
#
include          /usr/local/etc/openldap/schema/core.schema
include          /usr/local/etc/openldap/schema/cosine.schema
include          /usr/local/etc/openldap/schema/inetorgperson.schema

# НЕТ ОТСЫЛОК

# НЕ заботимся о файле ARGS, пока не обретём уверенность в своих силах
# stop-скрипту slapd следующая строка необходима для работы:
```

```

pidfile /var/run/slapd.pid

# Включаем большое количество вывода в журнал — нам это может понадобиться
loglevel -1

# НЕТ динамических модулей backend

# НЕТ соединений TLS

#####
# Определение базы данных bdb
#
# Замените ниже example и com на подходящее имя домена
#
# Если у Вас нет домена, то, поскольку example.com зарезервирован для
# экспериментов, можете оставить его, либо поменять на my.inc
#####

database bdb
suffix "dc=example, dc=com"
# Примечания к ACL
# Эти дополнительные примечания относятся к OpenLDAP версии 2.4:
# 1. В настоящий момент используется ключевое слово attrs вместо attr
# (это позволяет уменьшить количество предупреждающих сообщений).
# 2. При использовании регулярных выражений нужно удалить модификатор expand,
# поскольку OpenLDAP 2.4 отклоняет некоторые (но не все) ACL,
# содержащие этот модификатор.
# 3. Проверки, производимые OpenLDAP 2.4, гораздо более жесткие,
# они отклоняют ACL 8, так как в нём содержатся атрибуты,
# которые будут добавлены позже.
# 4. Если в условии бу используется ключевое слово exact и
# замена с применением регулярного выражения,
# то должен использоваться модификатор expand.
# ACL1
access to attrs=userpassword
    by self write
    by anonymous auth
    by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
    by * none

# ACL2
# разрешаем читать addressbook владельцу и itpeople; остальные не могут её просматривать
access to dn.regex="^ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
    attrs=entry
    by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" read
    by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
    by users none

# ACL3
# разрешаем itgroup создавать addressbook, но не просматривать записи
access to dn.regex="cn=([^\,]+),ou=people,dc=example,dc=com$"
    attrs=children
    by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
    by users none break

# ACL4
# разрешаем создавать записи в своей собственной addressbook;
# остальные не могут получить к ней доступ, требуются права на запись
# в атрибут ENTRY (ACL5 или ACL6A) и дочерние (CHILDREN) записи данной записи (ACL4)
access to dn.regex="ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
    attrs=children
    by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write
    by users none

# ACL5 — требуется только до версии 2.2
# разрешаем создание записей в своей addressbook;
# остальные не могут получить к ней доступ, требуются права на запись
# в атрибут ENTRY (ACL5 или ACL6A) и дочерние (CHILDREN) записи данной записи (ACL4)
#access to dn.regex="ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
#   attrs=entry
#   by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write
#   by users none

# ACL6 — требуется только до версии 2.2
# разрешаем создание записей в своей addressbook;
# остальные не могут получить к ней доступ
#access to dn.regex="ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
#   filter=(objectclass=inetorgperson)
#   by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write
#   by users none

# ACL6A — в версиях 2.2+ сразу два ACL, — ACL5 и ACL6, — заменяются данным ACL
access to dn.regex="ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
    attrs=entry,@inetorgperson
    by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write
    by users none

```

```

# ACL7
# разрешаем sales создание записей в customers
# пользователи, прошедшие аутентификацию получают доступ только на чтение
access to dn.one="ou=customers,dc=example,dc=com"
    attrs=children
    by group.exact="cn=salespeople,ou=groups,dc=example,dc=com" write
    by users read

# ACL8
access to attrs=carlicense,homepostaladdress,homephone
    by self write
    by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write
    by * none

# ACL8A — контроль доступа к equipment
access to dn.one="ou=equipment,dc=example,dc=com"
    by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
    by users read
    by * none

# ACL9
access to *
    by self write
    by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write
    by users read
    by * none

# root или администратор
rootdn "cn=jimbob, dc=example, dc=com"
rootpw dirtysecret
# Директория базы данных ДОЛЖНА существовать до того, как Вы запустите slapd
# Не забудьте поменять путь на подходящий Вам
directory /var/db/openldap/example-com

# Устанавливаемые для каталога индексы
# Только полное соответствие для unique id
index uid eq
# Стандартный поиск для commonname, givenname и email
index cn,gn,mail eq,sub
# Несколько вариантов для поиска surname
index sn eq,sub
# В индексах выше sub включает в себя subintial,subany,subfinal
# Оптимизируем поиск по подразделениям
index ou eq
# Если при поиске будет встречаться objectClass, раскомментируйте следующую строку
# index objectClass eq
# Продемонстрировано использование параметра индексирования default
index default eq,sub
# Настройка индексирования пропущена, по умолчанию используется eq,sub
index telephonenumber

# Другие параметры базы данных
# Дополнительная информация в разделе справочника по slapd.conf
cachesize 10000
checkpoint 128 15

```

Чтобы получить данный пример в виде отдельного текстового файла, используйте [эту ссылку](#) и функцию **сохранить как** Вашего браузера.

Теперь нам нужно остановить и снова запустить OpenLDAP, чтобы он принял новый файл slapd.conf. Это делается с помощью следующих команд:

```

# Остановка и запуск OpenLDAP (slapd)
# Linux/Redhat
/etc/rc.d/init.d/ldap restart

# BSD
[bsd] /usr/local/etc/openldap/slapd.sh stop
# затем
[bsd] /usr/local/etc/openldap/slapd.sh start

# убеждаемся, что slapd запущен
ps ax | grep slapd

```



5.3.5 Тестирование ACL

Теперь нам нужно проверить вновь установленную нами политику. Для проверки ACL используйте Ваш LDAP-браузер и следующие тесты:

1. Настройте Ваш LDAP-браузер на подключение с аутентификацией от имени dn: cn=Robert Smith, ou=people, dc=example, dc=com с паролем (**userpassword**) rJsmithH (регистр символов имеет значение). Поскольку у данной записи есть привилегии **hrpeople**, её владелец сможет просматривать и изменять все записи, в том числе атрибуты **carlicense**, **homepostaladdress** и **homephone**, но не атрибут **userpassword** (за исключением своей собственной записи), а также сможет читать записи **customers** и **equipment**, но не писать в них. Robert Smith также может просматривать и добавлять записи в свою собственную подзапись адресной книги (**addressbook**), но может только догадываться, есть ли у кого-нибудь еще адресная книга (то есть не может увидеть запись **addressbook** у кого-либо другого).
2. Настройте Ваш LDAP-браузер на подключение с аутентификацией от имени dn: cn=Sheri Smith, ou=people, dc=example, dc=com с паролем (**userpassword**) sSmithH (регистр символов имеет значение). Поскольку у данной записи есть привилегии **itpeople**, её владелица сможет просматривать и изменять атрибут **userpassword** всех записей, но не сможет просматривать атрибуты **carlicense**, **homepostaladdress** и **homephone** какой-либо записи, за исключением своей собственной. Она сможет читать записи в ветке **customers**, но не сможет писать туда. Она сможет читать и записывать записи в ветку **equipment**. Sheri Smith также может просматривать и добавлять записи в свою собственную подзапись адресной книги (**addressbook**), а также видеть подзапись **addressbook** (но не дочерние ей подзаписи) у любого человека. Она может также удалить подзапись **addressbook** у любой записи (и если Вы хотите это испытать, не переживайте, Вы также можете добавить её назад!)
3. Настройте Ваш LDAP-браузер на подключение с аутентификацией от имени dn: cn=John Smith, ou=people, dc=example, dc=com с паролем (**userpassword**) jSmithH (регистр символов имеет значение). Поскольку у данной записи нет привилегий, её владелец не сможет просматривать атрибуты **carlicense**, **homepostaladdress**, **homephone** и **userpassword** какой-либо записи, за исключением своей собственной (которую, также, сможет и модифицировать). Поскольку данная запись является членом группы **salespeople**, её владелец сможет читать и записывать записи в ветке **customers**, а записи в ветке **equipment** сможет только читать. John Smith также может просматривать и добавлять записи в свою собственную подзапись адресной книги (**addressbook**), но может только догадываться, есть ли у кого-нибудь еще адресная книга (то есть не может увидеть запись **addressbook** у кого-либо другого).
4. Поэкспериментируйте с добавлением записей в разные адресные книги, — общую и частные, — либо с помощью LDIF, либо с помощью Вашего LDAP-браузера.
5. Настройте Ваш LDAP-браузер на анонимное подключение и убедитесь, что доступ запрещён.
6. Пройдите аутентификацию от имени настроенного нами **rootdn** или **администратора** (определён в slapd.conf как cn=jimbob,dc=example,dc=com, пароль dirtysecret) и убедитесь, что в этом случае игнорируются все наши привилегии и можно просматривать и изменять всё что угодно.
7. На данном этапе может быть полезно экспортировать текущее состояние каталога в LDIF с помощью функции экспорта Вашего LDAP-браузера, либо [slapcat](#).

Примечание: Во всех приведённых выше тестах Вы должны быть в состоянии с помощью Вашего LDAP-браузера просматривать ветки **customers**, **equipment** и **groups**, а также записи в них. Если этого не происходит, то, вероятно, Вы установили значение Base DN (или **Root DN**) Вашего LDAP-браузера в **ou=people,dc=example,dc=com**. Установите его в **dc=example,dc=com** и тогда Вы сможете просматривать все эти записи.



5.4 Создание и добавление объектов

В этом разделе мы создадим несколько **атрибутов**, **объектный класс**, а также **набор схемы**, чтобы упаковать всё это на будущее.



5.4.1 Требования

Видя ещё более успешное внедрение LDAP, руководители нашей корпорации решили добавить в текущую реализацию нашего каталога следующие возможности:

1. Атрибут **dohicky**. Этот атрибут будет содержать логические значения. Его смогут просматривать и обновлять только владелец записи и любой член группы **hrpeople**.
2. Атрибут **ageAtBirth** (обратите внимание на особенно глупую псевдовенгерскую нотацию — Вы видите такое в последний раз). Этот атрибут будет содержать числовые значения. Его смогут просматривать и обновлять только владелец записи и любой член группы **hrpeople**.
3. Атрибут **gobbledegook**. Этот атрибут будет содержать строковые значения и будет публично доступен для просмотра всем пользователям, прошедшим аутентификацию. Его смогут обновлять только владелец записи и любой член группы **hrpeople**. Он должен быть доступен для поиска с помощью поисковых фильтров **<=** и **>=**.
4. В рамках нашего перехода на технологию единого входа (SSO), мы добавим каждому пользователю стандартный объектный класс **posixAccount**. Данная запись будет доступна для просмотра только группе **itpeople**.



5.4.2 Реализация

Мы очень старались найти среди существующих атрибутов объектного класса **inetorgperson** такие, которые можно было бы использовать под наши нужды (попросту, использовать существующие атрибуты с необходимым синтаксисом (SYNTAX), несмотря на то, что их оригинальное "предназначение" разрабатывалось для иных целей), но, увы, ничего не вышло. Итак, нам остаётся только создать свои собственные объекты. Вот что мы хотим, или точнее, что мы должны сделать:

1. Атрибут **dohicky**. Создадим свой собственный, используя OID существующего синтаксиса **boolean**.
2. Атрибут **ageAtBirth**. Создадим свой собственный, используя OID существующего синтаксиса **integer**.
3. Атрибут **gobbledegook**. Создадим свой собственный, используя OID существующего синтаксиса **PrintableString**. А для того, чтобы получить возможность пользоваться поисковыми фильтрами **<=** и **>=**, нужно использовать **правило соответствия ORDERING (caseIgnoreOrderingMatch)**.
4. Для этих трёх атрибутов мы создадим новый **объектный класс** и назовём его **ourObject** (довольно креативное имя). Поскольку в каждой записи у нас уже есть **структурный (STRUCTURAL)** объект (**inetorgperson**), мы будем использовать вспомогательный (AUXILIARY) **объектный класс**.

Если бы нам требовались структурные объекты, нам пришлось бы делать их всех **дочерними** по отношению к нашим текущим записям **inetorgperson**, как мы уже делали на предыдущем шаге с частными **адресными книгами**. Альтернативный путь: мы могли бы сделать **ourObject** структурным и использовать в его определении **sup inetOrgPerson**, добавляя его, таким образом, в конец иерархии объектных классов **Person -> organizationalPerson -> inetOrgPerson** (смотрите **определение объектного класса**).

5. При создании и **объектного класса** и **атрибутов** требуется, чтобы они были **глобально уникальными**. В данном примере используются глобально уникальные OID, которые можно безопасно использовать В ЦЕЛЯХ ТЕСТИРОВАНИЯ ЭТОГО ПРИМЕРА. НЕ ПОДДАВАЙТЕСЬ ИСКУШЕНИЮ ПОВТОРНО ИСПОЛЬЗОВАТЬ ЭТИ ИЛИ ЛЮБЫЕ ДРУГИЕ OID. Получение собственного номера для своего предприятия, позволяющего определять свои собственные **атрибуты** и **объектные классы** — простая и бесплатная процедура, предоставляемая **IANA** (следуйте по ссылке **Online Application for a Private Enterprise Number**). Использование существующих OID — очень скверная штука (VERY BAD THING™).
6. Поскольку наше любимое и уважаемое корпоративное руководство вряд ли остановится на этом, мы решили упаковать эти **фенечки** (атрибуты и объектный класс) в **набор схемы**, который мы назовём **ourco.schema** (опять же, весьма оригинально).
7. Слава Богу, объектный класс **posixAccount** — это вспомогательный (AUXILIARY) класс. Поскольку некоторые атрибуты данного объектного класса используются и в других объектных классах, мы пересмотрим ту драконовскую директиву, которая разрешает доступ к атрибутам **posixAccount**

только для **itpeople**. Теперь мы ограничим доступ только к уникальным атрибутам данного объектного класса, в частности — **homedirectory**, **gidnumber**, **uidnumber**, **loginshell** и **gecos**.

5.4.3 Определения атрибутов

Мы создали следующие определения атрибутов:

dohicky — boolean

```
attributetype ( 1.3.6.1.4.1.6863.2.3.107 NAME 'dohicky'
EQUALITY booleanMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 )
```

OID данного атрибута построен согласно [предполагаемому нами соглашению о нумерации](#). Мы используем OID стандартного синтаксиса типа данных **boolean**. [Правило соответствия](#) — только EQUALITY. Дополнительную информацию по определению атрибутов можно найти [здесь](#).

ageAtBirth — integer

```
attributetype ( 1.3.6.1.4.1.6863.2.3.108 NAME 'ageAtBirth'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )
```

OID данного атрибута построен согласно [предполагаемому нами соглашению о нумерации](#). Мы используем OID стандартного синтаксиса типа данных **integer**. [Правила соответствия](#) — EQUALITY и ORDERING. Дополнительную информацию по определению атрибутов можно найти [здесь](#).

gobbledegook — PrintableString

Порой можно встретить варианты **gobbledigook** или даже **gobbledygook** — не обращайте внимания, у нас самый точный вариант написания!

```
attributetype ( 1.3.6.1.4.1.6863.2.3.109 NAME 'gobbledegook'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
ORDERING caseIgnoreOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{200} )
```

OID данного атрибута построен согласно [предполагаемому нами соглашению о нумерации](#). Для данного атрибута мы используем OID стандартного синтаксиса типа данных **PrintableString** и определяем максимальную длину строки в 200 символов (в фигурных скобках). [Правила соответствия](#) — EQUALITY, SUBSTR и ORDERING. . Дополнительную информацию по определению атрибутов можно найти [здесь](#).

5.4.4 Определение объектного класса и набора схемы

```
objectclass ( 1.3.6.1.4.1.6863.2.4.57 NAME 'ourObject'
DESC 'A very useful object'
SUP top AUXILIARY
MUST ( dohicky $ gobbledegook )
MAY ageAtBirth )
```

OID данного объектного класса построен согласно [предполагаемому нами соглашению о нумерации](#).

SUP top ограничивает иерархию объектных классов с помощью удобного объектного класса **top**.

AUXILIARY позволяет нам использовать данный **объектный класс** совместно с любым структурным **объектным классом**.

MUST (dohicky \$ gobbledegook) определяет, что атрибуты **dohicky** и **gobbledegook** должны

присутствовать в той записи, куда добавляется данный объектный класс. А **MAY ageAtBirth**, напротив, говорит о том, что данный атрибут является необязательным.

Дополнительную информацию по определению объектных классов можно найти [здесь](#).

Наконец, мы добавим все вышеперечисленные элементы в файл и сохраним его как **ourco.schema** в той же директории, где находятся поставляемые с дистрибутивом наборы схемы. Конечно, можно сохранить его где угодно, но когда всё лежит в одном месте — сложнее запутаться.

Немного о белках: Пункт DESC большинства объектных классов и типов атрибутов, как правило, столь же значим и полезен, как и в примере нашего объектного класса. В этой жизни множество раздражающих вещей, и одна из них, — то, что мы называем "комплексом белки", — неестественное желание скрыть суть или предназначение того или иного предмета. Однажды поэта лорда Байрона спросили о значении строки одного из его стихотворений. Он ответил: "Когда я писал это, значение было известно мне и Богу, а теперь — одному только Богу". Всё рано или поздно забывается. Почему бы не добавить несколько слов осмысленного описательного текста? Особенно если разработчики предусматривают какие-нибудь серьёзные ограничения или особое использование своего объекта.

```
# ФАЙЛ НАБОРА СХЕМЫ EXAMPLE.COM
# атрибут принимает значение:
# true = надевает свежие носки по понедельникам
# false = не надевает свежие носки по понедельникам
attributetype ( 1.3.6.1.4.1.6863.2.3.107 NAME 'dohicky'
    EQUALITY booleanMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 )

# если присутствует, должен быть равен 0
attributetype ( 1.3.6.1.4.1.6863.2.3.108 NAME 'ageAtBirth'
    EQUALITY integerMatch
    ORDERING integerOrderingMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

# строка, используемая для описания роста человека в футах/дюймах или метрах
# може быть представлена в виде xfyi, например 5f7i (5 футов 7 дюймов) или x.ym, например
1.95m (1.95 метра)
# значения f, i, m не чувствительны к регистру
attributetype ( 1.3.6.1.4.1.6863.2.3.109 NAME 'gobbledegook'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    ORDERING caseIgnoreOrderingMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{200} )

# объектный класс, используемый в адресной книге для определения необходимой информации
# согласно Федеральному закону 5.7.3 — эти данные определены как конфиденциальные —
требуется ограниченный доступ
objectclass ( 1.3.6.1.4.1.6863.2.4.57 NAME 'ourObject'
    DESC 'A very useful object'
    SUP top AUXILIARY
    MUST ( dohicky $ gobbledegook )
    MAY ageAtBirth )
```



5.4.5 Изменения slapd.conf

Мы должны внести изменения в файл slapd.conf, во-первых, для того, чтобы подправить права доступа, а главное — для того, чтобы сообщить OpenLDAP о нашем новом объектном классе с помощью директивы **include /usr/local/etc/ldap/schema/ourco.schema**, чтобы он не задохнулся при виде атрибута **dohicky**. Вот наш изменённый slapd.conf:

```
#
##### ПРИМЕР 4 — КАТАЛОГ с замороженными объектным классом и атрибутами #####
#
# Примечание: inetorgperson использует атрибуты и
# объектные классы из всех трёх наборов схемы
# объектный класс device определён в core.schema
# объектный класс posixAccount — в nis.schema
# добавляется ourco.schema
#
# NB: В RH Linux наборы схемы располагаются в /etc/ldap
#
include /usr/local/etc/ldap/schema/core.schema
include /usr/local/etc/ldap/schema/cosine.schema
include /usr/local/etc/ldap/schema/inetorgperson.schema
include /usr/local/etc/ldap/schema/nis.schema
```

```

include /usr/local/etc/openldap/schema/ourco.schema
# НЕТ ОТСЫЛОК

# НЕ заботимся о файле ARGS, пока не обретём уверенность в своих силах
# stop-скрипту slapd следующая строка необходима для работы:
pidfile /var/run/slapd.pid

# Включаем большое количество вывода в журнал — нам это может понадобится
loglevel -1

# НЕТ динамических модулей backend

# НЕТ соединений TLS

#####
# Определение базы данных bdb
#
# Замените ниже example и com на подходящее имя домена
#
# Если у Вас нет домена, то, поскольку example.com зарезервирован для
# экспериментов, можете оставить его, либо поменять на my.inc
#####

database bdb
suffix "dc=example, dc=com"
# ACL1
# только владелец и члены itgroup могут просматривать и обновлять
access to attrs=userpassword
  by self write
  by anonymous auth
  by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
  by * none

# ACL1A
# только владелец и члены itgroup могут просматривать и обновлять
access to attrs=homedirectory,uidnumber,gidnumber,loginshell,gecos
  by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
  by * none

# ACL2
# разрешаем читать addressbook владельцу и itpeople; остальные не могут её просматривать
access to dn.regex="^ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
  attrs=entry
  by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" read
  by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
  by users none

# ACL3
# разрешаем itgroup создавать addressbook, но не просматривать записи
access to dn.regex="cn=([^\,]+),ou=people,dc=example,dc=com$"
  attrs=children
  by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
  by users none break

# ACL4
# разрешаем создавать записи в своей собственной addressbook;
# остальные не могут получить к ней доступ, требуются права на запись
# в атрибут ENTRY (ACL5) и дочерние (CHILDREN) записи данной записи (ACL4)
access to dn.regex="ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
  attrs=children
  by dn.regex,expand="cn=$1,ou=people,dc=example,dc=com" write
  by users none

# ACL5
# разрешаем создание записей в своей addressbook;
# остальные не могут получить к ней доступ, требуются права на запись
# в атрибут ENTRY (ACL5) и дочерние (CHILDREN) записи данной записи (ACL4)
access to dn.regex="ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
  attrs=entry
  by dn.regex,expand="cn=$1,ou=people,dc=example,dc=com" write
  by users none

# ACL6
# разрешаем создание записей в своей addressbook;
# остальные не могут получить к ней доступ
access to dn.regex="ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
  filter=(objectclass=inetorgperson)
  by dn.regex,expand="cn=$1,ou=people,dc=example,dc=com" write
  by users none

# В LDAP 2.2+ ACL5 и ACL6 заменяются на:
#access to dn.regex="ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com$"
#  attrs=entry,@inetorgperson
#  by dn.regex,expand="cn=$1,ou=people,dc=example,dc=com" write
#  by users none

# ACL7

```

```

# разрешаем sales создание записей в customers
# пользователи, прошедшие аутентификацию получают доступ только на чтение
access to dn.one="ou=customers,dc=zytrax,dc=com"
    attr=children
    by group.exact="cn=salespeople,ou=groups,dc=example,dc=com" write
    by users read

# ACL8
access to attr=carlicense,homepostaladdress,homephone,dohicky,ageatbirth
    by self write
    by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write
    by * none

# ACL8A — контроль доступа к equipment
access to dn.one="ou=equipment,dc=example,dc=com"
    by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
    by users read
    by * none

# ACL9
access to *
    by self write
    by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write
    by users read
    by * none

# root или администратор
rootdn "cn=jimbo, dc=example, dc=com"
rootpw dirtysecret
# Директория базы данных ДОЛЖНА существовать до того, как Вы запустите slapd
# Не забудьте поменять путь на подходящий Вам
directory /var/db/openldap/example-com

# Устанавливаемые для каталога индексы
# Только полное соответствие для unique id
index uid eq
# Стандартный поиск для commonname, givenname и email
index cn,gn,mail eq,sub
# Несколько вариантов для поиска surname
index sn eq,sub
# В индексах выше sub включает в себя subinitial,subany,subfinal
# Оптимизируем поиск по подразделениям
index ou eq
# Если при поиске будет встречаться objectClass, раскомментируйте следующую строку
# index objectClass eq
# Продемонстрировано использование параметра индексирования default
index default eq,sub
# Настройка индексирования пропущена, по умолчанию используется eq,sub
index telephonenumber

# Другие параметры базы данных
# Дополнительная информация в разделе справочника по slapd.conf
cachesize 10000
checkpoint 128 15

```

Чтобы получить данный пример в виде отдельного текстового файла, используйте [эту ссылку](#) и функцию **сохранить как** Вашего браузера.

Примечания:

1. Мы добавили директивы **include** для подключения наборов схемы **nis.schema** (для **posixAccount**) и **ourco.schema** (для **ourObject** и различных атрибутов).
2. Мы добавили директиву **access** (ACL1A), чтобы ограничения доступа касались только атрибутов **homedirectory**, **uidnumber** и **gidnumber** (к ним доступ получают только члены **itgroup** согласно требованиям нашей политики безопасности). В OpenLDAP версий 2.2+ мы можем использовать синтаксис **attr=@posixaccount** (все атрибуты объектного класса **posixAccount**), но тогда у нас будет незначительный побочный эффект ограничения доступа к **cd**! Не совсем то, что мы хотим.
3. Мы добавили **dohickey** и **ageatbirth** в директиву **access** (ACL8), предоставляя доступ к этим атрибутам только членам группы **hrgroup** и владельцу записи.

Теперь нам нужно остановить и снова запустить OpenLDAP, чтобы он принял новый файл slapd.conf. Это делается с помощью следующих команд:

```

# Остановка и запуск OpenLDAP (slapd)
# Linux/Redhat
/etc/rc.d/init.d/ldap restart

# BSD
[bsd] /usr/local/etc/openldap/slapd.sh stop

```

```
# затем
[bsd] /usr/local/etc/rc.d/slapd.sh start

# убеждаемся, что slapd запущен
ps ax | grep slapd
```

5.4.6 LDIF

С помощью приведённого ниже LDIF в наши записи вносятся изменения. Добавляется объектный класс **posixAccount** и его обязательные атрибуты, а также объектный класс **ourObject** со своими обязательными атрибутами.

Ни один из используемых нами LDAP-браузеров не позволяет добавлять новые вспомогательные (AUXILIARY) объектные классы. Видимо, LDIF — единственный способ сделать это.

```
version: 1

dn: cn=john smith,ou=people,dc=example,dc=com
changetype: modify
objectclass: inetorgperson
objectclass: posixaccount
uidnumber: 200
gidnumber: 203
homedirectory: /var/mail/example.com/jsmith
objectclass: ourObject
dohicky: false
gobbledegook: john
ageatbirth: 0

dn: cn=sheri smith,ou=people,dc=example,dc=com
changetype: modify
objectclass: inetorgperson
objectclass: posixaccount
uidnumber: 201
gidnumber: 203
homedirectory: /var/mail/example.com/ssmith
objectclass: ourObject
dohicky: true
gobbledegook: sheri

dn: cn=robert smith,ou=people,dc=example,dc=com
changetype: modify
objectclass: inetorgperson
objectclass: posixaccount
uidnumber: 202
gidnumber: 203
homedirectory: /var/mail/example.com/rsmith
objectclass: ourObject
dohicky: false
gobbledegook: robert
ageatbirth: 17
```

Примечания:

1. При использовании утилиты **ldapmodify** директива **changetype** не является строго обязательной, поскольку она подразумевается как значение по умолчанию.
2. Строка **objectclass: inetorgperson** необходима, поскольку она сообщает OpenLDAP с каким структурным (STRUCTURAL) объектным классом будут использоваться вспомогательные (AUXILIARY) объектные классы.
3. Обязательные (MUST) атрибуты каждого объектного класса должны присутствовать. Наличие необязательных (MAY) атрибутов — по желанию.

Предположим, что мы сохранили приведённый выше LDIF как `modify.ldif` в директории `/tmp`. Тогда мы будем загружать LDIF-файл с помощью примерно такого вызова **ldapmodify** (приведённая ниже команда разбита на две строки только из соображений форматирования HTML и должна записываться в одну строку):

```
ldapmodify -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"
-f /tmp/modify.ldif -w dirtysecret
```

Замените **ldaphost.example.com** на имя того хоста, на котором располагается Ваш каталог LDAP. Если

мы вызываем команду на том же хосте, на котором работает LDAP-сервер, параметр -H может быть опущен. Параметр -x указывает, что мы не будем использовать SASL-аутентификацию, он требуется только начиная с OpenLDAP версий 2.x. В параметре -D указан DN, от имени которого мы производим подключение, в данном случае **rootdn** — администратору приходится выполнять всю работу. Указывать пароль вслед за параметром -w, как мы это сделали в примере, мягко скажем, небезопасно, Вы можете использовать -W, тогда пароль будет запрошен после вызова команды.



5.4.7 Тестирование изменений

Теперь нужно проверить внесённые нами изменения. Для проверки ACL используйте Ваш LDAP-браузер и следующие тесты:

1. Настройте Ваш LDAP-браузер на подключение с аутентификацией от имени dn: cn=Robert Smith, ou=people, dc=example, dc=com с паролем (**userpassword**) rJsmithH (регистр символов имеет значение). Поскольку у данной записи есть привилегии **hrpeople**, её владелец сможет просматривать и изменять все записи, в том числе атрибуты **carlicense**, **homepostaladdress** и **homephone**, но не атрибут **userpassword** (за исключением своей собственной записи), а также сможет читать записи **customers** и **equipment**, но не писать в них. Robert Smith также может просматривать и добавлять записи в свою собственную подзапись адресной книги (**addressbook**), но может только догадываться, есть ли у кого-нибудь еще адресная книга (то есть не может увидеть запись **addressbook** у кого-либо другого). Он не может видеть атрибуты **homedirectory**, **guidnumber**, **gidnumber**, **loginshell** и **gecos** объектного класса posixAccount в своей собственной и в любой другой записи.
2. Настройте Ваш LDAP-браузер на подключение с аутентификацией от имени dn: cn=Sheri Smith, ou=people, dc=example, dc=com с паролем (**userpassword**) sSmithH (регистр символов имеет значение). Поскольку у данной записи есть привилегии **itpeople**, её владелица сможет просматривать и изменять атрибуты **userpassword**, **homedirectory**, **guidnumber**, **gidnumber**, **loginshell** и **gecos** всех записей, но не сможет просматривать атрибуты **carlicense**, **homepostaladdress** и **homephone** какой-либо записи, за исключением своей собственной. Она сможет читать записи в ветке **customers**, но не сможет писать туда. Она сможет читать и записывать записи в ветку **equipment**. Sheri Smith также может просматривать и добавлять записи в свою собственную подзапись адресной книги (**addressbook**), а также видеть подзапись **addressbook** (но не дочерние ей подзаписи) у любого человека. Она может также удалить подзапись **addressbook** у любой записи (и если Вы хотите это попробовать, не переживайте, Вы также можете добавить её назад!)
3. Настройте Ваш LDAP-браузер на подключение с аутентификацией от имени dn: cn=John Smith, ou=people, dc=example, dc=com с паролем (**userpassword**) jSmithH (регистр символов имеет значение). Поскольку у данной записи нет привилегий, её владелец не сможет просматривать атрибуты **carlicense**, **homepostaladdress**, **homephone** и **userpassword** какой-либо записи, за исключением своей собственной (которую, также, сможет и модифицировать). Поскольку данная запись является членом группы **salespeople**, её владелец сможет читать и записывать записи в ветке **customers**, а записи в ветке **equipment** сможет только читать. John Smith также может просматривать и добавлять записи в свою собственную подзапись адресной книги (**addressbook**), но может только догадываться, есть ли у кого-нибудь еще адресная книга (то есть не может увидеть запись **addressbook** у кого-либо другого). Он не может видеть атрибуты **homedirectory**, **guidnumber**, **gidnumber**, **loginshell** и **gecos** объектного класса posixAccount в своей собственной и в любой другой записи.
4. Настройте Ваш LDAP-браузер на анонимное подключение и убедитесь, что доступ запрещён.
5. Пройдите аутентификацию от имени настроенного нами **rootdn** или **администратора** (определён в slapd.conf как cn=jimbo,dc=example,dc=com, пароль dirtysecret) и убедитесь, что в этом случае игнорируются все наши привилегии и можно просматривать и изменять всё что угодно.
6. На данном этапе может быть полезно экспортировать текущее состояние каталога в LDIF с помощью функции экспорта Вашего LDAP-браузера, либо **slapcat**.

Примечание: Во всех приведённых выше тестах Вы должны быть в состоянии с помощью Вашего LDAP-браузера просматривать ветки **customers**, **equipment** и **groups**, а также записи в них. Если этого не

происходит, то, вероятно, Вы установили значение Base DN (или **Root DN**) Вашего LDAP-браузера в **ou=people,dc=example,dc=com**. Установите его в **dc=example,dc=com**, и тогда Вы сможете просматривать все эти записи.

5.5 Технология единого входа (Single Sign On, SSO)

В этой главе мы собираемся рассмотреть весьма важные для *nix систем и сетей возможности **технологии единого входа (Single Sign On, SSO)**, и их реализацию на основе OpenLDAP. В следующей таблице отражён потенциальный диапазон SSO, а также те сферы, где возможно или даже желательно применение SSO:

Функция	Тип учётной записи	Каким образом	ОС	Примечания
Система	Unix	pam_ldap.so nss_ldap	Linux, FreeBSD 5.x	Доступ к ресурсам сервера путём локального или удалённого входа в систему (logon). В FreeBSD 4.x не поддерживается система проверки пользователей pam/NSS, поэтому использование SSO для выполнения входа в систему невозможно.
Почта	App	courier-native	Любая	Доступ к электронной почте — локальные и виртуальные пользователи
ftp	App	ProFTP mod_ldap	Любая	Доступ к приватным (не анонимным) ftp-сервисам
Сеть	Unix	Samba3	Любая	Доступ к сетевым ресурсам с рабочих станций Windows и OS с помощью PDC на NT или Samba (PDC в стиле NT). Требуется учётная запись Unix.
web	App	Apache mod_ldap	Любая	Доступ к Intranet или приватным web-сервисам, например WEBDAV

Функциональные требования

Мы определим функциональные требования к системам SSO, которые на первый взгляд могут показаться очевидными, но...

Примечание: у консорциума OpenGroup есть [прекрасное введение по этой теме](#), составная часть их спецификации XSSO.

1. Единый пользовательский пароль для получения доступа ко всем функциям, разрешённым для данного пользователя или группы пользователей, в которую он входит. Это не обязательно означает, что под одним и тем же паролем пользователь получит доступ ко всем возможным сервисам, но LDAP-хранилище, в котором могут содержаться один или несколько паролей, будет доступно по единому паролю. С точки зрения пользователя технология единого входа соблюдается, даже если для разных приложений или подсистем фактически применяются разные пароли.
2. Единое с точки зрения администраторов место определения/управления доступом к различным функциям для каждого пользователя или группы пользователей.
3. Возможность создавать, изменять или удалять пользователей, а также назначать (ограничивать) пользователям права на доступ.

Уже совсем скоро (One day real soon now™).



5.6 Отсылки и репликация



Глава 6. Конфигурация LDAP

В этой главе упомянуто подробно описываются все параметры и **атрибуты/директивы**, с помощью которых управляются системы LDAP, охватываемые данным руководством (по крайней мере, так обязательно будет). В частности, речь здесь пойдет об OLC (cn=config) и файле slapd.conf (настройка сервера OpenLDAP), файле [ldap.conf](#) (настройка клиента и, частично, сервера OpenLDAP), а также о [настройке ApacheDS \(server.xml\)](#).

Если Вам нужно что-нибудь простенькое применительно к OpenLDAP, воспользуйтесь [примерами](#).

В версиях 2.3 и 2.4 OpenLDAP были представлены существенные изменения, самое значительное из которых состоит в том, что, хотя slapd.conf всё ещё поддерживается (по состоянию на 2.4), OpenLDAP постепенно движется в сторону On-Line конфигурации (OLC), часто упоминаемой в литературе как конфигурация cn=config или slapd.d. Этот метод позволяет производить большинство изменений настроек без необходимости останавливать и снова запускать LDAP-сервер. Конвертация slapd.conf и использование OLC (cn=config) описывается [здесь](#), и постепенно этот способ конфигурации станет основным при описании настроек OpenLDAP в данном руководстве. Обычно запись в cn=config состоит из атрибутов, имена которых эквивалентны названиям директив slapd.conf с добавлением префикса "olc", например, директива [access](#) при использовании в системах с cn=config становится атрибутом **olcAccess**.

Содержание

- 6.1 Обзор конфигурации
 - 6.1.1 Переход к использованию и использование OLC (cn=config/slapd.d)
 - 6.1.1.1 Обзор OLC (cn=config)
 - 6.1.1.2 Переконвертация slapd.conf в OLC (cn=config)
 - 6.1.1.3 Макет OLC (cn=config)
 - 6.1.1.4 Использование OLC (cn=config) (чтение, модификация)
 - 6.1.1.4.1 Общие замечания по конфигурации OLC (cn=config)
 - 6.1.1.4.2 Добавление/удаление наборов схемы данных при использовании OLC (cn=config)
 - 6.1.1.4.3 Добавление/удаление ACP/ACL при использовании OLC (cn=config)
 - 6.1.1.4.4 Добавление/удаление модулей при использовании OLC (cn=config)
 - 6.1.1.4.5 Добавление/удаление баз данных при использовании OLC (cn=config)
 - 6.2 Список директив (OLC (cn=config) и slapd.conf)
 - 6.3 Директивы глобального раздела (OLC (cn=config) и slapd.conf)
 - 6.3.1 Директивы TLS (OLC (cn=config) и slapd.conf)
 - 6.4 Директивы раздела backend (OLC (cn=config) и slapd.conf)
 - 6.5 Директивы раздела database (OLC (cn=config) и slapd.conf)
 - 6.5.1 Директивы overlay (OLC (cn=config) и slapd.conf)
 - 6.6 Директивы ldap.conf
 - 6.7 Конфигурация ApacheDS

Примечание: Параметры командной строки демона slapd приведены [здесь](#).

6.1 Обзор конфигурации

Параметры конфигурации по своей специфике подразделяются на глобальные (global), относящиеся к конкретному механизму манипуляции данными (backend) и относящиеся к конкретной базе данных или DIT (database). Следующие замечания могут оказаться полезными:

1. В OpenLDAP версии 2.3 представлено существенное, хотя и необязательное (в версии 2.4), изменение метода, по которому директивы конфигурации

могут быть применены к slapd. Начиная с версии 2.3 можно продолжать использовать существующий файл slapd.conf, ЛИБО один раз провести конвертацию этого файла и перейти к использованию конфигурации времени исполнения (on-line configuration, OLC), также называемой конфигурацией zero down-time, cn=config или slapd.d (немудрено и запутаться). Если коротко, обычно запись в cn=config состоит из атрибутов, имена которых эквивалентны названиям директив slapd.conf с добавлением префикса "olc", например, директива [access](#) при использовании в системах с cn=config становится атрибутом olcAccess. В последующих описаниях показаны обе формы: атрибут OLC (cn=config) и директива slapd.conf. Процедура конвертации, дальнейшее использование и эффекты от применения этого метода конфигурации описаны [здесь](#). Макет OLC (cn=config) и описание записей этого DIT даны [здесь](#).

2. Глобальные параметры применяются к LDAP-серверу в целом. Обычно они включают в себя параметры окружения, такие как местонахождение файлов.
3. Директивы формируют строгую иерархию: директивы глобального раздела могут быть переопределены директивами из разделов backend или database, в свою очередь директивы из раздела backend могут быть переопределены директивами из разделов database. Если директива была указана в глобальном разделе и в дальнейшем не подвергалась переопределению, сфера её действия (влияния) распространяется на все нижестоящие разделы (backend и database).
4. Строки, начинающиеся с #, считаются комментариями и игнорируются.
5. Пустые строки игнорируются. Если же строка начинается с пробельного символа, она считается продолжением предыдущей строки. Это может стать причиной неприятностей. Будьте очень осторожны. В целом, slapd.conf очень привередлив по отношению к пробелам и пустотам.
6. Каждое DIT со своими свойствами определяется в разделе database.
7. Для каждого DIT должен присутствовать отдельный раздел database; например, если требуется обслуживать два корневых DN, — dc=example,dc=com и dc=example,dc=net, — то должно быть два раздела database. Может присутствовать любое количество таких разделов. Первому экземпляру database присваивается номер базы данных 1, второму — 2, и так далее.
8. Обычно данные конфигурации находятся в:

```
# OLC (cn=config)
[fc] /etc/openldap/slapd.d
[bsd] /usr/local/etc/openldap/slapd.d

# slapd.conf
[fc] /etc/openldap
[bsd] /usr/local/etc/openldap
```

6.2 Список директив (OLC (cn=config) и slapd.conf)

Если не указано иное, все приведённые ниже директивы могут присутствовать как в записи/разделе глобальных настроек, так и в записях/разделах backend или database.

Глобальные директивы

[access to \(olcAccess\)](#)
[allow \(olcAllows\)](#)
[argsfile \(olcArgsFile\)](#)
[attributetype \(olcAttributeTypes\)](#)
[concurrency \(olcConcurrency\)](#)
[conn_max_pending \(olcConnMaxPending\)](#)
[conn_max_auth \(olcConnMaxPendingAuth\)](#)
[defaultaccess](#)
[defaultsearchbase \(olcDefaultSearchBase\)](#)
[disallow \(olcDisallows\)](#)
[gentlehup \(olcGentleHUP\)](#)
[idletimeout \(olcIdleTimeout\)](#)
[include](#)
[index \(olcDbIndex\)](#)
[logfile \(olcLogFile\)](#)
[loglevel \(olcLogLevel\)](#)
[moduleload \(olcModuleLoad\)](#)
[modulepath \(olcModulePath\)](#)
[objectclass \(olcObjectClasses\)](#)
[password-hash \(olcPasswordHash\)](#)
[pidfile \(olcPidFile\)](#)
[referral \(olcReferral\)](#)
[replicationinterval](#)
[require \(olcRequires\)](#)
[reverse-lookup \(olcReverseLookup\)](#)
[rootDSE \(olcRootDSE\)](#)
[schemadn \(olcSchemaDN\)](#)
[security \(olcSecurity\)](#)
[ServerID \(olcServerID\)](#)
[sizelimit \(olcSizeLimit\)](#)
[sockbuf_max_incoming \(olcSockBufMaxIncoming\)](#)
[sockbuf_max_incoming_auth \(olcSockBufMaxIncomingAuth\)](#)
[threads \(olcThreads\)](#)
[timelimit \(olcTimeLimit\)](#)

Директивы TLS/SSL (глобальный раздел)

[Обзор сервера TLS — что такое сервер TLS](#)

Директивы сервера (поставщика)

[TLSCACertificateFile \(olcTLSCACertificateFile\)](#)
[TLSCertificateFile \(olcTLSCertificateFile\)](#)
[TLSCertificateKeyFile \(olcTLSCertificateKeyFile\)](#)
[TLSCipherSuite \(olcTLSCipherSuite\)](#)
[TLSEphemeralDHPParamFile \(olcTLSDHPParamFile\)](#)
[TLSVerifyClient \(olcTLSVerifyClient\)](#)

Директивы клиента (потребителя)

[Обзор сервера TLS — что такое клиент TLS](#)
[TLS_CACERT](#)

Директивы backend

backend (olcBackend)

Директивы database

access to (olcAccess)
database (olcDatabase)
index
mirrormode (olcMirrorMode)
overlay (olcOverlay)
readonly (olcReadOnly)
replica (olcReplica)
repllogfile (olcReplLogFile)
require (olcRequires)
rootdn (olcRootDN)
rootpw (olcRootPW)
suffix (olcSuffix)
syncrepl (olcSyncrepl)
updatedn (olcUpdateDN)
updateref (olcUpdateref)

6.3 Директивы глобального раздела (OLC (cn=config) и slapd.conf)

Во всех случаях соответствия директив первой показана форма конфигурации OLC (cn=config), а затем (в скобках) форма slapd.conf. Мы сделали это намеренно, чтобы отразить переход к конфигурации в форме OLC (cn=config). В OLC (cn=config) глобальные атрибуты определяются в записи cn=config, основанной на объектном классе **olcGlobal**. Смотрите также **макет OLC и формат записей**.

olcAccess (access to)

Формат:

```
olcAccess: to <what> [ by <who> [<accesslevel>] [<control>] ]+  
access to <what> [ by <who> [<accesslevel>] [<control>] ]+
```

С помощью набора атрибутов **olcAccess** (директив **access to**) создаётся то, что принято называть **списками контроля доступа (Access Control List, ACL)** или **политиками контроля доступа (Access Control Policy, ACP)**.

В атрибуте **olcAccess** (директиве **access to**) предоставляется доступ (указываемый в <accesslevel>) к набору записей и/или атрибутов (указываемому в <what>) по одному или нескольким критериям того, кто производит запрос, либо по его характеристикам (определяются в **by <who>**). Необязательный параметр <control> определяет условие выхода для каждого элемента **by <who>**, при его отсутствии значение по умолчанию — **stop**.

Один или несколько атрибутов **olcAccess** (директив **access to**) могут быть включены либо в запись cn=config (раздел **глобальных** настроек), либо в настройки конкретного DIT (путём добавления в запись olcDatabase={Z}xxx,cn=config или в раздел **database** slapd.conf), либо и туда, и туда. Если атрибут **olcAccess** (директива **access to**) определен в записи (разделе) **глобальных** настроек, то он аддитивно применяется к любым ACL во всех последующих разделах **database** (DIT). Данная особенность может оказаться как полезным сокращением, так и ночным кошмаром.

Атрибуты **olcAccess** (директивы **access to**) обрабатываются по очереди до первого совпадения с условием <what>. Если такое совпадение было обнаружено, процесс продолжается поиском совпадений по всем элементам **by <who>** того атрибута (директивы), у которого найдено совпадение. Если какое-то из этих условий совпало, выполняется необязательная часть <control> (по умолчанию **stop**) и результат контроля доступа будет **успешным (success)**. Если же по окончании проверки атрибута **olcAccess** (директивы

access to) не было обнаружено ни одного совпадения с условиями **by <who>**, то выполняется действие **stop**, заданное в неявном элементе управления **<control>**, и результат контроля доступа будет **неудачным (fail)**. Если условие **<what>** атрибута **olcAccess** (директивы **access to**) не совпало, то управление передаётся следующему ACL и процесс повторяется. Если были проверены все ACL и условия **<what>** ни одного из них не совпали, то выполняется действие **stop**, заданное в неявном элементе управления **<control>**, и результат контроля доступа будет **неудачным (fail)**.

Директивы **access (ACL)** невероятно комплексны. Они позволяют установить очень тонкий контроль над тем, кто, к каким объектам и атрибутам, на каких условиях может получить доступ. Обратная сторона этой комплексности и мощи — то, что очень легко получить ужасающе неверные атрибуты **olcAccess** (директивы **access to**). Вы должны тщательно тестировать директивы ACL на доступ со всеми возможными правами. Для этого существует **slapacl** — автоматизированное средство для тестирования доступа к определенным атрибутам и записям на основе текущих атрибутов **olcAccess** (директив **access to**).

В следующих разделах сначала даётся описание параметров, как самих по себе, так и с некоторыми ограниченными примерами, а затем приводится несколько **рабочих примеров**, иллюстрирующих основные моменты. Возможно, лучший способ понять эту директиву — это, пройдя вскользь по деталям описаний, перейти к примерам, а затем вернуться и перечитать раздел с описаниями для более ясного понимания. Альтернативная стратегия — пройти от начала до конца по **разделу примеров (глава 5)**, в котором поочерёдно наращиваются и используются списки ACL, описанные в **рабочих примерах**.

to <what>

Сущность, к которой применяется директива контроля доступа. В одной директиве может быть несколько записей **what**, разделённых пробельными символами. Условия **what** могут принимать одну из следующих форм:

* Шаблон, означающий совпадение с любой записью.

dn[.type]=pattern Данная форма определяет запись на основании её **DN** (строка, заключённая в кавычки), например, **dn="ou=people,dc=example,dc=com"**. **type** — это необязательный квалификатор, который может принимать одно из следующих значений:

regex (значение по умолчанию для OpenLDAP вплоть до версии 2.2; начиная с 2.2 значением по умолчанию является **base** или **exact**) Регулярное выражение. Если часть его заключена в круглые скобки (), то та подстрока, которая в результате совпала с выражением в скобках (**подсовпадение**), может затем быть использована в последующих условиях **<who>** с помощью формы **\$digit**, где **digit** — цифра от 1 до 9, и **\$1** заменяется на первое подсовпадение, **\$2** — на второе, и так далее. Пример:

```
# Если тестовый dn - "ou=something,cn=my name,dc=example,dc=com"
# то в результате поиска совпадений $1 = 'my name'
# поскольку первая часть регулярного выражения не заключена в круглые скобки ( )
access to dn.regex="ou=[^, ]+,cn=([^\, ]+),dc=example,dc=com"
by dn.exact,expand="cn=$1,dc=example,dc=com"
```

[Учебник по регулярным выражениям.](#)

base **base** используется по умолчанию, если **type** пропущено (начиная с OpenLDAP версии 2.2). Определяет, что в сравнении участвует та запись, на которую указывает **pattern**.

exact **exact** — псевдоним для **base**.

one указывает на все записи непосредственно (на один уровень) ниже **pattern**

subtree указывает на все подзаписи (на всех уровнях) ниже **pattern**, включая и саму запись, определяемую **pattern**.

children указывает на все подзаписи (на всех уровнях) ниже **pattern**, но не включая саму запись, определяемую **pattern**.

Примечание: Значение по умолчанию, подразумеваемое в формате DN, изменилось с выходом OpenLDAP версии 2.2 с **regex** на **base**. Несмотря на то, что параметр **type** является необязательным, его следует всегда указывать, как для исключения двусмысленности, так и для обеспечения обратной и прямой совместимости (в случае, если значение по умолчанию вновь изменится).

attrs=attrlist Единичный атрибут или разделённый запятыми список атрибутов, к которым применяется директива контроля доступа. (Примечание: Ранее OpenLDAP принимал как **attr**, так и **attrs**. Текущие версии (2.4) теперь выдают предупреждения, что **attr** является устаревшим, поэтому в нашей документации и во всех файлах примеров мы заменили данный параметр на **attrs**).

Есть три дополнительных **псевдоатрибута**, которые можно использовать:

entry	область действия ограничивается данной записью
children	разрешается доступ к дочерним объектам идентифицированной записи
@objectclass	В OpenLDAP 2.2+ конструкция @ включает в себя все атрибуты указанного объектного класса; например, attrs=@inetOrgPerson будет включать все атрибуты данного объектного класса и всех его родительских объектных классов (organizationalPerson, person). Если используется форма !, то включаются только атрибуты, которых НЕТ в определениях данного объектного класса (и его родительских объектных классов), то есть attrs=!inetOrgPerson исключает все атрибуты в списках MUST и MAY для данного объектного класса (и его родительских объектных классов).

filter=**ldapfilters** Строковое представление [поискового фильтра](#).

Примеры:

```
# ПРИМЕЧАНИЕ: Это только фрагменты - точки (...)
#           указывают, что в этом месте может быть больше данных
#           точек не должно быть в итоговой директиве

# доступ к определённым атрибутам
access to attrs=userpassword,homephone ...

# OPENLDAP ДО ВЕРСИИ 2.1
# доступ к определённому DN и всем нижестоящим уровням
# type пропущено, таким образом по умолчанию regex
# распространяется на все DN ниже определённого DN (в том числе и сам этот DN)
access to dn="ou=people,dc=example,dc=com" ...

# OPENLDAP НАЧИНАЯ С ВЕРСИИ 2.2
# функциональный эквивалент предыдущего примера начиная с версии 2.2
# этот же формат (не являющийся значением по умолчанию) будет работать
# и с версиями до 2.2
# доступ к определённому DN и всем нижестоящим уровням
access to dn.subtree="ou=people,dc=example,dc=com" ...

# доступ только к одному уровню ниже определённого DN
access to dn.one="ou=people,dc=example,dc=com" ...

# доступ только к одному уровню ниже определённого DN
# и только к атрибуту userpassword
access to dn.one="ou=people,dc=example,dc=com" attrs=userpassword
...
```

by <who>

В выражении контроля доступа может быть несколько условий <who>. Они могут принимать следующие формы:

*	Шаблон, означающий совпадение с кем угодно.
anonymous	Доступ, предоставленный пользователям, не прошедшим проверку подлинности. Может быть использован в сочетании с auth , например: ... by anonymous auth Позволяет OpenLDAP получить доступ к аутентификационному атрибуту в пределах сервера от имени анонимного пользователя исключительно для целей аутентификации этого пользователя. За пределами сервера данный атрибут не разглашается.
users	Предоставляет права всем пользователям, прошедшим аутентификацию.
self	Доступ к записи разрешается только при условии, что пользователь прошёл аутентификацию от имени данной записи с паролем, используемым в этой записи.

dn[.type[,modifier]]=pattern

Доступ предоставляется совпавшему DN. Необязательный параметр **type** принимает те же значения, что и в форме **dn** условия **what**. При использовании же с данным условием (<who>) в **type** также разрешено значение **level{n}**, где n начинается от 0 и определяет единственный уровень в DIT, отсчитываемый от **pattern**. Так, **level{0}** — функциональный эквивалент **base** (или **exact**), **level{1}** — функциональный

эквивалент **one**, а **level{3}** указывает, что будут совпадать только записи на 3-м уровне ниже **pattern**. **pattern** представляет собой строку в кавычках. Ключевое слово **expand** может быть использовано совместно с **type** в качестве **модификатора**, указывающего, что значения в форме **\$<digit>** должны быть заменены на **подсовпадения** из условия <what>. **Примечание:** в версии 2.4 использование модификатора **expand** совместно с **regex** отклоняется, даже если присутствует выражение **подсовпадения**. Пример:

```
access to
dn.regex="^cn=( [^, ]+),ou=People,dc=example,dc=com$"
# Предположим, cn=my entry,ou=People,dc=example,dc=com
# У $1 будет значение 'my entry', которое будет подставлено ниже
by dn.exact,expand="cn=$1,ou=People,dc=example,dc=com" read
```

Замечание (от которого может произойти помрачение рассудка): **подсовпадения** в форме **\$n**, как правило, работают только когда предшествующее условие <what> типа **regex**. Однако, при использовании в условии <what> значений **type** типа **base**, **subtree**, **one** или **children** специальное значение **подсовпадения** **\$0** заменяется на совпавшую в условии <what> строку DN целиком. Хуже того, если в условии <what> **type** будет **subtree**, **one** или **children**, то специальное значение **подсовпадения** **\$1** заменяется на самое правое значение в условии <what>. Примеры:

```
# Две следующие директивы access функционально идентичны
# и разрешают доступ на чтение на всё, находящееся
# ниже уровня example.com
access to dn.children="dc=example,dc=com"
by dn.base,expand="$0" read

access to dn.children="dc=example,dc=com"
by dn.base,expand="$1" read
```

dnattr=attrname

Доступ предоставляется тем пользователям, чьи DN перечислены в атрибуте attrname той записи, к которой они собираются получить доступ.

group[/objectclass[/attrname]][.type]=pattern

Доступ предоставляется членам группы, DN записи которой указан в **pattern**. Необязательные параметры **objectclass** и **attrname** могут использоваться для указания атрибута, на основании которого определяется членство в группе (значение по умолчанию для **objectclass** — **groupOfNames**, для **attrname** — **member**). Необязательный параметр **type** может быть либо **expand** (что позволяет подстановку из регулярного выражения в предыдущих условиях), либо **exact** (по умолчанию). **pattern** представляет собой строку в кавычках.

peername[.type]=pattern
peername[.ip|ipv6|path]=value

Существует две формы директивы **peername**. При использовании вместе с **type**, **pattern** должен указывать тип поиска соответствия в форме **IP=ip[:port]** (для IPv4 и IPv6) или

PATH=/full/path (для имени файла именованного канала). **type** может быть либо **regex** (подразумевается, что будут осуществляться подстановки из регулярных выражений в предыдущих условиях), либо **exact** (псевдоним — **base**). **Примечание:** при типе **exact** (по умолчанию) индикаторы типа поиска соответствия **IP=** или **PATH=** включаются в совпадение.

Во второй форме явный тип канала определяется квалификатором в левой части выражения, а **value** задаётся в форматах:

```
peername.ip=ipv4[%netmask][{port}]
peername.ipv6=ipv6[%netmask][{port}]
peername.path= /path/to/named/pipe/file/name
```

Примечание: Необязательный номер порта заключается в фигурные скобки.

Примеры:

```
# Простой формат IPv4
peername.ip=192.168.2.0

# Формат IPv4 с маской сети
peername.ip=192.168.2.0%255.255.255.0

# IPv4 только от порта 5000
```

```
peername.ip=192.168.2.0{5000}

# Простой формат IPv6
peername.ipv6=2001:db8::1

# IPv6 только от порта 5000
peername.ipv6=2001:db8::1{5000}
```

Хотя поддержка формата IP-префикса (или слеш-нотации) и обсуждается в некоторых документациях, в ходе наших экспериментов (на 2.4.8) выяснилось, что указание **netmask** для IP-адресов в формате IP-префикса не поддерживается. Поскольку форматы адресации IPv6 поддерживают только формат IP-префикса для указания маски подсети, создаётся впечатление, что в настоящее время маски подсетей для IPv6 не поддерживаются.

```
sockname[.type]=pattern
sockurl[.type]=pattern
```

Для определения возможности доступа имя файла именованного канала (для формы **sockname**) или URL источника (для формы **sockurl**) сравнивается с **pattern**. Необязательный параметр **type** может быть либо **regex** (подразумевается, что будут осуществляться подстановки из регулярных выражений в предыдущих условиях), либо **exact** (по умолчанию).

```
domain[.type[,modifier]]=pattern
```

Доменное имя хоста, с которого производится подключение, сравнивается с **pattern** для определения возможности доступа. Необязательный параметр **type** может быть **expand** (подразумевается, что будут осуществляться подстановки из регулярных выражений в предыдущих условиях), **exact** (по умолчанию), либо **subtree**. **subtree** считается совпавшим, когда полностью определённое доменное имя точно соответствует шаблону домена, или одна из конечных частей (с правой стороны) при разделении по точкам полностью определённого доменного имени точно соответствует шаблону домена. Так, конструкции **domain.subtree=example.com** будут соответствовать example.com, ftp.example.com или ldap.example.com, но не www.anexample.com. Доменное имя хоста, с которого производится подключение, определяется путём выполнения обратного разрешения DNS-имени по IP-адресу хоста. В IPv4 нет строгого требования, чтобы для IP адресов устанавливалось обратное разрешение (такое требование есть в IPv6), поэтому подобный контроль доступа на основе доменного имени должен применяться с большой осторожностью. По умолчанию, обратное разрешение DNS-имени (требуемое данной функцией) отключено (смотрите [reverse-lookup](#)). Значение **modifier** может быть только **expand**, таким образом **domain.expand=** функционально эквивалентно **domain.exact,expand=**.

```
set[.style]=pattern
```

описание этой конструкции ещё не готово

```
ssf=n
transport_ssf=n
tls_ssf=n
sasl_ssf=n
```

Такая версия позволяет использовать фактор силы криптографической защиты, ассоциируемой с сессией пользователя, для определения возможности получения доступа. Это применимо только при использовании сессий, защищаемых с помощью [SASL](#) или [TLS/SSL](#). Значение **n** определяет минимальный требуемый уровень защиты информации, выражаемый как минимальное число бит в ключе какого-либо криптографического алгоритма — как правило, в диапазоне от 40 до 256, в зависимости от применяемого алгоритма (стандартные значения — 40, 56, 64, 128, 164 и 256). Так, значение 1 указывает, что может быть применён любой уровень криптографической защиты; значение 128 указывает, что будет пропущена сессия с применением криптографического алгоритма, длина ключа которого не менее 128 бит, а при ключе, скажем, длиной 56 бит сессия будет отклонена. Примеры:

```
# Указывает, что пользователь, запрашивающий доступ,
# ДОЛЖЕН использовать сессию TLS/SSL с минимальной длиной
# ключа шифрования 40 (разрешены шифры ЭКСПОРТНОГО класса)
by tls_ssf=40

# Указывает, что пользователь, запрашивающий доступ,
# ДОЛЖЕН использовать сессию TLS/SSL с минимальной длиной
# ключа шифрования 128 (исключаются шифры ЭКСПОРТНОГО класса
# и многие другие, например DES)
by tls_ssf=128

# Указывает, что пользователь, запрашивающий доступ,
# ДОЛЖЕН использовать SASL с любым шифрованием
# (без конкретных требований)
by sasl_ssf=1

# Указывает, что пользователь, запрашивающий доступ,
# ДОЛЖЕН использовать SASL или TLS/SSL с длиной
# ключа шифрования 64 или более
```

aci=attrname Контроль доступа определяется по значению атрибута attrname той записи, к которой осуществляется доступ. ACI являются экспериментальными; для того, чтобы эта функция работала, её нужно включить во время компиляции.

<accesslevel>

accesslevel определяет уровень доступа или конкретные привилегии доступа, которые будут предоставлены тому, у кого было найдено совпадение с полем who. Может быть в такой форме:

[self]{<level>|<priv>}

Описание параметров:

self Модификатор self позволяет выполнение специальных операций, таких как получение определённого уровня доступа или привилегий, только в случае, когда в операции участвует запись пользователя, от имени которого происходит запрос. Это означает, что пользователь, делающий запрос, произвёл подключение с проверкой подлинности. Примером может послужить право на запись самого себя в атрибут member записи группы, позволяющее кому-либо добавлять/удалять свой собственный DN из списка членов группы, не имея при этом возможности сделать это с другими членами группы.

level level определяет привилегии доступа и может принимать одно из приведённых значений. Каждый уровень доступа включает в себя все нижестоящие уровни; так, уровень search позволяет получать доступ search, compare, auth и disclose.

Может принимать одно из следующих значений:

- manage** Можно управлять объектами, определёнными в условии <what>. Это самый высокий уровень полномочий, аналогичный root в системе *nix или rootDN в OpenLDAP.
- write** Можно осуществлять запись в объекты, определённые в условии <what>.
- read** Можно читать содержимое атрибутов, определённых в условии <what>.
- search** Можно осуществлять поиск по атрибутам, определённым в условии <what>.
- compare** Можно осуществлять сравнение объектов, определённых в условии <what>.
- auth** Означает, что OpenLDAP позволяет осуществить внутренний доступ к атрибутам, определённым в условии <what>, исключительно в целях аутентификации/авторизации, например, в ходе операции подсоединения.
- disclose** Содержимое атрибутов, определённых в условии <what>, может быть раскрыто в сообщениях об ошибках.
- none** Доступ не разрешается.

priv Модель доступа priv опирается на явные настройки привилегий доступа для каждого условия. Знак = сбрасывает все предыдущие дополнительные настройки доступа, и, как следствие, окончательные привилегии доступа будут равны тем, которые определены в самом условии. Знаки + и - добавляют/удаляют привилегии доступа к существующим. Привилегии могут быть m = manage, w = write, r = read, s = search, d = disclose, c = compare и x = authentication. В одном выражении может быть добавлено более одной привилегии.

Формат:

{=|+| -}{w|r|s|c|x}+

<control>

control не является обязательным, и, если присутствует, принимает одно из следующих значений:

stop Значение по умолчанию; подразумевается, когда control отсутствует. В случае совпадения контроль доступа прекращается.

`continue` `continue` позволяет перейти к рассмотрению других условий `<who>` в том же самом условии `<access>`; таким образом можно попытаться осуществить последовательное изменение привилегий. Если больше не найдено ни одного совпадения с условиями `<who>`, будет применено последнее из совпавших условий `<who>`.

`break` `break` позволяет перейти к обработке других условий `<access>`, совпадающих с тем же целевым объектом.

Вот и всё, что касается директивы **access**. Довольно просто, не правда ли?!



Рабочие примеры `olcAccess` (`access to`)

1. [Вводные замечания](#)
2. [Доступ по умолчанию](#)
3. [Предоставление доступа только пользователям, прошедшим проверку подлинности](#)
4. [Использование групп для контроля доступа](#)
5. [Общие и персональные адресные книги](#)

Вводные замечания

Для начала несколько общих замечаний об атрибутах **olcAccess** (директивах **access to**):

1. При подключении от имени `olcRootDN (rootdn)` (**администратора**) того DIT, к которому происходит подключение, подразумевается наличие волшебных прав и все правила ACL игнорируются. **olcRootDN/rootdn** может делать что угодно с чем угодно. Запрещённых приёмов нет.
2. Если не установлено атрибутов **olcAccess** (директив **access to**), по умолчанию OpenLDAP неявно устанавливает:

```
access to *  
by anonymous read  
by * none
```

Пояснение:

1. **access to *** означает, что политика применяется ко всем атрибутам и записям в DIT.
2. **by anonymous read** означает, что кто угодно (не прошедший аутентификацию) может читать любые значения DIT, включая пароли.
3. **by * none** указывает, что ни у какого пользователя (*) нет прав на доступ. OpenLDAP также неявно оканчивает каждую директиву **access** этим правилом (даже если оно уже указано явно) для предотвращения случайных ошибок — всё, что не попадает под действие явно заданных условий, не имеет никаких прав. Если установлена только эта директива `access` (либо вообще никаких директив `access`, что по умолчанию означает установку этой директивы), то запись в DIT сможет осуществить только `rootdn` (**администратор**) со своим `rootpw`.
3. Во всех форматах параметров, содержащих выражения в **стиле dn**, и где **стиль dn** определён как **regex** (регулярное выражение), предоставляемый шаблон **pattern** используется для поиска совпадений с регулярным выражением; таким образом, при использовании в качестве шаблона `dn="dc=example,dc=com"` будет найдено совпадение со всеми записями поддерева, например, `"ou=people,dc=example,dc=com"`, но при этом будет использовано гораздо больше ресурсов CPU, чем при получении тех же результатов с помощью выражения `dn.subtree="dc=example,dc=com"`. Разумнее избегать использования **regex**, если можно применить другой формат, — даже если для этого потребуется более одной директивы.
4. Директивы **access** обрабатываются при каждом доступе к каталогу, начиная с той, которая определена первой — порядок очень важен. Эти директивы являются аддитивными: вторая добавляется к функциональности первой, третья — ко второй и так далее. Проверка ACL **останавливается**, как только найдено совпадение с правилом **by <who>** и доступ был либо **предоставлен**, либо **запрещён**.
5. Стиль составления директив **access**. Формат допускает свободную форму и, для облегчения понимания,

директива может быть записана как:

```
access to <parameters>
      by <parameters>
      by <parameters>
      ...
```

Примечание 1: Каждая новая строка в составе директивы должна начинаться с отступа хотя бы в один пробельный символ.

Примечание 2: Хотя формат записи атрибута **olcAccess** (директивы **access to**) очень гибок, не разрешается помещать комментарии (строки, начинающиеся с #) в каком-либо месте атрибута **olcAccess** (директивы **access to**).

6. Для каталога может быть установлен один или несколько атрибутов **olcAccess** (директив **access**), и каждое из правил **by <who>** может иметь параметр **<control>**. Пример:

```
# Первая директива access
# форма OLC (cn=config)
olcAccess: to <what>
      by <who1> break
      by <who2> read
      ...
# форма slapd.conf
access to <what>
      by <who1> break
      by <who2> read
      ...
# Вторая директива access
access to <what>
      by <who> write
```

Пояснения:

1. В первой директиве **access** при выполнении условия **by <who1> break** процесс обработки немедленно переходит ко второй директиве **access**. **break** означает 'перейти к следующему ACL'.
 2. В первой директиве **access** при выполнении условия **by <who2> read** процесс обработки останавливается (**stop**) с результатом **success** и вторая директива **access** не будет выполнена. **stop** является значением по умолчанию в отсутствие какого-либо явно заданного значения **<control>**. Эту строку можно было бы записать как **by <who2> read stop**, если для Вас это более понятно.
7. Каждый атрибут **olcAccess** (директива **access**) завершается (обычно) неявным условием **by * none stop**. Если не обнаружено совпадений со всеми предыдущими условиями **by <who>**, то это неявное условие прекращает процесс обработки ACL. В большей мере при использовании атрибутов **olcAccess** (директив **access**) в разделе глобальных настроек (хотя и в других случаях тоже), это может привести не к тем результатам, которых мы ожидаем, например:

```
# Раздел глобальных настроек
# Первая директива access
# форма OLC (cn=config)
olcAccess: to <what>
      by <who1> read
      ...
# форма slapd.conf
access to <what>
      by <who1> read
      ...
# Раздел database
# Вторая директива access
access to <what>
      by <who> write
```

В первой директиве **access** при выполнении условия **by <who1>** процесс обработки останавливается (**stop**) и обращение ко второй директиве не произойдёт никогда.

Чтобы обращение ко второй директиве **access** выполнялось для всех обращений, не совпавших с условием **by <who1>**, нужно использовать следующую форму:

```
# Раздел глобальных настроек
# Первая директива access
access to <what>
```

```

        by <who1> read
        by * break
        ...
# Раздел database
# Вторая директива access
access to <what>
    by <who> write

```

Параметр **break** в первой директиве access приведёт к тому, что всем обращениям, не совпавшим с условием **by <who1>**, не будет предоставлено никаких конкретных прав доступа (разрешающих или запрещающих) и обработка будет передана второй директиве access.

Примеры

Предоставление доступа только пользователям, прошедшим проверку подлинности

Приведённые ниже ACL поэтапно представлены в [примерах настройки каталога \(глава 5\)](#), с их помощью постепенно ограничивается доступ к целевой DIT.

Политика безопасности: Мы требуем, чтобы все пользователи проходили аутентификацию; запрещаем доступ к атрибуту, в котором хранится пароль, для всех, за исключением владельца записи; разрешаем только владельцу записывать (вносить изменения) в свою запись; все остальные пользователи, прошедшие аутентификацию, могут читать все записи (за исключением паролей, как было сказано выше). Данный пример подразумевает использование как минимум объектного класса [person](#) (для атрибута **userpassword**):

```

# ACL1
# форма OLC (cn=config)
olcAccess: to attrs=userpassword
    by self write
    by anonymous auth
    by * none
# форма slapd.conf
access to attrs=userpassword
    by self write
    by anonymous auth
    by * none
# ACL2
access to *
    by self write
    by users read
    by * none

```

Пояснения:

1. ACL1

1. **access to attrs=userpassword** в ACL1 означает, что данная политика применяется только к атрибуту **userpassword**.
2. **by self write** в ACL1 предоставляет только владельцу записи (прошедшему аутентификацию с использованием **userpassword** этой записи) права записывать в этот атрибут.
3. **by anonymous auth** в ACL1 предоставляет анонимному пользователю доступ к этому атрибуту только в целях аутентификации (используется внутри OpenLDAP для аутентификации).
4. **by * none** в ACL1 означает, что всё, что не попало под действие предыдущих условий, не разрешается; таким образом, любой пользователь, не являющийся владельцем записи, не может записывать в **userpassword**, и даже пользователи, прошедшие аутентификацию, не могут читать данный атрибут (нет прав на чтение).

2. ACL2

1. **access to *** в ACL2. Поскольку директивы access являются аддитивными, политика данной директивы применяется ко всем атрибутам всех записей, за исключением атрибута **userpassword**, для которого в предыдущей директиве определены собственные правила.
2. **by self write** в ACL2 предоставляет права производить запись в атрибуты, попадающие под действие данной директивы, только владельцу записи. Поскольку в ACL1 владельцу (**self**) предоставлен доступ к атрибуту **userpassword**, он может записывать в любые атрибуты своей

записи.

3. **by users read** в ACL2 предоставляет любому пользователю, прошедшему аутентификацию, права на чтение всех атрибутов, попадающих под действие данной директивы, (то есть всех, за исключением **userpassword**, политика доступа к которому определена в ACL1). Если бы мы хотели предоставить права на чтение всех атрибутов (за исключением **userpassword**) анонимным пользователям, можно было бы использовать условие **by anonymous read**.
4. **by * none** в ACL2. В данном случае это означает, что пользователи, не являющиеся владельцем записи, не могут производить запись в какой бы то ни было атрибут, а пользователи, не прошедшие аутентификацию, не могут читать, производить поиск или выполнять любую другую операцию над какой-либо записью или атрибутом.



Анонимный доступ для пользователей из локальной сети

В данном примере для всех пользователей, подключающихся вне пределов нашей локальной сети, требуется прохождение аутентификации; пользователям, подключающимся из локальной сети, разрешается анонимный доступ на чтение; запрещается доступ к атрибуту, в котором хранится пароль, всем, за исключением владельца записи; только владельцу разрешается записывать (вносить изменения) в свою запись; все остальные пользователи, прошедшие аутентификацию, могут читать все записи (за исключением паролей, как было сказано выше). Данный пример подразумевает использование как минимум объектного класса **person** (для атрибута **userpassword**), а также то, что в нашей локальной сети используются адреса из пространства частных локальных сетей класса C: 192.168.1.0-255 (192.168.1.0/24):

```
# ACL1
# форма OLC (cn=config)
olcAccess: to attrs=userpassword
    by self write
    by anonymous auth
    by * none
# форма slapd.conf
access to attrs=userpassword
    by self write
    by anonymous auth
    by * none
# ACL2
access to *
    by self write
    by users read
    by peername=192.168.1.* read
    by * none
```

Пояснения:

1. ACL1

1. **access to attrs=userpassword** в ACL1 означает, что данная политика применяется только к атрибуту **userpassword**.
2. **by self write** в ACL1 предоставляет только владельцу записи (прошедшему аутентификацию с использованием **userpassword** этой записи) права записывать в этот атрибут.
3. **by anonymous auth** в ACL1 предоставляет анонимному пользователю доступ к этому атрибуту только в целях аутентификации (используется внутри OpenLDAP для аутентификации).
4. **by * none** в ACL1. В данном случае это означает, что любой пользователь, не являющийся владельцем записи, не может записывать в **userpassword**, и даже пользователи, прошедшие аутентификацию, не могут читать данный атрибут (нет прав на чтение).

2. ACL2

1. **access to *** в ACL2. Поскольку директивы **access** являются аддитивными, политика данной директивы применяется ко всем атрибутам всех записей, за исключением атрибута **userpassword**, для которого в предыдущей директиве определены собственные правила.
2. **by self write** в ACL2 предоставляет права производить запись в атрибуты, попадающие под действие данной директивы (то есть все атрибуты), только владельцу записи. Поскольку в ACL1 владельцу (**self**) предоставлен доступ к атрибуту **userpassword**, он может записывать в любые атрибуты своей записи.
3. **by users read** в ACL2 предоставляет любому пользователю, прошедшему аутентификацию, права на чтение всех атрибутов, попадающих под действие данной директивы, (то есть всех, за

исключением **userpassword**, политика доступа к которому определена в ACL1). Если бы мы хотели предоставить права на чтение всех атрибутов (за исключением userpassword) анонимным пользователям, можно было бы использовать условие **by anonymous read**.

4. **by peername=192.168.1.*** в ACL2 предоставляет любому пользователю, подключающемуся из локальной сети (с ip-адреса из диапазона от 192.168.1.0 до 192.168.1.255), права анонимно читать все атрибуты, попадающие под действие данной директивы, (то есть все, за исключением **userpassword**). В данной директиве используется проверка на основе регулярного выражения, её можно записать как **peername.regex=192.168.1.***.
5. **by * none** в ACL2. В данном случае это означает, что пользователи, не являющиеся владельцем записи, не могут записывать в какой бы то ни было атрибут, а пользователи, не прошедшие аутентификацию, не могут читать, производить поиск или выполнять любую другую операцию над какой-либо записью или атрибутом.



Ограничение доступа к частям DIT

Построение политики контроля доступа (Access Control Policy, ACP, известную также как ACL) на основании корпоративной политики безопасности (круто!), которая гласит:

1. Владелец записи каталога имеет право просматривать ВСЕ атрибуты своей записи, включая пароли.
2. Сотрудники отдела Human Resources должны иметь право обновлять ЛЮБУЮ запись, но не должны иметь прав на чтение или запись паролей пользователей.
3. В записях каталога атрибуты **carlicense**, **homepostaladdress** и **homephone** не должны быть доступны для чтения кому бы то ни было, за исключением сотрудников отдела Human Resources, а также владельца записи каталога.
4. Все пользователи должны проходить аутентификацию (анонимный доступ не разрешается).
5. Сотрудники отдела IT должны иметь право обновлять или изменять атрибут, в котором хранится пароль, во всех записях каталога.

Данный пример подразумевает использование как минимум объектного класса **inetorgperson** (для **carlicense** и других атрибутов), а также то, что существует две **группы пользователей**, называемые **hrpeople** и **itpeople**:

```
# ACL1
# форма OLC (cn=config)
olcAccess: to attrs=userpassword
    by self write
    by anonymous auth
    by group.exact="cn=itpeople,ou=groups,dc=example,dc=com"
        write
    by * none
# форма slapd.conf
access to attrs=userpassword
    by self write
    by anonymous auth
    by group.exact="cn=itpeople,ou=groups,dc=example,dc=com"
        write
    by * none
# ACL2
access to attrs=carlicense,homepostaladdress,homephone
    by self write
    by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com"
        write
    by * none
# ACL3
access to *
    by self write
    by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com"
        write
    by users read
    by * none
```

Пояснения:

1. ACL1

1. **access to attrs=userpassword** в ACL1 означает, что данная политика применяется только к атрибуту **userpassword**.
2. **by self write** в ACL1 предоставляет владельцу записи (прошедшему аутентификацию с использованием **userpassword** этой записи) права производить запись в этот атрибут.
3. **by anonymous auth** в ACL1 предоставляет анонимному пользователю доступ к этому атрибуту только в целях аутентификации (используется внутри OpenLDAP для аутентификации и остаётся невидимым извне).
4. **by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write** в ACL1 предоставляет права на запись в данный атрибут любому члену группы **itpeople**. Тип **exact** указывается для повышения производительности: таким образом исключается использование типа **regex**, который является значением по умолчанию, если никакой конкретный тип не указан.
5. **by * none** в ACL1. В данном случае это означает, что любой пользователь, не являющийся владельцем записи и не состоящий в группе **itpeople**, не может записывать в **userpassword**, и даже пользователи, прошедшие аутентификацию, не могут читать данный атрибут (нет прав на чтение).

2. ACL2

1. **access to attrs=carlicense,homepostaladdress,homephone** в ACL2 означает, что данная политика применяется только к этим атрибутам.
2. **by self write** в ACL2 предоставляет владельцу записи (прошедшему аутентификацию с использованием **userpassword** этой записи) права записывать в эти атрибуты.
3. **by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write** в ACL2 предоставляет права на запись в данные атрибуты любому члену группы **hrpeople**.
4. **by * none** в ACL2. В данном случае это означает, что любой пользователь, не являющийся владельцем записи и не состоящий в группе **hrpeople**, не может записывать в эти атрибуты, и даже пользователи, прошедшие аутентификацию, не могут читать данные атрибуты (нет прав на чтение).

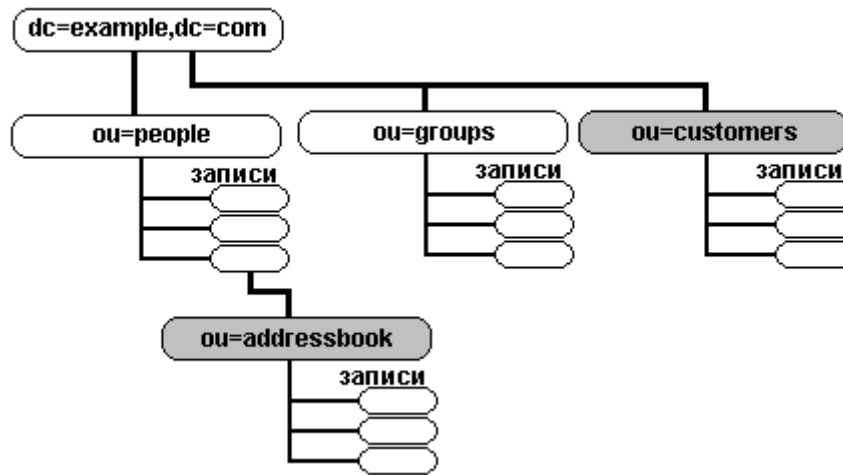
3. ACL3

1. **access to *** в ACL3. Поскольку директивы **access** являются аддитивными, политика данной директивы применяется ко всем атрибутам всех записей, за исключением атрибутов **userpassword**, **carlicense**, **homepostaladdress** и **homephone**, для которых в предыдущих директивах определены собственные правила.
2. **by self write** в ACL3 предоставляет владельцу записи права производить запись в атрибуты, попадающие под действие данной директивы. Поскольку в ACL1 и ACL2 владельцу (**self**) предоставлен доступ к другим атрибутам, он может производить запись в любые атрибуты своей записи.
3. **by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write** в ACL3 предоставляет любому члену группы **hrpeople** права на запись во все атрибуты, определённые в данном ACL, а также, согласно предыдущему ACL2, в атрибуты **carlicense**, **homepostaladdress** и **homephone**, но не даёт этой группе прав на чтение и запись в атрибут **userpassword** (это запрещено в ACL1).
4. **by users read** в ACL3 предоставляет любому пользователю, прошедшему аутентификацию, права на чтение всех атрибутов, попадающих под действие данной директивы, (то есть всех, за исключением тех, которые определены в ACL1 и ACL2). Если бы мы хотели предоставить права на чтение всех атрибутов (за исключением атрибутов, определённых в ACL1 и ACL2) анонимным пользователям, можно было бы использовать условие **by anonymous read**.
5. **by * none** в ACL3. В данном случае это означает, что пользователи, не являющиеся владельцем записи и не состоящие в группе **hrpeople**, не могут производить запись в какой бы то ни было атрибут, а пользователи, не прошедшие аутентификацию, не могут читать, производить поиск или выполнять любую другую операцию над какой-либо записью или атрибутом.



Общая и персональные адресные книги

В данном примере создаются общая и персональные адресные книги, как показано на рисунке:



Будет введена следующая политика:

1. Для доступа к каталогу все пользователи должны пройти аутентификацию.
2. Все пользователи, прошедшие аутентификацию, могут просматривать общую адресную книгу (в ветке **customers**).
3. Только сотрудники отдела продаж (члены группы **salespeople**) могут записывать в адресную книгу **customers**.
4. Члены группы **itpeople** могут создавать в каждой записи в ветке **people** дочернюю запись **addressbook**.
5. Владелец персональной адресной книги может читать и записывать в неё — никто больше не может даже просматривать **addressbook** (за исключением членов группы **itpeople**, которые могут создать **addressbook**, но не могут создавать записи внутри неё). Пользователю не разрешено удалять запись **addressbook**.
6. Сотрудники отдела IT должны иметь возможность обновлять или изменять пароли во всех записях каталога.
7. Сотрудники отдела Human resources (группа **hrpeople**) должны иметь возможность обновлять или изменять все атрибуты всех пользовательских записей, кроме атрибута `userpassword`, и не должны иметь возможности читать или изменять персональные адресные книги пользователей (**addressbook**).

Данный пример подразумевает использование как минимум объектного класса **inetorgperson** (для **carlicense** и других атрибутов), а также то, что существует две **группы пользователей**, называемые **hrpeople** и **itpeople**. Записи в адресных книгах обоих типов **addressbook** и **customers** используют объектный класс **inetorgperson**:

```

# Примечания к ACL
# Эти дополнительные примечания относятся к OpenLDAP версии 2.4:
# 1. В настоящий момент используется ключевое слово attrs вместо attr
#    (это позволяет уменьшить количество предупреждающих сообщений).
# 2. При использовании регулярных выражений нужно удалить модификатор expand,
#    поскольку OpenLDAP 2.4 отклоняет некоторые (но не все) ACL,
#    содержащие этот модификатор.
# 3. Проверки, производимые OpenLDAP 2.4, гораздо более жесткие,
#    они отклоняют ACL 8, так как в нём содержатся атрибуты,
#    которые будут добавлены позже.
# 4. Если в условии бу используется ключевое слово exact и
#    замена с применением регулярного выражения,
#    то должен использоваться модификатор expand.
# ACL1
# форма OLC (cn=config)
olcAccess: to attrs=userpassword
  by self write
  by anonymous auth
  by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
  by * none
# форма slapd.conf
access to attrs=userpassword
  by self write
  by anonymous auth

```

```

by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
by * none

# ACL2
# разрешаем читать addressbook владельцу и itpeople; остальные не могут её просматривать
access to dn.regex="^ou=addressbook,cn=([^\, ]+),ou=people,dc=example,dc=com$"
  attrs=entry
  by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" read
  by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
  by users none

# ACL3
# разрешаем itgroup создавать addressbook, но не просматривать записи
access to dn.regex="cn=([^\, ]+),ou=people,dc=example,dc=com$"
  attrs=children
  by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write
  by users none break

# ACL4
# разрешаем создавать записи в своей собственной addressbook;
# остальные не могут получить к ней доступ, требуются права на запись
# в атрибут ENTRY (ACL5 или ACL6A) и дочерние (CHILDREN) записи данной записи (ACL4)
access to dn.regex="ou=addressbook,cn=([^\, ]+),ou=people,dc=example,dc=com$"
  attrs=children
  by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write
  by users none

# ACL5 - требуется только до версии 2.2
# разрешаем создание записей в своей addressbook;
# остальные не могут получить к ней доступ, требуются права на запись
# в атрибут ENTRY (ACL5 или ACL6A) и дочерние (CHILDREN) записи данной записи (ACL4)
#access to dn.regex="ou=addressbook,cn=([^\, ]+),ou=people,dc=example,dc=com$"
#  attrs=entry
#  by dn.regex="cn=$1,ou=people,dc=example,dc=com" write
#  by users none

# ACL6 - требуется только до версии 2.2
# разрешаем создание записей в своей addressbook;
# остальные не могут получить к ней доступ
#access to dn.regex="ou=addressbook,cn=([^\, ]+),ou=people,dc=example,dc=com$"
#  filter=(objectclass=inetorgperson)
#  by dn.regex="cn=$1,ou=people,dc=example,dc=com" write
#  by users none

# ACL6A - в версиях 2.2+ сразу два ACL, - ACL5 и ACL6, - заменяются данным ACL
access to dn.regex="ou=addressbook,cn=([^\, ]+),ou=people,dc=example,dc=com$"
  attrs=entry,@inetorgperson
  by dn.exact,expand="cn=$1,ou=people,dc=example,dc=com" write
  by users none

# ACL7
# разрешаем sales создание записей в customers
# пользователи, прошедшие аутентификацию получают доступ только на чтение
access to dn.one="ou=customers,dc=example,dc=com"
  attrs=children
  by group.exact="cn=salespeople,ou=groups,dc=example,dc=com" write
  by users read

# ACL8
access to attrs=carlicense,homepostaladdress,homephone
  by self write
  by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write
  by * none

# ACL9
access to *
  by self write
  by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write
  by users read
  by * none

```

Пояснения:

1. ACL1

1. **access to attrs=userpassword** в ACL1 означает, что данная политика применяется только к атрибуту **userpassword**.
2. **by self write** в ACL1 предоставляет владельцу записи (прошедшему аутентификацию с использованием **userpassword** этой записи) права производить запись в этот атрибут.
3. **by anonymous auth** в ACL1 предоставляет анонимному пользователю доступ к этому атрибуту только в целях аутентификации (используется внутри OpenLDAP для аутентификации).
4. **by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write** в ACL1 предоставляет

права на запись в данный атрибут любому члену группы **itpeople**. Тип **exact** указывается для повышения производительности: таким образом исключается использование типа **regex**, который является значением по умолчанию, если никакой конкретный тип не указан.

5. **by * none** в ACL1 означает, что любой пользователь, не являющийся владельцем записи и не состоящий в группе **itpeople**, не может записывать в **userpassword**, и даже пользователи, прошедшие аутентификацию, не могут читать данный атрибут (нет прав на чтение).

2. ACL2

1. **access to dn.regex="^ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com\$" в ACL2** использует регулярное выражение, для того, чтобы данное правило применялось к записям, начинающимся с **addressbook** (^ — знак привязки к началу строки). В переменной \$1 будет сохраняться содержимое **cn** пользователя (выражение в скобках = **([^\,]+)**).
2. **attrs=entry** в ACL2 — это псевдоатрибут, указывающий на запись (то есть в данном условии речь идёт о самой **addressbook**, а не о записях внутри **addressbook**).
3. **by dn.exact,expand="cn=\$1,ou=people,dc=example,dc=com" read** в ACL2 предоставляет владельцу записи (прошедшему аутентификацию с использованием **userpassword** этой записи) права на чтение **addressbook**. \$1 выполняет подстановку **cn** пользователя (полученного из приведённого выше **regex**. Эффект от данного ACL — разрешение только самому владельцу записи просматривать персональную адресную книгу **addressbook**.
4. **by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write** в ACL2 предоставляет права записывать в данную запись любому члену группы **itpeople**. Это требуется для создания записи **addressbook**. Для создания записи требуются права на осуществление записи в псевдоатрибут **entry** и в псевдоатрибут **children** родительской записи (смотрите права на **children** в ACL3).
5. **by users none** в ACL2. В данном случае это означает, что пользователи, не являющиеся владельцем записи и не состоящие в группе **itpeople**, не могут читать данную запись (нет прав на чтение).

3. ACL3 — children-часть ACL2

1. **access to dn.regex="cn=([^\,]+),ou=people,dc=example,dc=com\$" в ACL3** использует регулярное выражение для соответствия ЛЮБЫМ записям в ветке **people** (у **regex** НЕТ привязки к началу строки с помощью знака ^). Это используется для предоставления **children**-доступа (доступа к записям-потомкам) к каждой записи в ветке **cn**.
2. **attrs=children** в ACL3 — это псевдоатрибут, указывающий на все записи-потомки записей, расположенных в ветке **people** (то есть, в данном случае, **addressbook**).
3. **by group.exact="cn=itpeople,ou=groups,dc=example,dc=com" write** в ACL3 предоставляет членам группы **itpeople** права записывать в записи-потомки любой записи в ветке **people**. Чтобы разрешить создание записи, требуются права производить запись для псевдоатрибута **entry** самой записи (ACL2) и для псевдоатрибута **children** родительской записи — это как раз и есть **children**-часть данного разрешения. Этот ACL и ACL2 позволяют только членам группы **itpeople** (но НЕ самому пользователю-владельцу) создавать и удалять запись **addressbook**.
4. **by users none break** в ACL3 означает, что пользователи, не являющиеся владельцем записи и не состоящие в группе **itpeople**, не могут читать запись **addressbook** (нет прав на чтение). **break** очень важен, поскольку он позволяет OpenLDAP продолжить тестирование. Без **break** тестирование данного ACL будет оканчиваться неудачей при прохождении OpenLDAP уровня **addressbook** и до перехода к записям в ветке **addressbook**, то есть владелец адресной книги (**cn**), так же, как и все остальные пользователи, получит права доступа **none** — и, следовательно, записать что-либо в запись в ветке **addressbook** не удастся.

4. ACL4 — children-часть ACL5/ACL6/ACL6A

1. **access to dn.regex="ou=addressbook,cn=([^\,]+),ou=people,dc=example,dc=com\$" в ACL4** использует регулярное выражение для соответствия ЛЮБЫМ записям в ветке **addressbook** (у **regex** НЕТ привязки к началу строки с помощью знака ^). Данный ACL предоставляет права производить запись в дочерние (**children**) записи ветки **addressbook** (записи в адресной книге), эти права требуются как составная часть для обеспечения возможности создания записи. Переменная \$1 будет содержать **cn** пользователя (выражение в скобках = **([^\,]+)**).
2. **attrs=children** в ACL4 — это псевдоатрибут, указывающий на все записи-потомки записей, расположенных в ветке **people** (то есть, в данном случае, **addressbook**).
3. **by dn.exact,expand="cn=\$1,ou=people,dc=example,dc=com" write** в ACL4. \$1 заменяется содержимым **cn** пользователя, полученным из **regex** в условии **to**. В этом условии данному **cn**

(владельцу) предоставляются права производить запись в записи-потомки любой записи в ветке **cn**. Чтобы разрешить создание записи, требуются права производить запись для псевдоатрибута **entry** самой записи (ACL5/ACL6 или ACL6A) и для псевдоатрибута **children** родительской записи — это как раз и есть children-часть данного разрешения. Данный ACL и ACL5/ACL6A позволяют только **cn** (владельцу) создавать и удалять записи в ветке **addressbook**.

4. **by users none** в ACL4 означает, что любой пользователь, не являющийся владельцем записи, не может читать **addressbook** (нет прав на чтение).

5. ACL5 — entry-часть ACL4

1. Требуется только в ранних версиях OpenLDAP — до 2.2. Начиная с версии 2.2 используйте ACL6A. ACL5 — это entry-часть, дополняющая ACL4. Она требуется, чтобы разрешить создание новой записи в ветке **addressbook**. Данный ACL полностью совпадает с ACL4, за исключением **attrs=entry**. Данный ACL вместе с ACL4 позволяет только **cn** (владельцу) создавать и удалять записи в **addressbook**. Порядок присутствия children и entry правил не важны.

6. ACL6 — добавление любых атрибутов в addressbook

1. Требуется только в ранних версиях OpenLDAP — до 2.2. Начиная с версии 2.2 используйте ACL6A. У ACL6 та же область действия и синтаксис, что и у ACL4 и ACL5, он разрешает **владельцу** добавлять любые поля из объектного класса **inetorgperson** с помощью конструкции **filter=(objectclass=inetorgperson)**. Это требуется только в версиях OpenLDAP до 2.2. В версиях 2.2+ ACL5 и ACL6 могут быть объединены с использованием синтаксиса **attrs=entry,@inetorgperson**. Данный синтаксис не поддерживается в версиях OpenLDAP до 2.2.

7. ACL6A — добавление любых атрибутов в addressbook

1. ACL6A можно использовать только начиная с OpenLDAP версии 2.2, в более ранних версиях необходимо использовать ACL5 и ACL6.
2. У ACL6A та же область действия и синтаксис, что и у ACL4, он разрешает **владельцу** добавлять любые поля из объектного класса **inetorgperson** с помощью конструкции **attrs=entry,@inetorgperson**. Данный синтаксис не поддерживается в версиях OpenLDAP до 2.2.

8. ACL7

1. **access to dn.one="ou=customers,dc=example,dc=com"** в ACL7 означает, что данная политика применяется к объектам, находящимся на один уровень ниже **customers** (записи в общей адресной книге).
2. **attrs=children** в ACL7 указывает только на дочерние для **customers** записи (записи в общей адресной книге).
3. **by group.exact="cn=salespeople,ou=groups,dc=example,dc=com" write** в ACL7 предоставляет любым членам группы **salespeople** права производить запись в записи в общей адресной книге.
4. **by users read** в ACL7 предоставляет любому пользователю, прошедшему аутентификацию, читать записи из общей адресной книги.

9. ACL8

1. **access to attr=carlicense,homepostaladdress,homephone** в ACL8 означает, что данная политика применяется только к этим атрибутам.
2. **by self write** в ACL8 предоставляет владельцу записи (прошедшему аутентификацию с использованием **userpassword** этой записи) права производить запись в эти атрибуты.
3. **by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write** в ACL8 предоставляет права на запись в данные атрибуты любому члену группы **hrpeople**.
4. **by * none** в ACL8. В данном случае это означает, что любой пользователь, не являющийся владельцем записи и не состоящий в группе **hrpeople**, не может записывать в эти атрибуты, и даже пользователи, прошедшие аутентификацию, не могут читать данные атрибуты (нет прав на чтение).

10. ACL9

1. **access to *** в ACL9. Поскольку директивы **access** являются аддитивными, политика данной директивы применяется ко всем атрибутам всех записей, за исключением атрибутов **userpassword**, **carlicense**, **homepostaladdress** и **homephone**, для которых в предыдущей

директиве определены собственные правила.

2. **by self write** в ACL9 предоставляет владельцу записи права производить запись в атрибуты, попадающие под действие данной директивы (то есть все атрибуты). Поскольку в ACL1 и ACL8 владельцу (**self**) предоставлен доступ к другим атрибутам, он может производить запись в любые атрибуты своей записи.
3. **by group.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write** в ACL9 предоставляет любому члену группы **hrpeople** права на запись во все атрибуты, определённые в данном ACL, а также, согласно предыдущему ACL8, в атрибуты `carlicense`, `homepostaladdress` и `homephone`, но не даёт этой группе прав на чтение и запись в атрибут **userpassword** (это запрещено в ACL1).
4. **by users read** в ACL9 предоставляет любому пользователю, прошедшему аутентификацию, права на чтение всех атрибутов, попадающих под действие данной директивы, (то есть всех, за исключением тех, которые определены в ACL1 и ACL8). Если бы мы хотели предоставить права на чтение всех атрибутов (за исключением атрибутов, определённых в ACL1 и ACL8) анонимным пользователям, можно было бы использовать условие **by anonymous read**.
5. **by * none** в ACL9. В данном случае это означает, что пользователи, не являющиеся владельцем записи и не состоящие в группе **hrpeople**, не могут производить запись в какой бы то ни было атрибут, а пользователи, не прошедшие аутентификацию, не могут читать, производить поиск или выполнять любую другую операцию над какой-либо записью или атрибутом.



olcAllows (allow)

```
allow feature [...]
```

Этот глобальный атрибут (директива) определяет характеристики поведения, которые будут поддерживаться данным сервером. В директиве указывается одно или несколько из перечисленных ниже значений (разделённых пробельными символами, например, пробелом или табуляцией):

<code>bind_anon_cred</code>	Разрешает простые соединения с предоставлением учётных данных (пароля), но без предоставления значения DN. Если данная характеристика не указана, все подобные подсоединения будут отклоняться (LDAP_INVALID_CREDENTIALS = 49, x'31).
<code>bind_anon_dn</code>	Разрешает простые соединения с предоставлением DN, но без предоставления учётных данных — технически это неавторизованное соединение . Если данная характеристика не указана, все подобные подсоединения будут отклоняться (LDAP_UNWILLING_TO_PERFORM = 53, x'35).
<code>bind_v2</code>	Разрешает серверу принимать соединения LDAP версии 2, в противном случае они будут отклоняться.
<code>prozy_authz_anon</code>	Описание будет позже.
<code>update_anon</code>	Позволяет под контролем ACL вносить изменения в DIT при анонимных подключениях. По умолчанию при анонимных подключениях, независимо от каких-либо настроек ACL, нельзя производить запись в DIT, если данная характеристика не включена.

Примеры:

```
# фрагмент slapd.conf
# глобальные настройки
allow bind_v2
...

# форма OLC (cn=config)
olcAllows: bind_v2
```



olcArgsFile (argsfile)

Заставляет OpenLDAP записывать аргументы командной строки, с которыми он был запущен, в указанный

файл. Пример:

```
# форма OLC (cn=config)
olcArgsFile: /var/run/ldap.args

# форма slapd.conf
argsfile /var/run/ldap.args
```

В качестве альтернативы можно использовать такую команду:

```
ps ax |grep slapd
```

В выводе будут указаны аргументы командной строки. Если директива **argsfile** не указывалась, файл с аргументами создаваться не будет.



olcAttributeTypes (attributetype)

В качестве артефакта обработки схемы данных OpenLDAP предоставляет возможность определить один или несколько типов атрибутов с помощью **attributetype** в файле slapd.conf или **olcAttributeTypes** в системах OLC (cn=config). Определение **типа атрибута** подробно описано [здесь](#). Не совсем понятно, что может подвигнуть Вас при использовании slapd.conf добавлять типы атрибутов таким способом — предпочтительнее создать файл пользовательского набора схемы и добавить его с помощью директивы [include](#). Тем не менее, Вы можете сделать и так, если Вас это забавляет. В OLC (cn=config) атрибуты **olcAttributeTypes** и **olcObjectClasses** применяются для загрузки всех наборов схемы данных, используемых конфигурацией OLC (cn=config).



olcConcurrency (concurrency)

Формат:

```
# форма OLC (cn=config)
olcConcurrency: integer

# форма slapd.conf
concurrency integer
```

Атрибут **olcConcurrency** (директива **concurrency**), если таковой определен, представляет собой "подсказку" системе потоков OpenLDAP. По умолчанию эта директива не содержит никакого значения. Примеры:

```
# форма OLC (cn=config)
olcConcurrency: 25

concurrency 25
# говорит о том, что будет использовано 25 потоков
# позволяет осуществлять 25 параллельных операций
```

Не совсем понятно, какова связь между атрибутом **olcConcurrency** (директивой **concurrency**) и [атрибутом olcThreads](#) (директивой [\(threads\)](#)).



olcConnMaxPending (conn_max_pending)

Формат:

```
# форма OLC (cn=config)
olcConnMaxPending: integer

# форма slapd.conf
conn_max_pending integer
```

Атрибут **olcConnMaxPending** (директива **conn_max_pending**) определяет количество запросов в режиме ожидания (стоящих в очереди) в пределах одной анонимной сессии. Если данный лимит превышен, сессия будет закрыта, однако стоящие в настоящий момент в очереди запросы будут обработаны. Значение по умолчанию — 100. Примеры:

```
# форма OLC (cn=config)
olcConnMaxPending: 10

# форма slapd.conf
conn_max_pending 10
# если в пределах одной анонимной сессии в очередь поставлены
# 10 неотработанных запросов, сессия будет завершена
```



olcConnMaxPendingAuth (conn_max_auth)

Формат:

```
# форма OLC (cn=config)
olcConnMaxPendingAuth: integer

# форма slapd.conf
conn_max_auth integer
```

Атрибут **olcConnMaxPendingAuth** (директива **conn_max_auth**) определяет количество запросов в режиме ожидания (стоящих в очереди) в пределах одной сессии, при установке которой пользователь прошёл аутентификацию. Если данный лимит превышен, сессия будет закрыта, однако стоящие в настоящий момент в очереди запросы будут обработаны. Значение по умолчанию — 1000. Примеры:

```
# форма OLC (cn=config)
olcConnMaxPendingAuth: 100

# форма slapd.conf
conn_max_auth 100
# если в пределах одной сессии от клиента, прошедшего аутентификацию,
# в очередь поставлены 100 неотработанных запросов, сессия будет завершена
```



defaultaccess

Начиная с OpenLDAP 2.1 данная директива считается устаревшей — используйте директиву [access to](#). Не поддерживается в OLC (cn=config), кроме как в целях переконвертации файлов slapd.conf.

Директива **defaultaccess** — глобальное значение по умолчанию для контроля доступа, распространяющаяся на все запросы, если Вы не определили директив [access to](#). Формат:

```
defaultaccess { none | compare | search | read | write }
```

Данный список иерархичен ВПРАВО, то есть уровень доступа read позволяет также search и compare, но не write; write позволяет read, search и compare. Значение по умолчанию, в случае если не указано ни директивы **defaultaccess**, ни директив **access to**:

```
defaultaccess read
# разрешено read, search и compare, но НЕ РАЗРЕШЕНО write
```



olcDefaultSearchBase (defaultsearchbase)

Формат:



```
# форма OLC (cn=config)
olcDefaultSearchBase: dn

# форма slapd.conf
defaultsearchbase dn
```

Атрибут **olcDefaultSearchBase** (директива **defaultsearchbase**), если таковой указан, определяет dn, который будет использоваться при поисковых запросах с **поисковым диапазоном** **one** или **sub** и пустым DN. Если эта директива не указана, то по умолчанию подобные поисковые запросы отклоняются с сообщением NoSuchObject. Примеры:

```
# форма OLC (cn=config)
olcDefaultSearchBase: dc=example,dc=com

# форма slapd.conf
defaultsearchbase dc=example,dc=com
# в поисковые запросы с пустым базовым dn
# будет подставлен приведённый выше dn
```

В конфигурации OLC (cn=config) данный атрибут добавляется в запись **olcDatabase={-1}frontend,cn=config**, в отличие от всех остальных глобальных атрибутов, которые добавляются в запись cn=config.



olcDisallows (disallow)

```
# форма OLC (cn=config)
olcDisallows: feature [...]

# форма slapd.conf
disallow feature [...]
```

Этот глобальный атрибут (директива) определяет характеристики поведения, которые не будут разрешены на данном сервере. В директиве указывается одно или несколько из перечисленных ниже значений (разделённых пробельными символами, например, пробелом или табуляцией):

bind_anon	Предотвращает анонимные подключения, то есть подключения без предоставления DN и учётных данных (пароля). В любом случае, даже без указания этой директивы, подключения с предоставлением DN, но без предоставления учётных данных (пароля) будут завершаться неудачей (но могут быть разрешены с помощью директивы allow bind_anon_dn). Точно также подключения без предоставления DN, но с предоставлением учётных данных (пароля) будут завершаться неудачей (но могут быть разрешены с помощью директивы allow bind_anon_cred).
bind_simple	Отключает возможность любых простых подключений — как анонимных, так и с проверкой подлинности. Использование данного ключевого слова означает, что сервер будет принимать только методы аутентификации SASL .
none	Значение по умолчанию.
tls_2_anon	Описание будет позже.
tls_authc	Описание будет позже.

Примеры:

```
# форма OLC (cn=config)
olcDisallows: bind_anon

# форма slapd.conf
disallow bind_anon
# сервер не допускает анонимные подключения
```



olcGentleHUP (gentlehub)

Формат:

```
# форма OLC (cn=config)
olcGentleHUP: TRUE | FALSE

# форма slapd.conf
gentlehub on | off
```

Если атрибут **olcGentleHUP** (директива **gentlehub**) установлен в **TRUE** (или **on**), серверу OpenLDAP разрешено выполнение корректного завершения работы при получении сигнала SIGHUP, например так:

```
kill -HUP PID
```

В этом случае OpenLDAP выполнит следующие действия:

1. прекратит ожидание новых входящих запросов;
2. продолжит обработку стоящих в очереди и текущих запросов;
3. вернёт 'unwilling to perform' (не желаю исполнять) на стоящие в очереди запросы на запись;
4. остановится по завершении всех текущих операций

Эта команда становится более эффективной, если установлены [атрибут olcIdleTimeout](#) (директива [idletimeout](#)) и [атрибут olcTimeLimit](#) (директива [timelimit](#)). OpenLDAP будет, как всегда, немедленно реагировать на сигнал SIGTERM. Значение по умолчанию — **FALSE** (или **off**).

Примеры:

```
# форма OLC (cn=config)
olcGentleHUP: TRUE

# форма slapd.conf
gentlehub on
# разрешена возможность корректного завершения при получении SIGHUP

# форма OLC (cn=config)
olcGentleHUP: FALSE

# форма slapd.conf
gentlehub off
# по умолчанию - выполнять SIGHUP аналогично SIGTERM
```



olcIdleTimeout (idletimeout)

Формат:

```
# форма OLC (cn=config)
olcIdleTimeout: integer

# форма slapd.conf
idletimeout integer
```

Атрибут **olcIdleTimeout** (директива **idletimeout**) определяет время в секундах, по истечении которого простаивающее клиентское соединение будет принудительно закрыто (выполнится принудительная операция unbind).

Если атрибут **olcIdleTimeout** (директива **idletimeout**) не указан или период установлен в 0, данная возможность отключена, то есть простаивающие клиентские соединения будут поддерживаться неограниченное время (сервер не будет выполнять принудительной операции unbind). Примеры:

```
# форма OLC (cn=config)
olcIdleTimeout: 0

# форма slapd.conf
idletimeout 0
# простаивающие клиенты не отключаются
```



```
# форма OLC (cn=config)
olcIdleTimeout: 30

# форма slapd.conf
idletimeout 30
# простаивающие клиенты отключаются через 30 секунд
```

Предостережение: Это значение распространяется на все подключения, обслуживаемые сервером. Если данный сервер является либо потребителем репликации (использует директиву [syncrepl](#) с типом, имеющим значение **refreshAndPersist**), либо поставщиком репликации (использует директиву [overlay syncprov](#) с одним или несколькими потребителями типа **refreshAndPersist**), то весьма вероятно, что установленные ими соединения будут бездействовать в течение длительного периода времени. В таких условиях необходимо соблюдать крайнюю осторожность при использовании атрибута `olcIdleTimeout` (директивы `idletimeout`), поскольку при разрыве соединения тип подключений репликации может смениться на **refreshOnly**, что, возможно, будет нежелательным побочным эффектом.



include

Данная директива по своей природе является избыточной в конфигурациях OLC (`cn=config`) и поддерживается только в целях переконвертации `slapd.conf`. Наборы схемы данных могут быть добавлены в конфигурацию OLC (`cn=config`) с помощью описанного [здесь](#) процесса.

Директива **include** позволяет считывать содержимое любого файла и добавлять это содержимое в качестве директив конфигурации. OpenLDAP не выполняет проверки содержимого при добавлении, поэтому могут возникнуть проблемы со вложенными директивами **include**. Формат директивы:

```
include /path/to/include/file
```

Два наиболее распространённых применения данной директивы:

1. подключения заранее подготовленных (существующих) файлов `.schema` (обычно находятся в `/usr/local/etc/openldap/schema` или `/etc/openldap/schema`).
2. считывание файлов, которые могут содержать конфиденциальную информацию, например пароли; эти файлы могут быть защищены путём ограничения доступа с помощью `chmod`.

Примеры:

```
include /usr/local/etc/openldap/schema/core.schema
# добавляет содержимое файла core.schema, поставляемого с дистрибутивом
include /var/myschemas/myschema.schema
# добавляет содержимое файла myschema.schema
include /var/passwords/userpass
# подключает файл с паролем пользователя, содержащим, к примеру,
# директиву rootpw и имеющим права доступа 0600
```



olcDbIndex (index)

Лучше определять директивы **index** файла `slapd.conf` перед первоначальной загрузкой каталога (с помощью [ldapadd](#)), тогда при первоначальной загрузке соответствующие индексы будут созданы автоматически. При внесении изменений в настройки индексов после первоначальной загрузки требуется переиндексирование (повторное создание индексов) каталога с помощью [slapindex](#) (**предупреждение:** перед этим `slapd` должен быть остановлен).

В OLC (`cn=config`) используется атрибут `olcDbIndex`, который может присутствовать только в записях `olcDatabase={Z}xxx,cn=config`. Кроме того, переиндексирование выполняется в режиме реального времени, то есть любое изменение какого-либо из атрибутов `olcDbIndex` применяется немедленно и сразу происходит переиндексирование (выполнение `slapindex` не требуется). Непонятно, однако, как при использовании OLC (`cn=config`) определить, когда переиндексирование выполнилось — нет каких-либо видимых индикаторов его завершения.

Формат:

```
# форма OLC (cn=config)
olcDbIndex: attrlist | default indices

# форма slapd.conf
index attrlist | default indices

# indices = [pres [,approx] [,eq] [,sub] [,special]]
```

Атрибуты **olcDbIndex** (директивы **index**) определяют, какие индексы будут обслуживаться OpenLDAP. Может быть включено любое количество параметров индексирования. Даже если атрибут не был указан в директивах **index**, его всё равно можно использовать в [поисковых фильтрах](#) — если это будет происходить часто, то, конечно, производительность будет страдать, а если раз в жизни — ничего страшного.

attrlist может быть единичным атрибутом, или списком атрибутов, разделённых запятыми.

Необязательный параметр **default** содержит те индексы, которые должны поддерживаться по умолчанию. Они применяются к атрибутам в последующих директивах **index**, в которых пропущен список индексов. Атрибут **olcDbIndex** (директива **index**) со значением **default** должен быть определен до появления атрибутов **olcDbIndex** (директив **index**) без списка индексов. Если определено несколько атрибутов **olcDbIndex** (директив **index**) со значением **default**, каждый из них будет влиять на атрибуты **olcDbIndex** (директивы **index**), следующие непосредственно за ним, до появления очередного атрибута **olcDbIndex** (директивы **index**) со значением **default**.

Индекс **pres** следует использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'objectclass=person' или 'attribute=mail'.

Индекс **approx** НЕОБХОДИМО использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'sn~=person' (поиск 'созвучных' значений).

Индекс **eq** следует использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'sn=smith', то есть без применения поисковых шаблонов (используется только правило EQUALITY).

Индекс **sub** следует использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'sn=sm*', то есть с применением поисковых шаблонов (используется правило SUBSTR). Данный индекс можно задать более конкретно, указывая **subinitial** (оптимизация под 'sn=*s'), **subany** (оптимизация под 'sn=*n*') или **subfinal** (оптимизация под 'sn=th*'). Для одного атрибута можно указать несколько индексов типа **sub**.

Параметр **special** связан с [подтипами \(subtype\)](#), может быть либо **nolang**, либо **nosubtypes**.

Будьте очень внимательны при выборе того, какие индексы будут поддерживаться каталогом. Лучше делать это на основании **требований приложений**; если их не учитывать, это может серьёзно повлиять на производительность каталога. И наоборот, нет никакого смысла индексировать те атрибуты, поиск по которым никогда не осуществляется. Если в поисковых запросах используются только правила EQUALITY, нет смысла устанавливать индексирование **sub**. Индексы потребляют память (чем больше индексов, тем больше они занимают оперативную память). Кроме того, операции записи и модификации при большом количестве индексов будут выполняться дольше, поскольку требуется время и ресурсы на обновление индексов.

Примеры:

```
# Простое использование значения default
# форма OLC (cn=config)
olcDbIndex: default pres,eq
olcDbIndex: cn,sn,uid

# форма slapd.conf
index default pres,eq
index cn,sn,uid

# Определяем индексы наличия и соответствия
# для атрибутов cn, sn и uid
# две приведённые выше директивы index выполняют
```

```
# абсолютно то же, что и три, приведённые ниже
index cn pres,eq
index sn pres,eq
index uid pres,eq

index cn eq,sub,subinitial
# Создаём индексы для атрибута cn (commonname)
# для поисков EQUALITY, SUBSTR, а также дальнейшая оптимизация
# для поисков типа sc=a*

index sn eq,approx,sub
# Создаём индексы для атрибута sn (surname)
# для поисков EQUALITY и SUBSTR
# Примечание: Индекс approx index - пустая трата времени,
# поскольку для sn не определено правило ORDERING,
# требуемое для поисков approx. Приводится лишь для
# иллюстрации возможности использования

index mail pres,eq,sub
# Создаём индексы для атрибута mail on
# для поисков на наличие, EQUALITY и SUBSTR

index objectclass eq
# Оптимизируем поиски по форме objectclass=person
```

Обзор других настроек производительности можно найти в соответствующей [главе](#).

olcLogFile (logfile)

OpenLDAP производит журналирование через syslogd (используя LOCAL4) во всех случаях (способ настройки syslogd для перенаправления сообщений LDAP в отдельный файл смотрите в описании [атрибута olcLogLevel](#) (директивы [loglevel](#))). В дополнение к этому можно использовать атрибут **olcLogFile** (директиву **logfile**) для создания отдельного файла, содержащего только журнал LDAP. Даже при использовании этой директивы OpenLDAP также будет производить журналирование той же информации через syslogd. Пример:

```
# форма OLC (cn=config)
olcLogFile: /path/to/ldap/log/file

# форма slapd.conf
logfile /path/to/ldap/log/file

# файл должен существовать до запуска OpenLDAP
touch /path/to/ldap/log/file
chown ldap:ldap /path/to/ldap/log/file
```

olcLogLevel (loglevel)

OpenLDAP производит журналирование через канал LOCAL4 syslogd. Чтобы направить журнал LDAP в отдельный файл средствами syslog, добавьте в **syslog.conf** (обычно /etc/syslog.conf) такую строку:

```
# добавляем в syslog.conf
local4.* /var/log/ldap.log

# создаём пустой log-файл
touch /var/log/ldap.log

# перезапускаем syslogd
killall -HUP syslogd
ИЛИ
/etc/rc.d/syslogd restart
```

С помощью этих действий журналы всех уровней канала local4 (то есть OpenLDAP) будут выводиться /var/log/ldap.log. Другой способ — воспользоваться [атрибутом olcLogFile](#) (директивой [logfile](#)).

Уровни журналирования OpenLDAP задаются следующей директивой:

Возможные значения **number**, **hex-value** и **log-name**:

number	hex-value	log-name	Описание уровня журналирования
-1	0xFFFF	any	включить всю отладочную информацию
0	0x0000	-	журналирование отключено — в журнал ничего не выводится, в том числе критические ошибки. Не рекомендуется.
1	0x1	trace	отслеживание вызовов функций
2	0x2	packets	отладка обработки пакетов
4	0x4	args	усиленная отладка
8	0x8	conns	управление соединениями
16	0x10	BER	вывод посылки и приёма пакетов
32	0x20	filter	обработка поисковых фильтров
64	0x40	config	обработка конфигурационного файла
128	0x80	ACL	обработка списков контроля доступа
256	0x100	stats	статистика соединений/операций/результатов (значение по умолчанию)
512	0x200	stats2	статистика посылки записей журнала
1024	0x400	shell	вывод взаимодействия с механизмами манипуляции данными shell
2048	0x800	parse	вывод отладочной информации разбора записей
4096	0x1000	cache	кеширование (не используется)
8192	0x2000	index	индексирование (не используется)
16384	0x4000	sync	вывод журнала syncrpl (репликации)
32768	0x8000	none	none — неверное наименование, на данном уровне выводятся сообщения, не попавшие в другие категории, включая, в том числе, высокоприоритетные сообщения

Директива loglevel принимает единичное значение или список значений, разделённых пробелами. Каждое значение может быть комбинацией **number**, **hex-value** или **log-name** из приведённой выше таблицы. Результаты соединяются друг с другом через логическое ИЛИ. Кроме того, можно задать множественное значение, состоящее из нескольких объединённых в одну записей типа **number** или **hex-value**, как показано в следующих примерах:

```
loglevel 255
# задаёт уровни 1, 2, 4, 8, 16, 32, 64 и 128
# добавляются все эти значения

loglevel 2176
# 2048 + 128
loglevel 296
# 256 + 32 + 8

# использование единичного hex-value (128)
loglevel 0x80

# множественное hex-value (1 + 128)
loglevel 0x81
# аналогично такой директиве
loglevel 0x1 0x80

# использование log-name (единичное значение)
loglevel acl

# несколько значений log-name
loglevel acl sync

# комбинация различных типов
loglevel 1 0x40 conns
```

Если атрибут **olcLogLevel** (директива **loglevel**) не определен, уровень журналирования по умолчанию — 256 (только stats).

Примечание: Если уровень журналирования установлен в -1, slapd выдаёт просто огромное количество отладочной информации. Понизьте это значение как можно скорее для вывода только тех сведений, которые Вас интересуют, либо покупайте новые диски, много новых дисков.



olcModuleLoad (moduleload)

Сногшибательный атрибут (директива), представляющий собой косвенный ущерб от плохо реализованных функций — в данном случае, наложений. То, что должно быть задействовано автоматически, исходя из настроек конфигурации, требует ручного определения, да ещё и основанного на ужасающем количестве опций сборки и скрипта configure, которые, при использовании RPM, обычно оказываются недоступными для пользователя. Но только не для пользователя OpenLDAP.

Атрибут **olcModuleLoad** (директива **moduleload**) указывается в разделе глобальных настроек и определяет имя **динамически** загружаемого модуля, который будет использоваться сервером исходя из настроек конфигурации. Указание этой директивы имеет очень важное значение, если: а) наложение является динамическим (смотрите ниже), и б) наложение используется (например, в директиве database) или указывается в директиве [overlay](#). Название наложения может задаваться как с указанием полного пути к файлу модуля, так и просто именем файла и должно содержать суффикс библиотеки .la (библиотеки, собранные в libtool) или .so (библиотеки разделяемых объектов). Для имен, не содержащих абсолютного пути, поиск файлов производится в директориях, указанных в атрибуте **olcModulePath** (директиве **modulepath**) или, если **olcModulePath/modulepath** не заданы, в директориях, указанных в **PATH**, **LTDL_LIBRARY_PATH** и **LD_LIBRARY_PATH**. Данный атрибут (директива) и [атрибут olcModulePath \(директива modulepath\)](#) могут применяться только если при сборке была указана опция --enable-modules (в большинстве дистрибутивов указана).

Следует отметить, что модули OpenLDAP могут быть статическими или динамическими (только динамические наложения требуют использования атрибутов **olcModuleLoad** (директив **moduleload**)). Покажем разницу между статическими и динамическими модулями на примере. Если при запуске скрипта configure OpenLDAP задано, скажем, --enable-accesslog=mod, то наложение accesslog будет собрано отдельным модулем и, следовательно, ДОЛЖНО подключаться с помощью атрибута **olcModuleLoad** (директивы **moduleload**). Если скрипту configure передавался параметр в форме --enable-accesslog, то данное наложение при сборке будет встроено в демон slapd, то есть оно будет собрано как статическое наложение. В таком случае указание атрибута **olcModuleLoad** (директивы **moduleload**) для accesslog вызовет ошибку, поскольку при --enable-accesslog отдельного модуля для данного наложения не создаётся (для получения полного списка опций скрипта configure выполните ./configure --help из директории сборки OpenLDAP). Чтобы наверняка узнать, является ли нужный модуль динамическим, можно помотреть в директорию, где хранятся собранные модули([bsd] /usr/local/libexec/openldap) [fc] /usr/libexec/openldap или /usr/sbin/openldap). Если этот модуль там есть и он используется при конфигурации OpenLDAP, его необходимо указать в атрибуте **olcModuleLoad** (директиве **moduleload**), а если его там нет — можно предположить, что он собран статически. Если есть сборка модуля с обоими суффиксами .la и .so — используйте файл с суффиксом .la, а если он не работает — тогда с суффиксом .so. Если не работают оба файла, мы можем порекомендовать только ритуальное самоубийство. Имена модулей можно посмотреть в описании директив [overlay](#) и [database](#).

```
# форма OLC (cn=config)
olcModuleLoad: module-name

# форма slapd.conf
moduleload module-name

# пример - загрузка наложения syncprov
# форма OLC (cn=config)
olcModuleLoad: syncprov.la

# форма slapd.conf
loadmodule syncprov.la

# формат с указанием полного пути
# форма OLC (cn=config)
olcModuleLoad: /usr/local/libexec/openldap/syncprov.la

# форма slapd.conf
loadmodule /usr/local/libexec/openldap/syncprov.la
```

Примечание: Атрибуты **olcModuleLoad** и **olcModulePath** определяются в записи `cn=modules{0},cn=config`, использующей объектный класс **olcModuleList**.



olcModulePath (modulepath)

Определяет одну или несколько директорий файловой системы, в которых будут находиться загружаемые модули (наложения). В одной директиве можно задать несколько путей, в этом случае они разделяются двоеточием (*nix) или точкой с запятой (windows). Если эта директива не определялась, поиск осуществляется в директориях, заданных в переменных окружения **PATH**, **LTDL_LIBRARY_PATH** и **LD_LIBRARY_PATH**. Смотрите также описание [атрибута olcModuleLoad \(директивы moduleload\)](#).

Примечание: В конфигурации OLC (cn=config) атрибут **olcModulePath** может иметь только одно значение (SINGLE-VALUE). Атрибуты **olcModuleLoad** и **olcModulePath** определяются в записи `cn=modules{0},cn=config`, использующей объектный класс **olcModuleList**.



olcObjectClasses (objectclass)

В качестве артефакта обработки схемы данных OpenLDAP предоставляет возможность определить один или несколько **объектных классов (objectclass)** в файле slapd.conf. Определение **объектного класса** описано [здесь](#). Не совсем понятно, что может подвигнуть Вас добавлять объектные классы таким способом — предпочтительнее создать файл пользовательского набора схемы и добавить его с помощью директивы **include**. Тем не менее, Вы можете сделать и так, если Вас это забавляет.

В конфигурации OLC (cn=config) атрибут **olcObjectClasses** используется для загрузки наборов схемы данных, требуемых системой OLC (cn=config).



olcPasswordHash (password-hash)

Позволяет определить один или несколько методов хэширования, используемых при хранении новых паролей, генерируемых с применением расширенной операции изменения пароля (Extended Modify Password Operation, RFC 3602), в атрибуте **userPassword**. Формат:

```
# форма OLC (cn=config)
olcPasswordHash: {hash}[,{hash} [, ...]]

# форма slapd.conf
password-hash {hash}[,{hash} [, ...]]
```

Значения hash должны быть из набора поддерживаемых методов: {SSHA}, {SHA}, {SMD5}, {MD5}, {CRYPT} или {CLEARTXT}. Значение по умолчанию — {SSHA}. Пример:

```
# форма OLC (cn=config)
olcPasswordHash: {SHA},{SSHA}

# форма slapd.conf
password-hash {SHA},{SSHA}
```

Здесь методом хэширования по умолчанию для всей серверной части задаётся {SHA} (FIPS-160 без "соли"). При задании нескольких значений они должны разделяться запятыми, при этом нельзя помещать между разделителями пробелы — slapd просто не загрузится.

В конфигурации OLC (cn=config) данный атрибут указывается/добавляется в запись `olcDatabase={-1}frontend,cn=config`, а не в запись глобальных настроек cn=config.



olcPidFile (pidfile)

Заставляет OpenLDAP (slapd) записывать PID в указанный файл. Пример:

```
# форма OLC (cn=config)
olcPidFile: /var/run/ldap.pid

# форма slapd.conf
pidfile /var/run/ldap.pid
```

Данный PID может быть использован в команде **kill** или для посылки других сигналов в ldap. В качестве альтернативы можно использовать такую команду:

```
ps ax |grep slapd
```

В выводе будет указана информация о PID. Если атрибут **olcPidFile** (директива **pidfile**) не указывался, файл с PID создаваться не будет. Большинство скриптов запуска/остановки/перезапуска slapd будут работать лишь при наличии PID-файла, причём в том месте, где они его ожидают найти. У Вас есть два варианта: использовать атрибут olcPidFile (директиву pidfile) для того, чтобы определить тот путь к файлу pid, который ожидают стандартные скрипты и пользоваться этими скриптами, либо не использовать olcPidFile/pidfile (или задать в них нестандартные для Вашей системы пути) и запускать/останавливать slapd вручную.

olcReferral (referral)

Формат:

```
# форма OLC (cn=config)
olcReferral: uri

# форма slapd.conf
referral uri
```

Атрибут **olcReferral** (директива **referral**) позволяет OpenLDAP возвращать указанный **uri** в качестве отсылки (**referral**) в случае, когда запрашиваемое значение DN не входит ни в одну базу данных/DIT (суффикс не совпадает с указанными в [атрибутах olcSuffix](#) (директивах [suffix](#)) разделов database) этого сервера. Не все клиенты способны обработать возвращённый **uri**, который должен быть сформирован как полный LDAP URL, содержащий максимально возможное количество информации, и должен правильно указывать на сайт, позволяющий анонимный доступ к требуемому корневому DN. Примеры:

```
# форма OLC (cn=config)
olcReferral: ldap://ldap.openldap.org/

# форма slapd.conf
referral ldap://ldap.openldap.org/
# в данном URL предпринимается попытка получить доступ к rootDSE системы OpenLDAP,
# находящейся по данному адресу, которая может завершиться неудачей!

# форма OLC (cn=config)
olcReferral: ldap://ldap.example.com/dc=example,dc=com??one?(objectclass=*)

# форма slapd.conf
referral ldap://ldap.example.com/dc=example,dc=com??one?(objectclass=*)
# система по адресу ldap.example.com должна уметь обслуживать запросы такого рода
# и возвращать все записи первого уровня, поддерживаемые всеми LDAP-серверами
# в данной системе - своего рода обзор общих сервисов/предметных областей
```

Примечание: Атрибут olcReferral (директива referral) применяется, когда сервер не может найти запрашиваемый DN внутри структуры своих DIT. Отсылки как делегирование полномочий на обслуживание подмножества DIT стандартным способом настраиваются с помощью специальных записей с объектным классом referral внутри структуры DIT. Дополнительная информация и примеры конфигурации отсылок [здесь](#).

replicationinterval

Начиная с версии 2.4 эта директива считается устаревшей и в OLC (cn=config) служит лишь в целях переконвертации slapd.conf. Определяется на главном (master) сервере, используется в [репликации в стиле slurpd](#). Данная директива читается демоном slurpd и устанавливает интервал (в секундах), по истечении которого slurpd будет очередной раз проверять файл [repllogfile](#) на предмет обновления подчинённых (slave) серверов.

```
replicationinterval seconds  
# пример: проверять repllogfile каждые 60 секунд  
replicationinterval 60
```

Происхождение этой директивы покрыто мраком тайны, но, кажется, она появилась в OpenLDAP версии 2.2. Не совсем понятно, как впрочем, часто бывает в OpenLDAP, каково значение по умолчанию, то есть какой будет интервал репликации, если не указывать данную директиву.



olcRequires (require)

```
# форма OLC (cn=config)  
olcRequires: policy [...]  
  
# форма slapd.conf  
require policy [...]
```

Данная директива, которая может указываться как в разделе глобальных настроек, так и в разделе database, определяет требования, которые должны быть соблюдены перед получением любого рода доступа к серверу. При использовании в разделе database установленное значение требований сохраняется, пока не будет сброшено (**olcRequires: none / require none**). Следующие друг за другом атрибуты **olcRequires** (директивы **require**), если это не атрибут/директива сброса **olcRequires: none / require none**, применяются аддитивно, то есть все их требования должны быть соблюдены. В директиве может быть указано одно или несколько из перечисленных ниже значений (разделённых пробельными символами, например, пробелом или табуляцией):

authc	Требуется, чтобы при любом обращении к DIT, перед тем как выполнить какую-либо операцию с каталогом, была пройдена LDAP-аутентификация. Использование этого значения также подразумевает неявное использование ключевого слова bind .
bind	Требуется, чтобы при любом обращении к DIT, перед тем как выполнить какую-либо операцию с каталогом, было выполнено подсоединение (bind). Данное ключевое слово просто требует подсоединения, которое может быть и анонимным подсоединением, и не подразумевает требования прохождения LDAP-аутентификации.
LDAPv3	Требуется, чтобы при любом обращении к DIT использовались процедуры 3-ей версии протокола LDAP.
none	Значение по умолчанию. Указывает, что никаких требований не предъявляется, и может применяться для сброса требований после их использования в разделе database. При отсутствии подобного сброса, последнее значение атрибута olcRequires (директивы require), определённого ранее, распространяется на все последующие разделы database. Если же сброса не произошло и объявляются следующие атрибуты olcRequires (директивы require), то требования каждой последующей директивы складываются с требованиями всех предыдущих директив с момента последнего сброса (то есть директивы являются аддитивными).
SASL	Требуется, чтобы перед выполнением операций с DIT была выполнена SASL-аутентификация. Использование данного значения не подразумевает прохождения LDAP-аутентификации или подсоединения; таким образом, неявного использования ключевых слов authc и bind не подразумевается.
strong	Требуется, чтобы перед выполнением операций с DIT были выполнены либо SASL, либо TLS процедуры. Использование данного значения не подразумевает прохождения LDAP-аутентификации или подсоединения; таким образом, неявного использования ключевых слов authc и bind не подразумевается.

Примеры:

```
# сбрасывает значение предыдущих требований olcRequires/require и
# устанавливает требование прохождения LDAP-аутентификации для данного DIT
# форма OLC (cn=config)
olcRequires: none authc

# форма slapd.conf
require none authc
```

olcReverseLookup (reverse-lookup)

Формат:

```
# форма OLC (cn=config)
olcReverseLookup: TRUE | FALSE

# форма slapd.conf
reverse-lookup on | off
```

Если атрибут **olcReverseLookup** (директива **reverse-lookup**) установлен в **TRUE** (или **on**), то OpenLDAP будет использовать обратное разрешение имён DNS при каждом обращении клиента. Значение по умолчанию — **FALSE** (или **off**). Этот атрибут (директива) будет работать лишь в том случае, если при сборке использовалась опция `--enable-rlookups`. Примеры:

```
# форма OLC (cn=config)
oldReverseLookup: TRUE

# форма slapd.conf
reverse-lookup on
# заставляет OpenLDAP выполнять обратное разрешение имён DNS
# при каждом клиентском доступе
```

olcRootDSE (rootDSE)

Формат:

```
# форма OLC (cn=config)
olcRootDSE: /path/to/ldif/file

# форма slapd.conf
rootDSE /path/to/ldif/file
```

В атрибуте **olcRootDSE** (директиве **rootDSE**) указывается LDIF-файл, содержащий определённые пользователем операционные атрибуты, которые будут добавлены к существующей записи rootDSE (доступна для просмотра в [subschema subentry](#)). Эти атрибуты являются аддитивными — в дополнение к стандартным (встроенным) атрибутам.

olcSecurity (security)

```
# форма OLC (cn=config)
olcSecurity: ssf-policy

# форма slapd.conf
security ssf-policy [...]
```

Данный атрибут (директива) может быть определен в записи (разделе) глобальных настроек, в этом случае областью его действия будут все разделы database, либо он может быть указан в конкретной записи (разделе) database, в этом случае область его действия ограничивается только этим DIT (при этом переопределяются политики любых директив из раздела глобальных настроек, если таковые имеются). При использовании защищенных соединений ([TLS](#) или [SASL](#)) атрибут olcSecurity (директива security) определяет

минимальный фактор силы безопасности (Security Strength Factor, SSF), применяющийся к любому типу безопасности (TLS или SASL) или требующий выполнения транзакции конкретного типа. Может быть определено одно или несколько (разделённых пробельными символами) значений **ssf-policy** из следующего списка:

sasl=n	Если устанавливается сессия SASL, то минимальная длина ключа должна быть не менее n .
simple_bind=n	Сервер будет отказываться принимать любые операции подсоединения, не защищённые сессиями TLS или SASL с SSF не менее n (ошибка LDAP_CONFIDENTIALITY_REQUIRED, 13, x'0D).
ssf=n	При установке соединений SASL или TLS требуется, чтобы минимальная длина ключа была не менее n .
tls=n	Если устанавливается сессия TLS, то минимальная длина ключа должна быть не менее n .
update_sasl=n	Сервер будет отказываться принимать любые операции обновления каталога (изменения, модификации, и т.д.), не защищённые сессией SASL с SSF не менее n .
update_ssf=n	Сервер будет отказываться принимать любые операции обновления каталога (изменения, модификации, и т.д.), не защищённые сессиями SASL или TLS с SSF не менее n .
update_tls=n	Сервер будет отказываться принимать любые операции обновления каталога (изменения, модификации, и т.д.), не защищённые сессией TLS с SSF не менее n .

Значение **n** указывает требуемый уровень безопасности, выражаемый как минимальное количество бит в криптографическом ключе, который будет использоваться. Как правило, это значение берётся из диапазона от 40 до 256 (стандартные значения: 40, 56, 64, 128, 164 и 256) в зависимости от алгоритма, который будет задействован. Таким образом, значение 1 показывает, что может применяться любой уровень криптографической защиты, а значение 128 — что сессии с применением криптографических алгоритмов с длиной ключа 128 и более будут пропускаться, а с длиной ключа, скажем, 56 — отклоняться.



olcSchemaDN (schemadn)

Формат:

```
# форма OLC (cn=config)
olcSchemaDN: cn=name

# форма slapd.conf
schemadn cn=name
```

Атрибут **olcSchemaDN** (директива **schemadn**) определяет **название subschema subentry**, используемое OpenLDAP. Значение по умолчанию — 'cn=subschema'. Примеры:

```
# форма OLC (cn=config)
olcSchemaDN: cn=ldapsubschema

# форма slapd.conf
schemadn cn=ldapsubschema
# изменяет название subschema subentry с
# subschema (по умолчанию) на ldapsubschema
```

Не совсем понятно, зачем Вам может понадобиться использовать эту функцию, за исключением, пожалуй, поддержки совместимости с другими/предыдущими реализациями LDAP.



olcServerID (ServerID)

Формат:

```
# форма OLC (cn=config)
olcServerID: number [URL]

# форма slapd.conf
serverid number [URL]
```

Только для OpenLDAP версий 2.4+. Атрибут **olcServerID** (директива **serverid**) определяет метод, по которому данный сервер можно уникально идентифицировать с помощью либо **числа (number)** (известного как SID), либо **URL** (или нескольких URL). Этот атрибут (директива) используется в [репликации с несколькими главными серверами](#), поддерживаемой начиная с OpenLDAP 2.4. Параметр **number** — это произвольное число, содержащее от 1 до 3 цифр, которое уникально идентифицирует сервер в группе нескольких главных серверов репликации (multi-master group). Значение по умолчанию — 000. Идентификатор сервера (server ID, SID) используется (как и значение rid обновлённого (в версии 2.4) [CSN](#)) для уникальной идентификации хоста в качестве источника обновлений. Если параметр директивы передаётся в опциональной форме URL, то данная директива может присутствовать несколько раз. Во многих примерах конфигурации с несколькими главными серверами показано, что все серверы в группе главных серверов репликации идентифицируются с помощью ассоциированных с ними URL. Эксперименты показывают, что это необязательное требование (информация о поставщике указывается в директиве **sync repl**), числовой формат также прекрасно работает. Ключевой характеристикой является то, что указанное значение (не важно, число или URL) в любой момент уникально идентифицирует данный сервер. При выборе того, каким форматом Вы будете пользоваться, имейте в виду два момента. Во первых, даже если число ServerID в данный момент уникально, оно может не всегда оставаться уникальным, например, если Вы впоследствии захотите производить репликацию с сервера, которому уже выделен SID 001. Во вторых, если DIT конфигурации сервера **cn=config** впоследствии реплицируется на другой сервер с тем же самым SID, результат может быть непредсказуемым. Использование в качестве SID URL сервера может обеспечить гарантированно уникальное значение данного идентификатора. Ну, а если нет — кроме ритуального самоубийства ничего не остаётся. Наконец, нет никакой связи между SID (server ID) и параметром **rid** директивы **sync repl**. Примеры:

```
# форма OLC (cn=config)
olcServerID: 001

# форма slapd.conf
serverid 001
# то же, что и предыдущее, но с одним символом
serverid 1

# в форме URI
# форма OLC (cn=config)
olcServerID: ldap1.example.com

# форма slapd.conf
serverid ldap1.example.com
```

olcSizeLimit (sizelimit)

Атрибут **olcSizeLimit** (директива **sizelimit**) определяет количество записей, которые будут возвращаться на каждый поисковый запрос. Существует две формы данной директивы, первая из них:

```
# форма OLC (cn=config)
olcSizeLimit: integer | unlimited

# форма slapd.conf
sizelimit integer | unlimited
```

Здесь **integer** — значение от 1 до 65435. **unlimited** (или -1) говорит о том, что ограничений на количество возвращаемых результатов не накладывается.

Вторая форма предоставляет больше контроля над количеством возвращаемых результатов и имеет следующий формат:

```
# форма OLC (cn=config)
olcSizeLimit: size[.{soft|hard|unchecked}]=integer

# форма slapd.conf
sizelimit size[.{soft|hard|unchecked}]=integer
```

Здесь **integer** — это максимальное число записей, которые slapd будет возвращать в ответ на поисковый запрос. Поведение директивы зависит от необязательного квалификатора **soft**, **hard** или **unchecked** следующим образом:

1. Если при запросе клиент явно не указывал ограничений по размеру, используется **мягкое (soft)** ограничение.
2. Если указанное клиентом ограничение по размеру превысило **жёсткое (hard)** ограничение, возвращается ошибка "Administrative limit exceeded" (превышено административное ограничение).
3. Если ограничение **hard** равно 0 или ключевому слову "soft", ограничение **soft** используется в любом случае.
4. Если ограничение **hard** равно -1 или ключевому слову "none", жёстких ограничений не накладывается.
5. Явно указанное при запросе ограничение по размеру меньше или равное жёсткому ограничению будет удовлетворено.
6. Квалификатор **unchecked** устанавливает ограничение на количество записей-кандидатов, которые могут быть рассмотрены в ходе обработки поискового запроса. Если количество отобранных записей-кандидатов превысило ограничение **unchecked**, поиск будет прерван с ошибкой LDAP_UNWILLING_TO_PERFORM (53, x'35) (не желаю исполнять). Если ограничение **unchecked** равно -1 или ключевому слову "none", ограничений на количество обрабатываемых записей-кандидатов не накладывается (поведение по умолчанию).
7. Если квалификаторы не указаны, значение **integer** присваивается мягкому ограничению, а жёсткое ограничение устанавливается в ноль, таким образом сохраняется оригинальное поведение, как при первой форме определения директивы sizelimit.

Если директива **sizelimit** не была определена, значение по умолчанию — 500. Примеры:

```
# форма OLC (cn=config)
olcSizeLimit: 0

# форма slapd.conf
sizelimit 0
# не отключать простаивающих клиентов

# форма OLC (cn=config)
olcSizeLimit: 100

# форма slapd.conf
sizelimit 100
# в ответ возвращается не более 100 записей

# расширенная форма
# форма OLC (cn=config)
olcSizeLimit: size=100

# форма slapd.conf
sizelimit size=100
# то же самое, что и sizelimit 100

# форма OLC (cn=config)
olcSizeLimit: size.soft=100 size.hard=200

# форма slapd.conf
sizelimit size.soft=100 size.hard=200
# если клиент не установил ограничений, то ограничения равны 100
# если клиент установил ограничения, превышающие 200 - запрос отклоняется
# нет ограничений на количество записей-кандидатов при поиске

# форма OLC (cn=config)
olcSizeLimit: size.unchecked=1000

# форма slapd.conf
sizelimit size.unchecked=1000
# если клиент не установил ограничений, то будет возвращено
# максимум 500 записей (значение по умолчанию)
# если же клиент установил ограничение, то
# если клиентское ограничение превысило 500 (значение по умолчанию) - запрос отклоняется
# если при обработке выбрано более 1000 записей-кандидатов - запрос отклоняется
```



olcSockbufMaxIncoming (sockbuf_max_incoming)

Формат:

```
# форма OLC (cn=config)
olcSockbufMaxIncoming: bytes

# форма slapd.conf
sockbuf_max_incoming bytes
```


Атрибут **olcSockbufMaxIncoming** (директива **sockbuf_max_incoming**) определяет максимальный PDU (размер блока) в байтах для входящих анонимных сессий. Значение по умолчанию — 262143. Примеры:

```
# форма OLC (cn=config)
olcSockbufMaxIncoming: 5000

# форма slapd.conf
sockbuf_max_incoming 5000
# максимальный размер PDU (блока данных) - 5000 байт,
# при превышении соединение отклоняется
```



olcSockbufMaxIncomingAuth (sockbuf_max_incoming_auth)

Формат:

```
# форма OLC (cn=config)
olcSockbufMaxIncomingAuth: integer

# форма slapd.conf
sockbuf_max_incoming_auth integer
```

Атрибут **olcSockbufMaxIncomingAuth** (директива **sockbuf_max_incoming_auth**) определяет максимальный PDU (размер блока) в байтах для входящих сессий с проверкой подлинности. Значение по умолчанию — 4194303. Примеры:

```
sockbuf_max_incoming_auth 5000
# максимальный размер PDU - 5000 байт,
# при превышении соединение отклоняется
```



olcThreads (threads)

Формат:

```
# форма OLC (cn=config)
olcThreads: integer

# форма slapd.conf
threads integer
```

Атрибут **olcThreads** (директива **threads**) определяет количество потоков, которые может использовать OpenLDAP. Чем выше это значение, тем больше одновременных запросов может обработать сервер. Значение по умолчанию — 16. Примеры:

```
# форма OLC (cn=config)
olcThreads: 25

# форма slapd.conf
threads 25
# разрешено максимум 25 потоков
```

Не совсем понятно, какова связь между атрибутом **olcThreads** (директивой **threads**) и [атрибутом olcConcurrency](#) (директивой **concurrency**).



olcTimeLimit (timelimit)

Атрибут **olcTimeLimit** (директива **timelimit**) определяет время в секундах, которое slapd может потратить на формирование ответа на поисковый запрос. Если за это время обработка запроса не будет завершена, клиенту возвращается ошибка о превышении ограничения времени. Существует две формы данной директивы, первая из них:

```
# форма OLC (cn=config)
olcTimeLimit: seconds | unlimited

# форма slapd.conf
timelimit seconds | unlimited
```

Здесь **seconds** — число секунд, которое slapd будет тратить на формирование ответа на поисковый запрос, **unlimited** или **-1** говорит о том, что ограничений на время обработки поискового запроса не накладывается.

Вторая (расширенная) форма предоставляет больше контроля над временем обработки запроса и имеет следующий формат:

```
# форма OLC (cn=config)
olcTimeLimit: time[.{soft|hard}]=seconds [...]

# форма slapd.conf
timelimit time[.{soft|hard}]=seconds [...]
```

Здесь **seconds** — число секунд, которое slapd будет тратить на формирование ответа на поисковый запрос. Поведение директивы определяется необязательным квалификатором **soft** или **hard** следующим образом:

1. Если при запросе клиент явно не указывал ограничений времени, используется **мягкое (soft)** ограничение.
2. Если указанное клиентом ограничение времени превысило **жёсткое (hard)** ограничение, возвращается ошибка "Administrative limit exceeded" (превышено административное ограничение).
3. Если ограничение **hard** равно 0 или ключевому слову "soft", ограничение **soft** используется в любом случае.
4. Если ограничение **hard** равно -1 или ключевому слову "none", жёстких ограничений не накладывается.
5. Явно указанное при запросе ограничение времени меньше или равно жёсткому ограничению будет удовлетворено.
6. Если квалификаторы не указаны, значение **integer** присваивается мягкому ограничению, а жёсткое ограничение устанавливается в ноль, таким образом сохраняется оригинальное поведение, как при первой форме определения директивы **timelimit**.

Если директива **timelimit** не была определена, устанавливается ограничение в 3600 секунд (1 час).
Примеры:

```
# форма OLC (cn=config)
olcTimeLimit: 15

# форма slapd.conf
timelimit 15
# для выполнения поискового запроса отводится 15 секунд
# при превышении возвращается ошибка time limit exceeded

# форма OLC (cn=config)
olcTimeLimit: 0

# форма slapd.conf
timelimit 0
# отключает ранее установленные ограничения
# (становится равным значению по умолчанию 3600)
timelimit unlimited
# ограничений по времени не накладывается

# расширенный формат
# форма OLC (cn=config)
olcTimeLimit: time=15

# форма slapd.conf
timelimit time=15
# то же самое, что и timelimit 15

# форма OLC (cn=config)
olcTimeLimit: time.soft=15 time.hard=20

# форма slapd.conf
timelimit time.soft=15 time.hard=20
# если клиент не установил ограничение времени, то ограничение равно 15 секундам
# если клиент установил ограничение времени, превышающее 20 - запрос отклоняется

# форма OLC (cn=config)
olcTimeLimit: time.soft=15

# форма slapd.conf
```

```
timelimit time.soft=15
# если клиент не установил ограничение времени, то ограничение равно 15 секундам
# если клиент установил какое-либо ограничение времени, оно будет принято

# форма OLC (cn=config)
olcTimeLimit: time.soft=15 time.hard=none

# форма slapd.conf
timelimit time.soft=15 time.hard=none
# если клиент не установил ограничение времени, то ограничение равно 15 секундам
# если клиент установил какое-либо ограничение времени, оно будет принято
```

6.3.1 Директивы TLS/SSL (OLC (cn=config) и slapd.conf)

Обзор TLS — что такое TLS сервер/клиент

В OpenLDAP атрибуты/директивы, необходимые для сервера (в частности, TLS-сервера), и для клиента (в частности, TLS-клиента) различаются. Термин LDAP-клиент применяется, скажем, по отношению к LDAP-браузеру или почтовому клиенту, считывающему адресную книгу из LDAP, тогда как системы OpenLDAP являются серверами. Однако TLS несколько размывает границы между этими понятиями. То, что мы называем сервером LDAP, может оказаться как TLS-сервером, так и TLS-клиентом, а может быть сразу и TLS-клиентом, и TLS-сервером, — в последнем случае для него необходимо определять как серверные, так и клиентские атрибуты/директивы TLS. Ключевым различием между сервером и клиентом TLS является то, что TLS-сервер никогда не является инициатором соединения, им всегда является TLS-клиент. Данные различия определяются на уровне записи/раздела database; таким образом, один из разделов database в slapd.conf (или olcDatabase в cn=config) может быть TLS-сервером, а другой — TLS-клиентом. Вот несколько примеров:

1. Система LDAP, которая занимается только предоставлением доступа внешним клиентам, не имеющая [атрибутов olcSyncrepl](#) (директив [syncrepl](#)) в какой-либо из записей (разделов) database и не имеющая записей (директив) [overlay chain](#), является TLS-сервером.
2. Запись (раздел) database, содержащая дочернюю запись (директиву) [overlay syncprov](#) и являющаяся поставщиком репликации в конфигурации типа [главный-подчинённый \(master-slave\)](#), всегда является TLS-сервером (потребитель репликации всегда подключается к поставщику).
3. Запись (раздел) database, содержащая [атрибут olcSyncrepl](#) (директиву [syncrepl](#)) и являющаяся потребителем репликации в конфигурации типа [главный-подчинённый \(master-slave\)](#), всегда является TLS-клиентом (потребитель репликации всегда подключается к поставщику).
4. Запись (раздел) database, содержащая [атрибут olcSyncrepl](#) (директиву [syncrepl](#)) и дочернюю запись (директиву) [overlay syncprov](#) в конфигурации репликации [с несколькими главными серверами \(multi-master\)](#), является одновременно и TLS-клиентом (как потребитель репликации, настраиваемый атрибутом (атрибутами) [olcSyncrepl](#) (директивой (директивами) [syncrepl](#))), и TLS-сервером (как поставщик репликации [syncprov](#)).
5. Сервер LDAP, содержащий запись/директиву [overlay chain](#), может быть одновременно и TLS-клиентом (для соединения сцепления (chaining)), и TLS-сервером (для обычного пользовательского доступа).
6. Вы будете приятно удивлены, узнав, что LDAP-клиенты, такие как LDAP-браузер, почтовый клиент и т.п., всегда являются TLS-клиентами!

Несколько рабочих [примеров конфигурации TLS/SSL](#) представлены в [главе 15](#). Все серверные директивы TLS указываются в записи cn=config или разделе глобальных настроек slapd.conf. Все клиентские директивы TLS указываются в [ldap.conf](#).

olcTLSCACertificateFile (TLSCACertificateFile)

```
# форма OLC (cn=config)
olcTLSCACertificateFile: /path/to/CA/cert/file.pem

# форма slapd.conf
```

```
TLSCACertificateFile /path/to/CA/cert/file.pem
```

Атрибут (директива) сервера TLS. Определяет путь до файла и имя файла сертификата Удостоверяющего центра (Certificate Authority, CA), известного также как корневой сертификат. Этот сертификат используется только при проверке входящих сертификатов. Данная директива требуется лишь в случае, когда сервер ожидает поступления или требует предъявления сертификата клиента ([атрибут `olcTLSVerifyClient` \(директива `TLSVerifyClient`\)](#) установлен в одно из значений `try`, `demand` или `allow`). Если используется значение по умолчанию атрибута `olcTLSVerifyClient` (директивы `TLSVerifyClient`) `never`, определять директиву `TLSCACertificateFile` не нужно. Если данная директива определяется, она указывает на корневой (CA) сертификат, который должен быть получен у поставщика сертификатов X.509 (то есть у Удостоверяющего центра, CA). Как правило, это файл в формате PEM (Privacy enhanced Mail) с расширением `.pem` или, если он был получен из установки браузера MSIE, с расширением `.cer`. Если рабочий сертификат X.509 (указанный в [атрибуте `olcTLSCertificateFile` \(директиве `TLSCertificateFile`\)](#)) подписан промежуточным удостоверяющим центром, то в файле формата PEM должны присутствовать все сертификаты удостоверяющих центров вплоть до центра корневого уровня. PEM — это текстовый формат, и несколько сертификатов могут помещаться в один и тот же файл в произвольном порядке — смотрите [заметки и примеры по формату PEM](#). В данном файле нет конфиденциальной информации (сертификат X.509 содержит только открытый ключ), поэтому для него не требуется установки специальных разрешений.

Примечание: Если LDAP-сервер выступает в роли TLS-клиента, например, как потребитель репликации с установленным [атрибутом `olcSyncrepl` \(директивой `syncrepl`\)](#), то требуемый корневой (CA) сертификат определяется в директиве `TLS_CACERT` в файле `ldap.conf`.



olcTLSCertificateFile (TLSCertificateFile)

```
# форма OLC (cn=config)
olcTLSCertificateFile: /path/to/cert/file.pem

# форма slapd.conf
TLSCertificateFile /path/to/cert/file.pem
```

Атрибут (директива) сервера TLS. Определяет путь до файла и имя файла рабочего сертификата X.509 сервера (в атрибуте `cn=` DN в поле [subject](#) данного сертификата указан URL этого LDAP-сервера, например, `ldap.example.com`). Данный сертификат содержит открытый ключ, используемый при обмене ключами в [протоколе TLS](#). Указываемый в директиве файл, как правило, в формате PEM (Privacy Enhanced Mail) обычно имеет суффикс `.pem`. PEM — это текстовый формат (смотрите [заметки и примеры по формату PEM](#)). Если данный сертификат является самоподписанным, то в зависимости от того, как он был создан, может потребоваться (или не потребоваться) [атрибут `olcTLSCACertificateFile` \(директива `TLSCACertificateFile`\)](#). Примеры конфигурации OpenLDAP с самоподписанным сертификатом можно посмотреть [здесь](#). В данном файле нет конфиденциальной информации (сертификат X.509 содержит только открытый ключ), поэтому для него не требуется установки специальных разрешений.



olcTLSCertificateKeyFile (TLSCertificateKeyFile)

```
# форма OLC (cn=config)
olcTLSCertificateKeyFile: /path/to/private/key/file.pem

# форма slapd.conf
TLSCertificateKeyFile /path/to/private/key/file.pem
```

Атрибут (директива) сервера TLS. Определяет расположение секретного ключа, открытый ключ которого содержится в сертификате сервера (определённом в [атрибуте `olcTLSCertificateFile` \(директиве `TLSCertificateFile`\)](#)). Этот ключ используется для декодирования посылаемых клиентом TLS во время [переговоров об установке соединения TLS](#) сведений о ключе pre-master (на основании которого будет вычисляться общий секретный ключ, используемый впоследствии для шифрования данных). Как правило, файл с ключом имеет суффикс `.pem`. Данный секретный ключ — очень важная конфиденциальная информация и, в идеале, он должен храниться в отдельной структуре директорий файловой системы с ограниченными правами на чтение или, как минимум, сам файл с ключом должен иметь права только на чтение (0400) для учётной записи, от имени которой запускается OpenLDAP — обычно **ldap**. Примеры

конфигурации OpenLDAP с самоподписанным сертификатом можно посмотреть [здесь](#). Пример:

```
# форма OLC (cn=config)
olcTLSCertificateKeyFile: /certs/ldapcert.pem

# форма slapd.conf
TLSCertificateKeyFile /certs/ldapcert.pem

# минимальная защита - установка прав только на файл
# подразумевается учётная запись ldap
chown ldap:ldap /certs/ldapcert.pem
chmod 0400 /certs/ldapcert.pem

# установка защиты на директорию и файл
chown -R ldap:ldap /certs/*
chmod -R 0400 /certs/*
```

olcTLSCipherSuite (TLSCipherSuite)

```
# форма OLC (cn=config)
olcTLSCipherSuite: cipher-list

# форма slapd.conf
TLSCipherSuite cipher-list
```

Атрибут (директива) сервера TLS. Это необязательная директива и по умолчанию подразумевается значение ALL (смотрите ниже). Определяет один или несколько шифров, которые будут использоваться во время [переговоров об установке соединения TLS](#). В ходе этих переговоров клиент TLS предлагает список **шифров** и сервер TLS примет **первый** шифр из своего списка шифров, который совпадёт с одним из предлагаемых клиентом. Используемый в описании данной директивы термин **cipher-list** определяет список (в формате OpenSSL), который будет перекодирован библиотеками OpenSSL в список **шифров** в формате TLS/SSL. Дополнительную информацию о формате cipher-list можно получить в [документации по шифрам OpenSSL](#). Примеры конфигурации OpenLDAP с самоподписанным сертификатом можно посмотреть [здесь](#).

Список доступных шифров (и, следовательно, cipher-list) определяется форматом открытого ключа, содержащегося в сертификате X.509, указанного в [атрибуте olcTLSCertificateFile](#) (директиве [TLSCertificateFile](#)). Так, если данный сертификат содержит открытый ключ RSA, то только шифры открытого ключа RSA могут быть использованы на этапах обмена ключами/аутентификации при [установке соединения TLS](#).

Если требуется обоюдная проверка сертификатов (и сервер и клиент TLS посылают сертификаты), то либо должен быть известен формат открытого ключа обоих сертификатов сервера и клиента TLS, либо должно использоваться значение ALL (смотрите команды ниже).

Отдельные пункты в cipher-list разделяются двоеточием, запятой или пробельным символом. Далее приводится подмножество имён RSA TLSv1, которые могут фигурировать в cipher-list, и эквивалентных им текстовых значений шифров TLS (при отправке по каналу они перекодировываются в шестнадцатеричные значения). **Примечание:** Слово EXPORT (или EXP), встречающееся в некоторых именах, указывает, что эти шифры являются экспортируемыми, то есть некоторые шифры разрешено использовать только в определённых странах (смотрите документацию [Бюро промышленности и безопасности Министерства торговли США](#) и [Вассенарские соглашения](#)), и это необходимо учитывать при настройке системы TLS, которая будет использоваться на международном уровне.

НАЗВАНИЕ ШИФРА TLS	НАЗВАНИЕ CIPHER-LIST OPENSSL
=====	=====
TLS_RSA_WITH_NULL_MD5	NULL-MD5
TLS_RSA_WITH_NULL_SHA	NULL-SHA
TLS_RSA_EXPORT_WITH_RC4_40_MD5	EXP-RC4-MD5
TLS_RSA_WITH_RC4_128_MD5	RC4-MD5
TLS_RSA_WITH_RC4_128_SHA	RC4-SHA
TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5	EXP-RC2-CBC-MD5
TLS_RSA_WITH_IDEA_CBC_SHA	IDEA-CBC-SHA
TLS_RSA_EXPORT_WITH_DES40_CBC_SHA	EXP-DES-CBC-SHA
TLS_RSA_WITH_DES_CBC_SHA	DES-CBC-SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA	DES-CBC3-SHA
TLS_RSA_WITH_AES_128_CBC_SHA	AES128-SHA
TLS_RSA_WITH_AES_256_CBC_SHA	AES256-SHA
TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA	EXP1024-DES-CBC-SHA
TLS_RSA_EXPORT1024_WITH_RC4_56_SHA	EXP1024-RC4-SHA

Чтобы вывести список значений cipher-list, поддерживаемых локальной инсталляцией OpenSSL, используйте:

```
# Все действительные шифры (ALL)
openssl ciphers -v ALL

# Все действительные шифры только для TLSv1
openssl ciphers -v -tls1 ALL

# Действительные шифры только для TLSv1, использующие
# алгоритм обмена ключами/аутентификации RSA
openssl ciphers -v -tls1 RSA

# Действительные шифры только для TLSv1, использующие
# алгоритм обмена ключами/аутентификации RSA
# исключая экспортируемые шифры
openssl ciphers -v -tls1 RSA:!EXP
# ПРИМЕЧАНИЕ: в некоторых оболочках знак ! требует экранирования
openssl ciphers -v -tls1 RSA:\!EXP

# То же, что и предыдущая команда, но исключаются NULL-шифры
openssl ciphers -v -tls1 RSA:!EXP:!NULL
# ПРИМЕЧАНИЕ: в некоторых оболочках знак ! требует экранирования
openssl ciphers -v -tls1 RSA:\!EXP:\!NULL

# Действительные шифры только для TLSv1, использующие
# алгоритм обмена ключами/аутентификации RSA
# только экспортируемые шифры
openssl ciphers -v -tls1 RSA:EXP
# ИЛИ
openssl ciphers -v TLSv1+RSA:EXP
```

В атрибуте olcTLSCipherSuite (директиве TLSCipherSuite) в качестве cipher-list могут указываться либо общие параметры (используемые в команде openssl ciphers), тогда для получения списка конкретных шифров может быть использована команда openssl, как показано выше (в этом случае порядок предпочтения шифров определяется openssl), либо явно заданный список шифров в порядке их предпочтения. Одно или несколько поддерживаемых значений в cipher-list должны поддерживаться клиентом TLS. Алгоритм поиска совпадений шифров (выбора того, какой шифр будет использован) таков: первый (наиболее предпочтительный) шифр из предоставляемых клиентом, который также поддерживается и сервером, становится шифром соединения (сессии). В следующих примерах используется только подмножество TLSv1 (SSLv3):

```
# Cipher-list содержит только основанные на RSA
# шифры аутентификации и обмена ключами
# поддерживаемые TLSv1 (и SSLv3)

# форма OLC (cn=config)
olcTLSCipherSuite: TLSv1+RSA

# форма slapd.conf
TLSCipherSuite TLSv1+RSA

# Cipher-list содержит только основанные на RSA
# шифры аутентификации и обмена ключами
# поддерживаемые TLSv1 (и SSLv3)
# исключая экспортируемые и NULL-шифры
# форма OLC (cn=config)
olcTLSCipherSuite: TLSv1+RSA:!EXPORT:!NULL

# форма slapd.conf
TLSCipherSuite TLSv1+RSA:!EXPORT:!NULL

# Упорядоченный список основанных на RSA
# шифров аутентификации и обмена ключами
# форма OLC (cn=config)
olcTLSCipherSuite: DES-CBC-SHA:DES-CBC3-SHA:RC4-SHA:RC4-MD5

# форма slapd.conf
TLSCipherSuite DES-CBC-SHA:DES-CBC3-SHA:RC4-SHA:RC4-MD5

# Все шифры за исключением NULL-шифров
# форма OLC (cn=config)
olcTLSCipherSuite: ALL:!NULL

# форма slapd.conf
TLSCipherSuite ALL:!NULL

# Значение по умолчанию, эквивалентно случаю
# когда директива не задана
# форма OLC (cn=config)
olcTLSCipherSuite: ALL
```



```
# форма slapd.conf
TLSCipherSuite ALL
```

Примечание: OpenSSL поддерживает ряд шифров, в результате применения которых может произойти NULL-шифрование данных большого объема и имитовставок MAC (если этот шифр будет единственным, который взаимно поддерживается и сервером, и клиентом). Это означает, что безопасно выполняется только процесс аутентификации, а вся последующая передача данных происходит в открытом виде. Чтобы предотвратить использование таких шифров, используйте значение !NULL в cipher-list или явно задайте список шифров, не включая в них NULL-шифры.



olcTLSRandFile (TLSRandFile)

```
# форма OLC (cn=config)
olcTLSRandFile: /path/to/random/file/name

# форма slapd.conf
TLSRandFile /path/to/random/file/name
```

Атрибут (директива) сервера TLS. Большинство *nix систем, таких как Linux и BSD, поддерживают либо /dev/urandom, либо /dev/random, поэтому данный атрибут (директива) обычно не требуется. Также можно использовать этот атрибут (директиву) для указания альтернативного источника случайных данных в случае, когда к ним выдвигаются специальные требования, либо /dev/urandom (/dev/random) невозможно прочитать, поскольку он занят другими приложениями. Если директива определяется, она указывает файл, содержащий случайные данные. Альтернативные формы, приемлемые для OpenSSL (который используется в реализации TLS-функций OpenLDAP), можно найти в [OpenSSL FAQ](#).



olcTLSHDParamFile (TLSEphemeralDHParamFile)

```
# форма OLC (cn=config)
olcTLSHDParamFile: /path/to/file

# форма slapd.conf
TLSEphemeralDHParamFile /path/to/file
```

Атрибут (директива) сервера TLS. Указывает файл, содержащий параметры для обмена ключами по алгоритму Диффи-Хеллмана (Diffie-Hellman ephemeral key exchange). Данная директива требуется только когда в файле [olcTLSCertificateFile/TLSCertificateFile](#) содержится сертификат DSA/DSS (и [olcTLSCertificateKeyFile/TLSCertificateKeyFile](#) указывает на ключ DSA/DSS). Параметры можно сгенерировать следующей командой:

```
openssl dhparam [-dsaparam] -out <filename> <numbits>
```

Дополнительную информацию о параметрах DH (**dhparam**) можно получить на [сайте openssl](#).



olcTLSVerifyClient (TLSVerifyClient)

```
# форма OLC (cn=config)
olcTLSVerifyClient: never | allow | try | demand

# форма slapd.conf
TLSVerifyClient never | allow | try | demand
```

Атрибут (директива) сервера TLS. TLS позволяет производить как аутентификацию только сервера (только у сервера есть сертификат X.509), так и обоюдную аутентификацию (и у клиента, и у сервера есть сертификаты X.509). Атрибут **olcTLSVerifyClient** (директива **TLSVerifyClient**) определяет, каким образом сервер TLS будет осуществлять обоюдную аутентификацию на основе сертификатов X.509. Значение (по умолчанию) **never** указывает, что сервер никогда не будет запрашивать сертификат клиента. Значение

allow указывает, что сервер будет запрашивать сертификат клиента, и если таковой не будет предоставлен, сессия будет продолжена нормальным образом; если же клиентский сертификат предоставлен, но не может быть проверен (не удалось найти соответствующий корневой (CA) сертификат), то данный клиентский сертификат будет проигнорирован и сессия будет продолжена нормальным образом. Значение **try** указывает, что сервер будет запрашивать сертификат клиента, и если таковой не будет предоставлен, сессия будет продолжена нормальным образом; если же клиентский сертификат предоставлен, но не может быть проверен (не удалось найти соответствующий корневой (CA) сертификат), то сессия будет прервана. Значение **demand** указывает, что сервер будет запрашивать сертификат клиента, и если таковой не будет предоставлен, сессия будет прервана; если клиентский сертификат не может быть проверен (не удалось найти соответствующий корневой (CA) сертификат), то сессия также будет прервана. Значение по умолчанию — **never**.



6.4 Директивы раздела backend (OLC (cn=config) и slapd.conf)

backend

Формат:

```
backend type
```

Директива **backend** определяет название механизма манипуляции данными, который будет использоваться. В конфигурации OLC (cn=config) механизм манипуляции данными определяется записью в форме `olcBackend={Z}type,cn=config` (где Z представляет собой порядковый номер данной записи в упорядоченном множестве таких записей), использующей объектный класс [olcBackendConfig](#). Список поддерживаемых OpenLDAP (slapd) значений **type**:

Тип	Описание
bdb	База данных Berkeley — предпочитаемый механизм OpenLDAP.
dnssrv	Это не база данных, а механизм, позволяющий slapd генерировать отсылки путём опроса записей DNS на основании указанного суффикса.
ldap	Позволяет LDAP следовать по отсылкам (разрешать отсылки), вместо того, чтобы просто возвращать отсылки клиенту.
ldbm	LDAP DBM — может использоваться любая из баз данных BDB, GNU DBM, MDBM или NDBM. Механизм считается устаревшим, начиная с OpenLDAP 2.4+.
meta	Позволяет LDAP следовать по отсылкам (разрешать отсылки), вместо того, чтобы просто возвращать отсылки клиенту. Расширенная версия механизма ldap , позволяющая использовать несколько серверов, а также маскировку контекстов именования (DIT).
monitor	Механизм LDAP, позволяющий получать статусную информацию slapd по протоколу LDAP.
null	Эквивалент /dev/null для LDAP.
passwd	Очень специфическая конфигурация, позволяющая делать поисковые запросы к системному файлу passwd .
perl	Позволяет LDAP проецировать запросы напрямую на объекты PERL.
shell	Позволяет LDAP проецировать запросы в команды оболочки (shell) и запускать внешние программы — API для бедных.
sql	Позволяет LDAP использовать ODBC для проецирования LDAP-запросов в реляционные СУБД — оболочка для реляционных СУБД.

Примеры:

```
backend bdb
# будут использоваться базы данных Berkeley (sleepy cat)
```

Примечание: Нет строгой необходимости в присутствии данной записи (директивы) — можно обойтись без

неё и конфигурации OLC (cn=config) и slapd.conf останутся вполне жизнеспособными, тем более в том случае, если наша конфигурация OLC (cn=config) (или slapd.conf) вообще не использует механизмов манипуляции данными. Поскольку данная директива должна указываться непосредственно перед определением записей/директив database того же самого типа, наличие двух практически одинаковых директив может вызвать путаницу. Возможно, разработчикам OpenLDAP пришла в голову идея указания 'общих параметров', которые будут применяться ко всем разделам databases одного типа, что избавляет от необходимости печатать одно и то же (хорошая мысль). Однако, поскольку в большинстве систем LDAP используется одно-единственное DIT (и, следовательно, одна запись/раздел database), практической пользы от записи/директивы backend немного. Используемый в OLC (cn=config) объектный класс, по существу, представляет собой пустую заглушку.



6.5 Директивы раздела database (OLC (cn=config) и slapd.conf)

Атрибуты (директивы) в разделе database состоят из общих атрибутов (директив), которые касаются почти всех типов данных и специфичных атрибутов (директив), которые относятся к определённому типу, например, bdb.

olcAccess (access to)
Запись (директива) database
olcDbIndex (index)
olcMirrorMode (mirrormode)
Запись (директива) overlay
olcReadOnly (readonly)
replica
repllogfile
olcRootDN (rootdn)
olcRootPW (rootpw)
olcSuffix (suffix)
olcSyncrepl (syncrepl)
updatedn
olcUpdateRef (updateref)

Запись (директива) database

Формат:

```
# форма slapd.conf
database type
```

Запись (директива) **database** указывает на начало определения DIT, а также то, каким образом данное DIT будет отображаться в файловой системе.

В конфигурации OLC (cn=config) базы данных определяются записями в форме **olcDatabase={Z}type,cn=config**. Описание макета/организации cn=config DIT смотрите [здесь](#). Описание добавления/удаления записей database смотрите [здесь](#).

Список поддерживаемых OpenLDAP (slapd) значений **type**:

Тип	Описание
bdb	База данных Berkeley — предпочитаемый механизм OpenLDAP. Директивы, специфичные для bdb . Имена динамически загружаемых модулей — back_bdb.la или back_bdb.so. Будьте осторожны: судя по всему, есть тенденция добавлять номер версии к имени загружаемого модуля, например, back_bdb-2.4.so
config	Специальная запись, позволяющая производить конфигурирование OpenLDAP во время исполнения с помощью LDAP-браузера или файлов LDIF. Использует DIT с жёстко установленным именем cn=config. О настройках и использовании cn=config читайте здесь .
dnssrv	Это не база данных, а механизм, позволяющий slapd генерировать отсылки путём опроса записей DNS на основании указанного суффикса. Директивы, специфичные для dnssrv .

hdb	База данных Berkeley — предпочитаемый механизм OpenLDAP. Абсолютно то же самое, что и bdb, только используется иерархическое размещение данных. Если Вам часто приходится выполнять переименование целых поддеревьев — это механизм манипуляции данными для Вас! Если Вы не поняли, о чём говорится в последнем предложении — используйте bdb. Директивы, специфичные для hdb (те же самые, что и для bdb) . Имена динамически загружаемых модулей — back_hdb.la или back_hdb.so. Будьте осторожны: судя по всему, есть тенденция добавлять номер версии к имени загружаемого модуля, например, back_hdb-2.4.so
ldap	Позволяет LDAP следовать по отсылкам (разрешать отсылки), вместо того, чтобы просто возвращать отсылки клиенту. Директивы, специфичные для ldap . Имена динамически загружаемых модулей — back_ldap.la или back_ldap.so. Будьте осторожны: судя по всему, есть тенденция добавлять номер версии к имени загружаемого модуля, например, back_ldap-2.4.so
ldbm	LDAP DBM — может использоваться любая из баз данных BDB, GNU DBM, MDBM или NDBM. Директивы, специфичные для ldbm . Механизм считается устаревшим, начиная с OpenLDAP 2.4.
meta	Позволяет LDAP следовать по отсылкам (разрешать отсылки), вместо того, чтобы просто возвращать отсылки клиенту. Расширенная версия механизма ldap , позволяющая использовать несколько серверов, а также маскировку контекстов именования (DIT). Директивы, специфичные для meta . Имена динамически загружаемых модулей — back_meta.la или back_meta.so. Будьте осторожны: судя по всему, есть тенденция добавлять номер версии к имени загружаемого модуля, например, back_meta-2.4.so
monitor	Механизм LDAP, позволяющий получать статусную информацию slapd по протоколу LDAP. Директивы, специфичные для monitor . Имена динамически загружаемых модулей — back_monitor.la или back_monitor.so. Будьте осторожны: судя по всему, есть тенденция добавлять номер версии к имени загружаемого модуля, например, back_monitor-2.4.so
null	Ничего. Эквивалент /dev/null для LDAP.
passwd	Очень специфическая конфигурация, позволяющая делать поисковые запросы к системному файлу passwd .
perl	Позволяет LDAP проецировать запросы напрямую на объекты PERL.
shell	Позволяет LDAP проецировать запросы в команды оболочки (shell) и запускать внешние программы — API для бедных.
sql	Позволяет LDAP использовать ODBC для проецирования LDAP-запросов в реляционные СУБД — оболочка для реляционных СУБД. Директивы, специфичные для sql .

Примеры:

```
# форма slapd.conf
database bdb
# будет использоваться база данных Berkeley (sleepy cat)
```



olcMirrorMode (mirrormode)

```
# форма OLC (cn=config)
olcMirrorMode: TRUE | FALSE

# форма slapd.conf
mirrormode TRUE | FALSE
```

Атрибут **olcMirrorMode** (директива **mirrormode**) теоретически используется для указания, что база данных запущена в режиме зеркала ("mirrormode") — вариант репликации с несколькими главными серверами (multi-mastering), разработанный до появления в 2.4+ разнонаправленной репликации с несколькими главными серверами (N-way multi-mastering). В конфигурации с несколькими главными серверами должен присутствовать атрибут **olcMirrorMode: TRUE** (директива **mirrormode TRUE**). В случае slapd.conf данная директива должна быть определена во всех разделах database главного сервера, выставляемых для репликации, ПОСЛЕ ВСЕХ ДИРЕКТИВ SYNCREPL в каждом из разделов database. Это необходимо, чтобы разрешить запись в эти разделы database. Во всех случаях (slapd.conf и OLC (cn=config)), если не добавить данную директиву, то операции записи в DIT, выставляемое для репликации с несколькими главными

серверами (у которого сразу определены [атрибут olcSyncRepl](#) (директива `syncrepl`) и [запись](#) (директива `overlay syncprov`), будут завершаться неудачей. Примеры:

```
# пример размещения директивы в slapd.conf
# при репликации с несколькими главными серверами
# директивы глобальных настроек
...

# фрагмент раздела DIT (database)
database bdb
...
# реплика от первого главного сервера (поставщика)
syncrepl
...
# реплика от второго главного сервера (поставщика)
syncrepl
...
# собственное определение поставщика
overlay syncprov
...
# после BCEX директив syncrepl
mirrormode TRUE
```



olcOverlay (overlay)

Запись (директива) `overlay` указывает на использование наложения (или расширения). В случае конфигурации `slapd.conf` она определяется в разделе `database` и всегда должна указываться после всех остальных директив раздела `database`. За каждой директивой `overlay` следуют одна или несколько директив, специфичных для данного наложения, которые описаны в документации к наложению (смотрите ниже). Для каждого раздела `database` могут быть определены несколько директив `overlay` (за каждой из которых следует одна или несколько директив, специфичных для конкретного наложения). В этом случае они будут применяться в обратном (от их фактического указания в `slapd.conf`) порядке — то есть наложение, определённое последним, будет применено первым.

В конфигурации OLC (`cn=config`) каждое наложение — это дочерняя запись по отношению к той записи `database`, к которой данное наложение будет применяться. Эта запись имеет форму

olcOverlay = { Z } name, olcDatabase = { z } type, cn = config. Описание макета/организации `cn=config` DIT смотрите [здесь](#). Описание добавления/удаления записей `overlay` смотрите [здесь](#).

Формат директивы `overlay` в `slapd.conf`:

```
# форма slapd.conf
overlay overlay-name

# пример с двумя директивами overlay
# (наложение accesslog применяется первым)

overlay syncprov
syncprov-checkpoint 10 100
...
overlay accesslog
logdb cn=accesslog
...
```

Далее приведён неполный список имён наложений (полный список можно найти в `man`-странице `slapd.overlays` для версий 2.4+):

- [accesslog](#) Ведение журнала доступа. Сохраняет сведения об изменениях DIT в виде серии записей в отдельном DIT, к которому потом можно обратиться. Данное наложение требуется при репликации типа [delta-syncrepl](#). Параметры и настройки данного наложения [здесь](#). Дополнительную информацию можно получить в `man`-странице `slapo-accesslog`. Имена динамически загружаемых модулей — `accesslog.la` или `accesslog.so`.
- [chain](#) Сцепление (chaining). Позволяет разрешать (или сцеплять) объекты `referral` в DIT, таким образом клиенту может возвращаться результат запроса, а не просто отсылка (подробнее об этом [здесь](#)). Параметры и настройки данного наложения [здесь](#). Дополнительную информацию можно получить в `man`-странице `slapo-chain`. Имена динамически загружаемых модулей — `chain.la` или `chain.so`.

- dynlist** Динамические группы. Нестандартная (нет соответствующего RFC) функция, получившая, однако, широкое распространение. С её помощью можно динамически создавать группы на основе LDAP-запросов (в отличие от статических групп, использующих объектный класс [groupOfNames](#)). Параметры и настройки данного наложения [здесь](#). Примеры конфигурации и использования [здесь](#). Дополнительную информацию можно получить в map-странице slapo-dynlist. Имена динамически загружаемых модулей — dynlist.la или dynlist.so.
- pcache** Кэширующий прокси. Позволяет серверу LDAP принимать запросы от клиентов и перенаправлять их на несколько других серверов LDAP. Параметры и настройки данного наложения [здесь](#). Дополнительную информацию можно получить в map-странице slapo-pcache. Имена динамически загружаемых модулей — pcache.la или pcache.so.
- ppolicy** Политики паролей. Предоставляет механизмы контроля за использованием паролей: контроль устаревания пароля, обязательное переназначение пароля и т.п. Параметры и настройки данного наложения [здесь](#). Дополнительную информацию можно получить в map-странице slapo-ppolicy. Имена динамически загружаемых модулей — rpolicy.la или rpolicy.so.
- rwm** Перезапись DN. Предоставляет возможность перезаписи DN перед использованием. Параметры и настройки данного наложения [здесь](#). Дополнительную информацию можно получить в map-странице slapo-rwm.
- syncprov** Контроль за обеспечением управления синхронизацией на стороне сервера-поставщика syncrepl. Данное наложение требуется на главном сервере (поставщике) репликации в стиле [syncrepl](#), потребители которого настраиваются [атрибутом olcSyncrepl](#) (директивой [syncrepl](#)). Параметры и настройки данного наложения [здесь](#). Дополнительную информацию можно получить в map-странице slapo-syncprov. Имена динамически загружаемых модулей — syncprov.la или syncprov.so.

olcReadOnly (readonly)

```
# форма OLC (cn=config)
olcReadOnly: TRUE | FALSE

# форма slapd.conf
readonly on | off
```

Атрибут olcReadOnly (директива readonly) запрещает выполнение запросов на модификацию (в ответ на них будет возвращаться "не желают исполнять" ("unwilling to perform")), переводя DIT в режим "только для чтения". Если DIT работает как подчинённое (смотрите информацию и примеры конфигурации репликации [здесь](#)), то такое DIT автоматически переводится в режим "только для чтения" на всех серверах, за исключением тех, которые указаны в [атрибуте olcUpdateDN](#) (директиве [updatedn](#)) (при репликации в стиле slurp) или тех, на которых не определялось [атрибута olcSyncrepl](#) (директивы [syncrepl](#)) (при репликации в стиле syncrepl). В этом случае данная директива НЕ ДОЛЖНА присутствовать. Пример:

```
# запретить операции модификации для этого DIT
# форма OLC (cn=config)
olcReadOnly: TRUE

# форма slapd.conf
readonly on
```

replica

Устарела с выходом OpenLDAP 2.4, в конфигурациях OLC (cn=config) поддерживается только в целях переконвертации slapd.conf. Директива replica используется (совместно с директивой [repllogfile](#)) для описания одного или нескольких подчинённых серверов, на которые производится репликация DIT в стиле slurpd. Данная директива может присутствовать только в определении ГЛАВНОЙ копии DIT (раздел database) на главном сервере репликации. В директиве replica указываются расположение и особенности доступа к каждой ПОДЧИНЁННОЙ копии данного DIT. Дополнительная информация и примеры конфигурации

репликации [здесь](#). Директиву `replica` можно определять только при настройке OpenLDAP вплоть до версии 2.3 (при использовании репликации в стиле `slurpd`). В версии 2.4 директива `replica` стала устаревшей и была заменена на [атрибут `olcSyncrepl` \(директиву `syncrepl`\)](#). Общий формат директивы `replica`:

```
replica uri=ldap[s]://hostname[:port]
[bindmethod={simple|kerberos|sasl}]
[binddn="DN"]
[saslmech=mech]
[authcid=identity]
[authzid=identity]
[credentials=password]
[srvtab=filename]
```

В параметре `uri` указывается схема, хост и, опционально, порт сервера, на котором расположен подчинённый экземпляр DIT. В качестве имени хоста (**hostname**) можно указать как доменное имя, так и IP-адрес. Если порт (**port**) не задан, используются стандартные номера портов LDAP: 389 для схемы `ldap` или 636 — для `ldaps`.

В параметре `binddn` указывается DN (строка в кавычках), от имени которого происходит подключение к подчинённому серверу при внесении изменений в подчинённую копию DIT. Это должно быть DN с правами на чтение и запись в базу данных подчинённого `slapd`, а также оно должно совпадать с DN, указанным в директиве [updatedn](#) файла `slapd.conf` на подчинённом сервере. Из соображений безопасности не рекомендуется, чтобы данный DN совпадал с `rootdn` главной базы данных (тем не менее, это возможно). Если `binddn` — это не `rootdn`, в таком случае он должен указывать на действительную запись в DIT с атрибутом `userPassword` (или `authPassword`).

Значением параметра `bindmethod` может быть **simple**, **kerberos** или **sasl**. Этот параметр указывает метод аутентификации, который будет использоваться при обновлении подчинённой копии DIT. Простая (`simple`) аутентификация не рекомендуется, если не были соблюдены адекватные меры защиты целостности и конфиденциальности данных (например, TLS или IPSEC). Для простой аутентификации требуется определение параметров **binddn** и **credentials**.

Аутентификация `kerberos` считается устаревшей в пользу механизмов аутентификации SASL, таких как `KERBEROS_V4` и `GSSAPI`. Для аутентификации `kerberos` требуется определение параметров **binddn** и **srvtab**. Подробнее эту тему мы раскрывать не будем.

В общем случае рекомендуется аутентификация SASL. Аутентификация SASL требует указания механизма аутентификации в параметре `saslmech`. В зависимости от данного механизма может потребоваться определение аутентификационной идентификационной сущности и/или сведений для проверки полномочий с помощью, соответственно, параметров `authcid` и `credentials`. Параметр `authzid` может использоваться для указания авторизационной идентификационной сущности.

Если присутствует несколько подчинённых серверов, директива **replica** должна быть определена для каждого экземпляра подчинённого сервера.

Примеры:

```
# простой уровень безопасности, подключение к подчинённому
# серверу ldap-repl.example.com с паролем в открытом виде
replica uri=ldap://ldap-repl.example.com bindmethod=simple
binddn="uid=admin,ou=admin,dc=example,dc=com" credentials=guess

# простой уровень безопасности, подключение к подчинённому
# серверу ldap-repl.example.com с паролем в открытом виде
replica uri=ldap://ldap-repl.example.com bindmethod=simple
binddn="uid=admin,ou=admin,dc=example,dc=com"
credentials=dirtysecret
```



repllogfile

Устарела с выходом OpenLDAP 2.4, в конфигурациях OLC (`cn=config`) поддерживается только в целях переконвертации `slapd.conf`. Директива `repllogfile` используется (совместно с директивой [replica](#)) для описания одного или нескольких подчинённых серверов, на которые производится репликация DIT в стиле `slurpd`. Данная директива может присутствовать только в определении ГЛАВНОЙ копии DIT (раздел `database`) на

главном сервере репликации. Директива `replogfile` задаёт расположение файла, используемого для хранения обновлений каталога, которые `slurpd` будет рассылать на один или несколько подчинённых серверов. Дополнительная информация и примеры конфигурации репликации [здесь](#). Директиву `replogfile` можно определять только при настройке OpenLDAP вплоть до версии 2.3 (при использовании репликации в стиле `slurpd`). В версии 2.4 директива `replogfile` стала устаревшей, для неё нет эквивалентной директивы при репликации в стиле [syncrepl](#). Общий формат директивы `replogfile`:

```
replogfile path/to/logfile
```

Здесь путь **`path/to/logfile`** может быть абсолютным или относительным.

При репликации в стиле `slurpd` данная директива считывается демоном `slurpd` для определения источника транзакций, которые он будет рассылать на подчинённые серверы, указанные в директивах `replica`. `slurpd` вносит изменения в файл, указанный данной директивой, поэтому должны быть установлены соответствующие права доступа к файлу.

Примеры:

```
# относительный формат пути
replogfile ../log/slavedit.log

# абсолютный формат пути
replogfile /var/log/ldap/slavedit.log
```

Директива `replogfile` может также использоваться без соответствующей директивы [replica](#) и без запуска `slurpd`. В этом случае просто генерируется журнал транзакций. Такой файл требует периодического обслуживания, поскольку имеет тенденцию быстро разрастаться.



olcRootDN (rootdn)

Определяет **DN root** (несчастливый и вводящий в заблуждение в данном контексте термин) или **администратора** для данного DIT. При подключении от имени **rootdn** автоматически предоставляются права производить запись во все записи, объектные классы и атрибуты данного DIT — глобальные и локальные ACL не применяются (обходятся любые правила, определённые в [атрибутах olcAccess \(директивах access\)](#)). Если директива **rootdn** отсутствует, права **администратора** не предоставляются. Как правило, **rootdn** требуется на этапе первоначального построения каталога, но после его запуска в работу **rootdn** может быть удалена по соображениям безопасности. Для **olcRootDN/rootdn** разрешены только подключения к LDAP с проверкой подлинности.

Запись администратора "**волшебная**": если пароль для неё определен в **olcRootPW/rootpw**, то существование в каталоге нормальной [записи](#) с таким DN не требуется, то есть нет необходимости делать эту запись видимой в структуре DIT. По соображениям безопасности пользователь может решить не определять соответствующий атрибут **olcRootPW** (директиву **rootpw**). В таком случае, поскольку пароль всегда требуется для получения привилегий **rootdn/администратора**, этот пароль должен храниться в записи DIT, на которую указывает **olcRootDN/rootdn**. Таким образом, в этом случае данная запись должна быть видимой в структуре DIT и иметь корректный атрибут пароля, например, **userPassword** из объектного класса **person**.

DN, указываемый в **olcRootDN/rootdn**, как правило, находится в [пространстве имён](#) того DIT, в записи (разделе) **database** которого он определяется. Так, если суффикс (**olcSuffix/suffix**) DIT — `dc=example,dc=com`, то **olcRootDN/rootdn** может быть любым DN в пространстве имён `dc=example,dc=com`, к примеру, `cn=anything,dc=example,dc=com`. Если же DN, указываемый в **olcRootDN/rootdn**, находится вне пределов пространства имён данного DIT, то, в зависимости от других параметров конфигурации, при попытке подключения от имени этого DN может быть сгенерирована отсылка в соответствующий DIT для получения пароля. Необязательный [атрибут olcRootPW \(директива rootpw\)](#) обеспечивает аутентификацию при подключении от имени **olcRootDN/rootdn**.

Во многих дистрибутивах есть файл-пример, в котором **olcrootDN/rootdn** определён как `"cn=Manager,dc=example,dc=com"` или что-то в этом роде. Выбор имени `cn=Manager` совершенно произволен — с таким же успехом это может быть **cn=gobbledegook**.

```
# форма OLC (cn=config)
olcSuffix: dc=example,dc=com
# olcRootDN в пространстве имён данного DIT
olcRootDN: cn=joeschmo,dc=example,dc=com

# форма slapd.conf
suffix "dc=example,dc=com"
# rootdn DN в пространстве имён данного DIT
rootdn "cn=joeschmo,dc=example,dc=com"
```

olcRootPW (rootpw)

Определяет пароль, который будет применяться к учётной записи **root** или **администратора** того **DIT**, которое описывается в данной записи (разделе) **database**. Эта директива имеет эффект лишь в том случае, когда **olcRootDN/rootdn** находится в **пространстве имён** данной записи (раздела) **database**. Пароль может быть определён как в открытом виде, так и, в целях его защиты, в формате хэш-пароля (создаётся утилитой **slappasswd**). Пароль можно дополнительно заключать в кавычки для удобства или на случай, если в нём есть пробельные символы. Следует отметить, что использование хэша — это всего лишь способ защиты пароля от непреднамеренной утечки. В качестве альтернативного метода (только для slapd.conf) можно установить права доступа к файлу slapd.conf так, чтобы только у группы ldap (обычно пользователь ldap и группа ldap) были права на чтение с маской доступа 0740 или ниже (slapd автоматически обеспечивает корректные разрешения на slapd.conf, изменяя их при загрузке в случае необходимости). Даже если пароль в OLC (cn=config) или slapd.conf защищён с помощью хэша, в случае, когда LDAP-клиент посылает по сети пароль администратора для проверки подлинности, он отправляется в открытом виде, затем к нему применяется соответствующий механизм хэширования, и результат сравнивается со значением, хранящимся в **olcRootPW / rootpw**. Таким образом, пароль по-прежнему уязвим к перехватам по сети (**TLS снимает данную угрозу**).

Если атрибут olcRootPW (директива rootpw) опущен, то могут быть использованы другие методы обеспечения безопасности (например, SASL). Примеры:

```
# форма OLC (cn=config)
olcRootDN: cn=good,dc=example,dc=com
# простая версия с паролем в открытом виде
# не ставьте пробелов после пароля, если они действительно не нужны!
olcRootPW: secret
# версия с ограничителями
olcRootPW: "secret"
# версия с хэшем SSHA
olcRootPW: {SSHA}HZvHCuZL/YcwFpi/WkqVWASiCxQnEzfw
# то же с ограничителями
olcRootPW: "{SSHA}HZvHCuZL/YcwFpi/WkqVWASiCxQnEzfw"

# форма slapd.conf
rootdn "cn=good,dc=example,dc=com"
# простая версия с паролем в открытом виде
# не ставьте пробелов после пароля, если они действительно не нужны!
rootpw secret
# версия с ограничителями
rootpw "secret"
# версия с хэшем SSHA
rootpw {SSHA}HZvHCuZL/YcwFpi/WkqVWASiCxQnEzfw
# то же с ограничителями
rootpw "{SSHA}HZvHCuZL/YcwFpi/WkqVWASiCxQnEzfw"
```

Примечания:

1. О том, как добавить хэшированный пароль в OLC (cn=config) или файл slapd.conf, смотрите примечания к описанию утилиты **slappasswd** в соответствующей **главе** данного руководства.
2. У многих алгоритмов хэширования, поддерживаемых **slappasswd**, есть версия с "солью". В данном случае "соль" — добавление некоторого количества случайных данных к паролю, что значительно снижает эффективность применения атак, основанных на подборе по словарю. В общем случае, если используется хэширование, нужно выбирать версии алгоритмов хэширования с "солью".
3. Чтобы защитить пароль при передаче по сети, используйте **либо методы SASL, либо TLS/SSL**.

olcSuffix (suffix)

Суффикс (suffix) — один из многих путанных терминов в LDAP (известный также как [root](#) или [base](#)). Атрибут `olcSuffix` (директива `suffix`) указывает [Distinguished Name](#) узла самого верхнего (корневого) уровня в [DIT](#), описываемого в данной записи (разделе) **database**. У Вас может быть [несколько DIT](#), каждый из которых описывается в отдельной записи (разделе) **database**. **Примечание:** Суффиксы для баз данных [config](#) и [monitor](#) жестко установлены соответственно в `cn=config` и `cn=monitor`, и не могут быть изменены с помощью атрибута **olcSuffix** (директивы `suffix`). Именованное корневое DN может вызвать [головную боль](#).

```
# форма OLC (cn=config)
# формат RFC 2377
olcSuffix: dc=example,dc=com

# формат X.500
olcSuffix: ou=example,c=us

# форма slapd.conf
# формат RFC 2377
suffix "dc=example,dc=com"

# формат X.500
suffix "ou=example,c=us"
```



olcSyncrepl (syncrepl)

Доступна начиная с версий 2.2+. Указывает, что текущая база данных (DIT) является репликой, синхронизирующейся с содержимым другого DIT, путём придания данному серверу статуса [потребителя репликации](#) и запуска механизма репликации `syncrepl`. Соответствующий главный сервер ([поставщик репликации](#)) настраивается с помощью записи (директивы) `overlay syncprov`. Содержимое реплики синхронизируется с содержимым главного сервера по протоколу LDAP Content Synchronization protocol ([RFC 4533 \(рус.\)](#), [RFC 4533 \(ориг.\)](#)). Начиная с версии 2.4, OpenLDAP поддерживает оба классических варианта репликации: главный-подчинённый сервер (Master-Slave) и несколько главных серверов (Multi-Master). Дополнительная информация и примеры конфигурации [здесь](#).

```
# форма OLC (cn=config)
olcSyncrepl:

# форма slapd.conf
syncrepl
[attrs= attr list>]
[attrsonly]
[authcid=identity]
[authzid= identity]
[binddn=dn]
[bindmethod=simple|sasl]
[credentials=passwd]
[filter=filter str]
[interval=dd:hh:mm:ss]
[logbase=base DN]
[logfilter=filter str]
provider=ldap[s]://hostname[:port]
[realm=realm]
[retry=[retry-interval num-retries | + ]
rid=replica-ID
[saslmech=mech]
[schemachecking=on|off]
[scope=sub|one|base]
searchbase=base DN
[secprops=properties]
[sizelimit=number-of-entries | unlimited]
[starttls=yes|critical]
[syncdata=default|accesslog|changelog]
[timeLimit=time-in-secs | unlimited]
[type=refreshOnly|refreshAndPersist]
```

Обязательные параметры: `rid`, `provider` и `searchbase`; все остальные не являются обязательными.

bindmethod	Может быть либо simple , либо sasl . Если bindmethod установлен в simple , то должны быть определены параметры binddn (строка в кавычках) и credentials . При использовании simple все данные, в том числе, возможно, и пароли посылаются в открытом виде и, следовательно, могут быть перехвачены при передаче по сети. Если bindmethod установлен в sasl , то должен быть определён параметр saslmech . В зависимости от

этого механизма, аутентификационная идентификационная сущность и/или учётные данные могут быть указаны в параметрах **authcid** и **credentials**. Для указания авторизационной идентификационной сущности может быть использован параметр **authzid**. Специфичные для соединения SASL свойства безопасности (такие как ключевое слово **sasl-secprops**) могут быть заданы параметром **secprops**. Отличный от заданного по умолчанию SASL realm может быть задан параметром **realm**.

binddn Необязательный параметр **binddn** определяет DN (и любые требуемые для аутентификации сведения в зависимости от выбранного метода **bindmethod**), которые позволят получить доступ к **searchbase**. Хотя есть соблазн использовать **rootdn** целевого DIT, этого следует избегать там, где это только возможно. Указываемому в **binddn** пользователю требуются только права на чтение (осуществление поиска) в том DIT или в том фрагменте DIT, который будет реплицироваться. В общем случае, нужно использовать DN с минимально возможным уровнем полномочий, либо создать отдельный ACL, предоставляющий минимальный уровень доступа (только чтение). Если параметр не указан, осуществляется анонимное подсоединение — как-то страшновато.

credentials Параметр **credentials** требуется всегда при **bindmethod=simple** а также, в зависимости от метода SASL, может потребоваться и при **bindmethod=sasl**. Он указывает пароль, который будет использоваться при аутентификации во время подсоединения от имени **binddn**. По умолчанию пароль задаётся в открытом виде (CLEARTEXT), в этом случае он посылается по сети также в открытом виде и, соответственно, может быть перехвачен по сети. Примеры:

```
# Отправка пароля аутентификации в открытом виде
# форма OLC (cn=config)
olcSyncrepl:
    bindmethod=simple
    binddn="cn=goodguy,dc=example,dc=com"
    credentials=dirtysecret
...
# форма slapd.conf
syncrepl
    bindmethod=simple
    binddn="cn=goodguy,dc=example,dc=com"
    credentials=dirtysecret
...
```

interval Этот параметр имеет смысл только при типе репликации **RefreshOnly**. Он определяет промежуток времени между процессами синхронизации. Время задаётся либо в формате dd:hh:mm:ss, либо количеством секунд. Так, параметр *interval=00:01:00:00* (или в секундах *interval=3600*) говорит о том, что попытки синхронизации будут предприниматься с интервалом в 1 час. Значение по умолчанию — 1 день или 86400 секунд.

logbase Требуется только в случае, когда установлен параметр **syncdata=accesslog** и включена **delta-синхронизация** (о том, что это такое, а также примеры конфигурации смотрите [здесь](#)). Параметр определяет суффикс DIT, в котором хранится accesslog (он должен соответствовать значению **logdb**, указанному для наложения **overlay accesslog** на сервере-поставщике, DIT которого будет реплицироваться).

logfilter Требуется только в случае, когда установлен параметр **syncdata=accesslog** и включена **delta-синхронизация** (о том, что это такое, а также примеры конфигурации смотрите [здесь](#)). Параметр определяет поисковый фильтр, используемый при чтении accesslog. Типичное значение logfilter при операциях delta-синхронизации:

```
# форма OLC (cn=config)
olcSyncrepl:
    syncdata=accesslog
    logfilter=(&(objectclass=auditWriteObject)(reqResult=0))
...
# форма slapd.conf
syncrepl
    syncdata=accesslog
    logfilter=(&(objectclass=auditWriteObject)(reqResult=0))
...
```

Объектный класс **auditWriteObject** используется при сохранении [записей журнала](#) в accesslog. Атрибут reqResult=0 указывает на сохранённый код возврата операции; соответственно, значение 0 говорит о том, что в ответ на поисковый запрос будут возвращаться только успешные операции.

provider Указывает сайт поставщика репликации, содержащий главную копию реплицируемого контента, в виде LDAP URI. Если в URI не задан **порт (port)**, используется стандартные порты LDAP (389) или LDAPS (636).

retry Если во время репликации (как в режиме **refreshOnly**, так и в режиме **refreshAndPersist**) возникнет ошибка,

потребитель будет пытаться произвести пересоединение согласно значению параметра **retry**. Это значение представляет собой список пар **интервал между повторами (retry-interval)** и **количество повторов (num-retries)**. Например, `retry="60 10 300 3"` указывает потребителю повторять попытки подключения через каждые 60 секунд первые 10 раз, а затем через каждые 300 секунд ещё 3 раза, после этого остановить попытки подключения. **num-retries** может быть установлено в '+', тогда потребитель будет пытаться подключиться неограниченное количество раз, то есть `retry="300 +"` указывает повторять попытки подключения через каждые 300 секунд бесконечное количество раз.

rid	Идентифицирует текущую директиву <code>syncrepl</code> в составе сервера-потребителя репликации. Это произвольное, уникальное, положительное целое число длиной не более трёх цифр. Так, если на сервере один атрибут <code>olcSyncrepl</code> (директива <code>syncrepl</code>), то для этой цели можно использовать <code>rid=000</code> (или <code>rid=001</code> , или любое число от 1 до 3 цифр), если на сервере более одного атрибута <code>olcSyncrepl</code> (директивы <code>syncrepl</code>), первый из них может иметь <code>rid=000</code> , второй — <code>rid=001</code> , и так далее. Таким образом, параметр <code>rid</code> должен быть уникальным в составе сервера. Связи между значениями <code>rid</code> и <code>olcServerID/ServerID</code> (SID), который требуется в конфигурации репликации с несколькими главными серверами (multi-master), нет.
schemachecking	schemachecking предписывает выполнять стандартные проверки соблюдения схемы данных. Значение по умолчанию — <code>off</code> .
searchbase	Содержимое реплики <code>syncrepl</code> определяется с помощью спецификации поиска LDAP — таким образом для потребителя существует возможность реплицировать любое DIT как целиком, так и частично. Сервер-потребитель отправляет поисковый запрос серверу-поставщику согласно спецификации поиска, которая определяется параметрами searchbase (строка в кавычках), scope (значение по умолчанию — <code>sub</code>), filter (значение по умолчанию — <code>(objectclass=*)</code>), attrs (значение по умолчанию — <code>"*,+"</code> , то есть все пользовательские и операционные атрибуты), attrsonly (значение по умолчанию — <code>false</code>), sizelimit (значение по умолчанию — <code>unlimited</code>), и timelimit (значение по умолчанию — <code>unlimited</code>) Эти параметры имеют тот же смысл, что и в обычной спецификации поискового запроса.
starttls	Параметр starttls определяет использование расширенной операции StartTLS для установки сессии TLS перед тем, как производить подсоединение к поставщику. Если запрос StartTLS завершился неудачей и значение параметра starttls было <code>critical</code> , сессия будет прервана; если же значение параметра starttls было <code>yes</code> , сессия репликации будет продолжена без TLS. Этот параметр требуется указывать только в случае, когда в URL в параметре <code>provider</code> использовалась схема <code>ldap</code> , например, <code>provider=ldap://ldap1.example.com</code> . Если же использовалась схема <code>ldaps</code> , например, <code>provider=ldaps://ldap1.example.com</code> , то определять параметр starttls не требуется, поскольку в этих условиях операция StartTLS выполняется автоматически.
syncdata	Необязательный параметр syncdata используется для включения delta-синхронизации (о том, что это такое, а также примеры конфигурации смотрите здесь). В нём указывается источник изменений, которые будут применяться к реплицируемому DIT. Значения, которые можно использовать: default (delta-синхронизация не используется), changelog (устаревший формат) или accesslog (включается delta-синхронизация). При этом, для указания соответствующего журнала изменений, необходимо задать параметры logbase и logfilter .
type	Протокол LDAP Content Synchronization protocol поддерживает два типа операций. Требуемый тип операции указывается в параметре type . При типе <code>refreshOnly</code> операции синхронизации (поиска) повторяются по расписанию, задаваемому в параметру <code>interval</code> . При завершении сессии синхронизации поставщик возвращает syncCookie и соединение закрывается. По прошествии периода времени, заданного в <code>interval</code> , потребитель инициирует следующую сессию, при этом syncCookie прошлой сессии передаётся для определения диапазона изменений — тогда (если это возможно) потребителю предоставляются только изменения, произошедшие с момента окончания предыдущей сессии. При типе <code>refreshAndPersist</code> начальные этапы сессии такие же, что и при типе <code>refreshOnly</code> , но, после завершения первоначальной синхронизации, соединение не разрывается и изменения (в рамках поискового диапазона синхронизации <code>searchbase</code>) посылаются от поставщика потребителю по мере их возникновения, обеспечивая тем самым практически мгновенное распространение обновлений. При первоначальной репликации syncCookie отсутствует, поэтому диапазон синхронизации будет совпадать с <code>searchbase</code> целиком (обычно это DIT полностью), таким образом можно начать новую репликацию с пустым DIT на сервере-потребителе. Если при этом целевое DIT большого размера, синхронизация может занять значительный промежуток времени.

Механизм репликации `syncrepl` поддерживается только механизмами манипуляции данными `back-bdb` и `back-hdb` (по состоянию на 2.4.31).

Примеры:

Потребитель будет запрашивать обновления с хоста master-ldap.example.com (используя порт по умолчанию 389) с интервалом в 1 час. Если подключиться к поставщику не удалось, потребитель будет повторно пытаться установить соединения через каждые 5 секунд 5 раз, а затем каждые 300 секунд (5 минут) неограниченное количество раз. Реплицируется DIT целиком (подразумевается, что суффикс главной копии DIT dc=example,dc=com). Передаются все записи (параметр filter не задан) со всеми пользовательскими и операционными атрибутами (attrs="*,+"). Используется простое подключение от имени cn=admin,ou=people,dc=example,dc=com с паролем в открытом виде. Дополнительная информация и примеры конфигурации [здесь](#).

```
# форма OLC (cn=config)
olcSuffix: rid=000
provider=ldap://master-ldap.example.com
type=refreshOnly
interval=00:1:00:00
retry="5 5 300 +"
searchbase="dc=example,dc=com"
attrs="*,+"
bindmethod=simple
binddn="cn=admin,ou=people,dc=example,dc=com"
credentials=guess

# форма slapd.conf
syncrepl rid=000
provider=ldap://master-ldap.example.com
type=refreshOnly
interval=00:1:00:00
retry="5 5 300 +"
searchbase="dc=example,dc=com"
attrs="*,+"
bindmethod=simple
binddn="cn=admin,ou=people,dc=example,dc=com"
credentials=guess
```

Дополнительная информация и примеры конфигурации [здесь](#).



updatedn

Считается устаревшей, начиная с OpenLDAP 2.4, в конфигурациях OLC (cn=config) поддерживается только в целях переконвертации slapd.conf. Данная директива применяется только с экземплярами DIT на подчинённом сервере в случае репликации в стиле slurpd для OpenLDAP вплоть до версии 2.3 (в версии 2.4 стала устаревшей и заменена на директиву [syncrepl](#)). Дополнительная информация о репликации и примеры конфигурации [здесь](#).

Директива определяет DN, которому разрешено вносить изменения в реплику.

```
updatedn DN | SASL-identity
```

Аргументом директивы может быть DN, от имени которого при внесении изменений в реплику подключается slurpd (применяется, когда в директиве [replica](#) на главном сервере **bindmethod** установлен в simple), либо DN, ассоциированный с идентификационной сущностью SASL (если в директиве replica **bindmethod** установлен в sasl).

```
# Пример с DN:
updatedn "cn=admin,dc=example,dc=com"

# Пример с SASL:
updatedn "uid=slurpd,cn=example.com,cn=digest-md5,cn=auth"
```



olcUpdateRef (updateref)

Атрибут **olcUpdateRef** (директива **updateref**) применяется только в конфигурации подчинённого сервера (или сервера-потребителя) и используется как при репликации в стиле slurpd, так и в стиле syncrepl. Дополнительную информацию о репликации и примеры конфигурации можно найти [здесь](#). Поскольку подчинённый сервер (или сервер-потребитель) доступен только для чтения, данная директива определяет

URL, который будет возвращаться клиентам, пытающимся выполнить запрос модификации (изменения) к подчинённому экземпляру DIT.

```
# форма OLC (cn=config)
olcUpdateRef: uri

# форма slapd.conf
updateref uri
```

Здесь **uri** указывается в обыкновенном формате `scheme://host[:port]`. Если директива `updateref` определена несколько раз, в возвращаемой отсылке будут указаны все URL.

Примеры:

```
# порт по умолчанию 389
# форма OLC (cn=config)
olcUpdateRef: ldap://ldap.example.com

# форма slapd.conf
updateref ldap://ldap.example.com

# нестандартный порт
# форма OLC (cn=config)
olcUpdateRef: ldap://ldap.example.com:10389

# форма slapd.conf
updateref ldap://ldap.example.com:10389

# ldap-порт по умолчанию для защищённых соединений (636)
# форма OLC (cn=config)
olcUpdateRef: ldaps://ldaps.example.com

# форма slapd.conf
updateref ldaps://ldaps.example.com
```



6.5.7 Директивы `slapd.conf`, специфичные для базы данных `null`

Описание скоро будет.



6.5.8 Директивы `slapd.conf`, специфичные для базы данных `passwd`

Описание скоро будет.



Глава 6. Использование OLC (cn=config) в OpenLDAP

- 6.1.1.1 Обзор OLC (cn=config)
- 6.1.1.2 Переконвертация `slapd.conf` в OLC (cn=config)
- 6.1.1.3 Макет OLC (cn=config)
- 6.1.1.4 Использование OLC (cn=config) (чтение, модификация)
 - 6.1.1.4.1 Общие замечания по конфигурации OLC (cn=config)
 - 6.1.1.4.2 Добавление/удаление наборов схемы данных при использовании OLC (cn=config)
 - 6.1.1.4.3 Добавление/удаление ACP/ACL при использовании OLC (cn=config)
 - 6.1.1.4.4 Добавление/удаление модулей при использовании OLC (cn=config)
 - 6.1.1.4.5 Добавление/удаление баз данных при использовании OLC (cn=config)
 - 6.1.1.4.6 Добавление/удаление наложений при использовании OLC (cn=config)

Обзор `cn=config`

Исторически OpenLDAP настраивался статически, то есть, чтобы произвести изменение, нужно было править конфигурационный файл `slapd.conf`, а затем останавливать и запускать `slapd`. При увеличении количества записей в каталоге перезапуск будет занимать всё более продолжительное время, и, в случае высоконагруженных каталогов, такой метод переконфигурации становится всё более неприемлемым. В OpenLDAP версии 2.3 была представлена опциональная новая возможность, посредством которой конфигурация может быть произведена во время исполнения с помощью модификации записей специального DIT, называемого `cn=config`. У данной возможности несколько разных имён (что создаёт некоторую путаницу), в том числе: **конфигурация времени исполнения (on-line configuration, OLC)** (наше любимое), конфигурация с нулевым временем простоя (zero down-time configuration), `cn=config` и `slapd.d`. Для начала, несколько тезисов:

1. В настоящий момент (OpenLDAP версии 2.4) данная возможность всё ещё остаётся опциональной. Это означает, что конфигурация с помощью `slapd.conf` продолжает работать, хотя формально и считается устаревшей.
2. Исторически сложилось так, что OpenLDAP может резко и неожиданно прекращать поддержку своих старых функций. Это значит, что переход на конфигурацию времени исполнения (OLC) нужно произвести, по возможности, в кратчайшие сроки. Лучше произвести это в спокойных условиях, когда Вам не требуется переходить на новый релиз, чем, экстренно перейдя на новый релиз из-за критической ошибки, Вы будете вынуждены переходить на новый метод конфигурации в авральном режиме.
3. Для управления настройками рабочей среды в конфигурации времени исполнения (OLC) используется конфигурационный DIT с жёстко установленным суффиксом `cn=config`. Концептуально, модификация записей в этом DIT (с помощью LDAP-браузера или файлов LDIF) приводит к немедленному изменению поведения рабочей среды `slapd` без необходимости перезапуска `slapd` (как это приходилось делать после изменения `slapd.conf`).



Переход к использованию OLC (cn=config)

Для создания DIT `cn=config` (OLC) можно ЛИБО создать серию файлов LDIF, описывающих конфигурацию, и применить их с помощью `slapadd`, ЛИБО переконвертировать Ваш существующий файл `slapd.conf`. Если Вы мазохист — мы рекомендуем использовать первый подход. Если же Вы нормальный человек — переконвертируйте `slapd.conf` так, как описано ниже. Переконвертация — это процесс, который выполняется один раз (кроме того, существует возможность отката назад). После того, как Вы переконвертировали файл `slapd.conf`, он больше не требуется. Во время загрузки `slapd` ищет конфигурационную директорию (по умолчанию `slapd.d`), считывает оттуда LDIF-файлы и инициализирует DIT `cn=config` (OLC). Если директория `slapd.d` не найдена, `slapd` ищет файл `slapd.conf`.

Чтобы перейти от `slapd.conf` к конфигурации времени исполнения (`cn=config`), выполните следующие действия:

1. Убедитесь в том, что конфигурационный файл `slapd.conf` является рабочим и отражает необходимую функциональность. Это делается в качестве меры предосторожности. Подразумевается, что Вы разбираетесь в текущей версии конфигурации, основанной на `slapd.conf`. Последнее, что неплохо бы сделать до перехода, — это внедрить те новые функции, которые Вам могут понадобиться, и сразу же их испробовать. Пускай это несколько замедлит процесс перехода, зато Вы приобретёте необходимые новые знания и опыт в спокойной обстановке.
2. Остановите LDAP-сервер.
3. Отредактируйте файл `slapd.conf`, добавив в него следующие строки:

```
# Перед первым определением database
database config
# ПРИМЕЧАНИЕ: суффикс жёстко определён как cn=config
# и директивы suffix НЕ ДОЛЖНО быть
# применяются стандартные правила - rootdn может быть любым DN
# но ДОЛЖЕН быть определён в разделе cn=config
rootdn "cn=admin,cn=config"
# Используйте любой поддерживаемый формат пароля, такой как {SSHA} и т.п.
# или оставьте пароль в открытом виде, как показано
rootpw config
```

Существует много материалов, описывающих тонкости подготовки конвертации в slapd.d (cn=config); по большому счёту, всё что в них написано — правда. Однако, для того, чтобы просто использовать функцию cn=config (читать, добавлять и изменять атрибуты) с помощью, скажем, LDAP-браузера, необходимо и достаточно лишь добавить те строки, которые приведены выше.

4. Переконвертируйте текущий файл slapd.conf в конфигурацию времени исполнения (OLC, cn=config), выполнив следующие команды:

```
# Будем использовать slaptest (наиболее разумный способ),
# однако конвертацию может выполнить любая утилита,
# поддерживающая аргументы -f file и -F dir,
# например, slapcat, slapadd и другие.

# Останавливаем slapd
[fc]/etc/rc.d/init.d/slapd stop
# ИЛИ вручную
killall slapd
# ИЛИ
[bsd]/usr/local/etc/rc.d/slapd.sh stop

[fc]cd /etc/openldap
[bsd]cd /usr/local/etc/openldap
# ОБЯЗАТЕЛЬНО - создаём стандартную директорию по умолчанию
mkdir slapd.d
# Конвертируем slapd.conf
slaptest -f slapd.conf -F slapd.d

# В зависимости от того, под каким пользователем Вы выполняли slaptest
# может потребоваться изменить владельца slapd.d и всех файлов внутри
chown -R ldap:ldap *

# Переименовываем slapd.conf
# Данный шаг не строго обязателен, но полезен
# чтобы убедиться, что используется именно slapd.d
mv slapd.conf slapd.conf.bak

# Запускаем slapd
[fc]/etc/rc.d/init.d/slapd start
# ИЛИ вручную
slapd -u ldap -g ldap

# Если slapd не загрузился, используем
slapd -d -1 -u ldap -g ldap
# и смотрим отладочные сообщения
```

Примечания:

1. В отличие от slapd.conf, для файлов cn=config, содержащихся в slapd.d, требуются права на чтение и запись для пользователя и группы, от имени которых Вы запускаете slapd (обычно ldap). Для конвертации мы запускали slaptest от имени root; таким образом, slapd при запуске не сможет получить доступа к файлам и, по умолчанию, примет настройки из файла slapd.conf, а не из cn=config. Подразумевая, что Вы запускаете slapd от имени пользователя ldap (-u ldap), выполните смену владельца (chown) на ldap:ldap, как было показано. Права доступа на файлы 0600 обеспечивают хороший контроль доступа к файлам и не требуют изменения.
2. Нет строгой необходимости переименовывать файл slapd.conf после переконвертации с помощью slaptest, но это даёт уверенность в том, что мы будем использовать именно cn=config, а не переходить по умолчанию на использование slapd.conf в случае возникновения ошибок при загрузке. Недостатком такого переименования можно считать то, что некоторые стандартные скрипты запуска могут подразумевать присутствие именно файла slapd.conf. Так было со скриптом запуска на нашей FreeBSD (/usr/local/etc/rc.d/slapd.sh) — он аварийно завершал работу, поскольку считал, что файл slapd.conf должен присутствовать. Мы закомментировали строку required_files и строку изменения владельца (chown) slapd.conf в скрипте, и снова воцарился мир и покой.
3. **Будьте осторожны:** Вы можете сконфигурировать OLC (cn=config) так, что он придёт в нерабочее состояние. Мы поменяли rootdn базы данных cn=config с помощью LDAP-браузера с cn=config на cn=admin — неверное изменение, поскольку все элементы конфигурационной базы данных должны заканчиваться (иметь своим корнем) cn=config. Однако, данное изменение было принято. Соединение было немедленно разорвано (как и должно было быть), но мы не смогли вновь подсоединиться ни под каким значением — ни старым, ни новым. Мы остановили и попытались запустить slapd, что тоже закончилось неудачей, поскольку slapd отказался

загружаться с нашим новым модифицированным rootdn (cn=admin). Единственным решением стало редактирование slapd.d/cn=config/olcDatabase={0}config.ldif и восстановление атрибута olcRootDn в cn=config. Затем мы загрузили slapd, с помощью LDAP-браузера изменили атрибут olcRootDn на cn=admin,cn=config и всё работало прекрасно.

rootDn базы данных cn=config может быть любая запись, но она ДОЛЖНА иметь своим корнем cn=config. Например, cn=manager,cn=config будет работать, а cn=manager,cn=admin будет принята, но не даст получить доступ к cn=config и, вследствие этого, slapd не будет загружаться.

4. **Почти переход:** Если Вам хочется посмотреть, что из себя представляет будущее, но морально Вы еще не готовы к переходу, либо Вам просто хочется поэкспериментировать, можете отредактировать slapd.conf как показано выше в шаге 3 (добавить то, что касается database config), а затем, не выполняя переконвертации, показанной в шаге 4, просто загрузите slapd. Вы сможете подключиться к cn=config и изменять атрибуты во время исполнения, однако, когда Вы остановите и вновь запустите slapd, все ваши изменения потеряются. Мы ведь говорили, что это почти переход.
5. Если Вы любознательны, стоит посмотреть файлы и директории, созданные в директории slapd.d после конвертации. Возможно, однажды Вам придётся их восстанавливать или ремонтировать.
5. Для доступа к возможностям OLC (cn=config) из [LDAP-браузера](#) настройте его соответствующим образом (подразумевается, что Вы использовали приведённые выше настройки, в противном случае — измените на свои):

```
# Используются нормальные установки порта и имени хоста
# Значения для LDAPBrowser/Editor
Basedn cn=config
Binddn cn=admin,cn=config
password config

# либо можно использовать ldapsearch
ldapsearch -w config -x -D cn=admin,cn=config -b cn=config
```

6. Если всё пошло не так, либо Вы хотите вернуться к старомодной(!) конфигурации slapd.conf (Вы увидели будущее, но оно Вас пока не устроило), есть возможность всё вернуть:

```
# Останавливаем slapd
killall slapd
# ИЛИ
[fc]/etc/rc.d/init.d/slapd stop

[fc]cd /etc/openldap
[bsd]cd /usr/local/etc/openldap
# Удаляем директорию slapd.d
rm -r slapd.d

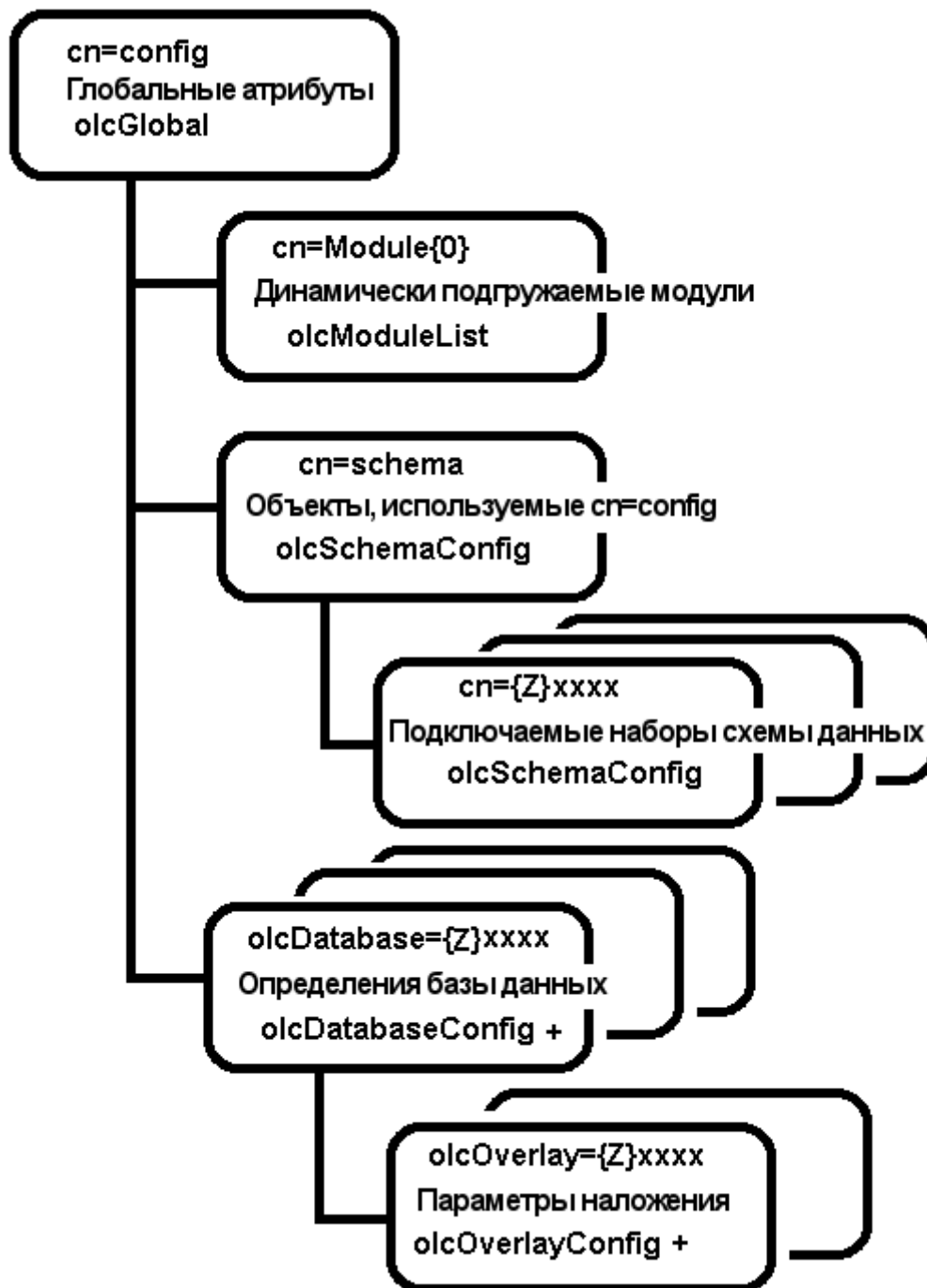
# Если Вы переименовывали slapd.conf, как было
# показано в процедуре переконвертации выше,
# необходимо вновь назвать его slapd.conf
# Закомментируйте наши строки database config
# в slapd.conf
# Теперь перезапустим slapd
[fc]/etc/rc.d/init.d/slapd start
# ИЛИ вручную
slapd -u ldap -g ldap
```

Когда Вы почувствуете себя готовыми и захотите попробовать снова, просто повторите процесс конвертации. Его можно повторять сколько угодно раз. **Примечание:** Если Вы изменяли конфигурацию в режиме OLC (cn=config), удаление директории slapd.d (как показано выше) приведёт к потере всех изменений. В зависимости от того, какие изменения Вы проводили, например, изменения индексов в режиме cn=config, возможно, при возврате к slapd.conf придётся выполнять некоторые действия по восстановлению каталога. В целом, пока Вы не решились на окончательный переход, лучше ограничить свои эксперименты с возможностями OLC (cn=config).



Макет OLC (cn=config)

Расположение элементов OLC (cn=config) DIT:



Примечания:

1. Корневая запись `cn=config` основывается на объектном классе **olcGlobal** и содержит глобальные атрибуты, влияющие на работу slapd. После первоначального перекодирования она содержит все глобальные директивы из файла `slapd.conf` (а также экзотический набор значений по умолчанию для тех директив, которых не было в файле), за исключением подключённых наборов схемы данных (смотрите ниже `cn=schema`) и динамически загружаемых модулей (смотрите `cn=module{0}`). Полный список атрибутов `olcGlobal` смотрите [здесь](#).
2. Запись `cn=module{0},cn=config` основывается на объектном классе **olcModuleList** и содержит список всех динамически загружаемых объектов и путей к ним. После первоначального перекодирования она содержит элементы, определенные в директивах **moduleload** и **modulepath** файла `slapd.conf`. Полный список атрибутов `olcModuleList` смотрите [здесь](#).
3. Запись `cn=schema,cn=config` использует объектный класс **olcSchemaConfig** и содержит все объекты, используемые системой `cn=config`. Полный список атрибутов `olcSchemaConfig` смотрите [здесь](#).
Примечание: Атрибуты и объектные классы определяются с помощью атрибутов **olcAttributeTypes** и **olcObjectClasses**. Каждая дочерняя запись определяет подключаемый набор схемы данных.

4. Дочерние записи `cn={Z}xxx,cn=schema,cn=config` используют объектный класс **olcSchemaConfig** и содержат все объекты (атрибуты и объектные классы), определённые в данном наборе схемы. Здесь `{Z}` — числовой счётчик, нумерация которого начинается с 0, а `xxxx` — имя набора схемы данных. То есть, если первым подключается набор схемы `core.schema`, то данное значение будет `{0}core`. Полный список атрибутов **olcSchemaConfig** смотрите [здесь](#). **Примечание:** При загрузке в `cn=config` определённые в наборе схемы данных атрибуты трансформируются из **attributetype** (как они определены в файле набора схемы) в атрибуты **olcAttributeTypes**, а объектные классы — из **objectclass** (как они определены в файле набора схемы) в атрибуты **olcObjectClasses**. Процедура добавления набора схемы данных в OLC (`cn=config`) описана [здесь](#).
 5. Запись `olcDatabase={Z}xxx,cn=config` использует объектный класс **olcDatabaseConfig** (содержащий общие атрибуты, определяемые для всех типов баз данных), а также объектный класс, специфичный для конкретного типа базы данных. Здесь `{Z}` — числовой счётчик, нумерация которого начинается с 0, а `xxxx` — имя типа базы данных. То есть, если первой определяется база данных типа `bdb`, то данное значение будет `{0}bdb`, и эта запись будет строиться на объектном классе **olcBdbConfig**, содержащем атрибуты, специфичные для этого типа баз данных. Полный список атрибутов **olcDatabaseConfig** смотрите [здесь](#). Полный список атрибутов **olcBdbConfig** смотрите [здесь](#). Могут присутствовать дочерние записи для каждого наложения, используемого этой базой данных (описание смотрите ниже).
- Примечание по поводу записи `olcDatabase={-1}frontend,cn=config`:** Данная запись присутствует всегда, выглядит ужасно и содержит специфические атрибуты из объектного класса **olcFrontendConfig**, определяющие глобальные установки или свойства.
6. Запись `olcOverlay={Z}xxx,olcDatabase={Z}xxx,cn=config` использует объектный класс **olcOverlayConfig**, содержащий общие атрибуты, определяемые для всех наложений, а также объектный класс, специфичный для конкретного наложения. Здесь `{Z}` — числовой счётчик, нумерация которого начинается с 0, а `xxxx` — имя наложения. То есть, если первым из наложений определяется наложение `syncprov`, то данное значение будет `{0}syncprov`, и эта запись будет строиться на объектном классе **olcSyncProvConfig**, содержащем атрибуты, специфичные для этого наложения. Точно также другие наложения будут использовать специфичные для них объектные классы, содержащие их уникальные атрибуты. Полный список атрибутов **olcOverlayConfig** смотрите [здесь](#). Полный список атрибутов **olcSyncProvConfig** смотрите [здесь](#).



6.1.1.4 Использование `cn=config`

DIT `cn=config` можно прочитать с помощью стандартных инструментов командной строки LDAP, таких как `ldapsearch`, и изменять с помощью `ldapadd` или `ldapmodify`, что, по нашему скромному мнению, в наибольшей степени соответствует концепции конфигурации времени исполнения (OLC). Альтернативный метод — использовать LDAP-браузер для интерактивного чтения и записи атрибутов и записей данного DIT. О том, как получить доступ к `cn=config` из `LDAPBrowser/Editor`, читайте [здесь](#).

6.1.1.4.1 Общие замечания по конфигурации OLC (`cn=config`)

Следующие замечания могут оказаться полезными (или нет) при использовании OLC (`cn=config`):

1. В конфигурации OLC (`cn=config`) в значениях атрибутов широко используется синтаксис `{X}value` (в описании этих [атрибутов](#) есть элемент X-ORDERED). С помощью данного синтаксиса создаётся представление упорядоченного множества атрибутов. Примеры: при наличии нескольких наборов схемы данных в `cn=schema,cn=config` они будут иметь значения `cn={Z}xxxx,cn=schema,cn=config`, где `Z` — порядковый номер, начинающийся с 0, а `xxxx` — уникальное имя набора схемы. При наличии нескольких ACP/ACL в разделе глобальных настроек (`cn=config`) или в разделе `database` (`olcDatabase={Z}yyyy,cn=config` — в данном случае `Z` будет иметь то же значение, что и выше, а `yyyy` будет типом базы данных) каждый ACP/ACL будет определяться атрибутом **olcAccess** и иметь формат `{Z}to .....`. В большинстве случаев (но не во всех) действует следующий принцип редактирования: если при добавлении атрибута указывается значение в фигурных скобках, то атрибут будет добавлен в позицию, определённую этим значением, а все последующие (сдвигаемые) атрибуты будут по мере необходимости перенумерованы для поддержания упорядоченного множества. Если число в фигурных скобках не используется, то атрибут будет добавлен в конец упорядоченного множества атрибутов и в

начало значения этого атрибута будет добавлен следующий порядковый номер в фигурных скобках. Это очень полезное правило, здорово упрощающее вставку и удаление атрибутов/записей.

2. Напомним, что OLC (cn=config) по своей природе оперирует с живыми данными. Изменения системы производятся во время того, как пользователи осуществляют к ней доступ. Предположим, выполняется изменение из двух этапов, скажем (исключительно для иллюстрации), удаления одного атрибута и модификация другого, в результате чего будет введён новый режим с высокой степенью безопасности. В какой-то очень или не очень небольшой (в зависимости от природы изменения) период будет применено только первое изменение. Пользователи же, однако, будут продолжать пользоваться системой. Очень важно учитывать состояние системы после каждого изменения, так как, в течение этого переходного периода, они могут привести к непреднамеренному воздействию на данные или к другим не менее ужасным последствиям. Вспомните закон Мерфи, закон непредвиденных последствий и все остальные законы техногенных катастроф: стоит только зазеваться, и ... Вполне возможно, что для преодоления такой проблемы "временного статуса" подобные двухэтапные изменения потребуются произвести в три, четыре, а то и более этапов.
3. По состоянию на OpenLDAP версии 2.4.31, невозможно удалить какую-либо **запись** (удаления атрибутов в основном допускаются) в OLC (cn=config) с помощью нормальных процедур LDAP, таких как `ldapdelete` или используя LDAP-браузер. В некоторых случаях, таких как записи наборов схемы данных, это ограничение, похоже, будет присутствовать постоянно, в других — (по слухам) такое положение может измениться. Во всех случаях такие записи могут быть удалены либо путём возврата к [файлу `slapd.conf`](#), внесения соответствующих изменений и повторного переконвертирования данного файла, либо путём прямого редактирования определений `slapd.d` (как правило, лучше не пробовать так делать, если Вы не эксперт, однако для искателей приключений процесс [описан ниже](#)). В целом, в связи с существующими ныне формальными ограничениями на удаление, имеет смысл: (а) перед переконвертацией файла `slapd.conf` убедиться, что он находится в хорошей форме, (б) перед окончательным переходом к OLC (cn=config) поэкспериментировать с описанным выше процессом **почти перехода**, (с) молиться своим богам, и почаще.

6.1.1.4.2 Добавление/удаление наборов схемы данных в OLC (cn=config)

Добавление наборов схемы данных

Примечание: При первоначальном переходе к использованию OLC (cn=config), когда производится описанный выше [процесс переконвертации](#), все подключенные наборы схемы данных преобразуются автоматически. Приводимый ниже процесс требуется только при добавлении нового файла набора схемы к существующей конфигурации OLC (cn=config).

При использовании с OLC (cn=config) стандартные наборы схемы данных LDAP должны быть преобразованы в формат LDIF. Получившийся в результате файл LDIF затем импортируется с помощью `ldapadd` или подходящего LDAP-браузера. Преобразование из формата `schema` может быть выполнено вручную (если файл невелик), либо с помощью утилиты [slaptest](#) с парой ручных правок, что является самым быстрым способом для больших файлов. Оба метода будут продемонстрированы ниже на примере тривиального файла набора схемы с одним атрибутом и одним объектным классом, выбранного исключительно в целях иллюстрации процесса (этот файл представляет собой отредактированную версию стандартного файла `java.schema`, так что если Вы уже используете данный набор схемы и попытаетесь добавить получившийся в примере LDIF, это приведёт к ошибке, поскольку такие объекты уже будут существовать — в этом случае попробуйте поэкспериментировать с другим файлом). Даже если у Вас большой файл, будет полезно почитать раздел о ручном преобразовании, если Вы хотите глубже понимать данный процесс.

Внимание: После того, как Вы добавили набор схемы, Вы не можете удалить его иначе как используя специальный и потенциально опасный процесс ручного удаления. Начиная с версии 2.4 Вы можете модифицировать существующие определения атрибутов и объектных классов в установленных или добавленных наборах схемы данных.

Ручное преобразование

Целевой файл набора схемы, который будет добавляться в данном примере (`test.schema`):

```
# модифицированный файл java.schema - Java Object Schema - используется только для примера
# Copyright Notice
#
```

```
# Copyright (C) The Internet Society (1999). All Rights Reserved.
#
# Определение атрибута
attributetype ( 1.3.6.1.4.1.42.2.27.4.1.6
    NAME 'javaClassName'
    DESC 'Fully qualified name of distinguished Java class or interface'
    EQUALITY caseExactMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
    SINGLE-VALUE )

# Определение объектного класса
objectclass ( 1.3.6.1.4.1.42.2.27.4.2.1
    NAME 'javaContainer'
    DESC 'Container for a Java object'
    SUP top
    STRUCTURAL
    MUST cn )
```

Этот файл будет отредактирован вручную, в результате чего будет создан файл LDIF для добавления записи, содержащей оба определения атрибута и объектного класса из набора схемы. Подобные записи наборов схемы размещаются в ветке `cn=schema,cn=config`. Мы решили использовать `cn=test` в качестве уникального имени данного набора схемы (для реального набора схемы, очевидно, нужно подобрать более описательное имя). Исходя из этого предположения, мы будем загружать новый набор схемы как **dn: cn=test,cn=schema,cn=config**. Для добавления набора схемы применяется объектный класс **olcSchemaConfig** (не имеющий, как ни странно, обязательных (MUST) атрибутов, хотя необходимость определения атрибута `cn`, по существу, делает его обязательным), использующий атрибуты **olcAttributeTypes** (заменяющий **attributetype**) и **olcObjectClasses** (заменяющий **objectclass**) для добавления соответствующих определений объектов. Получившийся в результате файл сохраним как `test.ldif` или под каким-нибудь более подходящим именем.

После ручного редактирования файл выглядит так:

```
# отредактированный вручную файл
# красные строки были добавлены, зелёные - изменены
# в начале и в конце каждой строки определений olcAttributeTypes и
# olcObjectClasses должно быть по одному пробельному символу

dn: cn=test,cn=schema,cn=config
objectClass: olcSchemaConfig
cn: test
olcAttributeTypes: ( 1.3.6.1.4.1.42.2.27.4.1.6
    NAME 'javaClassName'
    DESC 'Fully qualified name of distinguished Java class or interface'
    EQUALITY caseExactMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
olcObjectClasses: ( 1.3.6.1.4.1.42.2.27.4.2.1
    NAME 'javaContainer'
    DESC 'Container for a Java object'
    SUP top
    STRUCTURAL
    MUST cn )
```

Для добавления файла `test.ldif` (или как Вы его назвали) в OLC (`cn=config`) используйте LDAP-браузер или `ldapadd`.

Преобразование с помощью `slaptest`

Исходный файл набора схемы данных в точности тот же, что мы использовали для примера с ручным преобразованием:

```
# модифицированный файл java.schema - Java Object Schema - используется только для примера
# Copyright Notice
#
# Copyright (C) The Internet Society (1999). All Rights Reserved.
#
# Определение атрибута
attributetype ( 1.3.6.1.4.1.42.2.27.4.1.6
    NAME 'javaClassName'
    DESC 'Fully qualified name of distinguished Java class or interface'
    EQUALITY caseExactMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
```

```
SINGLE-VALUE )
```

```
# Определение объектного класса
```

```
objectclass ( 1.3.6.1.4.1.42.2.27.4.2.1
    NAME 'javaContainer'
    DESC 'Container for a Java object'
    SUP top
    STRUCTURAL
    MUST cn )
```

Теперь нам нужно выбрать подходящее место и создать пустой файл slapd.conf с подключением нашего набора схемы test.schema. Пустой файл (назовём его test.conf или что-то в этом роде) выглядит так:

```
include /path/to/test.schema
```

В этот файл можно добавить сразу несколько подключений файлов наборов схемы — каждое из них создаст отдельный файл ldif. Теперь создадим директорию (testdir) и запустим slaptest для преобразования нашего test.conf в эту директорию:

```
# создаём директорию
mkdir testdir
# запускаем slaptest
slaptest -f test.conf -F testdir
```

В результате получится файл testdir/cn=config/cn=schema/cn={0}test.ldif, содержащий следующее:

```
# AUTO-GENERATED FILE - DO NOT EDIT!! Use ldapmodify.
# CRC32 ed5d248b
dn: cn={0}test
objectClass: olcSchemaConfig
cn: {0}test
olcAttributeTypes: {0}( 1.3.6.1.4.1.42.2.27.4.1.6 NAME 'javaClassName' DESC 'Fully qualified name of distinguished Java class or interface' EQUALITY caseExactMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
olcObjectClasses: {0}( 1.3.6.1.4.1.42.2.27.4.2.1 NAME 'javaContainer' DESC 'Container for a Java object' SUP top STRUCTURAL MUST cn )
structuralObjectClass: olcSchemaConfig
entryUUID: 1cf25098-2f16-1031-9bb3-07590c97897d
creatorsName: cn=config
createTimestamp: 20120510180255Z
entryCSN: 20120510180255.489367Z#000000#000#000000
modifiersName: cn=config
modifyTimestamp: 20120510180255Z
```

Отредактируем данный файл как показано ниже и сохраним его, скажем, как test.ldif или как Вам больше нравится:

```
# AUTO-GENERATED FILE - DO NOT EDIT!! Use ldapmodify.
# CRC32 ed5d248b
# зелёные строки были изменены, а красные требуется удалить
dn: cn=test,cn=schema,cn=config
objectClass: olcSchemaConfig
cn: test
olcAttributeTypes: {0}( 1.3.6.1.4.1.42.2.27.4.1.6 NAME 'javaClassName' DESC 'Fully qualified name of distinguished Java class or interface' EQUALITY caseExactMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
olcObjectClasses: {0}( 1.3.6.1.4.1.42.2.27.4.2.1 NAME 'javaContainer' DESC 'Container for a Java object' SUP top STRUCTURAL MUST cn )
structuralObjectClass: olcSchemaConfig
entryUUID: 1cf25098-2f16-1031-9bb3-07590c97897d
creatorsName: cn=config
createTimestamp: 20120510180255Z
entryCSN: 20120510180255.489367Z#000000#000#000000
modifiersName: cn=config
modifyTimestamp: 20120510180255Z
```

Остаётся только добавить test.ldif (или как Вы его там назвали) в OLC (cn=config) с помощью LDAP-браузера или [ldapadd](#).

Примечание: Личности, обладающие ненасытным любопытством (или склонные к цинизму), наверняка захотят разобраться, для чего же нужно редактировать этот глупый LDIF-файл. Вспомните, что мы хотели добавить этот новый набор схемы к существующей рабочей конфигурации. Вспомните также, что файлы наборов схемы уже присутствуют в рабочей конфигурации в виде [упорядоченной последовательности](#). Когда

slaptest вершит свои тёмные делишки над подключенным набором схемы в нашем фиктивном файле slapd.conf и конвертирует его в LDIF-файл, он создаёт новое отдельное упорядоченное множество из подключенных наборов схемы, в приведённом выше примере подразумевается, что это будет {0}test. В рабочей конфигурации у нас уже есть запись с порядковым номером {0}, поэтому мы получим конфликт, который либо приведёт к отклонению попытки загрузки и перезаписи существующей записи с номером {0}, либо нанесёт непоправимый урон всей системе электропитания западного полушария (в зависимости от того, какую версию OpenLDAP Вы используете). Удаляя порядковый номер {0}, мы тем самым вынуждаем OpenLDAP добавить данный набор схемы в конец текущего множества записей наборов схемы в рабочей конфигурации и присвоить ему следующий доступный порядковый номер. Строки, выделенные красным, — просто операционные атрибуты, которые при добавлении записи в рабочую конфигурацию будут немедленно сгенерированы заново и в них будут помещены текущие значения. Вот и всё.

Удаление наборов схемы данных

С помощью стандартных функций OLC (cn=config) невозможно удалить набор схемы данных целиком, но теперь — начиная с версии 2.4 — есть возможность отредактировать специальные атрибуты (содержащие определения атрибутов или объектных классов, как показано выше в примере с ручным преобразованием набора схемы) любого загруженного набора схемы. Невозможность удаления целого набора схемы — преднамеренное конструктивное решение OpenLDAP, и, видимо, положение дел в ближайшее время не изменится (если вообще когда-нибудь изменится). Причина такого решения в том, что после удаления набора схемы будет практически невозможно осуществлять поиск в базах данных по атрибутам и объектным классам, содержащимся в удалённом наборе схемы, и принимать решения об обслуживании или отклонении таких запросов. Есть возможность, при соблюдении крайней аккуратности и точности, удалить любой набор схемы данных вручную, как описано ниже.

<Предупреждение> Если какой-либо атрибут или объектный класс, определённый в удаляемом с помощью описанного ниже процесса (или любого другого процесса) наборе схемы данных используется в какой-нибудь базе данных, то эта база данных придёт в нерабочее состояние. Повторяем — не будет работать. Совсем не будет. Нужно быть абсолютно уверенным (и держать наготове резервные копии), что ни в одной из баз данных не используются какие-либо атрибуты и объектные классы из того набора схемы, который Вы собираетесь удалять, перед тем как пытаться удалить этот набор схемы с помощью описанного ниже (или любого другого) метода. **</Предупреждение>**

Для удаления набора схемы из рабочей системы OLC (cn=config) выполните следующее:

1. Определите DN того набора схемы, который Вы хотите удалить, путём просмотра поддерева cn=schema,cn=config с помощью LDAP-браузера или ldapsearch. У всех наборов схемы DN имеет форму cn={Z}xxxx,cn=schema,cn=config. В качестве примера, если Вы добавляли приведённый выше набор схемы, он будет иметь DN что-то вроде cn={6}test,cn=schema,cn=config (значение 6 в фигурных скобках может быть больше или меньше, в зависимости от того, сколько наборов схемы было загружено). Запомните это значение.
2. Остановите slapd. Перейдите в директорию slapd.d (обычно [fc]/etc/openldap/slapd.d или [bsd] /usr/local/etc/openldap/slapd.d) и скопируйте всю директорию целиком вместе с поддиректориями и файлами в подходящее место (данная копия может быть использована для восстановления всего в случае возникновения проблем).
3. Перейдите из директории slapd.d в поддиректорию cn=config, а затем в cn=schema. В этой директории Вы найдёте все наборы схемы в виде LDIF-файлов, в том числе файл (исходя из сделанного выше предположения) cn={6}test.ldif (напомним, что значение 6 может отличаться в зависимости от того, сколько наборов схемы было загружено). Удалите данный файл (Вы ведь сделали полную копию директории slapd.d на шаге 2, правда?).
4. Если у удалённого Вами ldif-файла в фигурных скобках было наибольшее значение из всех в этой директории, то данный шаг можно пропустить. Если нет, переименуйте все файлы, у которых значения в фигурных скобках в пределах данной директории были выше, для поддержания непрерывной последовательности нумерации, начиная с 0. Так, если Вы удалили cn={6}test.ldif, но в директории есть файл, скажем, cn={7}splodge.ldif, то для поддержания непрерывной последовательности нумерации его нужно переименовать в cn={6}splodge.ldif.
5. Сделайте глубокий вдох и запустите slapd. Убедитесь в том, что удалённый набор схемы действительно был удалён.

6. Если что-то пошло не так, немедленно остановите slapd и проверяйте сообщения об ошибках в log-файлах. Восстановите директорию slapd.d целиком со всеми поддиректориями и файлами из резервной копии, сделанной на шаге 2, и снова запустите slapd. Затем попытайтесь выяснить, что же там произошло. Мы предупреждали, что резать по живому — не самое удачное решение.

6.1.1.4.3 Добавление/удаление ACP/ACL при использовании OLC (cn=config)

ACP/ACL могут присутствовать, как и положено, в разделе глобальных настроек (в записи cn=config) или в разделе database (olcDatabase={Z}type,cn=config). В обоих случаях каждый ACP/ACL содержится в атрибуте **olcAccess**, которые вместе составляют упорядоченное множество: в начале каждого атрибута указывается порядковый номер (начиная с 0) в фигурных скобках, как показано в примере ниже (разделение строк выполнено только для удобства чтения):

```
olcAccess: {0}to attrs=userpassword by self write by anonymous auth by group/groupOfNames/member.exact="cn=itpeople,ou=groups,dc=example,dc=com" write by * none
olcAccess: {1}to attrs=carlicense,homepostaladdress,homephone by self write by group/groupOfNames/member.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write by * none
olcAccess: {2}to * by self write by group/groupOfNames/member.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write by users read by * none
```

Если теперь добавить (с помощью ldapmodify или LDAP-браузера) новый ACL/ACP, используя атрибут **olcAccess** начинающийся с {1}to..., то он будет помещён непосредственно за атрибутом, начинающимся с {0}, а последующие атрибуты будут перенумерованы:

```
olcAccess: {0}to attrs=userpassword by self write by anonymous auth by group/groupOfNames/member.exact="cn=itpeople,ou=groups,dc=example,dc=com" write by * none
olcAccess: {1}to ...
olcAccess: {2}to attrs=carlicense,homepostaladdress,homephone by self write by group/groupOfNames/member.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write by * none
olcAccess: {3}to * by self write by group/groupOfNames/member.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write by users read by * none
```

Если новый атрибут добавляется без использования синтаксиса {}, то он будет помещён в конец множества атрибутов и ему будет присвоен следующий порядковый номер:

```
olcAccess: {0}to attrs=userpassword by self write by anonymous auth by group/groupOfNames/member.exact="cn=itpeople,ou=groups,dc=example,dc=com" write by * none
olcAccess: {1}to attrs=carlicense,homepostaladdress,homephone by self write by group/groupOfNames/member.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write by * none
olcAccess: {2}to * by self write by group/groupOfNames/member.exact="cn=hrpeople,ou=groups,dc=example,dc=com" write by users read by * none
olcAccess: {4}to ...
```

При изменении или удалении любого атрибута **olcAccess** Вы должны использовать синтаксис {} для идентификации конкретного атрибута.

6.1.1.4.4 Добавление/удаление модулей при использовании OLC (cn=config)

Добавление модулей

Динамически подгружаемые модули определяются в записи cn=Module{0},cn=config, использующей объектный класс **olcModuleList**. Конкретный модуль определяется с помощью атрибута **olcModuleLoad**. Для загрузки нового модуля требуется просто определить атрибут с необходимым модулем (без использования синтаксиса {}). При этом подразумевается, что должны быть удовлетворены две зависимости. Во-первых, этот модуль должен существовать (быть создан при установке программы), и, во-вторых, он должен быть доступен по одному из путей по умолчанию, либо по тому пути, который определён в атрибуте **olcModulePath** (этот атрибут SINGLE-VALUE, то есть может иметь только одно значение, однако в этом значении может быть список путей, разделённых двоеточиями — смотрите [спецификацию формата](#)). Атрибуты **olcModuleLoad** добавляются без использования синтаксиса {} (нет никакой разницы в том, в каком порядке загружаются динамически подгружаемые модули), поэтому новый модуль помещается в конец множества атрибутов и ему присваивается следующий порядковый номер. Если имя модуля неверное (то есть

не входит в состав модулей OpenLDAP), то запрос на добавление данного атрибута будет отклонён, а если модуль вкомпилирован в саму программу (и нет соответствующего динамически подгружаемого модуля), атрибут будет добавлен (что, по существу, избыточно).

Удаление модулей

Вы не можете удалить какой-либо атрибут **olcModuleLoad**, что логично для существующей политики OpenLDAP (по крайней мере на текущий момент). Несколько странно, что при попытке удалить данный атрибут выдаётся ошибка 80 (неизвестная ошибка (unknown error), снабжённая хорошим текстовым описанием), а не ошибка 53 (сервер не желает выполнять (server unwilling to perform)).

6.1.1.4.5 Добавление/удаление баз данных при использовании OLC (cn=config)

Добавление баз данных

Базы данных определяются с использованием объектного класса **olcDatabaseConfig** с обязательным (MUST) атрибутом **olcDatabase**, а также объектного класса для конкретного типа базы данных, позволяющего задавать специфичные атрибуты, используемые в определении базы данных. Например, при определении базы данных типа **bdb** используется объектный класс **olcBdbConfig** (с обязательным атрибутом **olcDbDirectory**). Все записи баз данных создают упорядоченное (с помощью синтаксиса { }) множество **типов баз данных** в форме **olcDatabase={Z}xxx,cn=config**. Здесь {Z} — порядковый номер, а xxxx — имя типа базы данных. Если при добавлении новой базы данных используется синтаксис { }, то она (начиная с версии 2.4) будет добавлена в указанное место множества, а все последующие базы данных будут перенумерованы. Если синтаксис { } не указывался, база данных будет добавлена в конец текущего списка и ей будет присвоен следующий порядковый номер.

Возможно, не стоило тратить драгоценное время на такую ерунду, но: на практике, хотя он и не является обязательным, атрибут **olcSuffix** должен присутствовать, иначе создать запись базы данных не удастся (ошибка **error 80 fail setup**). Вот.

В качестве примера, предполагая, что в текущей конфигурации присутствуют записи баз данных **olcDatabase={0}config,cn=config** и **olcDatabase={1}bdb,cn=config**, с помощью приведённой ниже минимальной конфигурации создадим запись с DN **olcDatabase={2}bdb,cn=config**.

```
# порядковый номер не указан, поэтому запись будет
# добавлена в конец текущего списка баз данных

dn: olcDatabase=bdb,cn=config
objectClass: olcBdbConfig
olcDatabase: bdb
olcDbDirectory: /var/db/new-db
olcSuffix: dc=example,dc=net
```

Поскольку запись создаётся без использования синтаксиса упорядочения (нет значения {Z}), она будет добавлена в конец текущего упорядоченного множества и ей будет присвоен следующий порядковый номер, в данном случае {2}. Не возбраняется и явно указывать значение **olcDatabase: {2}bdb** (внеся соответствующие изменения и в строку **dn:**), результат будет тот же. Если же Вы укажете значение **olcDatabase: {1}bdb** (опять таки, внеся соответствующие изменения в строку **dn:**), то при добавлении в подразумеваемую нами конфигурацию, текущая база данных с номером {1} будет перенумерована в {2}, а новая база данных будет добавлена на её место в упорядоченном множестве. Дополнительные атрибуты (в том числе такие очевидные, как **olcRootDN** и **olcRootPW**) можно добавить как на этапе создания записи, так и позже. Напоминаем, что директория, в которой размещаются файлы базы данных (определяемая атрибутом **olcDbDirectory**), должна существовать (и иметь приемлемые права доступа) до того, как её необходимо будет использовать, то есть перед созданием записи базы данных.

Удаление баз данных

По состоянию на OpenLDAP версии 2.4.31 невозможно удалить запись Database. Любые попытки это сделать отклоняются и выдаётся ошибка 53 (Server unwilling to perform (сервер не желает исполнять)), как видите, описание вполне информативно. Ходят слухи, что есть какой-то экспериментальный код, реализующий удаление, который якобы можно получить, но пока дела обстоят так. Есть вариант использовать для удаления какой-либо неверно добавленной базы данных технику, сходную с той, что мы рассматривали для

удаления лишних наборов схемы данных. Будьте осторожны, поскольку в этом случае при операциях с отсылками на удалённую базу данных могут быть негативные последствия, то есть если в какой-либо базе данных содержатся отсылки на удалённую (несуществующую) базу данных LDAP, переход по этим отсылкам будет вызывать ошибки (положение усугубляется ещё и тем, что большинство LDAP-браузеров по умолчанию пытаются разрешать отсылки автоматически).

6.1.1.4.6 Добавление/удаление наложений при использовании OLC (cn=config)

Добавление наложений

Наложения описываются в записях, дочерних по отношению к соответствующей записи **olcDatabase={Z}xxx,cn=config**, к которой они применяются. Записи наложений используют общий объектный класс **olcOverlayConfig** (обязательный атрибут **olcOverlay**) и специфичный для наложения объектный класс (всегда являющийся потомком **olcOverlayConfig**), представляющий собой контейнер для специфичных атрибутов данного наложения. Например, наложение **syncprov** (поставщик репликации) использует объектный класс **olcSyncProvConfig** (нет обязательных атрибутов). Если наложение использует динамически подгружаемый модуль, то он должен быть **добавлен** до определения наложения. С помощью приведённой ниже минимальной конфигурации можно создать запись **syncprov**, как дочернюю к записи базы данных **olcDatabase={1}bdb,cn=config**:

```
dn: olcOverlay=syncprov,olcDatabase={1}bdb,cn=config
objectclass: olcSyncProvConfig
olcOverlay: syncprov
```

Поскольку запись создаётся без использования синтаксиса упорядочения (нет значения {Z}), она будет добавлена в конец текущего упорядоченного множества и ей будет присвоен следующий порядковый номер, в данном случае {0}. Не возбраняется, подразумевая, что это первая запись, и явно указывать значение **olcOverlay: {0}syncprov** (внеся соответствующие изменения и в строку **dn:**), результат будет тот же. Напомним, что порядок определения наложений определяет порядок их обработки, что во многих случаях может иметь важное значение. Для добавления наложений в корректном порядке как раз подойдёт синтаксис упорядоченного множества {Z} (все последующие записи будут перенумерованы для поддержания правильного упорядоченного множества). Дополнительные атрибуты можно добавить как на этапе создания записи, так и позже.

Удаление наложений

Возможно, для Вас это будет сюрпризом, но... по состоянию на OpenLDAP версии 2.4.31 невозможно удалить запись Overlay. Любые попытки это сделать отклоняются и выдаётся ошибка 53 (Server unwilling to perform (сервер не желает исполнять)), как видите, описание вполне информативно. Ходят слухи, что есть какой-то экспериментальный код, реализующий удаление, который якобы можно получить, но пока дела обстоят так. Есть вариант использовать для удаления какого-либо неверно добавленного наложения технику, сходную с той, что мы рассматривали для [удаления лишних наборов схемы данных](#). Как всегда, будьте осторожны.



Глава 6. OpenLDAP Idap.conf

Конфигурационный файл **Idap.conf** содержит информацию и директивы конфигурации, используемые клиентами OpenLDAP. При необходимости эти директивы могут использоваться и инструментами OpenLDAP.

Директивы TLS

Какие именно клиентские директивы TLS будут использоваться, зависит от того, будет ли клиент TLS отправлять сертификат X.509 и проверять сертификат TLS сервера (в этом случае требуется большинство директив), либо только проверять сертификат TLS сервера (в этом случае требуется только директива **TLS_CACERT** и, опционально, директива **TLS_CIPHER_SUITE**). Далее, для удобства, введены следующие обозначения: если директива используется для посылки TLS-сертификата клиента, это указывается ключевым словом **КЛИЕНТ**, если используется при обоюдной аутентификации — ключевым словом **ОБОЮДНО**, а в случае только проверки TLS-сертификата сервера — ключевым словом **СЕРВЕР**.

TLS_CACERT

```
TLS_CACERT /path/to/CA/cert/file.pem
```

Клиентская директива TLS (СЕРВЕР). Определяет путь до файла и имя файла сертификата Удостоверяющего центра (Certificate Authority), известного также как корневой сертификат. Позволяет клиенту проверить сертификат сервера LDAP. Данный файл требуется при использовании как самоподписанного, так и коммерческого сертификата. Корневой сертификат должен быть получен у поставщика сертификатов X.509 (или, в случае самоподписанного сертификата, скопирован с LDAP-сервера каким-либо безопасным способом). Как правило, это файл в формате PEM (Privacy enhanced Mail) с расширением .pem или, если он был получен из установки браузера MSIE, с расширением .cer. Если рабочий сертификат X.509 (указанный в директиве [TLSCertificateFile](#) slapd.conf) подписан промежуточным удостоверяющим центром, то в файле формата PEM должны присутствовать все сертификаты удостоверяющих центров вплоть до центра корневого уровня. PEM — это текстовый формат, и несколько сертификатов могут помещаться в один и тот же файл в произвольном порядке — смотрите [заметки и примеры по формату PEM](#). Пример конфигурации OpenLDAP с самоподписанным сертификатом можно посмотреть [здесь](#). В данном файле нет конфиденциальной информации (сертификат X.509 содержит только открытый ключ).

TLS_CIPHER_SUITE

```
TLS_CIPHER_SUITE cipher-list
```

Клиентская директива TLS (СЕРВЕР+КЛИЕНТ+ОБОЮДНО). Это необязательная директива и по умолчанию подразумевается значение ALL (что эквивалентно **openssl ciphers -v ALL**). Определяет один или несколько шифров, которые будут использоваться во время [переговоров об установке соединения TLS](#). В ходе этих переговоров клиент TLS предлагает список **шифров** и сервер TLS примет **первый** шифр из своего списка шифров, который совпадёт с одним из предлагаемых клиентом. Используемый в описании данной директивы термин **cipher-list** определяет список (в формате OpenSSL), который будет перекодирован библиотеками OpenSSL в список **шифров** в формате TLS/SSL. Дополнительную информацию о формате cipher-list можно получить в [документации по шифрам OpenSSL](#). Примеры конфигурации OpenLDAP с самоподписанным сертификатом можно посмотреть [здесь](#).

Список доступных шифров (и, следовательно, cipher-list) определяется форматом открытого ключа, содержащегося в сертификате X.509. В случае отправки сертификата клиентом TLS открытый ключ извлекается из сертификата, указанного в директиве [TLS_CERT](#), в случае только выполнения проверки сертификата сервера — извлекается из сертификата сервера TLS, в случае обоюдного обмена сертификатами — извлекается из обоих сертификатов. Так, если сертификат (сертификаты) содержат открытый ключ RSA, то только шифры открытого ключа RSA могут быть использованы на этапах обмена ключами/аутентификации при [установке соединения TLS](#). Если криптоалгоритм открытого ключа входящего сертификата сервера TLS не удастся опознать, то в качестве списка будет использоваться ALL (смотрите команды ниже). Отдельные пункты в cipher-list разделяются двоеточием, запятой или пробельным символом. Далее приводится подмножество имён RSA TLSv1, которые могут фигурировать в cipher-list, и эквивалентных им текстовых значений шифров TLS (при отправке по каналу они перекодировываются в шестнадцатеричные значения).

Примечание: Слово EXPORT (или EXP), встречающееся в некоторых именах, указывает, что эти шифры являются экспортируемыми, то есть некоторые шифры разрешено использовать только в определённых странах (смотрите документацию [Бюро промышленности и безопасности Министерства торговли США](#) и [Вассенарские соглашения](#)), и это необходимо учитывать при настройке системы TLS, которая будет использоваться на международном уровне.

НАЗВАНИЕ ШИФРА TLS	НАЗВАНИЕ CIPHER-LIST OPENSSL
TLS_RSA_WITH_NULL_MD5	NULL-MD5
TLS_RSA_WITH_NULL_SHA	NULL-SHA
TLS_RSA_EXPORT_WITH_RC4_40_MD5	EXP-RC4-MD5
TLS_RSA_WITH_RC4_128_MD5	RC4-MD5
TLS_RSA_WITH_RC4_128_SHA	RC4-SHA
TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5	EXP-RC2-CBC-MD5
TLS_RSA_WITH_IDEA_CBC_SHA	IDEA-CBC-SHA
TLS_RSA_EXPORT_WITH_DES40_CBC_SHA	EXP-DES-CBC-SHA
TLS_RSA_WITH_DES_CBC_SHA	DES-CBC-SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA	DES-CBC3-SHA

TLS_RSA_WITH_AES_128_CBC_SHA	AES128-SHA
TLS_RSA_WITH_AES_256_CBC_SHA	AES256-SHA
TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA	EXP1024-DES-CBC-SHA
TLS_RSA_EXPORT1024_WITH_RC4_56_SHA	EXP1024-RC4-SHA

Чтобы вывести список значений cipher-list, поддерживаемых локальной инсталляцией OpenSSL, используйте:

```
# Все действительные шифры (ALL)
openssl ciphers -v ALL

# Все действительные шифры только для TLSv1
openssl ciphers -v -tls1 ALL

# Действительные шифры только для TLSv1, использующие
# алгоритм обмена ключами/аутентификации RSA
openssl ciphers -v -tls1 RSA

# Действительные шифры только для TLSv1, использующие
# алгоритм обмена ключами/аутентификации RSA
# исключая экспортируемые шифры
openssl ciphers -v -tls1 RSA:!EXP
# ПРИМЕЧАНИЕ: в некоторых оболочках знак ! требует экранирования
openssl ciphers -v -tls1 RSA:\!EXP

# То же, что и предыдущая команда, но исключаются NULL-шифры
openssl ciphers -v -tls1 RSA:!EXP:!NULL
# ПРИМЕЧАНИЕ: в некоторых оболочках знак ! требует экранирования
openssl ciphers -v -tls1 RSA:\!EXP:\!NULL

# Действительные шифры только для TLSv1, использующие
# алгоритм обмена ключами/аутентификации RSA
# только экспортируемые шифры
openssl ciphers -v -tls1 RSA:EXP
# ИЛИ
openssl ciphers -v TLSv1+RSA:EXP
```

В директиве TLS_CIPHER_SUITE в качестве cipher-list могут указываться либо общие параметры, например, RSA, тогда для получения списка конкретных шифров может быть использована команда openssl, как показано выше (в этом случае порядок предпочтения шифров определяется openssl), либо явно заданный список шифров в порядке их предпочтения. Одно или несколько поддерживаемых значений в cipher-list должны поддерживаться сервером TLS. Алгоритм поиска совпадений шифров (выбора того, какой шифр будет использован) таков: первый (наиболее предпочтительный) шифр из предоставляемых клиентом, который также поддерживается и сервером, становится шифром соединения (сессии). В следующих примерах используется только подмножество TLSv1 (SSLv3):

```
# Cipher-list содержит только основанные на RSA
# шифры аутентификации и обмена ключами
# поддерживаемые TLSv1 (и SSLv3)
TLS_CIPHER_SUITE TLSv1+RSA

# Cipher-list содержит только основанные на RSA
# шифры аутентификации и обмена ключами
# поддерживаемые TLSv1 (и SSLv3)
# исключая экспортируемые и NULL-шифры
TLS_CIPHER_SUITE TLSv1+RSA:!EXPORT:!NULL

# Упорядоченный список основанных на RSA
# шифров аутентификации и обмена ключами
TLS_CIPHER_SUITE DES-CBC-SHA:DES-CBC3-SHA:RC4-SHA:RC4-MD5

# Все шифры за исключением NULL-шифров
TLS_CIPHER_SUITE ALL:!NULL

# Значение по умолчанию, эквивалентно случаю
# когда директива не задана
TLS_CIPHER_SUITE ALL
```

Примечание: OpenSSL поддерживает ряд шифров, в результате применения которых происходит NULL-шифрование данных большого объема и имитовставок MAC. Это означает, что безопасно выполняется только процесс аутентификации, а вся последующая передача данных происходит в открытом виде. Чтобы предотвратить использование таких шифров либо используйте значение !NULL в cipher-list, либо явно задайте список шифров, не включая в них NULL-шифры.



Уже совсем скоро (One day real soon now ™)

Глава 6. ApacheDS server.xml

В этой главе рассматривается вопрос конфигурации LDAP-систем ApacheDS.

Уже совсем скоро (One day real soon now TM)

Глава 6. OpenLDAP slapd.conf, базы данных bdb и hdb

Этот раздел документации посвящён директивам, специфичным для баз данных **bdb** (и **hdb**). Общие директивы раздела database описаны [здесь](#). [Berkeley DataBase \(BDB или HDB\)](#) (ныне принадлежит Oracle) — предпочтительный и рекомендуемый механизм манипуляции данными OpenLDAP, требующий, однако, тщательной настройки в случае, когда приложению необходима высокая производительность. Указанные здесь параметры — только часть настроек производительности. Другие настройки могут содержаться в файле [DB_CONFIG file](#) (смотрите также [главу, посвящённую производительности](#)).

Предупреждение: Файл журнала транзакций BDB (log.xxxxxxxxxxxx в директории базы данных LDAP, определяемой директивой [directory](#)) может очень быстро и очень сильно разрастаться, и потому требует регулярного обслуживания — смотрите описание процедур обслуживания в [разделе 17 главы 12 документации BDB](#).

Часть параметров данного раздела файла slapd.conf напрямую обрабатывается базой данных BDB. Такие параметры могут быть заменены соответствующими директивами файла настройки базы данных DB_CONFIG и **удалены** из файла slapd.conf. Видимо, разработчики OpenLDAP всё больше склоняются к использованию директив DB_CONFIG, нежели их эквивалентов в slapd.conf. Если такая замена возможна, это указывается в описании каждой конкретной директивы.

Содержание

- [cachesize](#)
- [checkpoint](#)
- [dbnosync](#)
- [directory](#)
- [dirtyread](#)
- [index](#)
- [lockdetect](#)
- [mode](#)
- [searchstack](#)
- [Пример файла DB_CONFIG](#)
- [Фрагменты-примеры раздела database bdb](#)

cachesize

Формат:

```
cachesize integer
```

Директива **cachesize** определяет количество **записей**, которое механизм манипуляции данными LDAP будет обслуживать в памяти. Не путайте данную директиву с директивой BDB **set_cachesize** — они управляют различным поведением.

Для достижения максимальной производительности этот показатель должен соответствовать, либо быть максимально приближен к количеству записей, содержащихся в каталоге. Значение по умолчанию — 1000. Примеры:

```
cacheSize 10000
# LDAP обслуживает в памяти 10,000 записей
```

Смотрите также [главу о настройках производительности](#).



checkpoint

Формат:

```
checkpoint kbyte min
```

Директива **checkpoint** определяет промежуток времени между выполнением операции сброса данных в контрольной точке (checkpoint) в базе данных BDB, то есть записи данных из оперативной памяти на диск. Восстановить базу данных можно только из последней контрольной точки.

По сути, частота выполнения операции сброса данных в контрольной точке определяет промежуток времени; если за этот промежуток времени в каталог будут внесены изменения, то, в случае краха системы, их невозможно будет восстановить средствами BDB. Если **dbnosync** НЕ ИСПОЛЬЗУЕТСЯ, в качестве данного промежутка времени может быть установлен достаточно длительный период, скажем, 10 минут или больше; если же директива **dbnosync** ИСПОЛЬЗУЕТСЯ, то лучше установить его в 5 — 15 минут или менее. Параметр **kbytes** — количество килобайт, **записанных** в каталог, а параметр **min** — время в минутах. То значение, которое будет достигнуто первым, и определяет окончание периода между операциями сброса в контрольной точке.

По умолчанию OpenLDAP НЕ ПРОИЗВОДИТ ОПЕРАЦИЙ СБРОСА В КОНТРОЛЬНОЙ ТОЧКЕ, таким образом, нужно всегда явно указывать директиву **checkpoint**. Дополнительную информацию можно найти в [разделе 15 главы 12 документации BDB](#). Данную директиву можно заменить директивой **txn_checkpoint** файла [DB_CONFIG](#). Примеры:

```
checkpoint 128 15
# сброс данных в контрольной точке происходит либо после записи 128k данных,
# либо по прошествии 15 минут, в зависимости от того, что произойдёт раньше
```



dbnosync

Формат:

```
dbnosync
```

Директива **dbnosync** указывает, что для базы данных НЕ требуется немедленная синхронизация при изменении какой-либо записи в памяти. Данная опция увеличивает производительность при операциях записи, но есть и недостаток: если крах системы произошёл до того, как данные на диске и в памяти синхронизировались, то часть данных может быть утеряна. Синхронизация данных между памятью и диском управляется директивой **checkpoint**. По умолчанию (при отсутствии данной директивы) обновление данных на диске происходит сразу же. Примеры:

```
dbnosync
# разрешено отложенное обновление данных на диске
```

Данная директива аналогична опции BDB **set_flag** DB_TXN_NOSYNC — смотрите [главу о настройках производительности](#).



directory

Формат:

```
directory /path/to/directory
```

Директива **directory** определяет местонахождение файлов базы данных BDB. По соглашению эти файлы размещаются в /var/db/openldap-data (BSD) или /var/lib/ldap (FC/Linux), но их можно размещать в любом удобном месте. Указанная в директиве директория ДОЛЖНА существовать перед первым запуском OpenLDAP.

Если Вы не определили данную директиву, OpenLDAP по умолчанию будет использовать директорию **/var/db/openldap-data** (BSD) или /var/lib/ldap (FC). Если Вы указали относительное имя для директории (без '/' в начале), BDB будет подставлять в начало пути содержимое своей переменной окружения DB_HOME, которая может указывать на любую точку вселенной.

Хотя значения по умолчанию предусмотрены, документация OpenLDAP утверждает (но не настаивает), что директива **directory** является ОБЯЗАТЕЛЬНОЙ. Мы рекомендуем всегда указывать эту директиву хотя бы затем, чтобы не тратить время на поиски месторасположения файлов базы данных.

Если Вы собираетесь создавать или обслуживать каталог с несколькими DIT (для этого нужно несколько разделов **database**), целесообразно заранее продумать и создать систему именования поддиректорий для хранения файлов этих баз данных. Примеры:

```
directory /var/ldap/example-com
# директория example-com должна существовать, на неё должны
# быть назначены права на чтение и запись пользователю ldap

directory example-com
# в начало пути добавляется значение переменной BDB DB_HOME
# и директория будет /user/local/share/openldap/example-com (BSD)
```



dirtyread

Формат:

```
dirtyread
```

Директива **dirtyread** позволяет OpenLDAP при поисковых запросах возвращать данные из памяти, которые могут быть ещё не записаны на диск. Если последующая операция записи на диск завершится неудачей, эти данные будут потеряны. Таким образом, пользователь может получить результаты, отличающиеся от конечного состояния каталога.

Указание данной директивы может увеличить производительность каталога, повышая риск некоторого нарушения целостности. По умолчанию (при отсутствии данной директивы) данные в памяти, еще не записанные на диск, возвращаться не будут. Примеры:

```
dirtyread
# пользователь МОЖЕТ получить данные, которые
# не соответствуют конечному состоянию каталога
```



olcDbIndex (index)

Лучше определять директивы **index** файла slapd.conf перед первоначальной загрузкой каталога (с помощью **ldapadd**), тогда при первоначальной загрузке соответствующие индексы будут созданы

автоматически. При внесении изменений в настройки индексов после первоначальной загрузки требуется переиндексирование (повторное создание индексов) каталога с помощью [slapindex](#) (**предупреждение:** перед этим slapd должен быть остановлен).

В конфигурации OLC (cn=config) используется атрибут `olcDbIndex`, который может присутствовать только в записи `olcDatabase={Z}xxx,cn=config`. Кроме того, переиндексирование происходит в режиме реального времени, то есть любые изменения данного атрибута срабатывают немедленно и индексирование изменяется (выполнять `slapindex` не нужно). Однако, при использовании OLC (cn=config) трудно понять, когда переиндексирование завершается (никаких видимых признаков окончания процесса нет).

Формат:

```
# форма OLC (cn=config)
olcDbIndex: attrlist | default indices

# форма slapd.conf
index attrlist | default indices

# indices = [pres [,approx] [,eq] [,sub] [,special]]
```

Атрибуты **olcDbIndex** (директивы **index**) определяют, какие индексы будут обслуживаться OpenLDAP. Может быть включено любое количество параметров индексирования. Даже если атрибут не был указан в директивах **index**, его всё равно можно использовать в [поисковых фильтрах](#) — если это будет происходить часто, то конечно, производительность будет страдать, а если раз в жизни — ничего страшного.

attrlist может быть единичным атрибутом, или списком атрибутов, разделённых запятыми.

Необязательный параметр **default** содержит те индексы, которые должны поддерживаться по умолчанию. Они применяются к атрибутам в последующих атрибутах **olcDbIndex** (директивах **index**), в которых пропущен список индексов. Атрибут **olcDbIndex** (директива **index**) со значением **default** должен быть определён до появления атрибутов **olcDbIndex** (директив **index**) без списка индексов. Если определено несколько атрибутов **olcDbIndex** (директив **index**) со значением **default**, каждая из них будет влиять на атрибуты **olcDbIndex** (директивы **index**), следующие непосредственно за ней, до появления очередного атрибута **olcDbIndex** (директивы **index**) со значением **default**.

Индекс **pres** следует использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'objectclass=person' или 'attribute=mail'.

Индекс **approx** НЕОБХОДИМО использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'sn~=person' (поиск 'созвучных' значений).

Индекс **eq** следует использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'sn=smith', то есть без применения поисковых шаблонов (используется только правило EQUALITY).

Индекс **sub** следует использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'sn=sm*', то есть с применением поисковых шаблонов (используется правило SUBSTR). Данный индекс можно задать более конкретно, указывая **subinitial** (оптимизация под 'sn=*s'), **subany** (оптимизация под 'sn=*n*') или **subfinal** (оптимизация под 'sn=th*'). Для одного атрибута можно указать несколько индексов типа **sub**.

Параметр **special** связан с [подтипами \(subtype\)](#) и может быть либо **nolang**, либо **nosubtypes**.

Будьте очень внимательны при выборе того, какие индексы будут поддерживаться каталогом. Лучше делать это на основании **требований приложений**; если их не учитывать, это может серьёзно повлиять на производительность каталога. И наоборот, нет никакого смысла индексировать те атрибуты, поиск по которым никогда не осуществляется. Если в поисковых запросах используются только правила EQUALITY, нет смысла устанавливать индексирование **sub**. Индексы потребляют память (чем больше индексов, тем больше они занимают оперативную память). Кроме того, операции записи и модификации при большом количестве индексов будут выполняться дольше, поскольку требуется время и ресурсы на обновление индексов.

Примеры:

```
# Простое использование значения default
```

```
# форма OLC (cn=config)
olcDbIndex: default pres,eq
olcDbIndex: cn,sn,uid

# форма slapd.conf
index default pres,eq
index cn,sn,uid

# Определяем индексы наличия и соответствия
# для атрибутов cn, sn и uid
# две приведённые выше директивы index выполняют
# абсолютно то же, что и три, приведённые ниже
index cn pres,eq
index sn pres,eq
index uid pres,eq

index cn eq,sub,subinitial
# Создаём индексы для атрибута cn (commonname)
# для поисков EQUALITY, SUBSTR, а также дальнейшая оптимизация
# для поисков типа sc=a*

index sn eq,approx,sub
# Создаём индексы для атрибута sn (surname)
# для поисков EQUALITY и SUBSTR
# Примечание: Индекс approx index - пустая трата времени,
# поскольку для sn не определено правило ORDERING,
# требуемое для поисков approx. Приводится лишь для
# иллюстрации возможности использования

index mail pres,eq,sub
# Создаём индексы для атрибута mail on
# для поисков на наличие, EQUALITY и SUBSTR

index objectclass eq
# Оптимизируем поиски по форме objectclass=person
```

Обзор других настроек производительности можно найти в соответствующей [главе](#).

lockdetect

Считается устаревшей в пользу параметра **set_ik_detect** файла [DB_CONFIG](#).



mode

Формат:

```
mode mask
```

Директива **mode** определяет права на файлы, назначаемые при первоначальном создании файлов базы данных BDB. Значение по умолчанию — 0600 (чтение и запись разрешены только пользователю ldap, остальным доступ запрещён). Примеры:

```
mode 0660
# пользователь ldap - чтение и запись
# группа ldap - чтение и запись
# остальные - нет доступа
```



searchstack

Формат:

```
searchstack depth
```

Директива **searchstack** определяет **глубину (depth)** стека, используемого для оценки поискового фильтра. Оценка поисковых фильтров происходит по стеку, в который помещаются вложенные условия AND / OR. Для каждого потока сервера выделяется собственный стек. Глубина стека определяет, насколько комплексные фильтры могут быть оценены без необходимости выделения дополнительной памяти.

Применение поискового фильтра с глубиной вложенности большей, чем глубина поискового стека, приведёт к тому, что для этой конкретной операции поиска будет выделен отдельный стек. Таким образом, если количество вложенных условий превышает указанное значение (или значение по умолчанию), происходит значительное падение производительности.

Каждый поисковый стек использует 512Kb для одного вложенного уровня условий. Глубина стека по умолчанию — 16, то есть используется 8MB памяти для каждого потока. Примеры:

```
searchstack 5
# позволяет размещать в стек до 5-ти вложенных инструкций поиска
```



Пример файла DB_CONFIG

Ответ на вопрос, зачем использовать файл DB_CONFIG, прост: если Вы заботитесь о производительности — используйте его, если не заботитесь — забудьте о нём. ОДНАКО, если Вы решили не использовать DB_CONFIG, то Вы ДОЛЖНЫ определить директиву **checkpoint** в файле slapd.conf. Разработчики OpenLDAP всё чаще советуют использовать DB_CONFIG вместо определения аналогичных директив в slapd.conf.

Файл DB_CONFIG располагается в директории, указанной в директиве **directory**. Используемые в этом файле директивы полностью описаны в [руководстве по BDB](#). Можно заменить директивы **checkpoint**, **lockdetect**, **dbnosync** файла slapd.conf эквивалентными директивами файла DB_CONFIG, а также добавить дополнительные директивы настройки производительности по мере необходимости.

Пример файла DB_CONFIG:

```
# замена директивы lockdetect
set_lk_detect DB_LOCK_EXPIRE
# ЛИБО set lk_detect DB_LOCK_DEFAULT

# раскомментируйте, если требуется dbnosync
# set_flags DB_TXN_NOSYNC
# разрешено использовать несколько директив set_flags

# устанавливает максимальный размер журнала в 5Mb (по умолчанию для BDB - 10Mb)
set_lg_max 5242880

set_cachesize 0 5242880 1
# устанавливает кэш базы данных в 5Mb
# разрешается фрагментация
# НЕ заменяет директиву cachesize в slapd.conf
# это параметр базы данных

txn_checkpoint 128 15 0
# замена директивы checkpoint в slapd.conf
# сброс в контрольной точке при записи 128K или каждые 15 минут
# 0 говорит о том, что если не было операций записи, то обновлений не производится
```

Примечание:

1. Если у Вас **несколько DIT** (несколько разделов database в slapd.conf), файл DB_CONFIG требуется для каждой директории, содержащей файлы базы данных.
2. Для проверки некоторых показателей статистики базы данных используйте:

```
cd /to/bdb/directory
db_stats -m
```

```
# ИЛИ на BSD
db4_stat -m
```

3. Из документации не совсем понятно, что произойдёт, если параметры в `slapd.conf` и в `DB_CONFIG` будут конфликтовать друг с другом. В общем случае — НЕ ДОПУСКАЙТЕ этого! Например, если у Вас в `slapd.conf` присутствует директива **checkpoint**, а в `DB_CONFIG` — директива **txn_checkpoint**, — удалите одну из них.



Фрагменты-примеры раздела database bdb

Следующие фрагменты раздела `database` иллюстрируют настройку базы данных BDB. Цель оптимизации производительности не ставится.

Пример 1 — файл `DB_CONFIG` не используется:

```
database bdb
...
# только фрагмент раздела database
# порядок директив не важен
cachesize 10000
checkpoint 128 15
directory /var/ldap/first-dit
dbnosync
dirtyread
index mail pres,eq
index objectclass pres
index default eq,sub
index sn eq,sub,subinitial
index telephone number
index cn
searchstack 5
```

Та же конфигурация с использованием файла `DB_CONFIG`:

Фрагмент `slapd.conf`:

```
database bdb
...
# только фрагмент раздела database
# порядок директив не важен
cachesize 10000
directory /var/ldap/first-dit
dirtyread
index mail pres,eq
index objectclass pres
index default eq,sub
index sn eq,sub,subinitial
index telephone number
index cn
searchstack 5
```

Файл `DB_CONFIG`:

Файл `DB_CONFIG` располагается в директории, указанной в директиве `directory`. Используемые в этом файле директивы полностью описаны в [руководстве по BDB](#). Можно заменить директивы `checkpoint`, `lockdetect`, `dbnosync` файла `slapd.conf` эквивалентными директивами файла `DB_CONFIG`, а также добавить дополнительные директивы настройки производительности по мере необходимости.

```
set lk_detect DB_LOCK_DEFAULT
set_flags DB_TXN_NOSYNC
set_lg_max 5242880
set_cachesize 0 5242880 1
txn_checkpoint 128 15 0
```



Глава 6. OpenLDAP slapd.conf, база данных dnssrv

Уже совсем скоро (One day real soon now TM).



Глава 6. OpenLDAP slapd.conf, база данных ldap

Уже совсем скоро (One day real soon now TM).



Глава 6. OpenLDAP slapd.conf, база данных ldbm

Уже совсем скоро (One day real soon now TM).



Глава 6. OpenLDAP slapd.conf, база данных meta

Уже совсем скоро (One day real soon now TM).



Глава 6. OpenLDAP slapd.conf, база данных monitor

Уже совсем скоро (One day real soon now TM).



Глава 6. OpenLDAP slapd.conf, база данных sql

Уже совсем скоро (One day real soon now TM).



Глава 6. Наложение OpenLDAP accesslog

Наложение **accesslog** используется для отслеживания всех или избранных операций, производимых в конкретном DIT (целевое DIT) путём записывания деталей этих операций в виде записей в другое DIT (accesslog DIT). Это accesslog DIT можно исследовать с помощью стандартных запросов LDAP. Параметры наложения accesslog управляют тем, нужно ли помещать в журнал сведения о всех, либо только о части операций LDAP в целевом DIT ([logops](#)), следует ли сохранять связанную информацию, такую как предыдущее содержимое атрибутов или записей ([logold](#) и [logoldattr](#)), а также когда удалять записи из accesslog DIT. Записи accesslog DIT сохраняются с использованием объектных классов и атрибутов из специализированного [набора схемы данных аудита](#).

Данное наложение создаёт accesslog DIT как DIT общего назначения, которое может использоваться, например, для операций LDAP или аудита журнала операций целевого DIT. Кроме того, accesslog DIT может специализированно применяться в директиве [syncrepl](#) для [delta-репликации](#) или [delta-синхронизации](#). В последнем случае, поскольку accesslog DIT сконфигурировано только на хранение информации об операциях изменения LDAP, выполнявшихся в целевом DIT, оно представляет собой очень точный и подробный отчёт об изменениях атрибутов целевого DIT. При нормальной синхронизации syncrepl, когда изменяется любой из атрибутов записи целевого DIT, во время цикла репликации изменённая запись передаётся целиком. Когда syncrepl используется совместно с accesslog DIT, во время цикла репликации требуется передавать только эти изменения атрибута (а также связанные с ними операционные атрибуты, такие как entryCSN и т.п.).

Обзор использования

Наложение accesslog определяется в [разделе database](#) целевого DIT — таким образом, создаётся привязка целевого DIT к конкретному accesslog DIT (это означает, что на одном сервере LDAP может быть несколько accesslog DIT). Затем accesslog DIT определяется как отдельный раздел database и, если данное accesslog DIT будет использоваться для организации delta-репликации, оно должно быть настроено как поставщик репликации путём использования [наложения syncprov](#). Пример:

```
# раздел глобальных параметров slapd.conf
...
# раздел database целевого DIT
database bdb
suffix "dc=example,dc=com"
...
# указываем, что данное целевое DIT использует журналирование accesslog
overlay accesslog
# определяем, какое accesslog DIT будет использоваться данным целевым DIT
logdb "cn=accesslogname"
# остальные необходимые директивы
...
# если это целевое DIT реплицируется в стиле syncrepl
# необходимо определить наложение syncprov
overlay syncprov
...

# accesslog DIT
database bdb
# суффикс должен совпадать со значением, указанным
# в директиве logdb ассоциированного целевого DIT
suffix "cn=accesslogname"
...
# требуется, если accesslog используется в delta-syncrepl
overlay syncprov
...
```

Примечание: При использовании в конфигурации delta-синхронизации syncrepl и целевое DIT, и accesslog DIT должны быть поставщиками syncrepl (overlay syncprov). Это необходимо для того, чтобы разрешить процедурам синхронизации пользоваться обоими этими DIT как источниками репликации. Например, если потребитель syncrepl пуст, то для первоначальной синхронизации потребуется целевое DIT, после чего в нормальном режиме будет использоваться accesslog DIT.

Подробное описание конфигурации delta-репликации [здесь](#).

Директивы конфигурации

Приведённые ниже директивы slapd.conf применяются к наложению журнала доступа (accesslog). Они

должны определяться после директивы *overlay accesslog*.

Директива logdb

logdb suffix

Директива определяет суффикс accesslog DIT, которое будет использоваться для хранения записей журнала, относящихся к DIT, определяемому в том разделе database, в котором присутствует данная директива *overlay accesslog* (то есть к целевому DIT). accesslog DIT может определяться для одного или нескольких разделов database (целевых DIT), каждый из которых будет ссылаться на различный суффикс accesslog. accesslog DIT должно быть определено в каком-либо месте конфигурации с помощью другой директивы и раздела [database](#). Списки контроля доступа **access to** для определения accesslog DIT должны ограничивать доступ только к этому DIT. Наложение accesslog автоматически создаст запись суффикса (корневую запись) в accesslog DIT, и потому для его инициализации не требуется применение файла LDIF. Каждая операция, сохраняемая в accesslog DIT, будет добавляться в виде дочерней записи в корневую запись. Пример:

```
# фрагмент slapd.conf
...
# целевое DIT
database bdb
suffix "dc=example,dc=com"
...
overlay accesslog
logdb "cn=mylog"
...

# accesslog DIT
database bdb
suffix "cn=mylog"
...
```

Директива logops

logops operations

Директива определяет, какие типы операций в целевом DIT будут помещаться в accesslog DIT. Аргументы директивы задаются в виде списка, разделяемого пробельными символами. Допустимые типы операций: abandon (отказ от подключения), add (добавление), bind (подсоединение), compare (сравнение), delete (удаление), extended (расширенная операция), modify (модификация), modrdn (модификация rdn), search (поиск) и unbind (отсоединение). Для общепринятых наборов операций могут использоваться псевдонимы:

```
writes – add, delete, modify, modrdn
reads – compare, search
session – abandon, bind, unbind
all – все операции
```

Если наложение используется в целях delta-синхронизации, как правило, настраивается журналирование только тех операций, которые изменяют содержимое DIT, то есть используется только **writes**. Как видно из списка поддерживаемых типов операций, с помощью данного наложения можно организовать всеобъемлющее журналирование в целях аудита. Примеры:

```
# при использовании с delta-синхронизацией
logops writes

# всеобъемлющее журналирование
logops writes reads session
# ИЛИ
logops all

# только избранные операции
logops delete add bind
```

Директива logold

logold filter

Директива определяет фильтр для сопоставления с записями, которые удаляются или модифицируются. Если запись соответствует фильтру, старое содержимое данной записи будет сохранено в журнале наряду с текущей операцией. Данная возможность не используется в delta-синхронизации, но может быть важным требованием при использовании accesslog для аудита. Пример:

```
# при операциях delete или modify, перед тем,
# как помещать в журнал сведения об операции,
# в него помещается содержимое записи,
# если в ней есть объектный класс inetorgperson
# и определён атрибут uid
logold (&(objectclass=inetorgperson)(uid=*))
```

Директива logoldattr

logoldattr attr [...]

Директива определяет список атрибутов, прежнее содержимое которых будет всегда помещаться в журнал при операциях modify и modRDN. По умолчанию в журнал помещается только новое содержимое атрибутов, изменённых в операции modify, а при запросах modRDN сведения об атрибутах в журнал вообще не помещаются. Для delta-синхронизации не используется. Пример:

```
# всегда сохраняется текущее состояние атрибутов uid и cn
# до выполнения любой операции modify или modRDN
logoldattr uid cn
```

Директива logpurge

logpurge age interval

Директива определяет сразу максимальный возраст записей журнала, которые будут храниться в базе данных, и частоту сканирования базы данных на предмет устаревших записей. Оба аргумента **age** и **interval** указываются как промежуток времени в днях, часах, минутах и секундах. Формат времени: [ddd+]hh:mm[:ss], дни и секунды — необязательные компоненты, а часы и минуты — обязательные. Число дней может содержать до пяти цифр, все остальные числовые поля должны содержать ровно две цифры. Пример:

```
# база данных журнала будет сканироваться каждый день
# записи старше двух дней будут удалены
logpurge 2+00:00 1+00:00
```

При использовании базы данных журнала, поддерживающей упорядоченное индексирование атрибутов типа generalizedTime, задание индекса **eq** на атрибут reqStart увеличит производительность операций очистки.

Директива logsuccess

logsuccess TRUE | FALSE

Если эта директива установлена в TRUE, то в журнал accesslog будут помещаться только записи об успешно выполненных запросах, то есть запросах, возвративших результирующий код 0 (LDAP_SUCCESS). Если директива установлена в FALSE, в accesslog будут добавляться записи об указанных операциях независимо от их результирующего кода. Значение по умолчанию — FALSE. При использовании с delta-синхронизацией logsuccess должна быть установлена в TRUE.

Примеры конфигурации:

```
# фрагмент slapd.conf
# раздел глобальных настроек
...
# определение целевой DIT
```

```

database bdb
suffix dc=example,dc=com
...
# определяем журнал аудита общего назначения,
# содержащий все операции (включая завершившиеся неудачей)
# в данном целевом DIT
overlay accesslog
logdb "cn=auditlog"
logops writes reads
# перечитывать журнал каждые 5 дней и производить чистку
# записей которые старше 30 дней
logpurge 30+00:00 5+00:00
# дополнительно - сохраняем предыдущее содержимое записей с
# объектным классом person до выполнения операций записи
logold (objectclass=person)
# если требуется только содержимое конкретных атрибутов,
# нужно использовать logoldattr
logoldattr uid cn

# определение accesslog DIT
database bdb
suffix "cn=auditlog"
...
index reqStart eq
# ACL - разрешаем выполнять поиск конкретному клиенту
access to *
by dn.base="cn=admin,dc=example,dc=com" read

```

Пример конфигурации поставщика при использовании delta-синхронизации:

```

# фрагмент slapd.conf
# раздел глобальных настроек
...
# определение целевой DIT
database bdb
suffix dc=example,dc=com
...
# определяем журнал, приемлемый для использования с delta-syncrepl
# в журнал помещаются лишь успешные операции, модифицирующие содержимое
# данного целевого DIT
overlay accesslog
logdb "cn=deltalog"
logops writes
# заносим в журнал только успешные операции
# операции записи, завершившиеся неудачей, не меняют содержимого данного DIT
logsuccess TRUE
# проверяем журнал ежедневно и чистим записи старше двух дней
logpurge 2+00:00 1+00:00

# определение accesslog DIT
database bdb
suffix "cn=deltalog"
...

# ACL - разрешаем выполнять поиск для delta-синхронизации
# данный DN должен совпадать с указываемым в параметре binddn
# директивы syncrepl потребителя
access to *
by dn.base="cn=admin,dc=example,dc=com" read

```

Набор схемы данных accesslog

Наложение accesslog использует следующий специализированный набор схемы данных. Этот набор схемы загружается в двоичном формате с наложением accesslog и потому не поставляется в виде отдельного файла набора схемы. Мы публикуем его для того, чтобы, изучив его, Вы могли конструировать поисковые запросы для изучения содержимого accesslog DIT.

Существует основной объектный класс auditObject, от которого унаследованы два дополнительных объектных класса — auditReadObject и auditWriteObject. Далее от этих объектных классов унаследованы объектные классы для всех типов операций LDAP. Данная иерархия объектных классов разработана для того, чтобы позволить осуществлять гибкие и эффективные поисковые запросы записей журнала, основываясь либо на классе конкретного типа операции, либо на более обобщённой классификации. Определение класса auditObject выглядит следующим образом:

```

( 1.3.6.1.4.1.4203.666.11.5.2.1
  NAME 'auditObject'
  DESC 'OpenLDAP request auditing'
  SUP top STRUCTURAL

```

```

MUST ( reqStart $ reqType $ reqSession )
MAY ( reqDN $ reqAuthzID $ reqControls $ reqRespControls
      $ reqEnd $ reqResult $ reqMessage $ reqReferral ) )
# Имейте ввиду, что все OID, используемые в наборе схемы журнала,
# в настоящее время принадлежат ветке OpenLDAP Experimental.
# Предполагается, что в будущем они будут перемещены в ветку Standard.
# Обзор атрибутов:
# reqStart и reqEnd представляют собой, соответственно, время начала
# и конца операции. Они используют синтаксис generalizedTime.
# Атрибут reqStart также используется в качестве RDN для каждой записи журнала.
# Атрибут reqType - простая строка, содержащая тип операции, помещаемой в журнал,
# например, add, delete, search, и т.д. Для расширенных операций,
# тип также включает OID расширенной операции, например: extended(1.1.1.1)
# Атрибут reqSession - зависящий от реализации идентификатор, общий для всех операций,
# ассоциированных с одной и той же сессией LDAP. В настоящее время - это внутренний
# идентификатор соединения slapd, хранящийся в виде десятичного числа.
# Атрибут reqDN - это distinguishedName целевой записи для производимой операции.
# Например, для запросов Bind - это Bind DN, для запросов Add - это DN добавляемой записи,
# для запросов Search - это базовый DN поиска.
# Атрибут reqAuthzID - это distinguishedName пользователя, выполняющего операцию.
# Обычно это то же самое имя, которое ассоциировалось с сессией при её установке
# с помощью запроса Bind (если таковой был), но оно может быть изменено в зависимости
# от различных обстоятельств.
# Атрибуты reqControls и reqRespControls содержат любые управляющие последовательности,
# посылаемые клиентом при запросе и возвращаемые сервером в ответе соответственно.
# Значения атрибута - это просто неинтерпретируемые строки байт.
# Атрибут reqResult - это числовой результирующий код LDAP операции, указывающий
# либо на успешное завершение операции, либо на код конкретной ошибки LDAP.
# Код ошибки может сопровождаться текстовым сообщением об ошибке, которое будет
# записываться в атрибут reqMessage.
# Атрибут reqReferral содержит любые отсылки, которые были возвращены с результатом запроса.

# Специфические для операций классы определяются с дополнительными атрибутами
# для хранения всех соответствующих параметров, связанных с операцией:

( 1.3.6.1.4.1.4203.666.11.5.2.4
  NAME 'auditAbandon'
  DESC 'Abandon operation'
  SUP auditObject STRUCTURAL
  MUST reqId )
# Для операции Abandon атрибут reqId содержит идентификатор сообщения того запроса,
# который был отменён.

( 1.3.6.1.4.1.4203.666.11.5.2.5
  NAME 'auditAdd'
  DESC 'Add operation'
  SUP auditWriteObject STRUCTURAL
  MUST reqMod )
# Класс Add унаследован от класса auditWriteObject. Классы Add и Modify очень похожи.
# Атрибут reqMod содержит все атрибуты добавляемой записи (или, в случае
# операции Modify, все выполняемые модификации). Формат значений:
# attribute:<+|-|=|#> [ value]
# Здесь '+' указывает, что значение добавляется (add), '-' - что значение удаляется
# (delete), '=' - что значение заменяется (replace), а '#' - что значение
# инкрементируется (increment). В операциях Add все значения атрибута reqMod
# будут иметь обозначение '+'.

( 1.3.6.1.4.1.4203.666.11.5.2.6
  NAME 'auditBind'
  DESC 'Bind operation'
  SUP auditObject STRUCTURAL
  MUST ( reqVersion $ reqMethod ) )
# Класс Bind включает атрибут reqVersion, содержащий версию протокола LDAP,
# указываемую в операции Bind, а также атрибут reqMethod, содержащий Bind Method,
# используемый в операции Bind. Это будет строка SIMPLE для LDAP Simple Bind
# или SASL(mech) для SASL Bind. Имейте ввиду, без специальных настроек
# (переопределения на глобальном уровне), будут приниматься только
# Simple Bind подсоединения с использованием DN из текущей базы данных.

( 1.3.6.1.4.1.4203.666.11.5.2.7
  NAME 'auditCompare'
  DESC 'Compare operation'
  SUP auditObject STRUCTURAL
  MUST reqAssertion )
# Для операций Compare атрибут reqAssertion содержит Attribute Value Assertion,
# используемое в запросах сравнения (compare).

( 1.3.6.1.4.1.4203.666.11.5.2.8
  NAME 'auditDelete'
  DESC 'Delete operation'
  SUP auditWriteObject STRUCTURAL
  MAY reqOld )
# Для операций Delete не требуется дополнительных параметров. Однако,
# атрибут reqOld может опционально использоваться для сохранения содержимого
# записи перед её удалением. Формат значений:
# attribute: value
# Атрибут reqOld заполняется только в случае, если удаляемая запись

```



```
# совпадает с фильтром, указанным в директиве logold.

( 1.3.6.1.4.1.4203.666.11.5.2.9
  NAME 'auditModify'
  DESC 'Modify operation'
  SUP auditWriteObject STRUCTURAL
  MAY reqOld
  MUST reqMod )
# Операция Modify содержит описание модификаций в атрибуте reqMod, который уже
# был описан выше в операции Add. Данная операция опционально может содержать
# предыдущие значения любых изменившихся атрибутов в атрибуте reqOld
# с использованием того же формата, что был описан выше для операции Delete.
# Атрибут reqOld заполняется только в случае, если модифицируемая запись
# совпадает с фильтром, указанным в директиве logold.

( 1.3.6.1.4.1.4203.666.11.5.2.10
  NAME 'auditModRDN'
  DESC 'ModRDN operation'
  SUP auditWriteObject STRUCTURAL
  MUST ( reqNewRDN $ reqDeleteOldRDN )
  MAY ( reqNewSuperior $ reqOld ) )
# Класс ModRDN использует атрибут reqNewRDN для хранения нового RDN запроса.
# Атрибут reqDeleteOldRDN - значение типа Boolean, содержащее TRUE
# если старое RDN было удалено из записи, или FALSE если старое RDN было сохранено.
# Атрибут reqNewSuperior содержит DN новой родительской записи если запрос
# определяет новую родительскую запись. Атрибут reqOld заполняется только в случае,
# если модифицируемая запись совпадает с фильтром, указанным в директиве logold,
# и содержит атрибуты из списка, указанного в директиве logoldattr.

( 1.3.6.1.4.1.4203.666.11.5.2.11
  NAME 'auditSearch'
  DESC 'Search operation'
  SUP auditReadObject STRUCTURAL
  MUST ( reqScope $ reqDerefAliases $ reqAttrsOnly )
  MAY ( reqFilter $ reqAttr $ reqEntries $ reqSizeLimit $
    reqTimeLimit ) )
# В классе Search атрибут reqScope содержит диапазон оригинального поискового запроса
# и использует значения, указываемые в формате LDAP URL, например, base, one, sub
# или subord. Атрибут reqDerefAliases принимает одно из значений never, finding,
# searching или always, означающих каким образом во время поиска будут
# обрабатываться псевдонимы. Атрибут reqAttrsOnly - значение типа Boolean,
# устанавливаемое в TRUE если запрашивались только имена атрибутов, или в FALSE
# если запрашивались атрибуты со своими значениями. Атрибут reqFilter содержит фильтр,
# использовавшийся в поисковом запросе. В атрибуте reqAttr перечисляются запрашиваемые
# атрибуты, если таковые указывались при запросе. Атрибут reqEntries - целое число,
# означающее количество записей, которые были возвращены на данный поисковый запрос.
# Атрибуты reqSizeLimit и reqTimeLimit показывают, какие ограничения были запрошены
# клиентом на данную операцию поиска.

( 1.3.6.1.4.1.4203.666.11.5.2.12
  NAME 'auditExtended'
  DESC 'Extended operation'
  SUP auditObject STRUCTURAL
  MAY reqData )
# Класс Extended представляет LDAP Extended Operation. Как было сказано выше,
# в атрибут reqType родительского класса включается актуальный OID операции.
# Если с запросом предоставлялась какая-либо дополнительная информация,
# она будет помещена в атрибут reqData как неинтерпретируемая строка байт.
```

Глава 6. Наложение OpenLDAP chain

Уже совсем скоро (One day real soon now ™)



Глава 6. Наложение OpenLDAP dynlist

Наложение dynlist позволяет конкретному атрибуту конкретного объектного класса (оба указываются в параметрах наложения) дополнять результаты операции поиска при их возвращении. То есть, когда запись (DN) с явно заданным адресом, либо полученная в результате поиска, содержит указанный объектный класс, то в ответ на поисковый запрос будут возвращены результаты другого поискового

запроса по URL, определённом в указанном атрибуте этой записи. Самое распространённое применение данной функции - то, что называют динамическими группами (Dynamic Groups). В отличие от статических групп, определяемых с помощью объектного класса `groupOfMembers`, динамические группы обычно создаются с помощью объектного класса `groupOfURLs` с параметром поиска, определённом в атрибуте `memberURL`, содержащем стандартные [поисковые URL LDAP](#). Динамические группы - нестандартная (не определённая в RFC), но получившая широкую реализацию возможность LDAP. Динамические группы могут использоваться опосредствованно для нахождения специфических подмножеств DIT, либо непосредственно для создания групп, которым затем могут быть назначены полномочия и права на доступ (с помощью директив [access to](#)).

Как видно (по состоянию на апрель 2012 года), проводится активная работа по стандартизации динамических групп, и, как следствие, к окончанию этой работы в данной области могут произойти значительные изменения. В различных документах не рекомендуется использовать текущую реализацию в промышленных целях как раз по причине этого отсутствия стандартизации. Тем не менее, динамические группы могут быть невероятно полезны (а в некоторых случаях вообще могут оказаться единственно возможным вариантом решения) и потому применяются очень широко.

Конфигурация

В OpenLDAP dynlist реализован в виде наложения. Его конфигурация приведена в следующем фрагменте `slapd.conf`:

```
...
# подключение требуемого для dynlist набора схемы dyngroup
include dyngroup.schema
...
# требуется, если наложение было собрано динамически
# указание без полного пути требует определения директивы modulepath
# если модули не находятся в стандартном расположении PATH
loadmodule dynlist.la
# ИЛИ
loadmodule dynlist.so
...
database bdb
suffix "dc=example,dc=com"
...
# устанавливаем dynlist только для данного DIT
# это наложение можно также поместить в раздел глобальных настроек,
# в этом случае оно будет применяться ко всем DIT
overlay dynlist
dynlist-attrset dyn-objc [URI] url-attr [[map-dn:]member-attr]
...
# другие директивы overlay
# или следующая директива database
```

Далее следует описание каждого параметра директивы `dynlist-attrset`:

dyn-objc

Обязательный параметр. Определяет объектный класс, который будет использоваться для запуска процесса динамического расширения. Обычно это объектный класс `groupOfURLs`. При использовании данного объектного класса в DIT создаются записи в такой форме:

```
# создание групповой записи dyngroup
dn: cn=dyngroup,ou=groups,dc=example,dc=com
objectclass: groupOfURLs
cn: dyngroup
memberURL: ldap:///ou=people,dc=example,dc=com?cn,sn?one?(ou=it*)
```

В данном случае, когда при операции поиска встречается DN `cn=dyngroup,ou=groups,dc=example,dc=com`, это вызывает на сервере LDAP срабатывание функции динамической группы.

URI

Описание скоро будет.

url-attr

Обязательный параметр. Определяет атрибут записи, в котором содержится [поисковый URL](#). Если в записи есть объектный класс, указанный в параметре **dyn-objc**, то будет вызвана операция поиска по URL, хранящемуся в этом атрибуте, и результаты этого поиска будут возвращены обратно в запись. При использовании с объектным классом **groupOfURLs** этим атрибутом обычно является [memberURL](#). Вот пример такого атрибута memberURL:

```
memberURL: ldap:///ou=people,dc=example,dc=com?cn,sn?one?(ou=it*)
```

В данном случае указанный [поисковый LDAP URL](#) создаст поисковый запрос на локальном сервере (это следует из синтаксиса ///) с базой поиска ou=people,dc=example,dc=com только по записям на один уровень ниже (**one**), возвращающий атрибуты cn и sn любых записей, в которых атрибут ou (organizationalUnitName) начинается с "it", либо содержит только "it" (поиск без учёта регистра символов). Члены данной группы (которые будут возвращены в результате поиска) могут затем использоваться для получения каких-либо привилегий при помощи директив [access to](#) (или атрибутов olcAccess в случае cn=config).

Этот пример иллюстрирует использование локального поиска, но это не единственный вариант. Вот корректный LDAP URL для удалённого сервера:

```
memberURL: ldap://ldap.example.net/ou=people,dc=example,dc=com?cn,sn?one?(ou=it*)
```

В зависимости от скорости передачи и объёмов возвращаемых данных, в результате подобного поиска могут возникнуть значительные задержки времени при обслуживании.

map-dn

Описание скоро будет.

member-attr

Необязательный параметр. Если **member-attr** присутствует, поведение динамической группы будет отличаться. В этом случае поисковый запрос вернёт DN всех записей, удовлетворивших критериям поиска, в указанный атрибут (который должен присутствовать в записи, содержащей объектный класс "срабатывания", и тип которого должен быть distinguishedName). Если объектный класс "срабатывания" - **groupOfURLs**, то в нём есть два атрибута, позволяющих хранить DN: owner и seeAlso. Когда в определении dynlist-attrset используется **member-attr**, список возвращаемых атрибутов в [поисковом LDAP URL](#) должен быть пустым. Следующие фрагменты демонстрируют использование этого типа конфигурации:

```
# фрагмент slapd.conf
...
overlay dynlist
dynlist-attrset groupOfURLs memberURL owner
...

# создание групповой записи dyngroup
dn: cn=dyngroup,ou=groups,dc=example,dc=com
objectclass: groupOfURLs
cn: dyngroup
# пустой список атрибутов в поисковом URL
memberURL: ldap:///ou=people,dc=example,dc=com??one?(ou=it*)
```

В данном случае DN всех записей, удовлетворяющих критериям поиска, будут возвращены в атрибут **owner**.

Уже совсем скоро (One day real soon now ™).



Глава 6. Наложение политики паролей OpenLDAP

Модуль `ppolicy` предоставляет возможности расширенного управления паролями, применяемые к попыткам подключения к OpenLDAP не от имени `rootdn`. Стандартное наложение `ppolicy` предоставляет следующие контролируемые пользователем возможности:

1. Срок действия пароля (может быть определен как минимальный, так и максимальный срок действия). (не равно 0)
2. Поддержка истории паролей для предотвращения повторного использования одинакового пароля в течение определённого периода времени.
3. Качество пароля — новые пароли могут проверяться по различным характеристикам.
4. Максимальное количество последовательных неудачных попыток аутентификации.
5. Автоматическая блокировка учётной записи.
6. Автоматическая или административная разблокировка учётной записи.
7. Отсрочка блокировки (допускается использование просроченных паролей на ограниченное количество попыток подключения).
8. Политики паролей могут определяться так, что их действие будет распространяться на всё DIT, конкретных пользователей или группы пользователей, либо в любом сочетании существующих возможностей.

Спецификация такой функциональности описана в черновом RFC [draft-behera-ldap-password-policy-09.txt](#), которое, если память нам не изменяет, остаётся в таком состоянии с 2005 года.

Примечание: Стандартный модуль `ppolicy` при определённых условиях может вызывать чрезмерное количество блокировок, что добавляет работы администраторам и приводит к разочарованию пользователей. В частности, так происходит при внедрении функции устаревания пароля и работе с утилитами LDAP, кэширующими пароли. Если закешированный пароль не был изменён сразу же после того, как пользователь сменил свой пароль, то при очередной попытке подключения будет использоваться предыдущий (закешированный) пароль, что приведёт к сбою подключения. В большинстве случаев при ошибках подключения утилита выполняет несколько повторных попыток подключения (3, 5 и более попыток — не редкость), что, в свою очередь, может привести к превышению пороговых значений попыток неудачного подключения и блокировки учётной записи. Mozilla Corporation попросила Zytrax разработать модифицированную версию `ppolicy`, которая делает различия между повторными попытками входа с использованием одного и того же пароля и попытками входа с использованием различающихся паролей (как бывает при атаках подбора пароля по словарю), и использует разные стратегии блокировки для разных условий. Модифицированная версия в настоящий момент используется на промышленных серверах (версий 2.4.11 и 2.4.16) и господин Aravind Gottipati из Mozilla Corporation написал об этом [заметку в своём блоге](#). Mozilla Corporation сделала данные модификации свободно доступными и их можно скачать с сайта [zytrax.com](#) для [OpenLDAP версии 2.4.11](#) и [OpenLDAP версии 2.4.16](#). Полные инструкции и модифицированные map-страницы прилагаются. Документация на этой странице относится и к стандартной версии модуля от OpenLDAP, и к модифицированной версии. Части документации, относящиеся к модифицированной версии, заключены в теги `<MODIFIED>` `</MODIFIED>`.

Содержание

[Обзор использования](#)
[Директивы конфигурации OpenLDAP](#)
[Объектный класс `pwdPolicy` и его атрибуты](#)
[Операционные атрибуты](#)
[Набор схемы данных `ppolicy`](#)
[Разблокировка учётной записи](#)
[Примеры конфигурации](#)

Обзор использования

Политика паролей устанавливается на определённое DIT комбинацией подключения файла набора схемы данных `ppolicy.schema`, указания директивы `overlay ppolicy` и опциональных директив конфигурации `ppolicy`, а также определения одной или нескольких [записей политик паролей](#). Модуль политики паролей также генерирует несколько [операционных атрибутов](#), которые могут быть использованы для проверки статуса (а в некоторых случаях — для управления) политиками паролей пользовательских записей.

Установка наложения политики паролей при использовании OLC (`cn=config`):

1. Загрузить файл `ppolicy.schema` с помощью описанной [здесь](#) процедуры.
2. Если модуль `ppolicy` подгружается динамически, его необходимо добавить как атрибут **`olcModuleLoad`** с помощью описанной [здесь](#) процедуры.
3. Запись наложения `ppolicy` необходимо добавить для соответствующей базы данных с помощью описанной [здесь](#) процедуры.
4. Запись наложения `ppolicy` создаётся с использованием объектного класса [`olcPPolicyConfig`](#), а затем в данную запись добавляются специфичные для этого объектного класса атрибуты как показано ниже.

Установка наложения политики паролей на конкретное DIT при использовании `slapd.conf`:

```
...
# подключение набора схемы ppolicy
include ppolicy.schema
...
# требуется, если наложение было собрано динамически
loadmodule ppolicy.la
# ЛИБО
loadmodule ppolicy.so
...
database bdb
suffix "dc=example,dc=com"
...
# включение политик паролей только для данного DIT
overlay ppolicy
# опционально - директивы ppolicy
...
# другие директивы overlay
# либо следующая директива database
```

Атрибуты/директивы конфигурации OpenLDAP

`olcPPolicyDefault` (`ppolicy_default`)

```
# форма OLC (cn=config)
olcPPolicyDefault: DN

# форма slapd.conf
ppolicy_default DN
# пример
# форма OLC (cn=config)
olcPPolicyDefault: cn=defaultppolicy,dc=example.com

# форма slapd.conf
ppolicy_default "cn=defaultppolicy,dc=example.com"
```

Данный атрибут/директива определяет DN [политики паролей](#) по умолчанию, которая будет применяться при отсутствии у пользователя какой-либо конкретной политики паролей.

Таким образом, директива `ppolicy_default` определяет политику паролей для всего DIT и является самым простым путём задействования возможностей политики паролей. Как только задана политика паролей по умолчанию, дополнительная конфигурация не требуется. Если присутствует специфичная политика паролей для записи пользователя (определяется с помощью атрибута [`pwdPolicySubentry`](#)), всегда будет применяться она, а не политика по умолчанию.

Если атрибут **`olcPPolicyDefault`** (директива **`ppolicy_default`**) не определялся и для записи пользователя нет специфичной политики паролей, то для этой записи пользователя не будет применяться никакой политики

паролей. Таким образом существует возможность определить общую политику для всех пользователей, специфичную политику для каждого пользователя, либо применять политику паролей только для ограниченного количества пользователей или класса пользователей.

olcPPolicyHashCleartext (ppolicy_hash_cleartext)

```
# форма OLC (cn=config)
olcPPolicyHashCleartext: TRUE | FALSE

# форма slapd.conf
ppolicy_hash_cleartext
# данная директива не принимает никаких параметров
ppolicy_hash_cleartext
```

Данный атрибут/директива указывает серверу сохранять в DIT пароли, вводимые в открытом виде с помощью стандартных запросов Add или Modify, применяя к ним установленный по умолчанию для сервера алгоритм хэширования. Если эта директива не используется, такие пароли будут сохраняться в открытом виде, и это означает, что в ответ на простые поисковые запросы будут возвращаться пароли в открытом виде. Чтобы защититься от такой напасти, ко всем парольным атрибутам необходимо закрывать доступ на чтение для всех пользователей (кстати, неплохая политика даже при использовании хэшированных паролей). В данном атрибуте/директиве нет необходимости, если все LDAP-клиенты способны выполнять и выполняют расширенную операцию Password Modify.

olcPPolicyUseLockout (ppolicy_use_lockout)

```
# форма OLC (cn=config)
olcPPolicyUseLockout: TRUE | FALSE

# форма slapd.conf
ppolicy_use_lockout
# данная директива не принимает никаких параметров
ppolicy_use_lockout
```

Попытки подключения от имени заблокированной учётной записи всегда будут возвращать ошибку (по умолчанию **InvalidCredentials**). Данный атрибут/директива позволяет серверу возвращать специфичный код ошибки **AccountLocked**. Хотим предупредить, что возврат кода ошибки AccountLocked может оказать услугу хакерам. Если они исследуют реакцию на те или иные попытки получения доступа, выясняя тем самым рабочие установки сервера (такие, например, как пороговое значение неверных попыток до блокировки учётной записи), то возврат AccountLocked может помочь им в последующих атаках.

olcPPolicyForwardUpdates (ppolicy_forward_updates)

```
# форма OLC (cn=config)
olcPPolicyForwardUpdates: TRUE | FALSE

# форма slapd.conf
ppolicy_forward_updates
# данная директива не принимает никаких параметров
ppolicy_forward_updates
```

Применять данный атрибут/директиву можно только в конфигурации подчинённого сервера (потребителя) репликации. Наличие этого атрибута/директивы указывает, что нарушения политики паролей не будут помещаться на данный подчинённый сервер (потребитель), а будут сразу записываться на главный сервер (поставщик) репликации. При наличии данного атрибута/директивы необходимо, чтобы в конфигурации сервера присутствовал атрибут/директива [olcUpdateRef/updateref](#) и было настроено наложение [chain](#).

Объектный класс pwdPolicy и его атрибуты

Чтобы установить политики паролей, должны быть определены одна или несколько записей, основанных на вспомогательном (AUXILIARY) объектном классе pwdPolicy. Эти записи политик могут быть установлены как для конкретных учётных записей пользователей, так и для классов учётных записей, а также для всего DIT в целом. Если требуется установка на уровне конкретной учётной записи, то к записи пользователя, основанной, например, на объектном классе **inetOrgPerson**, требуется добавить атрибут [pwdPolicySubentry](#), в котором указывается DN записи той политики паролей, которая будет применяться к данной учётной

записи. Определение набора схемы данных для объектного класса `pwdPolicy` можно посмотреть [здесь](#). В последующих подразделах описывается каждый атрибут данного объектного класса. Несколько рабочих примеров представлены [далее](#).

pwdAllowUserChange

```
pwdAllowUserChange TRUE | FALSE
```

```
# пример  
pwdAllowUserChange TRUE
```

Данный атрибут управляет тем, разрешено ли пользователям менять собственные пароли. Если значение этого атрибута — `TRUE` (по умолчанию), то пользователям разрешено менять свои пароли. Если значение этого атрибута — `FALSE`, то пользователям не разрешено менять свои пароли (поменять им пароли может только администратор).

pwdAttribute

```
pwdAttribute attribute-name
```

```
# пример  
pwdAttribute secretCode
```

Данный атрибут позволяет дополнительно расширить политику паролей, чтобы она распространялась на разные парольные атрибуты. Значение по умолчанию (в данный момент поддерживается только оно) — **userPassword**. Таким образом, политика паролей (в настоящее время) применяется только к операциям подсоединения, использующим пароль, который хранится в атрибуте **userPassword**. Пока можно безболезненно не указывать этот атрибут.

pwdCheckModule

```
pwdCheckModule name-of-module
```

```
# пример  
pwdCheckModule pwcheck.la
```

Данный атрибут определяет название пользовательского модуля проверки качества пароля, который будет вызываться для выполнения проверки качества пароля и имеет смысл только в том случае, когда атрибут [pwdCheckQuality](#) установлен в 1 или 2, во всех остальных случаях данный атрибут можно не указывать.

Примечание: `pwdCheckModule` — нестандартное расширение политик паролей LDAP.

pwdCheckQuality

```
pwdCheckQuality value
```

```
# пример  
pwdCheckQuality 0
```

Если значение данного параметра — 0 (по умолчанию), то никаких проверок паролей не проводится. Если его значение — 1 и предоставляется пароль в открытом виде, то для проверки качества пароля будет вызвана пользовательская функция (если таковая определена в атрибуте [pwdCheckModule](#)). Если эта пользовательская функция недоступна, то пароль будет принят (если, опять же, он успешно пройдёт все остальные тесты, определённые в различных атрибутах **pwdPolicy**). Если значение данного атрибута — 2 и предоставляется пароль в открытом виде, то для проверки качества пароля будет вызвана пользовательская функция. Если эта пользовательская функция недоступна, то пароль будет отклонён. Если пользовательский модуль проверки качества паролей не предоставляется, данный атрибут можно безболезненно не указывать.

pwdExpireWarning

```
pwdExpireWarning time-in-seconds
```

```
# пример
```



```
pwdExpireWarning 3600
```

Данный атрибут управляет предупреждающими сообщениями об устаревании пароля, которые возвращаются пользователю при попытке подключения. Если его значение — 0 (по умолчанию), то при попытках подключения предупреждающих сообщений выдаваться не будет до того момента, пока пароль всё ещё действителен. Если значение больше 0, то оно определяет время (в секундах) до момента устаревания пароля, при наступлении которого предупреждающие сообщения об устаревании пароля начнут возвращаться при попытках подключения.

pwdFailureCountInterval

```
pwdFailureCountInterval time-in-seconds
```

```
# пример  
pwdFailureCountInterval 20
```

Данный атрибут управляет тем, когда сбрасывается счётчик последовательных неудачных попыток ввода пароля. Если его значение — 0 (по умолчанию), то счётчик последовательных неудачных попыток ввода пароля сбрасывается только в случае успешного прохождения аутентификации. Если значение больше 0, то оно определяет время (в секундах), по прошествии которого счётчик последовательных неудачных попыток ввода пароля сбрасывается, даже если не произошло удачной попытки подключения (аутентификации).

pwdGraceAuthNLimit

```
pwdGraceAuthNLimit number
```

```
# пример  
pwdGraceAuthNLimit 5
```

Данный атрибут управляет тем, будут ли разрешены дополнительные операции подсоединения после того, как пароль устарел — часто это называется периодом отложенной блокировки (дословный перевод — период милосердных подключений). Если его значение — 0 (по умолчанию), то любые попытки подсоединения с использованием устаревшего пароля будут отклоняться. Если значение больше 0, то оно определяет количество успешных подсоединений, которое пользователю разрешено произвести с использованием устаревшего пароля. Если этот лимит достигнут и пароль не был изменён, все последующие операции подсоединения будут отклоняться.

pwdInHistory

```
pwdInHistory no-of-passwords
```

```
# пример  
pwdInHistory 5
```

Данный атрибут управляет количеством паролей, которые будут содержаться в списке прежде использовавшихся паролей, — истории паролей, — для учётной записи. Когда пользователь меняет свой пароль, этот пароль сверяется с данным списком истории паролей, и, если будет найдено совпадение, такой пароль будет отклонён. Если пароль не совпал ни с одним из списка истории паролей, то он будет применён в качестве нового пароля и добавлен в список истории, а самое старое значение из этого списка будет удалено. Суть в том, чтобы заставить пользователя генерировать такие пароли, которые не использовались прежде, или, хотя бы за ограниченный историей паролей промежуток времени. Все пароли в списке истории паролей хранятся с применением установленного по умолчанию для сервера метода хэширования. Если значение данного атрибута — 0 (по умолчанию), значит история паролей вести не будет. В этом случае будет лишь проверяться, чтобы новый пароль не совпадал с текущим.

pwdLockout

```
pwdLockout TRUE | FALSE
```

```
# пример  
pwdLockout TRUE
```

Данный атрибут управляет тем, какое действие будет предпринято при превышении для учётной записи количества последовательных неудачных попыток подключения с неверным паролем значения, указанного в атрибуте `pwdMaxFailure`. Если его значение — TRUE (по умолчанию), то учётная запись будет заблокирована — дальнейшие попытки подключения будут отклоняться, пока учётная запись не будет разблокирована администратором (подробности о том, как это делается, смотрите в [подразделе о разблокировке учётной записи](#)). Если значение данного атрибута — FALSE, учётная запись не блокируется и разрешается любое количество последовательных попыток подключения с неверным паролем.

pwdLockoutDuration

```
pwdLockoutDuration number-of-seconds  
  
# пример  
pwdLockoutDuration 0
```

Данный атрибут управляет тем, как долго учётная запись остаётся заблокированной; имеет смысл только если атрибут `pwdLockout` установлен в TRUE. Если его значение — 0 (по умолчанию), пользователь не сможет получить доступ к заблокированной учётной записи, пока эта учётная запись не будет разблокирована администратором (подробности о том, как это делается, смотрите в [подразделе о разблокировке учётной записи](#)). Если значение больше 0, то оно определяет время (в секундах), в течение которого учётная запись остаётся заблокированной. По окончании данного периода учётная запись автоматически разблокируется.

pwdMaxAge

```
pwdMaxAge time-in-seconds  
  
# пример  
pwdMaxAge 2592000
```

Это один из тех атрибутов, значение которого требует сложных расчётов. Он определяет максимальное время (в секундах), в течение которого пароль считается действительным. По истечении данного промежутка времени считается, что пароль нельзя больше применять, и потому любые операции подключения с данным устаревшим паролем будут интерпретироваться как неверные. То ужасное значение, которое мы привели в примере, составляет 30 дней, это значит, что пароли необходимо менять каждые 30 дней. Значение по умолчанию — 0 означает, что пароли не устаревают никогда.

pwdMaxFailure

```
pwdMaxFailure number-of-attempts  
  
# пример  
pwdMaxFailure 5
```

Данный атрибут управляет тем, какое количество последовательных попыток ввода неверного пароля будет разрешено сделать перед тем, как будет выполнено действие, определённое в атрибуте `pwdLockout`. Если его значение — 0 (по умолчанию), будет разрешено сделать неограниченное количество последовательных попыток ввода неверного пароля. Если значение больше 0, то оно определяет максимальное количество последовательных попыток ввода неверного пароля, которые разрешено сделать перед выполнением действия, определённого в атрибуте `pwdLockout`. Любая успешная операция подключения сбрасывает данный счётчик.

<MODIFIED>

pwdMaxTotalAttempts

```
pwdMaxTotalAttempts number-of-attempts  
  
# пример  
pwdMaxTotalAttempts 50
```

Данный атрибут управляет тем, какое количество последовательных попыток ввода неверного пароля, в том

числе повторного ввода одного и того же пароля, будет разрешено сделать перед выполнением действия, определённого в атрибуте [pwdLockout](#). Если его значение — 0 (по умолчанию), проверка повторного ввода пароля осуществляться не будет. Если значение больше 0 (положительное число), то оно определяет максимальное количество последовательных попыток ввода неверного пароля (суммарное значение повторяющихся и уникальных неверных паролей), которые разрешено сделать перед выполнением действия, определённого в атрибуте [pwdLockout](#). Если значение — -1, то разрешено делать неограниченное количество повторных попыток ввода одного и того же неверного пароля. Во всех случаях количество попыток ввода уникального неверного пароля контролируется атрибутом [pwdMaxFailure](#).

</MODIFIED>

pwdMinAge

```
pwdMinAge time-in-seconds
# пример
pwdMinAge 3600
```

Данный атрибут управляет минимальным временем (в секундах) между сменами пароля. В приведённом выше примере пароль может изменяться не чаще чем раз в час (3600 секунд). Попытки изменить пароль до истечения этого времени будут отклоняться. Значение по умолчанию — 0 позволяет производить смену пароля в любое время с момента последнего изменения.

pwdMinLength

```
pwdMinLength bytes
# пример
pwdMinLength 6
```

Данный атрибут управляет тем, будет ли сервер выполнять проверку минимальной длины пароля. Если его значение — 0 (по умолчанию), проверка длины пароля осуществляться не будет. Если значение больше 0, то оно определяет минимальную длину предоставляемого пользователем пароля. Пароли, длина которых меньше этого значения, будут отклоняться. Если предоставляется пароль в хэшированном виде, проверка длины не может быть осуществлена, и выполняется действие по умолчанию, определённое атрибутом [pwdCheckQuality](#): если его значение — 0 (по умолчанию) или 1, пароль принимается, если 2 -отклоняется.

pwdMustChange

```
pwdMustChange TRUE | FALSE
# пример
pwdMustChange TRUE
```

Данный атрибут управляет тем, должен ли пользователь сменить свой пароль после того, как его учётная запись была разблокирована администратором после её блокировки; имеет смысл только если атрибут [pwdLockout](#) установлен в TRUE. Если значение атрибута [pwdMustChange](#) — FALSE (по умолчанию), то пользователю не обязательно менять свой пароль после того, как его учётная запись разблокирована. Если его значение — TRUE, то пользователь ДОЛЖЕН поменять свой пароль после того, как его учётная запись разблокирована. Если для [разблокировки учётной записи](#) используется атрибут [pwdReset](#), то его значение переопределяет значение данного атрибута.

pwdSafeModify

```
pwdSafeModify TRUE | FALSE
# пример
pwdSafeModify TRUE
```

Данный атрибут управляет тем, должен ли пользователь отправлять свой текущий пароль во время операции смены пароля. Если его значение — FALSE (по умолчанию), то пользователю не требуется отправлять свой текущий пароль. Если значение — TRUE, то пользователь ДОЛЖЕН отправлять свой текущий пароль при

модификации значения пароля.

Операционные атрибуты

Модуль `ppolicy` использует несколько операционных атрибутов в записи пользователя для того, чтобы обозначить статус учётной записи и позволить администратору разблокировать учётную запись, перешедшую в состояние блокировки.

`pwdAccountLockedTime`

```
pwdAccountLockedTime account-locked-time
```

Данный атрибут показывает время с момента блокировки учётной записи и будет появляться лишь в том случае, если атрибут `pwdLockout` установлен в `TRUE`. Если данный атрибут удалён администратором, то учётная запись становится разблокированной и существующий пароль может быть использован (подразумевается, что он не устарел). Смотрите подраздел о [разблокировке учётной записи](#).

`pwdChangedTime`

```
pwdChangedTime last-password-change-time
```

Атрибут только для чтения. Данный атрибут показывает время последнего изменения пароля для данной записи.

`pwdFailureTime`

```
pwdFailureTime invalid-password-attempt-time
```

Атрибут только для чтения. Этот атрибут, у которого может быть несколько значений, показывает время каждой попытки неудачного ввода пароля.

`pwdGraceUseTime`

```
pwdGraceUseTime grace-auth-time
```

Атрибут только для чтения. Этот атрибут, у которого может быть несколько значений, содержит время каждого удачного подсоединения (аутентификации) после того, как пароль стал устаревшим. Он присутствует только в том случае, если есть атрибут `pwdGraceAuthNLimit`, значение которого больше 0.

`pwdHistory`

```
pwdHistory hashed-user-password
```

Атрибут только для чтения. Этот атрибут, у которого может быть несколько значений, содержит хэшированные значения использованных ранее паролей (в том числе текущий пароль), количество которых ограничено значением атрибута `pwdInHistory`. Он присутствует только в том случае, если есть атрибут `pwdInHistory`, значение которого больше 0.

`pwdPolicySubentry`

```
pwdPolicySubentry DN
# пример
pwdPolicySubentry "cn=adminusers,ou=ppolicies,dc=example,dc=com"
```

Данный атрибут управляет тем, будет ли к пользовательской записи применяться уникальная политика паролей или политика паролей по умолчанию (в обоих случаях это записи с объектным классом `pwdPolicy`). Если проверка соответствия политике паролей активирована для данного DIT (определена директива `overlay`

ppolicy) и в пользовательской записи нет атрибута `pwdPolicySubentry`, то для данной записи применяется политика паролей по умолчанию, определённая директивой `ppolicy_default`. Если директива **ppolicy_default** не была определена, то для данной записи не применяется никакой политики паролей. Если атрибут `pwdPolicySubentry` присутствует в пользовательской записи, то политика паролей (запись с объектным классом **pwdPolicy**), DN которой указан в качестве значения данного атрибута, будет использоваться для данной пользовательской записи. **Примечание:** Подразумевается, что для данного DIT была определена директива конфигурации **overlay ppolicy**. Если это не так, то проверок паролей в данном DIT производиться не будет, независимо от значений данного атрибута.

pwdReset

```
pwdReset TRUE | FALSE
```

```
# пример  
pwdReset TRUE
```

Данный атрибут может использоваться администратором для сброса блокировки учётной записи. Если его значение — TRUE, пользователь должен сменить свой пароль до следующей попытки подключения. Если его значение — FALSE, то учётная запись разблокирована, но при этом пользователю не обязательно менять свой пароль. Значение данного атрибута переопределяет установку атрибута `pwdMustChange`.

<MODIFIED>

pwdUniqueAttempts

```
pwdUniqueAttempts hashed-invalid-password
```

Атрибут только для чтения. Этот атрибут, у которого может быть несколько значений, содержит хэшированные значения и время каждой попытки ввода уникального неверного пароля. Он присутствует только в том случае, если есть атрибут `pwdMaxTotalAttempts` с ненулевым (не равным 0) значением. Максимальное количество значений атрибута `pwdUniqueAttempts` определяется значением атрибута `pwdMaxFailure`. Если значение атрибута `pwdMaxTotalAttempts` больше 0, в значении атрибута `pwdUniqueAttempts` также содержится количество использований каждого неверного пароля.

</MODIFIED>

Разблокировка учётной записи

Если учётная запись пользователя заблокирована (атрибут `pwdLockout` должен быть установлен в TRUE), то она может быть разблокирована администратором с помощью одной из следующих процедур:

1. Удаления операционного атрибута `pwdAccountLockedTime`. Данная процедура позволяет пользователю продолжить использование текущего пароля и эффективна лишь в том случае, если срок действия пароля не истёк.
2. Добавления операционного атрибута `pwdReset` со значением TRUE или FALSE. Добавление данного атрибута со значением FALSE эффективно лишь в том случае, если срок действия пароля не истёк, и имеет тот же эффект, что и удаление атрибута `pwdAccountLockedTime`.

Примеры конфигурации

Для иллюстрации использования `ppolicy` рассматриваются два сценария. В первом единственная политика по умолчанию используется для всех пользовательских записей. Во втором используется отдельная политика для учётной записи.

Только политика паролей по умолчанию

Политика по умолчанию в виде записи с объектным классом **pwdPolicy** создаётся и добавляется в DIT с помощью следующего фрагмента LDIF:

```
# Добавление политики по умолчанию для DIT.
```

```
# Атрибуты, начинающиеся с #, показывают
# значения по умолчанию и могут быть пропущены.
# Пароли должны меняться каждые 30 дней,
# иметь минимальную длину 6 символов. Пользователи
# будут получать предупреждения об устаревании пароля
# за 1 час до устаревания. После того, как было произведено
# 5 последовательных попыток ввода неверного пароля,
# учётная запись будет заблокирована и разблокировать её
# может только администратор. От пользователей не требуется
# предъявлять старый пароль при смене пароля.
dn: cn=default,ou=pwpolicies,dc=example,dc=com
objectClass: pwdPolicy
cn: default
pwdMaxAge: 2592000
pwdExpireWarning: 3600
#pwdInHistory: 0
#pwdCheckQuality: 0
pwdMaxFailure: 5
pwdLockout: TRUE
#pwdLockoutDuration: 0
#pwdGraceAuthNLimit: 0
#pwdFailureCountInterval: 0
pwdMustChange: TRUE
pwdMinLength: 6
#pwdAllowUserChange: TRUE
pwdSafeModify: FALSE
```

Наложение `ppolicy` подключается к DIT путём добавления следующих директив в конфигурационный файл (или с помощью соответствующих значений **olc** при использовании **cn=config**):

```
...
# подключение набора схемы ppolicy
include ppolicy.schema
...
# требуется, если наложение было собрано динамически
loadmodule ppolicy.la
# ЛИБО
loadmodule ppolicy.so
...
database bdb
suffix "dc=example,dc=com"
...
# включение политик паролей только для данного DIT
overlay ppolicy
# определение политики по умолчанию
ppolicy_default "cn=default,cn=pwpolicies,dc=example,dc=com"
# опционально - директивы ppolicy
...
# другие директивы overlay
# либо следующая директива database
```

Политика паролей распространяется на все учётные записи в данном DIT. Других действий не требуется.

Политика паролей по умолчанию и отдельная политика паролей для конкретного пользователя

Политика паролей по умолчанию и отдельная политика для конкретного пользователя в виде записей с объектным классом **pwdPolicy** создаются и добавляются в DIT с помощью следующего фрагмента LDIF:

```
# Добавление политики по умолчанию для DIT.
# Атрибуты, начинающиеся с #, показывают
# значения по умолчанию и могут быть пропущены.
# Пароли должны меняться каждые 30 дней,
# иметь минимальную длину 6 символов. Пользователи
# будут получать предупреждения об устаревании пароля
# за 1 час до устаревания. После того, как было произведено
# 5 последовательных попыток ввода неверного пароля,
# учётная запись будет заблокирована и разблокировать её
# может только администратор. От пользователей не требуется
# предъявлять старый пароль при смене пароля.
dn: cn=default,ou=pwpolicies,dc=example,dc=com
objectClass: pwdPolicy
cn: default
pwdMaxAge: 2592000
pwdExpireWarning: 3600
#pwdInHistory: 0
#pwdCheckQuality: 0
pwdMaxFailure: 5
pwdLockout: TRUE
#pwdLockoutDuration: 0
```



```
#pwdGraceAuthNLimit: 0
#pwdFailureCountInterval: 0
pwdMustChange: TRUE
pwdMinLength: 6
#pwdAllowUserChange: TRUE
pwdSafeModify: FALSE

# Добавление в DIT специфичной для пользователя политики.
# Атрибуты, начинающиеся с #, показывают
# значения по умолчанию и могут быть пропущены.
# Пароли должны меняться каждые 7 дней,
# иметь минимальную длину 10 символов. Пользователи
# не будут получать предупреждения об устаревании пароля.
# После того, как было произведено
# 3 последовательных попытки ввода неверного пароля,
# учётная запись будет заблокирована и разблокировать её
# может только администратор. От пользователей требуется
# предъявлять старый пароль при смене пароля.
dn: cn=user,ou=pwpolicies,dc=example,dc=com
objectClass: pwdPolicy
cn: user
pwdMaxAge: 604800
# pwdExpireWarning: 0
pwdInHistory: 3
#pwdCheckQuality: 0
pwdMaxFailure: 3
pwdLockout: TRUE
#pwdLockoutDuration: 0
#pwdGraceAuthNLimit: 0
#pwdFailureCountInterval: 0
pwdMustChange: TRUE
pwdMinLength: 10
#pwdAllowUserChange: TRUE
pwdSafeModify: TRUE
```

Наложение `ppolicy` подключается к DIT путём добавления следующих директив в конфигурационный файл (или с помощью соответствующих значений **olc** при использовании **cn=config**):

```
...
# подключение набора схемы ppolicy
include ppolicy.schema
...
# требуется, если наложение было собрано динамически
loadmodule ppolicy.la
# ЛИБО
loadmodule ppolicy.so
...
database bdb
suffix "dc=example,dc=com"
...
# включение политик паролей только для данного DIT
overlay ppolicy
# определение политики по умолчанию
ppolicy_default "cn=default,cn=pwpolicies,dc=example,dc=com"
# опционально - директивы ppolicy
...
# другие директивы overlay
# либо следующая директива database
```

Наконец, учётные записи пользователей, к которым будет применяться отдельная политика, модифицируются, — добавляется указатель на специфичную политику, — с помощью следующего фрагмента LDIF:

```
# указываем пользовательской записи применять специфичную политику
dn: cn=John Smith,ou=people,dc=example,dc=com
changetype: modify
add: pwdPolicySubentry
pwdPolicySubentry: "cn=user,ou=pwpolicies,dc=example,dc=com"
```

Политика паролей по умолчанию распространяется на все учётные записи в данном DIT, за исключением учётной записи John Smith, которая использует политику, определённую в "cn=user,ou=pwpolicy,dc=example,dc=com". Других действий не требуется.



Уже совсем скоро (One day real soon now [™]).



Глава 6. Наложение OpenLDAP syncprov

Наложение **syncprov** (Sync Provider) должно быть определено для каждого DIT, выполняющего роль поставщика (главного DIT) при использовании [репликации в стиле syncrepl](#), то есть с помощью LDAP Content Synchronization protocol ([RFC 4533 \(рус.\)](#), [RFC 4533 \(ориг.\)](#)). Это наложение может использоваться с любым механизмом манипуляции данными, поддерживающим в записях атрибуты [entryCSN](#) и [entryUUID](#) (в настоящий момент только механизмы bdb и hdb). Для управления синхронизацией наложение создаёт в корневой записи базы данных (DIT) атрибут [contextCSN](#).

Для повышения производительности синхронизации рекомендуется при использовании данного наложения назначать индекс eq на атрибут entryCSN в том DIT, где используется это наложение. Пример для файла slapd.conf:

```
database bdb
...
index contextCSN eq
...
```

Эквивалент для OLC (cn=config):

```
olcDbIndex: contextCSN eq
```

Наложение syncprov (и любые его директивы) должно быть определено для каждого DIT (раздела database), выступающего в роли поставщика для операций репликации. Наложение syncprov определяется в конце раздела, но перед указанием параметров, специфичных для базы данных. Примеры использования для файла slapd.conf:

```
# определение первого DIT
database bdb
...
# DIT будет выполнять роль поставщика
overlay syncprov
...
# дополнительные параметры базы данных
cachesize 10000
...

# определение второго DIT
database bdb
...
# DIT НЕ будет выполнять роль поставщика

# дополнительные параметры базы данных
cachesize 10000
...

# определение третьего DIT
database bdb
...
# DIT будет выполнять роль поставщика
overlay syncprov
...
# дополнительные параметры базы данных
cachesize 10000
...
```

В OLC (cn=config) каждое наложение определяется как дочерняя запись к соответствующей записи базы данных (смотрите информацию по использованию наложений в OLC (cn=config) [здесь](#)).

В типичной репликации типа Главный-Подчинённый сервер LDAP, содержащий главное DIT (поставщик), определяется наложением **syncprov**, а подчинённый сервер LDAP (потребитель) — директивой [syncrepl](#) (атрибутом **olcSyncRepl** в OLC), в которой описывается способ доступа к поставщику. Дополнительная информация и примеры конфигурации репликации [здесь](#). Единственное исключение — это конфигурация

с [несколькими главными серверами \(multi-master\)](#), при которой каждое DIT является одновременно и главным (поставщиком), и подчинённым (потребителем).

Параметры конфигурации

Следующие атрибуты syncprov могут присутствовать в записи наложения syncprov в OLC (cn=config) или быть определены после директивы overlay syncprov в файле slapd.conf.

olcSpCheckPoint (syncprov-checkpoint)

```
olcSpCheckpoint: ops minutes
syncprov-checkpoint ops minutes
```

Данный атрибут (директива) управляет обслуживанием значения [contextCSN](#), которое обычно содержится и изменяется только в оперативной памяти, а запись в базу данных осуществляется лишь при нормальном завершении работы сервера; соответственно, загрузка значения из базы данных происходит во время операции старта сервера. Данный атрибут (директива) используется, чтобы обязать поставщика принудительно записывать значение contextCSN в соответствующую базу данных при условии выполнения успешных операций записи в эту базу данных, либо по прошествии определённого числа (**ops**) операций записи, либо по истечении определённого количества времени в минутах (**minutes**) после последнего обновления значения contextCSN в базе данных (или сброса в контрольной точке). По умолчанию **syncprov-checkpoint** отключена. Цель данной директивы — минимизировать количество предпринимаемых потребителем действий по синхронизации, которые потребуются в случае аварийного завершения работы сервера, на котором располагается главное DIT (поставщик). Пример:

```
overlay syncprov
# обновлять значение contextCSN в базе данных после ЛИБО
# 100 успешных операций записи, ЛИБО
# по прошествии более 5 минут
# с момента последней записи contextCSN в базу данных
olcSpCheckpoint: 100 5
syncprov-checkpoint 100 5
```

olcSpNoPresent (syncprov-nopresent)

```
olcSpNoPresent: TRUE | FALSE
syncprov-nopresent TRUE | FALSE
```

Если эта директива установлена в TRUE, при обновлении будет пропущена [фаза наличия \(Present phase\)](#). Для экземпляров syncprov, используемых совместно с базами данных журналов (например, управляемых с помощью наложения [accesslog](#)), данное значение должно быть установлено в TRUE. Значение по умолчанию — FALSE.

olcSpReloadHint (syncprov-reloadhint)

```
olcSpReloadHint: TRUE | FALSE
syncprov-reloadhint TRUE | FALSE
```

Указывает, что наложение должно соблюдать флаг reloadHint в Sync Control (примечание: некоторые клиенты версии 2.3 не устанавливают корректно этот флаг). При использовании наложения accesslog для [delta-синхронизации](#) данная директива должна быть установлена в TRUE. Значение по умолчанию — FALSE. Флаг **reloadhint** может использоваться потребителем, запрашивающим операцию репликации, для указания того, что он хочет принудительно выполнить загрузку полного DIT независимо от всех остальных параметров и значений, таких как Sync Cookie.

olcSpSessionlog (syncprov-sessionlog)

```
olcSpSessionlog: ops
syncprov-sessionlog ops
```

Указывает, что на сервере-поставщике должен вестись журнал, в который записывается информация об

операциях записи, производимых в базе данных. Аргумент **ops** определяет количество операций, которое можно поместить в данный журнал. Туда помещается информация о всех операциях записи (за исключением операций добавления add). Если такой журнал сессии используется, целесообразно назначить индекс eq на атрибут [entryUUID](#) в соответствующей базе данных поставщика.

Данный журнал сессии может использоваться при репликации во время операций синхронизации, чтобы минимизировать обновления потребителя, в первую очередь в режиме **refreshOnly**. Поскольку на поставщике не настраивается количество потребителей репликации, которое будет запрашивать синхронизацию, для наибольшей эффективности значение аргумента, задаваемого в данной директиве, должно позволять сохранять в журнале предполагаемый максимум числа изменений, которые могут произойти в промежутке между синхронизациями потребителя с самым длинным интервалом пересинхронизации (устанавливается параметром [interval](#) директивы syncrepl). Если в журнале сессии недостаточно информации, поставщик вынужден будет выполнять полную последовательность ресинхронизации, начиная с последней известной точки.



Глава 7. Репликация и отсылки

В данной главе рассматриваются вопросы настройки систем LDAP для выполнения [репликации](#), реализации [отсылок](#) и [псевдонимов](#). Репликация — это операционная характеристика, реализуемая с помощью настроек конфигурации, а отсылки могут быть как общими (операционная характеристика), так и явными, указываемыми в DIT (с помощью объектного класса [referral](#)). Будет ли LDAP-сервер разрешать отсылки (процесс, называемый [сцеплением](#)) или возвращать их клиенту, зависит от настроек сервера LDAP. Кроме того, у LDAP-браузеров обычно есть возможность настройки на автоматическое следование по отсылкам (разрешение отсылок), а также на отображение записей, содержащих объекты-отсылки, чтобы иметь возможность редактировать эти объекты. Псевдонимы включены в этот раздел по тем сомнительным соображениям, что они демонстрируют "подобное отсылкам" поведение в том смысле, что они могут использоваться для изменения строгого иерархического потока DIT, вызывая переход к определённой пользователем альтернативной (или псевдоименованной) записи, которая может существовать в совершенно несвязанной ветке данного DIT. Упрощённо **отсылка** может рассматриваться как **переход к внешнему серверу LDAP**, так как при её разыменовании используется LDAP URI, а **псевдоним** может рассматриваться как **переход внутри сервера LDAP**, поскольку при его разыменовании используется только DN.

Содержание

- 7.1 Обзор репликации и отсылок
- 7.2 Репликация
 - 7.2.1 Репликация OpenLDAP
 - 7.2.1.1 Репликация OpenLDAP в стиле slurpd (устарела, начиная с версии 2.4)
 - 7.2.1.1.1 Ошибки при репликации OpenLDAP в стиле slurpd (устарела, начиная с версии 2.4)
 - 7.2.1.2 Репликация OpenLDAP в стиле syncrepl
 - 7.2.1.2.1 Тип RefreshOnly репликации syncrepl в OpenLDAP
 - 7.2.1.2.2 Тип RefreshAndPersist репликации syncrepl в OpenLDAP
 - 7.2.1.2.3 Репликация syncrepl с несколькими главными серверами (Multi-Master) в OpenLDAP
 - 7.2.1.2.4 Журналы сессии, журналы доступа и delta-синхронизация при репликации syncrepl в OpenLDAP
 - 7.2.2 Репликация ApacheDS
- 7.3 Синхронизация DIT перед запуском репликации
 - 7.3.1 Синхронизация DIT перед запуском репликации в стиле syncrepl
 - 7.3.2 Синхронизация DIT перед запуском репликации в стиле slurpd
- 7.4 Отсылки
 - 7.4.4 Удаление отсылок
 - 7.4.5 Сцепление при отсылках
- 7.5 Псевдонимы

7.1 Обзор репликации и отсылок

Одним из наиболее мощных аспектов LDAP (и X.500) является вложенная в них способность делегировать ответственность за поддержание части каталога другому серверу, сохраняя при этом общую картину каталога как единого целого. Таким образом, можно создать в каталоге компании делегирование

ответственности (**отсылку (referral)** в терминологии LDAP) той части всего каталога (DIT), в которой описан конкретный отдел, LDAP-серверу этого отдела. Делегированное DIT называется **нижестоящим (subordinate)** по отношению к тому DIT, откуда на него происходит отсылка. А то DIT, откуда происходит отсылка, называется **вышестоящим (superior)**. В этом аспекте делегирование LDAP почти полностью отражает делегирование DNS.

В отличие от системы DNS, в стандартах не предусмотрена возможность указать LDAP-серверу проследовать по отсылке (разрешить отсылку) (в различных документах есть только ссылки на черновые RFC), LDAP-клиенту оставлена возможность напрямую обращаться к новому серверу, используя возвращённую отсылку. Равным образом, поскольку в стандарте не определена организация данных LDAP, возможность перехода LDAP-сервера по ссылке (разрешение ссылки) не противоречит стандартам и некоторые LDAP-серверы выполняют эту функцию автоматически, используя процесс, обычно называемый **сцеплением (chaining)**.

OpenLDAP буквально следует стандартам и по умолчанию не выполняет **сцепление**, а всегда возвращает отсылку. Однако OpenLDAP может быть настроен на выполнение **сцепления** с помощью директивы **overlay chain**.

Встроенная функция репликации LDAP позволяет создать одну или несколько подчинённых (или реплицированных) копий каталога (DIT) от одной главной, таким образом, по сути, создаётся устойчивая структура. В OpenLDAP версии 2.4 представлена возможность реализовать полноценную конфигурацию разнонаправленной репликации с несколькими главными серверами (N-Way Multi-Master).

Важно, однако, подчеркнуть разницу между LDAP и транзакционными базами данных. При выполнении обновления на главном каталоге LDAP для обновления всех подчинённых каталогов может потребоваться некоторое время (в компьютерной терминологии) — главный и подчинённый каталоги могут быть рассинхронизированы в течение какого-то периода времени.

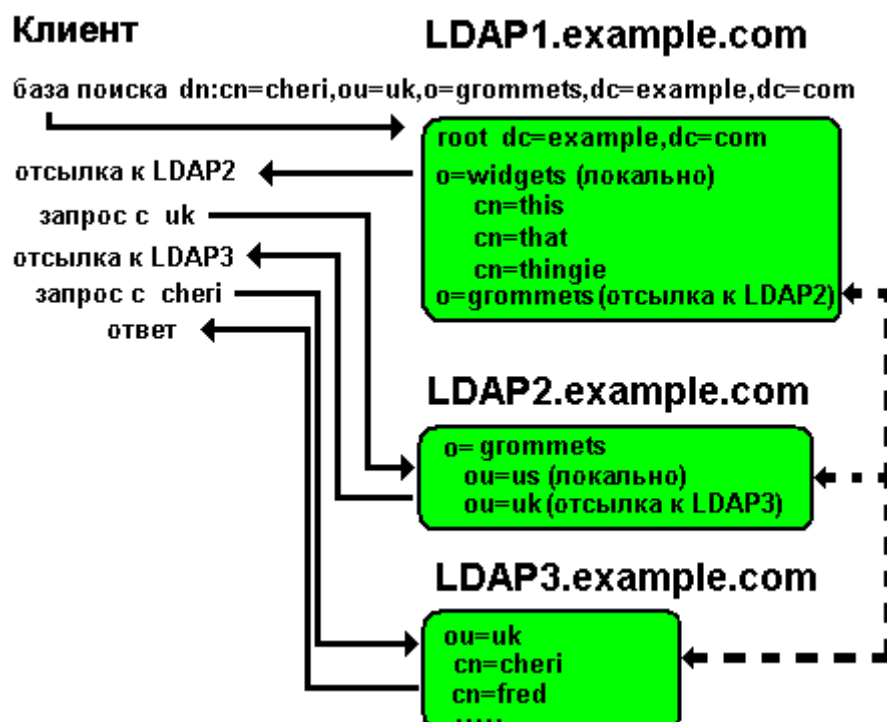
В контексте LDAP, временное отсутствие синхронизации DIT считается несущественным. В случае же транзакционной базы данных, даже временное отсутствие синхронизации считается катастрофическим. Это подчёркивает различия в характеристиках данных, которые должны храниться в каталоге LDAP и в транзакционной базе данных.

Конфигурация репликации (**OpenLDAP** и **ApacheDS**) и **отсылок** рассматривается далее в этой главе, а также в **приводимых примерах**.

7.1.1 Отсылки LDAP

Рисунок 7.1-1 демонстрирует поисковый запрос с базовым DN

dn:cn=cheri,ou=uk,o=grommets,dc=example,dc=com к LDAP-системе с отсылками, ответ на который возвращается в результате серии отсылок к серверам LDAP2 и LDAP3:



Примечания:

1. Все клиентские запросы начинаются с обращения к глобальному каталогу LDAP1.
2. На сервере LDAP1 запросы любых данных, содержащие **widgets** в качестве **RDN** в **DN**, удовлетворяются непосредственно из LDAP1, например:

```
dn: cn=thingie,o=widget,dc=example,dc=com
```

3. На сервере LDAP1 запросы любых данных, содержащие **grommets** в качестве **RDN** в **DN**, генерируют отсылку на сервер LDAP2, например:

```
dn: uid=cheri,ou=uk,o=grommets,dc=example,dc=com
```

4. На сервере LDAP2 запросы любых данных, содержащие **uk** в качестве **RDN** в **DN**, генерируют отсылку на сервер LDAP3, например:

```
dn: uid=cheri,ou=uk,o=grommets,dc=example,dc=com
```

5. Если LDAP-сервер сконфигурирован на выполнение сцепления (то есть на следование отсылкам, как показано альтернативными пунктирными линиями), то LDAP-клиенту будет отправлен один единственный ответ. Сцепление контролируется конфигурацией LDAP-сервера и значениями в поисковых запросах. Информация по сцеплению [здесь](#).
6. На рисунке показано явное сцепление с использованием объектного класса **referral**. Серверы OpenLDAP могут быть [настроены так](#), чтобы возвращать общую отсылку в случае, если запрашиваемый DN не был найден во время операции поиска.

7.1.2 Репликация LDAP

Функции репликации позволяют копировать обновления LDAP DIT на одну или несколько LDAP-систем в целях резервирования и/или повышения производительности. В этом контексте стоит подчеркнуть, что репликация работает на уровне DIT, а не на уровне LDAP-сервера. Так, если на одном сервере обслуживается несколько DIT, каждое из них может реплицироваться на разные серверы. Репликация происходит периодически, в течение промежутка времени, который мы называем **временем цикла репликации**. Исторически для выполнения репликации в OpenLDAP использовался отдельный демон (slurpd), но в версии 2.3 была представлена новая возможность (обычно называемая sync repl), и, начиная с версии 2.4, репликация в стиле slurpd стала устаревшей. Существует два возможных типа конфигураций репликации, у каждого из которых есть несколько вариантов.

1. **Главный-подчиненный (Master-Slave):** В конфигурации главный-подчиненный для выполнения обновлений доступно единственное (главное) DIT, и эти обновления реплицируются или копируются на один или несколько указанных серверов, на которых запущены подчиненные DIT. Подчиненные серверы оперируют с доступной только для чтения копией главного DIT. Пользователи, которые выполняют только чтение, будут обращаться к серверам, содержащим подчиненные DIT. Пользователи, которым нужно вносить изменения в каталог, смогут это сделать только обратившись к серверу, содержащему главное DIT. Чтобы ещё больше запутать своих несчастных пользователей, разработчики OpenLDAP ввели термины **поставщик (provider)** и **потребитель (consumer)** для репликации в стиле sync repl. Каталог может рассматриваться как источник (поставщик) сервисов репликации (то, что простые смертные называют главным (master) DIT), и как назначение (или потребитель) сервисов репликации (то, что простые смертные называют подчиненным (slave) DIT). У конфигурации главный-подчиненный (или поставщик-потребитель) есть два очевидных недостатка:
 - Несколько мест размещения. Если всем или большинству клиентов требуется вносить изменения в DIT, то им потребуется получать доступ к одному серверу (на котором запущено подчиненное DIT) для осуществления чтения и к другому серверу (на котором запущено главное DIT) для выполнения обновлений. Другой вариант — клиенты всегда будут обращаться к серверу, на котором запущено главное DIT. В последнем случае репликация выполняет только функцию резервного копирования.
 - Низкая отказоустойчивость. Поскольку существует только один сервер, содержащий главное DIT,

он представляет собой единую точку отказа, которая может повлиять на работоспособность всей системы.

2. **Несколько главных (Multi-Master):** В конфигурации с несколькими главными серверами могут обновляться несколько главных DIT, запущенных на одном или нескольких серверах, и результаты обновлений распространяются на остальные главные серверы.

Исторически OpenLDAP не поддерживал операции с несколькими главными серверами, но в версии 2.4 такие возможности введены. В этом контексте, наверное, стоит отметить две вариации общей проблемы **конкуренции обновлений**, специфичной для всех конфигураций с несколькими главными серверами, выявленные OpenLDAP и являющиеся верными для всех систем LDAP:

1. **Конкуренция значений.** Если выполняется два обновления одного и того же атрибута в одно и то же время (в пределах времени цикла репликации) и при этом атрибуту назначаются разные значения, то, в зависимости от типа атрибута (SINGLE или MULTI-VALUED), результирующая запись может быть в неправильном или непригодном для использования состоянии.
2. **Конкуренция удаления.** Если один пользователь добавляет дочернюю запись в то же самое время (в пределах времени цикла репликации), когда другой пользователь удаляет оригинальную запись, то удалённая запись вновь появится.

Рисунок 7.1-2 показывает несколько возможных конфигураций репликации.

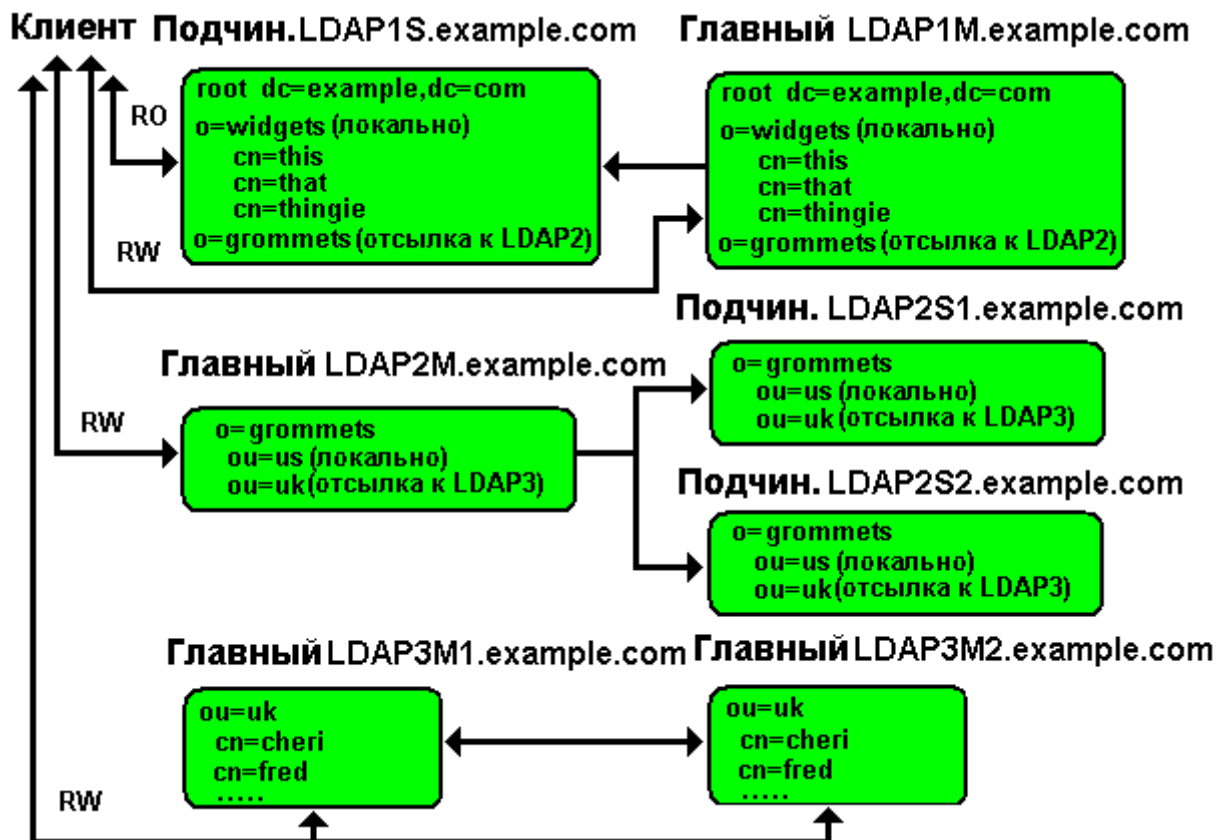


Рисунок 7.1-2 — Конфигурации репликации

Примечания:

1. **RO** = только чтение, **RW** = чтение-запись
2. В случае LDAP1 клиент взаимодействует с подчинённой системой, доступной только для чтения. Для выполнения операций записи клиенты должны обращаться к главной системе.
3. В случае LDAP2 клиент взаимодействует с главной системой, которая реплицируется на две подчинённые.
4. LDAP3 является системой с несколькими главными серверами и для выполнения операций чтения и записи клиенты могут обращаться к любому серверу. В данной конфигурации каждый главный сервер может, в свою очередь, иметь одно или несколько подчинённых DIT.

7.2 Репликация

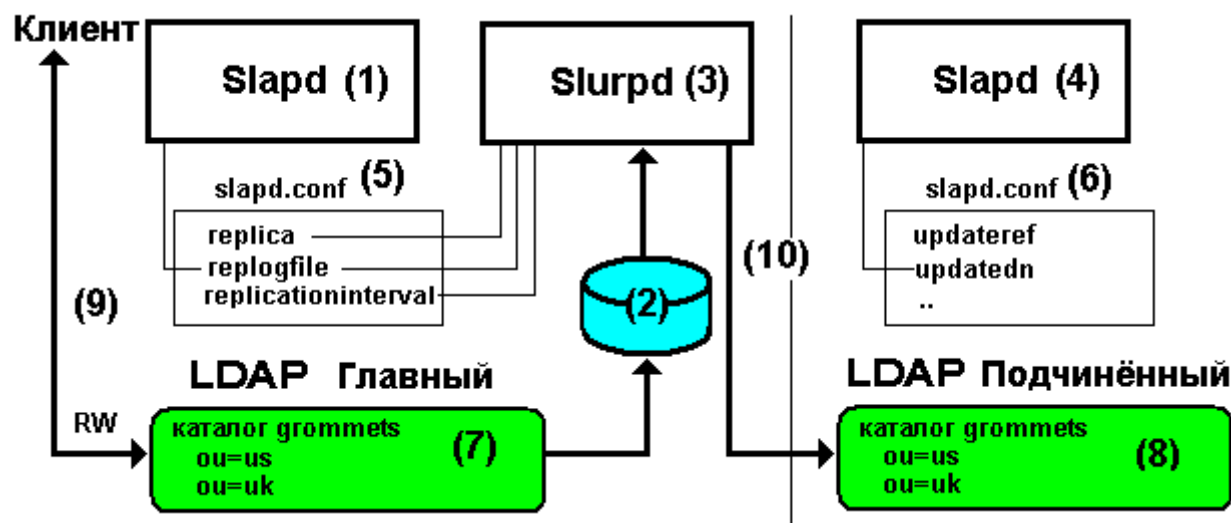
Репликация происходит на уровне DIT и описывает процесс копирования обновлений из DIT на одном сервере LDAP в такое же DIT на одном или нескольких других серверах. Конфигурации репликации могут быть типа "главный-подчинённый" (MASTER-SLAVE) (подчинённая копия всегда доступна только для чтения) и типа "несколько главных" (MULTI-MASTER). Репликация настраивается на уровне конфигурации сервера (то есть является операционной сущностью). Далее рассматриваются конфигурации репликации для [OpenLDAP](#) и [ApacheDS](#).

7.2.1 Репликация OpenLDAP

Когда-то жизнь была очень проста. В репликации OpenLDAP использовались slurpd и временный файл. В OpenLDAP версии 2.3 был представлен новый метод, известный как sync REPL ([RFC 4533 \(рус.\)](#), [RFC 4533 \(ориг.\)](#)), но поддержка репликации в стиле slurpd была продолжена. OpenLDAP версии 2.4 больше не поддерживает репликации в стиле slurpd. Поэтому в следующих разделах даётся описание как репликации в стиле [slurpd](#) для версий до 2.3 включительно, так и репликации в новом стиле [sync REPL](#) для версий 2.4+ (или 2.3+ для смельчаков).

Репликация OpenLDAP в стиле slurpd (устарела, начиная с версии 2.4)

Репликация в стиле slurpd использовала репликационную стратегию "посылок" ("push"). Она **устарела**, начиная с версии 2.4. Этот раздел документации размещается здесь по историческим причинам, а также для всех тех, кто застрял на старых версиях OpenLDAP. Настройка и управление такой репликацией показана на рисунке 7.2-1:



7.2-1 Репликация в стиле slurpd

Когда slapd (1), обслуживающий главное DIT (7), принимает запрос на операцию модификации (9), он обновляет своё DIT и помещает копию произошедшей транзакции в файл журнала (2), указанный в директиве [relogfile](#) файла slapd.conf (5) главного сервера.

При первоначальной загрузке slurpd (3) получает свои операционные параметры из файла slapd.conf (5). Периодически, через промежуток времени, указанный в директиве [replicationinterval](#), slurpd будет считывать содержимое файла журнала (2), указанного в директиве [relogfile](#), и записывает обновления (10) на один (или несколько) подчинённых DIT (8), указанных в директивах [replica](#) в файле slapd.conf (5).

Подчинённые DIT (8) являются копиями "только для чтения" для всех клиентов, за исключением клиента, подключающегося от имени DN, указанного в директиве [updatedn](#). В ответ на все поступающие от клиентов операции модификации, за исключением тех, которые поступают от клиента, подключающегося от имени DN, указанного в директиве [updatedn](#), подчинённый сервер (4) возвращает LDAP URI, указанное в директиве [updateref](#). Обе директивы updatedn и updateref определяются в файле slapd.conf (6). DN, указываемый в директиве [updatedn](#) в slapd.conf (6) ДОЛЖЕН совпадать с тем, который указан в директиве

`replica` (параметр `binddn`) в `slapd.conf` (5) для данного подчинённого сервера.

Ошибки при репликации OpenLDAP в стиле `slurpd` (устарела, начиная с версии 2.4)

Если `slurpd` (3) не удаётся обновить экземпляр подчинённого сервера, он создаёт файл отказов (REJECTION), имя которого совпадает с именем файла, указанного в директиве `repllogfile`, с добавлением суффикса `.rej`, как показано в примере:

```
# директива repllogfile slapd.conf
repllogfile /var/log/ldap/slave1.log

# файл REJECTION будет называться
# /var/log/ldap/slave1.log.rej
```

Каждое сообщение об ошибке в файле журнала REJECTION аналогично тому, которое используется в нормальном журнале транзакций, но оно предваряется строкой, начинающейся с ключевого слова `ERROR` и содержащей сообщение об ошибке. Вот пример:

```
ERROR: No such attribute
replica: slave1.example.com:389
time: 809618633
dn: uid=rsmith,dc=example,dc=com
changetype: modify
replace: description
description: clown
-
replace: modifiersName
modifiersName: uid=rsmith,dc=example,dc=com
-
replace: modifyTimestamp
modifyTimestamp: 20000805073308Z
```

Чтобы исправить ошибки, нужно либо внести изменения в содержимое каталога на подчинённом сервере (в случае с приведённым выше примером добавить в запись атрибут `description`), либо отредактировать журнал REJECTION, исправляя ошибки (в приведённом выше примере заменить строку `replace: description` на `add: description`). Нет необходимости удалять строки, начинающиеся с `ERROR`, поскольку они игнорируются. После внесения исправлений можно попытаться заново применить файл REJECTION путём запуска `slurpd` в режиме однократного прогона (после остановки работающего в данный момент `slurpd`) следующей командой:

```
slurpd -o -r /var/log/ldap/slave1.log.rej

# здесь -r указывает путь к файлу REJECTION
# а -o говорит о том, что запуск будет
# произведён в режиме однократного прогона
```

`slurpd` применит транзакции из указанного файла (`-r`) и завершит работу. После этого следует снова запустить `slurpd` в нормальном режиме работы.

Примеры конфигурации `slurpd`:

Конфигурация `slapd.conf` на главном сервере:

```
# главный сервер slapd
# раздел глобальных настроек - проверка файла каждые 5 минут
replicationinterval 300

# раздел database
database bdb
...
# подключение к подчинённому серверу ldap-rep1.example.com
# с простым уровнем безопасности и паролем в открытом виде
# директива используется только демоном slurpd
replica uri=ldap://ldap-rep1.example.com bindmethod=simple
binddn="uid=admin,ou=admin,dc=example,dc=com" credentials=guess

# записывать изменения в указанный файл
# директива используется обоими демонами slapd и slurpd
repllogfile /var/log/ldap/slavedit.log
```

Конфигурация `slapd.conf` на подчинённом сервере (хост `ldap-rep1.example.com`):

```
# подчинённый сервер slapd
# раздел глобальных настроек

# раздел database
database bdb
...

# указывается dn, который используется в
# директиве replica на главном сервере
# директива используется только демоном slurpd
updatedn "uid=admin,ou=admin,dc=example,dc=com"

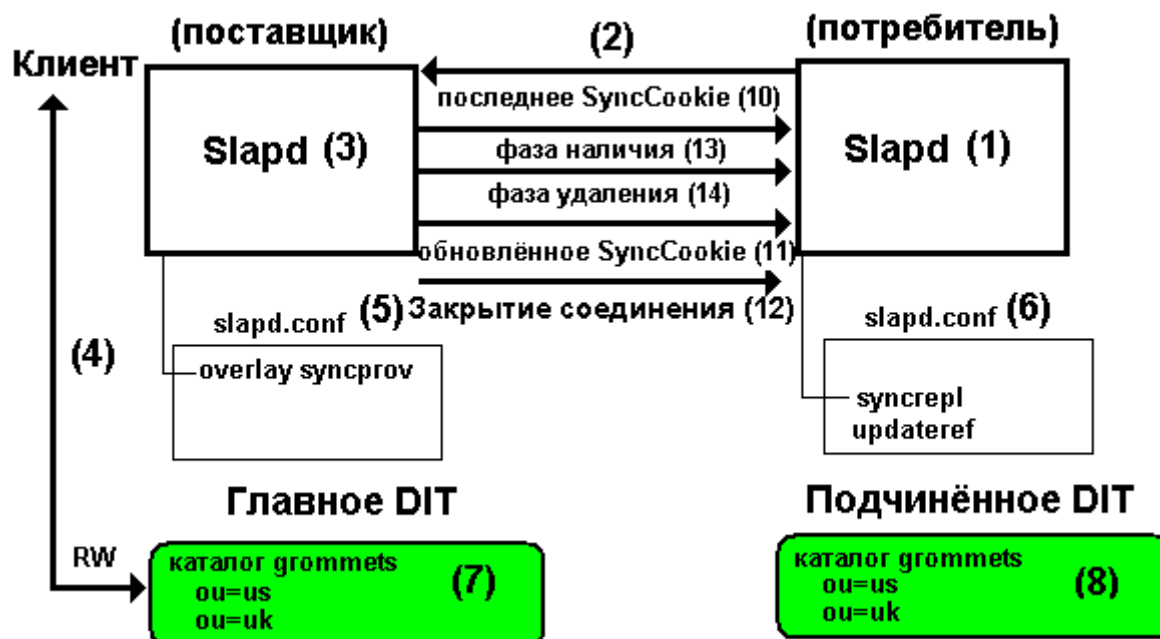
# отсылка, возвращаемая клиенту, пытающемуся выполнить обновление подчинённого каталога
updateref ldap://master-ldap.example.com
```

Репликация OpenLDAP в стиле syncrepl (начиная с версии 2.3)

В OpenLDAP версии 2.3 была представлена поддержка нового протокола синхронизации содержимого LDAP (LDAP Content Synchronization protocol), а начиная с версии 2.4 он стал единственной поддерживаемой возможностью репликации (теперь репликация в стиле slurpd является устаревшей). LDAP Content Synchronization protocol определён в [RFC 4533 \(рус.\)](#) ([RFC 4533 \(ориг.\)](#)) и широко известен под именем директивы файла slapd.conf, с помощью которой осуществляется управление им — [syncrepl](#). syncrepl предоставляет как классическую репликацию главный-подчинённый (master-slave), так и, начиная с версии 2.4, репликацию с несколькими главными серверами (multi-master). В протоколе используются термины [поставщик \(provider\)](#) (вместо главного (master)) для указания на источник реплицируемых обновлений, и [потребитель \(consumer\)](#) (вместо подчинённого (slave)) для указания назначения реплицируемых обновлений.

При репликации в стиле syncrepl процесс обновления **всегда** инициируется потребителем, тогда как при репликации в стиле slurpd обновления всегда инициируются главным сервером (поставщиком). Потребитель может быть настроен на периодические **запросы (pull)** обновлений у поставщика (**refreshOnly**), либо может запросить у провайдера выполнять **посылки (push)** обновлений (**refreshAndPersist**). В любом случае, для однозначного указания на какую-либо запись сервер должен поддерживать универсальный уникальный номер ([entryUUID](#)) для каждой записи в DIT. Процесс синхронизации показан на рисунках 7.2-2 (refreshOnly) и 7.2-3 (refreshAndPersist):

7.2.1.2.1 Репликация refreshOnly (по запросам потребителя)



7.2-2 Репликация syncrepl поставщик-потребитель — refreshOnly

Сервер slapd (1), желающий выполнить репликацию DIT (8) ([потребитель](#)), сконфигурирован с использованием директивы [syncrepl](#) в своём файле slapd.conf (6). В директиве syncrepl определяется

расположение (или имя) сервера slapd (3) ([поставщика](#)), содержащего главную копию DIT. Поставщик (3) сконфигурирован с использованием директивы [overlay syncprov](#) в своём файле slapd.conf (5).

При типе репликации **refreshOnly** потребитель (1) инициирует соединение (2) с поставщиком (3) — происходит синхронизация DIT и соединение разрывается. Периодически потребитель (1) устанавливает повторные соединения (2) с поставщиком (3) и пересинхронизируется. Синхронизацию **refreshOnly** можно рассматривать как операцию в повторяющемся режиме, а **время цикла репликации** — это время между повторными соединениями.

Подробности: Потребитель (1) устанавливает сессию (2) с поставщиком (3) и запрашивает синхронизацию **refreshOnly**. К поставщику может обращаться и запрашивать выполнение синхронизации любое количество потребителей — на поставщике нет никаких настроек относительно его потребителей (потребителей): если потребитель удовлетворяет требованиям безопасности поставщика, запросы синхронизации от этого потребителя будут удовлетворены. По существу, запрос на синхронизацию — это расширенная операция поиска LDAP, определяющая область репликации с помощью обычных поисковых параметров LDAP (базовый DN, диапазон, поисковый фильтр и атрибуты) — таким образом, во время сессии синхронизации реплики, в зависимости от критериев поиска, потребитель может быть обновлён до текущего состояния как всего содержимого DIT поставщика, так и его части.

Поставщику не нужно хранить у себя сведения о текущем состоянии каждого потребителя. Вместо этого в конце сессии синхронизации он отправляет синхронизационное куки (**SyncCookie** 11), в этом куки содержится порядковый номер изменения (Change Sequence Number [contextCSN](#)) — по сути, метка времени, указывающая на последнее изменение каталога, отправленное этому потребителю, и которую можно рассматривать как контрольную точку изменений или синхронизации. При установке сессии потребителем, он передает поставщику последнее полученное от него куки (10), чтобы указать поставщику пределы данной сессии синхронизации. В зависимости от того, каким образом был [инициализирован каталог потребителя](#), во время первого обращения к поставщику у потребителя может не быть SyncCookie, тогда первоначальная синхронизация охватит все записи в DIT поставщика (в пределах диапазона синхронизации). Побочный эффект этого процесса — то, что потребитель может быть первоначально запущен с пустым DIT, и, при первой репликации, полностью синхронизироваться с DIT поставщика.

В ответ на запрос синхронизации DIT поставщик (3) будет отвечать, используя одну из двух фаз или сразу обе фазы. Первая из них — **фаза наличия (present phase)** (13), указывающая на те записи, которые ещё остаются в DIT (или фрагменте DIT). В ней выполняются следующие действия:

1. Для каждой изменившейся записи (с момента последней синхронизации) посылается запись целиком, включая её DN и UUID ([entryUUID](#)). По этим данным потребитель может обновить свои записи без всяких разночтений.
2. Для каждой неизменившейся записи (с момента последней синхронизации) посылается пустая запись со своим UUID ([entryUUID](#)).
3. Для записей, которые были удалены, никаких сообщений не посылается. Теоретически, по окончании двух предыдущих процессов потребитель может удалить записи, не попавшие ни в один из них.

В фазе удаления (delete phase) (14):

1. Поставщик возвращает DN и UUID ([entryUUID](#)) для каждой записи, удалённой с момента последней синхронизации. Потребитель может удалить эти записи без всяких разночтений.

Необходимость использования при синхронизации обеих фаз определяется [рядом дополнительных методов](#).

По окончании фазы (фаз) синхронизации поставщик посылает SyncCookie (текущий contextCSN) и закрывает сессию. Потребитель сохраняет SyncCookie и в дальнейшем будет инициировать другую сессию синхронизации (через промежуток времени, указанный в параметре **interval** директивы [syncrepl](#)) путём отправки последнего полученного SyncCookie для ограничения диапазона следующей сессии синхронизации.

Примеры конфигурации syncrepl refreshOnly:

Конфигурация slapd.conf главного сервера (подразумевается, что имя хоста master-ldap.example.com):

```
# slapd поставщика
# раздел глобальных настроек
...
```

```
# раздел database
database bdb
...
# разрешаем потребителю доступ на чтение
# может понадобиться слияние с другими ACL
access to *
    by dn.base="cn=admin,ou=people,dc=example,dc=com" read
    by * break

# Примечание:
# конфигурация поставщика не содержит данных о потребителях

# указываем поставщику использовать наложение syncprov
# (заключительные директивы в разделе database)
overlay syncprov
# позволяем записывать contextCSN в базу данных
# через каждые 100 обновлений или 10 минут
syncprov-checkpoint 100 10
```

Конфигурация slapd.conf потребителя:

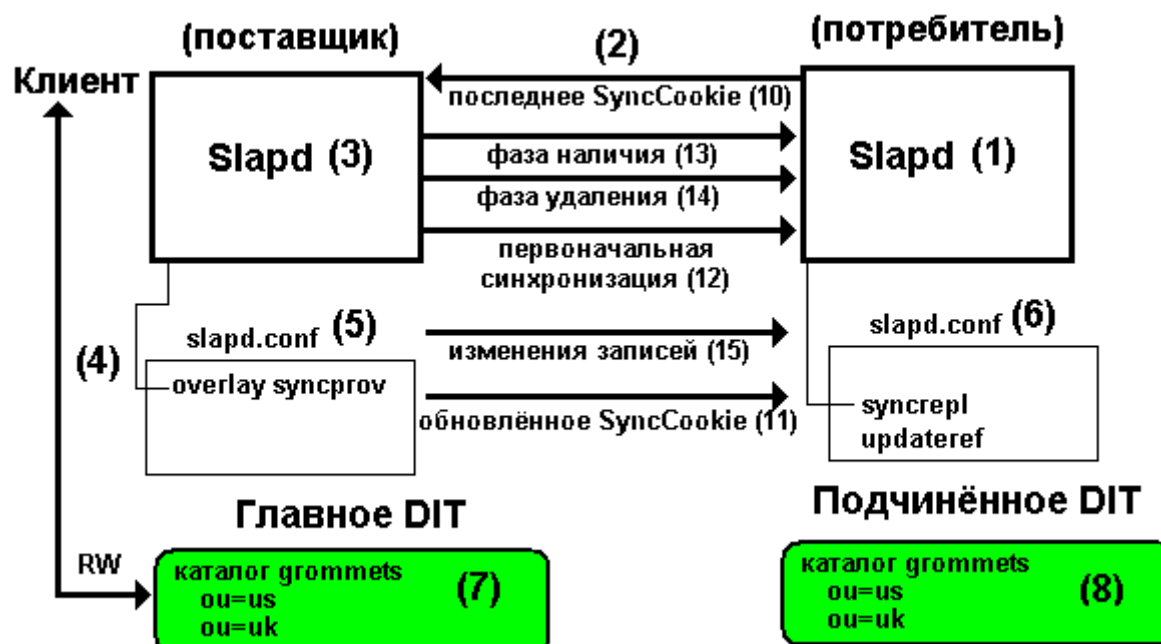
```
# slapd потребителя
# раздел глобальных настроек

# раздел database
database bdb
...

# поставщик - ldap://master-ldap.example.com:389, интервал синхронизации -
# раз в час, полное DIT (searchbase), синхронизируются все пользовательские атрибуты
# простой уровень безопасности с паролем в открытом виде
# Примечание: комментарии внутри директивы syncprov приводят к ошибкам OpenLDAP
# и приведены здесь лишь с целью дополнительных пояснений.
# Их НЕ ДОЛЖНО быть в рабочем файле
syncprov rid=000
provider=ldap://master-ldap.example.com
type=refreshOnly
# очередное соединение/пересинхронизация каждый час
interval=00:1:00:00
retry="5 5 300 +"
searchbase="dc=example,dc=com"
# синхронизируются и пользовательские (*), и операционные (+) атрибуты
attrs="*,+"
bindmethod=simple
binddn="cn=admin,ou=people,dc=example,dc=com"
# внимание: пароль посылается в открытом виде - небезопасно
credentials=dirtysecret
```



7.2.1.2.2 Репликация refreshAndPersist (посылки поставщика)



7.2-3 Репликация syncrepl поставщик-потребитель — refreshAndPersist

Сервер slapd (1), желающий выполнить репликацию DIT (7) с сервера (3) (**поставщика**), сконфигурирован с использованием директивы **syncrepl** в своём файле slapd.conf (6). В директиве syncrepl определяется расположение (или имя) сервера slapd (3) (**поставщика**), содержащего главную копию DIT. Поставщик (3) сконфигурирован с использованием директивы **overlay syncprov** в своём файле slapd.conf (5).

При типе репликации **refreshAndPersist** потребитель (1) инициирует соединение (2) с поставщиком (3) — сразу после этого происходит синхронизация (12) DIT и по окончании этого процесса соединение продолжает поддерживаться (становится **непрерывным (persist)**). Последующие изменения (4) на поставщике (3) немедленно передаются потребителю (1).

Подробности: Потребитель (1) устанавливает сессию (2) с поставщиком (3) и запрашивает синхронизацию **refreshAndPersist**. К поставщику может обращаться и запрашивать выполнение синхронизации любое количество потребителей — на поставщике нет никаких настроек относительно его потребителей (потребителей): если потребитель удовлетворяет требованиям безопасности поставщика, запросы синхронизации от этого потребителя будут удовлетворены. По существу, запрос на синхронизацию — это расширенная операция поиска LDAP, определяющая область репликации с помощью обычных поисковых параметров LDAP (базовый DN, диапазон, поисковый фильтр и атрибуты) — таким образом, во время сессии синхронизации реплики, в зависимости от критериев поиска, потребитель может быть обновлён до текущего состояния как всего содержимого DIT поставщика, так и его части.

Поставщику (3) не нужно хранить у себя сведения о текущем состоянии каждого потребителя. Вместо этого он периодически посылает синхронизационные куки (**SyncCookie** 11), в этом куки содержится порядковый номер изменения (Change Sequence Number **contextCSN**) — по сути, метка времени, указывающая на последнее изменение каталога, отправленное этому потребителю, и которую можно рассматривать как контрольную точку изменений или синхронизации. Когда потребитель репликации типа **refreshAndPersist** (1) устанавливает сессию (2) с поставщиком (3), они сначала должны синхронизировать состояния своих DIT (или фрагментов DIT). В зависимости от того, каким образом был **инициализирован каталог потребителя**, во время первоначального открытия сессии (2) с поставщиком (3) у потребителя может не быть **SyncCookie**, тогда областью изменений будет DIT целиком (или фрагмент DIT целиком). Побочный эффект этого процесса — то, что потребитель может быть первоначально запущен с пустым DIT, и, при первой репликации, полностью синхронизироваться с DIT поставщика. При последующих подключениях (2) потребителя (1) к поставщику (3), у потребителя уже будет **SyncCookie**. В случае репликации типа **refreshAndPersist** переподключения будут происходить только после сбоя на поставщике, потребителе или в сети, в результате которых соединение прерывается, в противном случае соединение будет поддерживаться постоянно.

Во время процесса синхронизации в ответ на запрос синхронизации DIT поставщик (3) будет отвечать, используя одну из двух фаз или сразу обе фазы. Первая из них — **фаза наличия (present phase)** (13), указывающая на те записи, которые ещё остаются в DIT (или фрагменте DIT). В ней выполняются следующие действия:

1. Для каждой изменившейся записи (с момента последней синхронизации) посылается запись целиком, включая её DN и UUID (**entryUUID**). По этим данным потребитель может обновить свои записи без всяких разночтений.
2. Для каждой неизменившейся записи (с момента последней синхронизации) посылается пустая запись со своим UUID (**entryUUID**).
3. Для записей, которые были удалены, никаких сообщений не посылается. Теоретически, по окончании двух предыдущих процессов потребитель может удалить записи, не попавшие ни в один из них.

В фазе удаления (delete phase) (14):

1. Поставщик возвращает DN и UUID (**entryUUID**) для каждой записи, удалённой с момента последней синхронизации. Потребитель может удалить эти записи без всяких разночтений.

Необходимость использования при синхронизации обеих фаз определяется **рядом дополнительных методов**.

По окончании фазы (фаз) синхронизации (12) поставщик обычно посылает SyncCookie (текущий contextCSN) и ПРОДОЛЖАЕТ ПОДДЕРЖИВАТЬ сессию. Последующие обновления (4) DIT поставщика (7) (изменения, добавления или удаления) будут немедленно отправляться (15) поставщиком (3) потребителю (1) для обновления DIT реплики (8). Изменения или дополнения посылаются в виде полной записи (со всеми атрибутами) и, как правило, также обновляется SyncCookie (11). DIT поставщика (7) и потребителя (8)

поддерживаются в состоянии синхронизации, а **время цикла репликации** в данном случае соответствует времени передачи данных между поставщиком и потребителем.

Примеры конфигурации `syncrepl refreshAndPersist`:

Конфигурация `slapd.conf` главного сервера (подразумевается, что имя хоста `master-ldap.example.com`):

```
# slapd поставщика (главного сервера)
# раздел глобальных настроек
...

# раздел database
database bdb
...
# разрешаем потребителю доступ на чтение
# может понадобиться слияние с другими ACL
# указанный dn.base должен совпадать с параметром binddn= поставщика
#
access to *
    by dn.base="cn=admin,ou=people,dc=example,dc=com" read
    by * break

# Примечание:
# конфигурация поставщика не содержит данных о потребителях

# указываем поставщику использовать наложение syncprov
# (заключительные директивы в разделе database)
overlay syncprov
# contextCSN сохраняется в базу данных
# через каждые 100 обновлений или 10 минут
syncprov-checkpoint 100 10
```

Конфигурация `slapd.conf` потребителя:

```
# slapd потребителя
# раздел глобальных настроек

# раздел database
database bdb
...

# поставщик - ldap://master-ldap.example.com:389,
# полное DIT (searchbase), синхронизируются все пользовательские атрибуты
# простой уровень безопасности с паролем в открытом виде
# Примечание: комментарии внутри директивы syncrepl приводят к ошибкам OpenLDAP
#             и приведены здесь лишь с целью дополнительных пояснений.
#             Их НЕ ДОЛЖНО быть в рабочем файле
syncrepl rid=000
    provider=ldap://master-ldap.example.com
    type=refreshAndPersist
    retry="5 5 300 +"
    searchbase="dc=example,dc=com"
# требуются и пользовательские (*), и операционные (+) атрибуты
    attrs="*,+"
    bindmethod=simple
    binddn="cn=admin,ou=people,dc=example,dc=com"
# внимание: пароль посылается в открытом виде - небезопасно
    credentials=dirtysecret
```



7.2.1.2.3 OpenLDAP разнонаправленная `syncrepl`-репликация с несколькими главными серверами (N-Way Multi-Master)

В OpenLDAP 2.4 представлена поддержка разнонаправленной репликации с несколькими главными серверами. При такой конфигурации любое количество главных серверов может синхронизироваться друг с другом. Функциональность репликации была описана ранее для типов `refreshOnly` и `refreshAndPersist` и здесь мы не станем повторяться. Приведённые далее заметки и примеры конфигурации относятся только к разнонаправленной репликации с несколькими главными серверами.

Примечание: Для разнонаправленной репликации с несколькими главными серверами жизненно необходимо, чтобы системное время на всех главных серверах (поставщиках) было синхронизировано с одним и тем же источником времени, например, на всех серверах должен быть запущен NTP (Network Time Protocol).

При разнонаправленной репликации с несколькими главными серверами каждый **поставщик** сервиса синхронизации является также **потребителем** сервиса синхронизации, как показано на рисунке 7.2-4:

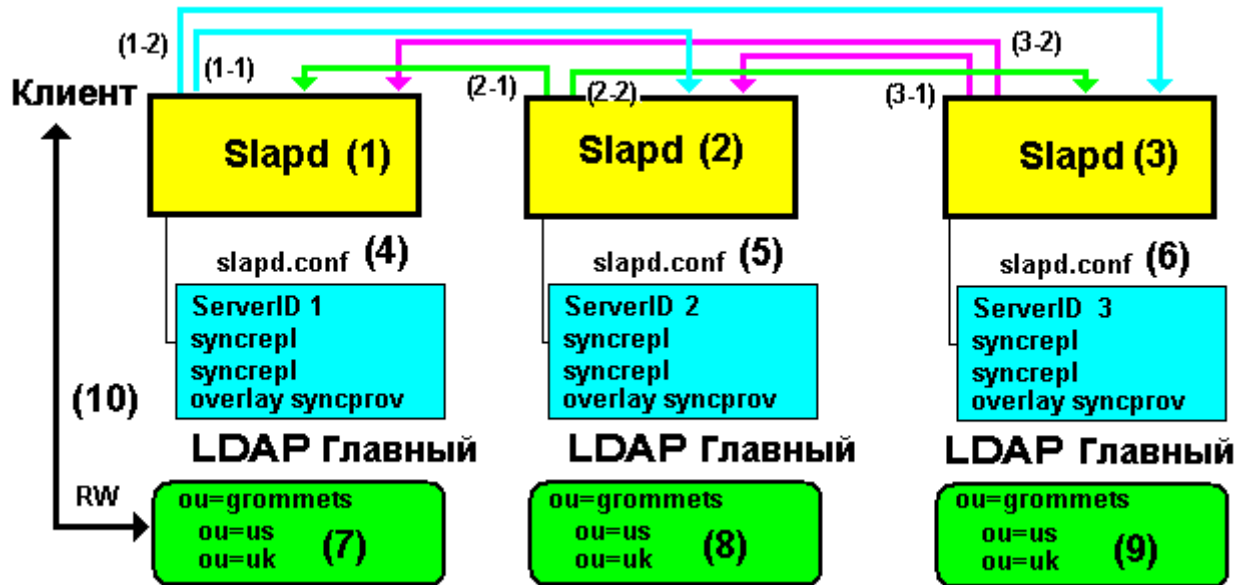


Рисунок 7.2-4: Разнонаправленная репликация syncrepl с несколькими главными серверами

На рисунке 7.2-4 показана конфигурация разнонаправленной репликации с тремя главными серверами (1, 2, 3). Каждый главный сервер сконфигурирован (в соответствующем файле slapd.conf (4, 5, 6)) как поставщик (с помощью директивы **overlay syncprov**) и как потребитель для всех остальных главных серверов (с помощью директив **syncrepl**). Каждый поставщик должен быть уникально идентифицирован с помощью директивы **ServerID**. Кроме того, все поставщики, как уже было сказано, должны быть синхронизированы с одним источником времени. Итак, в настройках поставщика (1) DIT (7) содержится директива **overlay syncprov** (наложение поставщика) и две директивы **syncrepl** типа **refreshAndPersist**, по одной для каждого из двух других поставщиков (2, 3), как показано соединительными линиями (1-1, 1-2). Два других поставщика имеют аналогичную конфигурацию — настройку поставщика и две директивы **syncrepl** типа **refreshAndPersist** для соответствующих других двух поставщиков.

В данной конфигурации подразумевается использование типа синхронизации **refreshAndPersist**, — непонятно, что может сподвигнуть Вас использовать тип **refreshOnly**, — поэтому изменения (10) содержимого каталога любого из главных серверов будут немедленно распространяться на все остальные главные серверы (серверы-поставщики).

Разнонаправленная репликация с несколькими главными серверами в версии 2.4 не поддерживает **delta-синхронизацию**.

Примеры конфигурации syncrepl refreshAndPersist:

Подразумевается три главных сервера (ldap1.example.com, ldap2.example.com и ldap3.example.com), использующих разнонаправленную репликацию syncrepl с несколькими главными серверами. У них будут такие файлы slapd.conf:

slapd.conf для ldap1.example.com:

```
# slapd главного сервера ldap1.example.com
# раздел глобальных настроек
...
# уникальный идентификатор данного сервера
serverID 001
# раздел database
database bdb
...
# разрешаем доступ на чтение всем потребителям
# подразумевается, что на всех главных серверах будет использоваться binddn с этим значением
# может понадобиться слияние с другими ACL
access to *
    by dn.base="cn=admin,ou=people,dc=example,dc=com" read
    by * break
```

```

# Примечание:
# директивы syncrepl для каждого из остальных главных серверов
# поставщик - ldap://ldap2.example.com:389,
# полное DIT (searchbase), синхронизируются все пользовательские и операционные атрибуты
# простой уровень безопасности с паролем в открытом виде
syncrepl rid=000
provider=ldap://ldap2.example.com
type=refreshAndPersist
retry="5 5 300 +"
searchbase="dc=example,dc=com"
attrs="*,+"
bindmethod=simple
binddn="cn=admin,ou=people,dc=example,dc=com"
credentials=dirtysecret

# поставщик - ldap://ldap3.example.com:389,
# полное DIT (searchbase), синхронизируются все пользовательские и операционные атрибуты
# простой уровень безопасности с паролем в открытом виде
syncrepl rid=001
provider=ldap://ldap3.example.com
type=refreshAndPersist
retry="5 5 300 +"
searchbase="dc=example,dc=com"
attrs="*,+"
bindmethod=simple
binddn="cn=admin,ou=people,dc=example,dc=com"
credentials=dirtysecret

...
# индексы, специфичные для syncprov (добавьте другие по необходимости)
index entryCSN eq
index entryUUID eq
...
# зеркальный режим необходим для того, чтобы разрешить запись
# эта директива должна определяться после всех директив syncrepl
mirrormode TRUE

# указываем поставщику использовать наложение syncprov
# (заключительные директивы в разделе database)
overlay syncprov
# contextCSN сохраняется в базу данных
# через каждые 100 обновлений или 10 минут
syncprov-checkpoint 100 10

```

slapd.conf для ldap2.example.com:

```

# slapd главного сервера ldap2.example.com
# раздел глобальных настроек
...
# уникальный идентификатор данного сервера
ServerID 002
# раздел database
database bdb
...
# разрешаем доступ на чтение всем потребителям
# подразумевается, что на всех главных серверах будет использоваться binddn с этим значением
# может понадобиться слияние с другими ACL
access to *
    by dn.base="cn=admin,ou=people,dc=example,dc=com" read
    by * break

# Примечание:
# директивы syncrepl для каждого из остальных главных серверов
# поставщик - ldap://ldap1.example.com:389,
# полное DIT (searchbase), синхронизируются все пользовательские и операционные атрибуты
# простой уровень безопасности с паролем в открытом виде
syncrepl rid=000
provider=ldap://ldap1.example.com
type=refreshAndPersist
retry="5 5 300 +"
searchbase="dc=example,dc=com"
attrs="*,+"
bindmethod=simple
binddn="cn=admin,ou=people,dc=example,dc=com"
credentials=dirtysecret

# поставщик - ldap://ldap3.example.com:389,
# полное DIT (searchbase), синхронизируются все пользовательские и операционные атрибуты
# простой уровень безопасности с паролем в открытом виде
syncrepl rid=001
provider=ldap://ldap3.example.com
type=refreshAndPersist
retry="5 5 300 +"
searchbase="dc=example,dc=com"
attrs="*,+"
bindmethod=simple

```

```

binddn="cn=admin,ou=people,dc=example,dc=com"
credentials=dirtysecret
...
# зеркальный режим необходим для того, чтобы разрешить запись
# эта директива должна определяться после всех директив syncsrepl
mirrormode TRUE

# индексы, специфичные для syncprov (добавьте другие по необходимости)
index entryCSN eq
index entryUUID eq
...
# указываем поставщику использовать наложение syncprov
# (заключительные директивы в разделе database)
overlay syncprov
# contextCSN сохраняется в базу данных
# через каждые 100 обновлений или 10 минут
syncprov-checkpoint 100 10

```

slapd.conf для ldap3.example.com:

```

# slapd главного сервера ldap3.example.com
# раздел глобальных настроек
...
# уникальный идентификатор данного сервера
ServerID 003
# раздел database
database bdb
...
# разрешаем доступ на чтение всем потребителям
# подразумевается, что на всех главных серверах будет использоваться binddn с этим значением
# может понадобиться слияние с другими ACL
access to *
    by dn.base="cn=admin,ou=people,dc=example,dc=com" read
    by * break

# Примечание:
# директивы syncsrepl для каждого из остальных главных серверов
# поставщик - ldap://ldap1.example.com:389,
# полное DIT (searchbase), синхронизируются все пользовательские и операционные атрибуты
# простой уровень безопасности с паролем в открытом виде
syncsrepl rid=000
    provider=ldap://ldap1.example.com
    type=refreshAndPersist
    retry="5 5 300 +"
    searchbase="dc=example,dc=com"
    attrs="*,+"
    bindmethod=simple
    binddn="cn=admin,ou=people,dc=example,dc=com"
    credentials=dirtysecret

# поставщик - ldap://ldap2.example.com:389,
# полное DIT (searchbase), синхронизируются все пользовательские и операционные атрибуты
# простой уровень безопасности с паролем в открытом виде
syncsrepl rid=001
    provider=ldap://ldap2.example.com
    type=refreshAndPersist
    retry="5 5 300 +"
    searchbase="dc=example,dc=com"
    attrs="*,+"
    bindmethod=simple
    binddn="cn=admin,ou=people,dc=example,dc=com"
    credentials=dirtysecret

# индексы, специфичные для syncprov (добавьте другие по необходимости)
index entryCSN eq
index entryUUID eq
# зеркальный режим необходим для того, чтобы разрешить запись
# эта директива должна определяться после всех директив syncsrepl
mirrormode TRUE

# указываем поставщику использовать наложение syncprov
# (заклучительные директивы в разделе database)
overlay syncprov
# contextCSN сохраняется в базу данных
# через каждые 100 обновлений или 10 минут
syncprov-checkpoint 100 10

```

Примечания:

1. Поскольку все главные серверы реплицируют одно и то же DIT, в данном примере показано, что значения параметра **binddn** в директивах **syncsrepl** одни и те же. Это вполне допустимо. Однако, при успешном взломе одного из серверов, все остальные также будут взломаны. Возможно, более безопасная стратегия — использовать разные записи в качестве **binddn** для каждого сервера. Для

этого потребуется внести изменения в директивы **syncrepl** и **access to**.

2. Каждый параметр **rid** директивы **syncrepl** должен быть уникальным в пределах файла `slapd.conf` (то есть в пределах одного сервера). Значение **serverid** должно быть уникальным для каждого сервера. Зависимостей между значениями **rid** и **serverid** нет.
3. Для конфигураций с несколькими главными серверами требуется директива **mirrormode true**. Она должна присутствовать после всех директив **syncrepl** в разделе `database`. Отсутствие этой директивы в какой-либо из конфигураций главного сервера приведёт к тому, что все операции записи будут завершаться неудачей!
4. Для конфигураций с несколькими главными серверами требуется, чтобы на всех серверах было одинаковое время. Все серверы должны быть клиентами NTP, ссылающимися на один и тот же источник времени. Недостаточно просто использовать команду **ntpdate** при загрузке сервера или другой подобный метод, поскольку погрешность часов может быть на удивление большой.
5. Внесение изменений в каталог — один из болезненных вопросов репликации с несколькими главными серверами. Для осуществления этой операции в OpenLDAP используются метки времени. Так, если обновление одного и того же атрибута (атрибутов) происходит примерно в одно и то же время (с учётом времени распространения) на разных серверах, то одно из обновлений для запрашиваемого атрибута будет потеряно. Потерянным будет обновление с меньшим значением отметки времени — хотя разница может составлять всего лишь миллисекунды. Это неизбежный побочный эффект репликации с несколькими главными серверами. NTP позволяет свести к минимуму возникновение этой проблемы.



7.2.1.2.4 Журналы сессии, журналы доступа и delta-синхронизация

Итак, до этого момента всё было на удивление просто. Теперь же добавим немного чёрной магии. И всё в интересах сокращения трафика между поставщиком и потребителем.

При синхронизации в режиме **refreshOnly** промежуток времени до распространения обновлений может быть значительным — в зависимости от значения параметра **interval** директивы **syncrepl**. Для уменьшения интервала между синхронизациями обновлений (чтобы минимизировать задержки распространения) эффективнее применять режим **refreshAndPersist**, но и в этом случае при каждом повторном подключении происходит начальная синхронизация всего каталога. Если в процессе синхронизации требуется выполнение **фазы наличия**, то, даже если с момента последней синхронизации изменения не происходили, каждая неизменённая запись всё равно будет отправлена в виде пустой записи (со своим идентификатором **entryUUID**), и такая синхронизация может занять значительное время.

В режиме **refreshAndPersist** пересинхронизация происходит только при начальном соединении (повторные соединения происходят только после крупных сбоев поставщика, потребителя или сети). Однако даже в этом режиме обновление какого-нибудь одного атрибута в записи приводит к тому, что данная запись передаётся целиком. Если в очень большой записи изменяется всего один атрибут, накладные расходы на пересылку получаются несоразмерно велики.

Наконец, непродуманные реализации LDAP могут создавать проблемы при обновлении. Для иллюстрации худшего варианта предположим, что есть приложение, которое выполняется раз в день и вносит изменения размером в 1 байт в каждую запись в DIT. Это приведёт к тому, что всем потребителям репликации будет рассылаться всё DIT целиком. Это, пожалуй, не слишком удобно, а порой даже и неосуществимо за промежуток в 24 часа.

Для решения этих проблем OpenLDAP предоставляет два метода: журнал сессии (session log) и журнал доступа (access log). Цель обоих методов — минимизировать передачу данных, а в случае журналов доступа — реализовать так называемую **delta-синхронизацию** (пересылку только изменений записи, а не всей записи).

Журналы сессии

Наложение **syncprov** предоставляет возможность вести журнал сессии. Параметр журнала сессии имеет следующую форму:


```
syncprov-sessionlog ops
# здесь ops определяет количество операций, которые могут быть сохранены
# при заполнении журнала старые операции удаляются
# Примечание: в версии 2.3 был параметр sid, но в 2.4 он был удалён

# пример определения syncprov
# с журналом сессии на 100 записей (изменений)
overlay syncprov
syncprov-checkpoint 100 10
syncprov-sessionlog 100
```

Журнал сессии — это журнал, размещаемый в оперативной памяти и содержащий все операции (за исключением операций добавления add). В зависимости от того, изменения за какой промежуток времени хранятся в журнале сессии на момент начала синхронизации, он может позволить поставщику пропустить необязательную **фазу наличия** и таким образом значительно ускорить процесс синхронизации. Например, если с момента последней синхронизации в журнале сессии не зафиксировано новых изменений, в **фазе наличия** нет нужды. Журнал сессии может использоваться с любым режимом репликации (**refreshOnly** или **refreshAndPersist**), но больше пользы он приносит при режиме **refreshOnly**. Если журнал сессии недостаточно велик, чтобы покрыть все изменения с момента последнего запроса на синхронизацию от потребителя, то будет выполняться полная последовательность действий по пересинхронизации (включая **фазу наличия**). Для использования журнала сессии не требуется указания никаких специальных параметров в директиве `syncrepl` на стороне потребителя.

Журналы доступа (delta-синхронизация)

Журнал доступа `accesslog` позволяет вести журнал операций LDAP целевого DIT в связанном, но отдельном `accesslog` DIT. Функциональность `accesslog` обеспечивается директивой [overlay accesslog](#).

В нормальном случае, операции синхронизации реплики производят обновления, используя информацию из того DIT, к которому выполняется поисковый запрос, содержащийся в запросе синхронизации. Вместо этого можно выполнять синхронизацию, перенацелив директиву [syncrepl](#) на журнал доступа `accesslog`. Поскольку объекты, хранящиеся в журнале доступа, содержат только изменения (в том числе операции удаления, добавления, переименования и изменения `rdn` записей), объём данных получается значительно ниже, нежели при выполнении полной операции синхронизации на основном DIT (или даже на фрагменте DIT), когда при изменении любого атрибута запись пересылается целиком. Использование `accesslog` известно как **delta-репликация** или **delta-синхронизация**, и даже как **delta-syncrepl**.

Для того, чтобы использовать данную форму репликации, требуется определить `accesslog` DIT на поставщике и параметры **logbase**, **logfilter** и **syncdata** директивы [syncrepl](#) на потребителе, как показано в примере ниже.

Примеры delta-репликации (accesslog):

Конфигурация поставщика (подразумевается, что имя хоста `ldap1.example.com`):

```
# slapd поставщика ldap1.example.com
# раздел глобальных настроек
...
# Разрешаем потребителю доступ на чтение к целевому DIT и accesslog DIT.
# В данной форме применяется глобальное правило доступа
# Указываемый DN ДОЛЖЕН совпадать с тем, который используется в
# параметре binddn директивы syncrepl потребителя
access to *
    by dn.base="cn=admin,ou=people,dc=example,dc=com" read
    by * break
...
# раздел database - целевое DIT
# с суффиксом dc=example,dc=com
database bdb
suffix "dc=example,dc=com"
...
# индексы, специфичные для syncprov (добавьте другие по необходимости)
# их определение не обязательно, но помогает повысить производительность
index entryCSN,entryUUID eq
...
# определяем наложение accesslog и его параметры
# ежедневная очистка accesslog:
# удаляем записи старше двух дней
# заносим в журнал операции записи writes (включаются add, delete, modify, modrdn)
# заносим в журнал только успешные операции
# accesslog DIT имеет суффикс cn=deltalog
overlay accesslog
```

```

logdb "cn=deltalog"
logops writes
logsuccess TRUE
logpurge 2+00:00 1+00:00

# указываем поставщику использовать наложение syncprov
# (заключительные директивы в разделе database)
overlay syncprov
# contextCSN сохраняется в базу данных
# через каждые 100 обновлений или 10 минут
syncprov-checkpoint 100 10

# Теперь определяем accesslog DIT
# нормальное определение database
database bdb

...
suffix "cn=deltalog"
# это рекомендуется для оптимизации accesslog
index default eq
index entryCSN,objectClass,reqEnd,reqResult,reqStart
...
# accesslog DIT также будет поставщиком
# syncprov-nopresent подавляет фазу наличия при синхронизации
# syncprov-reloadhint TRUE обязательно при delta-синхронизации
overlay syncprov
syncprov-nopresent TRUE
syncprov-reloadhint TRUE

```

Конфигурация потребителя:

```

# slapd потребителя ldap3.example.com
# раздел глобальных настроек
...

# раздел database
database bdb
suffix "dc=example,dc=com"
...

# Примечание:
# директива syncrepl будет использовать accesslog
# для delta-синхронизации
# поставщик - ldap://ldap1.example.com:389,
# полное DIT (searchbase), синхронизируются все пользовательские атрибуты
# простой уровень безопасности с паролем в открытом виде
# binddn используется для авторизации доступа к поставщику
# logbase ссылается на logdb (deltalog) поставщика
# logfilter позволяет запрашивать успешные операции add, delete, modify, modrdn
# syncdata указывает использовать формат accesslog
syncrepl rid=000
provider=ldap://ldap1.example.com
type=refreshAndPersist
retry="5 5 300 +"
searchbase="dc=example,dc=com"
attrs="*,+"
bindmethod=simple
binddn="cn=admin,ou=people,dc=example,dc=com"
credentials=dirtysecret
logbase="cn=deltalog"
logfilter="(&(objectClass=auditWriteObject)(reqResult=0))"
syncdata=accesslog
...

```

Примечания:

1. Поиск с фильтром, указанным в параметре logfilter ("(&(objectClass=auditWriteObject)(reqResult=0))") будет возвращать все стандартные записи в [accesslog](#), содержащие сведения об успешных операциях (reqResult=0). Поскольку в конфигурации поставщика мы определили, что в accesslog будут помещаться сведения только об успешных операциях (logsuccess TRUE), теоретически этот фильтр избыточен, но и вреда не приносит.
2. Определение database для accesslog (cn=deltalog) на поставщике не содержит директив **rootdn** или **rootpw**, поскольку необходимости в них нет. Там также нет директив **access to**, что означает применение глобального ACL, определённого в начале файла slapd.conf поставщика. Указанному в этом ACL пользователю предоставлены минимальные права доступа и к целевому, и к accesslog DIT, однако требуется, чтобы такая запись реально существовала в целевом DIT. Использование rootdn и rootpw целевого DIT в качестве параметра binddn директивы syncrepl поставщика также будет работать (при условии, что глобальная директива **access to** будет удалена, а соответствующая директива **access to** добавлена в определение accesslog DIT), однако это подвергает привилегированного пользователя

угрозе потенциальных атак прослушивания сетевого трафика и не рекомендуется.

3. Потребитель может быть запущен с пустым DIT, в этом случае сначала произойдёт нормальная синхронизация, по завершении которой последующие обновления будут синхронизироваться через механизм `accesslog`.
4. При передаче изменений, зафиксированных в `accesslog`, потребителю предоставляется значение нового атрибута, а также `entryCSN`, `modifiersName` (DN) и `ModifyTimestamp`. Последние три атрибута являются операционными и требуются для поддержания точной копии DIT у потребителя.

7.2.2 Репликация ApacheDS

Уже совсем скоро (One day real soon now™)



7.3.1 Синхронизация DIT перед запуском репликации в стиле syncrep1

Существует две возможные стратегии первоначального запуска репликации в стиле `syncrep1`:

1. Не делать ничего. После конфигурации потребителя нет необходимости предпринимать что-либо ещё. Первоначальный запрос синхронизации выполнит синхронизацию реплики из пустого состояния. Если DIT очень велико, это может занять неприемлемо долгое время.
2. Загрузка LDIF-копии реплики с сервера-поставщика с помощью `slapadd` до запуска репликации. В зависимости от того, как выполнялась эта процедура, первоначальная синхронизация может быть минимальной, либо её вообще может не быть. Следующая инструкция пошагово описывает данный процесс. Предполагается, что поставщик использует OpenLDAP 2.2+ и уже [настроен на репликацию](#):

1. Сохраните LDIF-копию DIT поставщика (с помощью LDAP-браузера или даже [slapcat](#), если используется механизм манипуляции данными BDB или HDB). Нет нужды останавливать сервер-поставщик, поскольку изменения, произошедшие во время сохранения или в промежутке между сохранением копии DIT и её загрузкой на сервер-потребитель, будут синхронизированы во время первоначальной репликации.
2. Скопируйте LDIF-файл на сервер-потребитель.
3. Сконфигурируйте сервер-потребитель на выполнение репликации.
4. Загрузите LDIF в каталог потребителя с помощью [slapadd](#) с опцией `-w` для создания **SyncCookie** с наиболее поздним временем внесения изменений. Пример:

```
slapadd -l /path/to/provider/copy/ldif -w
```

5. Запустите сервер-потребитель.
6. Выполните тестовую транзакцию либо на поставщике (в конфигурации главный-подчинённый), либо на одном из поставщиков (в конфигурации с несколькими главными серверами) и убедитесь, что изменения были распространены.
3. В случае конфигурации главный-подчинённый Вы, возможно, захотите добавить директиву [olcReferral](#) (`referral`) и/или директивы [olcUpdaterf](#) (`updaterf`).

7.3.2 Синхронизация DIT перед запуском репликации в стиле slurpd

Перед тем как репликация в стиле `slurpd` сможет функционировать, DIT (главное и подчинённое (подчинённые)) должны быть приведены в одинаковое состояние. Сначала должен быть выполнен процесс

ручной синхронизации, описанный ниже:

Примечание: slurpd считается устаревшим в OpenLDAP 2.4 и эта информация включена сюда лишь для полноты картины. В случае, когда OpenLDAP использует репликацию в стиле syncrepl, подчинённый сервер или один из главных серверов в конфигурации с несколькими главными серверами, может начать синхронизироваться, вообще не имея записей в DIT. Тем не менее, описанный ниже процесс также может быть использован, и, в зависимости от объёма каталога, он может обеспечить более эффективный (быстрый) запуск подчинённого DIT.

1. Остановите LDAP-сервер, который будет содержать главный экземпляр DIT. Это необходимо для предотвращения дальнейшего внесения изменений в DIT.
2. Создайте LDIF-копию того DIT, которое будет реплицироваться, с помощью соответствующего off-line инструмента для LDAP-сервера.

(В OpenLDAP используйте [slapcat](#)).

3. Сконфигурируйте данный сервер для запуска главного экземпляра DIT.

Для репликации в стиле slurpd OpenLDAP: добавьте директивы [replica](#), [repllogfile](#) и [replicationinterval](#) в файл slapd.conf. Пока не перезапускайте сервер.

Примечание: Если OpenLDAP использует [on-line конфигурацию \(cn=config\)](#), сервер должен быть активен.

4. Скопируйте LDIF-файл, созданный на шаге 2, на сервер (серверы), на которых будет запущен подчинённый экземпляр DIT (или один из главных серверов в конфигурации с несколькими главными серверами, применимо только для репликации в стиле syncrepl).
5. Остановите LDAP-сервер, на котором будет запущен подчинённый экземпляр DIT (или один из главных серверов в конфигурации с несколькими главными серверами, применимо только для репликации в стиле syncrepl).
6. Примените LDIF-файл, скопированный на шаге 4, с помощью соответствующего off-line инструмента для LDAP-сервера.

(В OpenLDAP используйте [slapadd](#). Поскольку сервер ещё не сконфигурирован, нужно использовать опцию -n (dbnum)).

7. Сконфигурируйте сервер, на котором будет запущен подчинённый экземпляр DIT (или один из главных серверов в конфигурации с несколькими главными серверами, применимо только для репликации в стиле syncrepl) на работу в качестве подчинённого или одного из главных серверов.

Для репликации в стиле slurpd OpenLDAP это означает определение директивы [database](#) и всех связанных с ней директив (поскольку при добавлении DIT использовалась опция -n dbnum, порядок, в котором определяются разделы database, очень важен), для спецификации репликации добавьте директиву [updatedn](#) и директивы [updateref](#).

Примечание: Если OpenLDAP использует [конфигурацию времени исполнения \(cn=config\)](#), сервер должен быть активен.

8. В конфигурации главный-подчинённый запустите сервер, на котором будет работать подчинённый экземпляр DIT. Убедитесь, что он заработал. В конфигурации с несколькими главными серверами (только для репликации в стиле syncrepl) запустите данную копию главного сервера и убедитесь, что он заработал.
9. Запустите сервер, на котором работает главный экземпляр DIT, или другой главный сервер в конфигурации с несколькими главными серверами (только для репликации в стиле syncrepl).
10. Выполните тестовую транзакцию на главном сервере (на одном из главных серверов в конфигурации с несколькими главными серверами, только для репликации в стиле syncrepl) и убедитесь, что она была распространена на подчинённый сервер (или другой главный сервер). Если нет — изучайте журналы. И начинайте паниковать — это всегда помогает.



Глава 7. Отсылки и псевдонимы

7.4 Отсылки

Отсылка - это процесс, посредством которого LDAP-сервер, вместо того, чтобы вернуть результат запроса, возвращает ссылку (отсылку, *referral*) на другой LDAP-сервер, который может содержать дополнительную информацию. Отсылка может рассматриваться как **переход к внешнему LDAP-серверу (DIT)**. В документации OpenLDAP в контексте отсылок используются запутанные термины **вышестоящий** (*superior*) и **нижестоящий** (*subordinate*). Условия, при которых OpenLDAP будет возвращать отсылки:

1. Общая отсылка возвращается в том случае, когда DN поискового запроса не входит в область ни одного из деревьев, указанных в директивах *suffix* данного LDAP-сервера. Эта функция настраивается с помощью директивы *referral* в разделе глобальных настроек *slapd.conf*. Подробнее об этом [здесь](#).
2. Если клиент пытается внести изменения в подчинённое DIT (или DIT потребителя), сервер может быть настроен на возврат отсылки с помощью **атрибута *olcUpdateref* (директивы *updateref*)** записи ***olcDatabase cn=config*** (раздела *database* *slapd.conf*). Подробнее об этом [здесь](#).
3. Использование объектного класса *referral* в DIT. Это позволяет делегировать ответственность за обслуживание части LDAP-системы одной или нескольким другим LDAP-системам. Подробнее об этом [здесь](#).

По умолчанию за переход по отсылкам отвечает LDAP-клиент (LDAP-браузер или утилита). Есть опциональная возможность настроить серверы OpenLDAP на следование по определениям объектных классов *referral* (или их разрешение), вместо того, чтобы просто возвращать отсылку. Данная возможность использует директиву *overlay chain*. Подробнее об этом [здесь](#).

7.4.1 Общие отсылки

Если в запросе LDAP-клиента LDAP-серверу фигурирует неверный DN (базовая часть DN не входит ни в одно DN из указанных в директивах *suffix* данного сервера), то сервер вернёт ошибку. В то же время, сервер может быть сконфигурирован на возвращение отсылки (LDAP URI) на один или несколько серверов, которые, вероятно, смогут предоставить дополнительную информацию. Данная возможность определяется с помощью директивы *referral* в разделе глобальных настроек *slapd.conf*, как показано в следующем примере:

```
# slapd.conf
# раздел глобальных настроек
...
# Если сервер получил поисковый запрос с DN
# который не входит ни в одно DN из указанных в директивах suffix
# какого-либо из разделов database, он вернёт оба LDAP URI.
# В данном случае отсылки будут возвращаться, если DN
# в поисковом запросе не заканчивается на dc=example,dc=net
referral ldap://ldap-master.example.com
referral ldap://ldap-services.example.com:10389
...
# раздел (разделы) database
database bdb
...
suffix "dc=example,dc=net"
...
```

7.4.2 Отсылки при попытках внести изменения в подчинённый сервер (сервер-потребитель)

Если LDAP-клиент пытается выполнить запрос на запись (модификацию) содержимого **подчинённого сервера при репликации в стиле *slurpd*** или **потребителя при репликации в стиле *sync repl*** (в простой конфигурации главный-подчинённый), этот запрос будет отклонён. В то же время, сервер может быть сконфигурирован на возвращение отсылки (LDAP URI), которая обычно указывает на главный сервер (на поставщика) репликации, с помощью **атрибута *oldUpdateref* (директивы *updateref*)**, как показано в следующем примере:

```
# slapd.conf подчинённого сервера (или потребителя)
# раздел глобальных настроек
...
...
```

```
# раздел (разделы) database
database bdb
...
# директива updatedn (для репликации в стиле slurpd)
# ИЛИ
# директива syncrepl

# отсылка на DIT главного сервера (поставщика)
updateref ldap://ldap-master.example.com
...
```



7.4.3 Объекты referral

Отсылки могут быть явно определены в DIT с помощью объектного класса [referral](#). У этого объектного класса единственный атрибут [ref](#) (LDAP URI).

Для иллюстрации этого процесса предположим, что есть LDAP-система с делегированием (основанная на отсылках), как показано на рисунке 7.4-1:

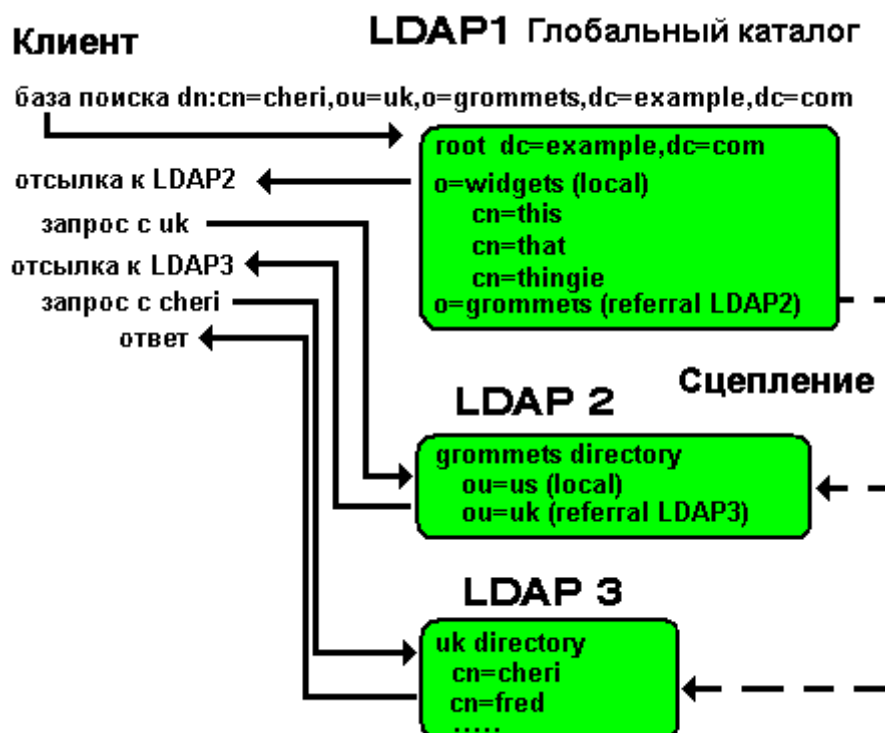


Рисунок 7.4-1 - Отсылки на LDAP2 и LDAP3

Чтобы определить отсылку от LDAP1 к LDAP2, используется следующий набор инструкций [LDIF](#):

```
# данное определение создаёт RDN o=grommets
# и устанавливает отсылку от него к ldap2
dn: o=grommets,dc=example,dc=com
objectClass: referral
objectClass: extensibleObject
o: grommets
ref: ldap://ldap2.example.com/o=grommets,dc=example,dc=net
```

Примечания:

1. Объектный класс [extensibleObject](#) позволяет добавить любой атрибут (в данном случае **o (organizationName)**) к объектному классу [referral](#).
2. Отсылка к LDAP2 будет возвращаться (или будет выполнено её [разрешение](#)), если сервер LDAP1.example.com получит поисковый запрос с таким DN:

```
cn=cheri,ou=uk,o=grommets,dc=example,dc=com
```

Чтобы определить отсылку от LDAP2 к LDAP3, используется следующий набор инструкций [LDIF](#):


```
# запись с объектным классом organizationalUnit
# и с атрибутом organizationName (o), имеющим значение grommets,
# должна быть определена на данном сервере,
# либо суффикс LDAP2 должен быть задан как o=grommets,dc=example,dc=com

# данное определение создаёт RDN o=uk
# и устанавливает отсылку от него к ldap3
dn: ou=uk,o=grommets,dc=example,dc=com
objectClass: referral
objectClass: extensibleObject
ou: uk
ref: ldap://ldap3.example.com/ou=uk,o=grommets,dc=example,dc=net
```

Примечания:

1. Объектный класс [extensibleObject](#) позволяет добавить любой атрибут (в данном случае **o (organizationName)**) к объектному классу referral.
2. Отсылка к LDAP3 будет возвращаться (или будет выполнено её [разрешение](#)), если сервер LDAP2.example.com получит поисковый запрос с таким DN:

```
cn=cheri,ou=uk,o=grommets,dc=example,dc=com
```



7.4.4 Удаление отсылок

При использовании стандартных утилит, таких как [ldapmodify](#), удаление отсылок выполняется несколько мудрёно. Напомним, что LDAP-клиенты всегда следуют по отсылкам, таким образом, при поиске экземпляра отсылки, которую требуется удалить (или модифицировать), LDAP-сервер определяет объектный класс referral и немедленно посылает отсылку, по которой потом следует клиент, в результате запись-отсылка так и не будет найдена. Для пресечения подобного автоматического поведения отсылок на LDAP-сервере, клиент должен использовать элемент управления Manage DSA IT (OID 2.16.840.1.113730.3.4.2, определён в [RFC 3296](#)). В случае с [ldapmodify](#) для посылки данного элемента управления при попытке удаления или модификации записи-отсылки должен быть использован аргумент -M.



7.4.5 Сцепление (следование по отсылкам)

Обычно, если в процессе поиска LDAP-серверу встречается [объект referral](#), он вернёт LDAP-клиенту отсылку. Если это требуется, сервер может быть настроен на следование по отсылкам (разрешение отсылок) и возвращение клиенту полного результата. Этот процесс часто называется **сцеплением (chaining)** и конфигурируется на сервере с помощью директивы [overlay chain](#). Сцепление показано на рисунке 7.4-2:

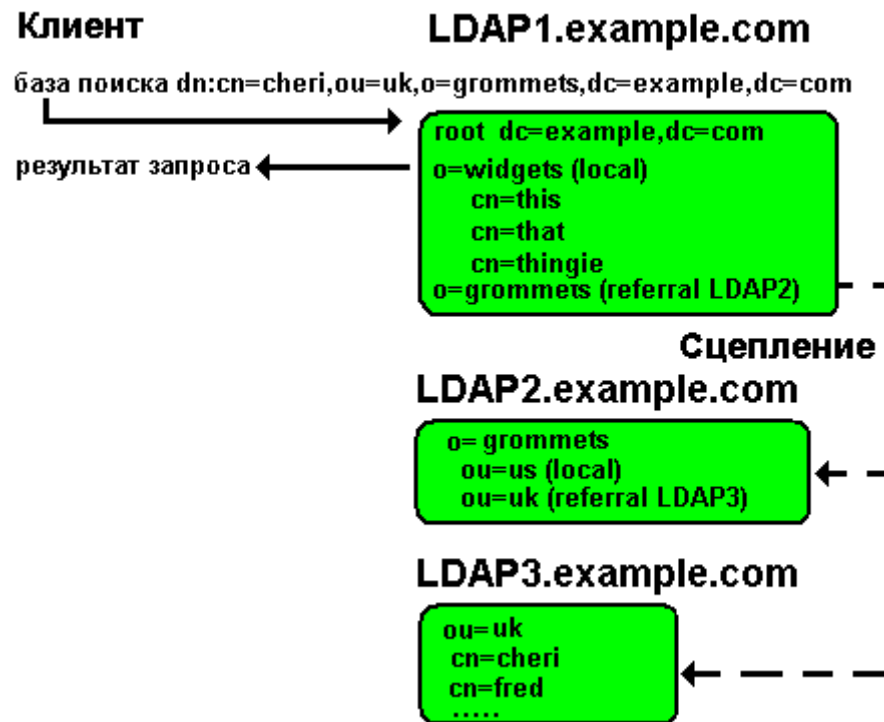


Рисунок 7.4-2 - Сцепление с LDAP2 и LDAP3

В следующем примере демонстрируется конфигурация, которую нужно произвести на сервере LDAP1 для выполнения сцепления (следования) по двум отсылкам, как показано на рисунке 7.4-2:

```
# slapd.conf
# раздел глобальных настроек
...
# общая отсылка для запросов с неверным DN
referral ldap:ldap-master.example.com
...
# раздел (разделы) databases
database bdb
...
suffix "dc=example,dc=com"
...

overlay chain
# разрешаем следовать по двум отсылкам: из данного DIT
# и из того DIT, на которое ссылается данное DIT
chain-max-depth 2
# в случае какой-либо ошибки возвращается отсылка
chain-return-error FALSE

overlay chain
chain-uri "ldap://ldap2.example.com"
chain-rebind-as-user yes
chain-idassert-bind bindmethod="simple"
binddn="cn=admin,dc=example,dc=com"
credentials="secret"
mode="self"
chain-uri "ldap://ldap3.example.com"
chain-rebind-as-user yes
chain-idassert-bind bindmethod="simple"
binddn="cn=admin,dc=example,dc=com"
credentials="secret"
mode="self"
```

Полное описание директив конфигурации наложения chain [здесь](#).



7.5 Псевдонимы

Псевдонимы LDAP (определены в [RFC 4512](#)) используют структурный (STRUCTURAL) объектный класс [alias](#) для определения позиции в текущем DIT и атрибут [aliasedObjectName](#) для определения DN записи, содержащей информацию. LDAP-сервер автоматически разыменовывает псевдоним и возвращает запись, на

которую указывает DN в атрибуте `aliasedObjectName`. Псевдоним может рассматриваться как **переход внутри LDAP-сервера**, поскольку при его разыменовании используется только DN. Примеры псевдонимов:

```
# Пример 1
# Классический псевдоним: два имени отображаются на одну запись.
# Используется единственное DIT (суффикс dc=example,dc=com).
# Любители танго возмутятся от такого псевдонима.

dn: ou=candombe,ou=tango,dc=example,dc=com
objectClass: alias
objectClass: extensibleObject
ou: candombe
aliasedObjectName: ou=milonga,ou=tango,dc=example,dc=com

# При ссылке на ou=candombe,ou=tango,dc=example,dc=com
# возвращаются данные из ou=milonga,ou=tango,dc=example,dc=com

# Пример 2
# Замена одного DN другим DN.
# Допустим, одна компания поглотила другую.
# Подразумевается, что на сервере два DIT (суффиксы dc=example,dc=de и dc=example,dc=kr)

dn: uid=jschmidt,ou=people,dc=example,dc=de
objectClass: alias
objectClass: extensibleObject
uid: jschmidt
aliasedObjectName: uid=jschmidt,ou=people,dc=example,dc=kr

# При ссылке на uid=jschmidt,ou=people,dc=example,dc=de
# возвращаются данные из uid=jschmidt,ou=people,dc=example,dc=kr
```

Примечания:

1. Объектный класс [extensibleObject](#) позволяет добавить к объектному классу `alias` любой атрибут (в первом примере - **ou** (`organizationalUnitName`), во втором - `uid`).
2. Атрибут `aliasedObjectName` существенно влияет на то, какие записи будут возвращаться сервером LDAP. Так, в первом из приведённых примеров при запросе записи `cn=Siga El Baile,ou=candombe,ou=tango,dc=example,dc=com` будет возвращена запись `cn=Siga El Baile,ou=milonga,ou=tango,dc=example,dc=com`.
3. В отличие от отсылок, для удаления записей с объектным классом `alias` не требуется предпринимать каких-либо специальных действий, то есть при выполнении операции `delete` LDAP-сервер автоматически подавляет разыменование.



Глава 8. LDAP LDIF и DSML

В данной главе описываются различные методы импорта и экспорта [записей](#) LDAP и [DIT](#) целиком, используя либо LDIF, либо [DSML](#).

Формат LDAP Data Interchange Files (LDIF) определён в [RFC 2849\(рус.\)](#) ([RFC 2849\(ориг.\)](#)).

Содержание

- 8.1 Обзор LDIF
- 8.2 Формат и директивы LDIF
 - 8.2.1 Формат файла LDIF
 - 8.2.1.1 Терминология LDIF и типы строк
 - 8.2.1.2 Пример LDIF
 - 8.2.2 Директивы LDIF
 - 8.2.2.1 Директива `add`
 - 8.2.2.2 Директива `attributename`
 - 8.2.2.3 Директива `changetype`
 - 8.2.2.4 Директива `control`
 - 8.2.2.5 Директива `delete`

- 8.2.2.6 Директива deleteoldrdn
- 8.2.2.7 Директива dn
- 8.2.2.8 Директива newrdn
- 8.2.2.9 Директива newsuperior
- 8.2.2.10 Директива objectclass
- 8.2.2.11 Директива replace
- 8.2.2.12 Директива version

8.3 Обработка двоичных данных (включая пароли) в LDIF

8.4 Импорт файлов LDIF

8.5 Примеры LDIF

8.1 Обзор LDIF

Файлы LDIF используются в пяти основных случаях:

1. Для первоначального создания структуры DIT.
2. Для добавления (импорта) больших записей в каталог.
3. Для восстановления (импорта) каталога.
4. Для архивирования (экспорта) каталога.
5. Для выполнения крупных изменений каталога.

OpenLDAP предоставляет ряд [инструментов](#) для импорта и экспорта LDIF-файлов.

LDIF-файл представляет собой простой текстовый файл, который может быть создан и отредактирован с помощью любого текстового редактора. Поскольку каждая строка ограничивается ЛИБО <LF> (unix-формат), ЛИБО <CR><LF> (windows-формат), такие файлы могут быть созданы в любой операционной системе.



8.2 Общий формат LDIF

8.2.1 Формат LDIF

LDIF может быть довольно придирчив - наличие или отсутствие пробелов является особенно важным. Формат LDIF состоит из нескольких ТИПОВ СТРОК, часть из которых может содержать **директивы** LDIF. Строки могут быть организованы в последовательности ЗАПИСИ и последовательности ОПЕРАТОРА. Каждая строка ограничивается ЛИБО <LF> (unix-формат), ЛИБО <CR><LF> (windows-формат). Чтобы упростить дальнейшие объяснения, мы провели классификацию и дали названия ТИПАМ СТРОК. Этих названий мы будем придерживаться при последующем описании директив.

8.2.1.1 Терминология LDIF и типы строк

В этом разделе определена терминология, которую мы используем при описании LDIF-файлов. Эта терминология была создана для упрощения объяснения некоторых наиболее таинственных особенностей LDIF-файлов.

- **Типы строк LDIF**

1. СТРОКА ДИРЕКТИВЫ (DIRECTIVE LINE) - это строка, которая начинается (первый символ в строке) с любого символа, ЗА ИСКЛЮЧЕНИЕМ ПРОБЕЛА или # (решётки).
2. СТРОКА ПРОДОЛЖЕНИЯ (CONTINUATION LINE) - это следующая за СТРОКОЙ ДИРЕКТИВЫ строка, которая начинается (первый символ в строке) С ПРОБЕЛА - последующие символы считаются частью предыдущей строки. Может быть любое количество СТРОК ПРОДОЛЖЕНИЯ, вплоть до предельного размера, установленного для атрибута.
3. ПУСТАЯ СТРОКА (BLANK LINE) - это строка без символов (обычно создаваемая клавишей ENTER). ПУСТЫЕ СТРОКИ обычно используются для разделения последовательностей ЗАПИСИ.

4. СТРОКА КОММЕНТАРИЯ (COMMENT LINE) - это строка, которая начинается (первый символ в строке) с символа # (решётка).
5. СТРОКА-РАЗДЕЛИТЕЛЬ (SEPARATOR LINE) - это строка, которая начинается (первый символ в строке) с символа - (тире). СТРОКИ-РАЗДЕЛИТЕЛИ обычно используются для ограничения последовательностей ОПЕРАТОРА.

- **Последовательности LDIF**

1. Последовательность ЗАПИСИ (ENTRY sequence) - это группа директив, обычно начинающаяся с директивы **dn:**, каждая директива в которой применяется к одной и той же записи в DIT. Последовательность ЗАПИСИ обычно начинается и заканчивается ПУСТОЙ строкой.
2. Последовательность ОПЕРАТОРА - это группа директив, используемых с директивой **changetype:** **modify**, каждая директива в которой применяется к одной и той же операции **add**, **delete** или **replace**. Последовательность ОПЕРАТОРА обычно ограничивается либо строкой-РАЗДЕЛИТЕЛЕМ, либо ПУСТОЙ строкой.



8.2.1.1 Пример LDIF

Данный пример LDIF иллюстрирует создание DIT с помощью [стандартного файла примера](#) и показывает синтаксис, используемый чаще всего. Такие файлы обычно загружают с помощью утилиты [ldapadd](#):

```
## DEFINE DIT ROOT/BASE/SUFFIX #####
## используется формат RFC 2377
## замените встречающиеся ниже example и com на требуемые
## или, для экспериментов, оставьте как есть

## dcObject - это ВСПОМОГАТЕЛЬНЫЙ объектный класс и, кроме него, запись
## ДОЛЖНА иметь СТРУКТУРНЫЙ объектный класс (в данном случае, organization)
# это последовательность ЗАПИСИ и ей предшествует ПУСТАЯ СТРОКА

dn: dc=example,dc=com
dc: example
description: Моя замечательная компания. В эту строку можно поместить столько текста, сколько
хотите
в этой строке продолжение информации из предыдущей строки вплоть до 32Kb
строка оканчивается либо на <CR>, либо на <CR><LF>, то есть отрабатывается ENTER
с систем как Windows, так и *nix - новая строка ДОЛЖНА начинаться с ОДНОГО ПРОБЕЛА
objectClass: dcObject
objectClass: organization
o: Example, Inc.

## ПЕРВЫЙ уровень иерархии - люди (people)
## для объектных классов используется смешанная форма записи в верхнем и нижнем регистре
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой

dn: ou=people, dc=example,dc=com
ou: people
description: All people in organisation
objectclass: organizationalunit

## ВТОРОЙ уровень иерархии
## ДОБАВЛЯЕМ одну запись в ПЕРВЫЙ уровень (people)
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой
# ou: Human Resources - это название подразделения

dn: cn=Robert Smith,ou=people,dc=example,dc=com
objectclass: inetOrgPerson
cn: Robert Smith
cn: Robert J Smith
cn: bob smith
sn: smith
uid: rjsmith
userpassword: rJsmith
carlicense: HISCAR 123
homephone: 555-111-2222
mail: r.smith@example.com
mail: rsmith@example.com
mail: bob.smith@example.com
description: swell guy
ou: Human Resources
```

Примечания:

1. **<предупреждение>** В версиях данного руководства до 0.1.2 в приведённом выше примере мы некорректно определяли **dn: dc=example,dc=com** как **dn: dc=example.com**. Это успешно загружалось в OpenLDAP 2.0 и 2.1, но стало отклоняться в OpenLDAP 2.2 (ошибка 64). **</предупреждение>**
2. Записи при добавлении начинаются со строки, первыми символами которой являются 'dn:'. В общем случае, для этих целей может использоваться любой атрибут при условии соблюдения уникальности 'dn:', и, во избежание излишней нагрузки при поиске, обычно им становится наиболее часто используемый при доступе к записи DN. Поэтому в последней записи в приведённом выше LDIF используется значение **cn=Robert Smith,ou=people,dc=example,dc=com**, подразумевая, что cn= будет наиболее часто используемым DN при доступе к записям. Однако, если запись будет использоваться для аутентификации, то данный DN ДОЛЖЕН быть таким, чтобы его можно было использовать во время операций подсоединения. В этом случае более подходящими могут быть DN типа **uid=rjsmith,ou=people,dc=example,dc=com**. Поскольку для данного начального DN (или DN 'создания') нет специального термина в стандартах LDAP, то, при использовании в целях идентификации, он иногда называется **DN принциала (Principal DN)**. [Дополнительная информация по этой теме](#).
3. Для упрощения описания некоторых концепций LDIF мы используем некоторые введённые нами [термины](#).
4. Как упоминалось в комментариях, директива **version:** не является строго обязательной. Если она присутствует, то (в настоящий момент) она должна быть установлена в 1 для указания версии 1 формата LDIF. Данная директива включена просто потому, что делать это полезно (Good Thing™ to do). Будущие версии могут быть несовместимы, либо могут налагать более строгие требования к корректности - лучше занять хорошие привычки сейчас. Во время нашего тестирования мы заметили, что некоторые версии OpenLDAP могут приводиться в ступор директивами **version**, выдавая сообщения об ошибках с предположениями об отсутствии DN. Если это произошло, удалите строку version и все соответствующие комментарии.
5. Комментарии обозначаются # только в первом символе строки. В следующей строке # интерпретируется как содержимое:

```
cn: my name #this is my name
```

В результате атрибут cn будет содержать 'my name #this is my name'.

6. Между записями должна быть КАК МИНИМУМ одна [ПУСТАЯ строка](#) (перед строками, начинающимися с **dn:**). Это ОЧЕНЬ важно - в противном случае могут возникать странные ошибки.
7. Предполагается, что строка является [строкой ПРОДОЛЖЕНИЯ](#), если предыдущая строка оканчивается разделителем строк, а текущая начинается РОВНО С ОДНОГО ПРОБЕЛА.
8. В именах атрибутов в приведённом выше файле непоследовательно используются символы в верхнем и нижнем регистрах - в частности, эта ужасная псевдовенгерская нотация **objectClass** или все буквы в нижнем регистре **objectclass**. Работает и то, и другое. Следуйте любому стилю на Ваш выбор.
9. Строка **cn: bob smith** содержит несколько **кажущихся** ошибок. Здесь есть два пробела между 'bob' и 'smith' и оба они в нижнем регистре. Ни одна из этих **кажущихся** ошибок не имеет никакого эффекта в работе каталога, поскольку атрибут **cn** (дочерний от атрибута **name**) использует нечувствительное к регистру [правило соответствия](#) и LDAP при поиске выполняет некоторые [интересные вещи](#).
10. В большом количестве документации Вы встретите **objectclass: top** и определение всех **объектных классов** в иерархии. Начиная с OpenLDAP 2.0, [это не является обязательным](#). Принимайте решение, будете ли Вы это делать, исходя из требований Вашей системы.
11. Пробел, следующий за : в каждой строке, ЖИЗНЕННО НЕОБХОДИМ.

Большинство систем каталогов могут быть **построены** с помощью приведённого выше набора директив LDIF.



8.2.2 Директивы LDIF

Далее следует описание директив LDIF, приводимых в алфавитном порядке.

8.2.2.1 Директива add

Формат:

```
add: attributename
```

Директива **add** следует за директивой **changetype: modify** и определяет имена атрибута (атрибутов), которые будут добавлены в существующую запись. Можно добавить несколько атрибутов с одним и тем же именем атрибута.

Примечания:

1. О том, как **добавить** к существующей записи вспомогательный (AUXILIARY) объектный класс, смотрите в примерах к **changetype: modify**.

Примеры:

```
# добавляем один атрибут
# текущие значения атрибута не меняются

dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: 123-111

# добавляем несколько атрибутов
# текущие значения атрибута не меняются

dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: 555-123-1111
telephonenumber: 111
```



8.2.2.2 Директива attributename

Формат:

```
attributename: value
```

Директива **attributename** позволяет задать значение атрибуту.

Примечания:

1. Если **атрибут** поддерживает несколько значений, любое количество директив **attributename** может быть добавлено в LDIF-файл.
2. Директива **attributename** может быть использована с **changetype add** или **replace**
3. В качестве **value** может быть:
 1. Строка
 2. Строка **Base 64**.
 3. URL файла, тогда значение будет получено из файла **< file://path/to/file**.

Примеры:

```
# добавляем атрибуты к новой записи
dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: add
objectclass: inetorgperson
cn: Robert Smith
cn: Robert J Smith
cn: Bob Smith
telephonenumber: 123-111
...

# добавляем атрибут с несколькими значениями к существующей записи
dn: cn=Robert Ssmith,ou=people,dc=example,dc=com
changetype: modify
add: telephonenumber
telephonenumber: 555-123-1111
telephonenumber: 111

# использование URL файла
dn: cn=Robert Ssmith,ou=people,dc=example,dc=com
changetype: modify
add: jpegphoto
# между : и < не должно быть пробела
jpegphoto:< file://tmp/my.jpg
```



8.2.2.3 Директива changetype

Формат:

```
changetype: type
```

Директива **changetype** следует сразу за директивой **dn:** и определяет операцию, которая должна быть выполнена над [записью](#).

type может принимать одно из следующих значений:

1. При использовании типа **add** последующие директивы будут использоваться для добавления (создания) записи. Если запись уже существует, то должна использоваться форма **changetype: modify**. Если в LDIF отсутствует директива **changetype** и он обрабатывается с помощью [ldapadd](#), то по умолчанию подразумевается **changetype: add**.

Примеры:

```
dn: cn=Robert Smith,ou=people,dc=example,dc=com
# добавить запись указанного в предыдущей строке dn
changetype: add
objectclass: inetorgperson
cn: Robert Smith
...
```

2. При использовании типа **delete** запись, указанная в предшествующей директиве **dn**, будет удалена.

```
dn: cn=Robert Smith,ou=people,dc=example,dc=com
# удалить запись, указанную dn в предыдущей строке
changetype: delete
```

3. При использовании типа **modify** последующие команды будут изменять существующую запись, указанную в предшествующей директиве **dn**. Следующие за **modify** атрибуты (или объектные классы) могут быть **добавлены**, **заменены** или **удалены**.

Примечания:

1. Несколько операций **modify** могут быть собраны в последовательность ОПЕРАТОРА с помощью **строка-РАЗДЕЛИТЕЛЕЙ**.
2. Для добавления нового вспомогательного (AUXILIARY) объектного класса к существующей записи перед добавлением нового объектного класса и последующих атрибутов требуется указать существующий объектный класс (вспомогательные объектные классы присоединяются к существующему объектному классу). Смотрите примеры:

Примеры:

```
dn: cn=Robert Smith,ou=people,dc=example,dc=com
# модифицировать запись, указанную dn в предыдущей строке
changetype: modify
# single operation
add: telephonenumber
telephonenumber: 555

dn: cn=Robert Smith,ou=people,dc=example,dc=com
# модифицировать запись, указанную dn в предыдущей строке
changetype: modify
# несколько операций
add: telephonenumber
telephonenumber: 555
# за которыми следует строка-РАЗДЕЛИТЕЛЬ
-
replace: mail
mail: bob.smith@example.com
-
delete: secretary

# чтобы ДОБАВИТЬ новый вспомогательный объектный класс
# к существующей записи

dn: cn=Robert Smith,ou=people,dc=example,dc=com
# модифицировать запись, указанную dn в предыдущей строке
changetype: modify
# данный объектный класс используется для присоединения
objectclass: inetorgperson
# к нему будет добавлен вспомогательный объектный класс
objectclass: posixaccount
# далее включаются обязательные атрибуты
# нового объектного класса
uidnumber: 200
gidnumber: 207
homedirectory: /home/rsmith
...
```

4. **modrdn** (и его псевдоним **moddn**) используется для изменения **RDN** записи (переименования или копирования записи), указанной в предшествующей директиве **dn:.** За ней ДОЛЖНА следовать директива **newrdn:**, а также могут следовать директивы **deleteoldrdn** и **newsuperior**.

Вы НЕ СМОЖЕТЕ переименовать запись, если у неё есть одна или несколько дочерних записей.

Пример:

```
dn: cn=Robert Ssmith,ou=people,dc=example,dc=com
# следующая последовательность переименовывает указанный в предыдущей
# строке DN в
# cn=Robert Smith,ou=people,dc=example,dc=com
# и удаляет запись
# cn=Robert Ssmith,ou=people,dc=example,dc=com
changetype: modrdn
newrdn: cn=Robert Smith
deleteoldrdn: 1
```



8.2.2.4 Директива control

Уже совсем скоро (One day Real Soon Now™)



8.2.2.5 Директива delete

Формат:

```
delete: attributename
```

Директива **delete** следует за директивой **changetype: modify**. Она выполняет операции с атрибутами и определяет список атрибутов для удаления. Чтобы удалить запись, используйте **changetype: delete**.

За директивой **delete** МОЖЕТ следовать одна или несколько директив, определяющих, какие атрибуты удалять. Если за ней не следуют такие директивы (то есть за ней следует конец файла (EOF), **ПУСТАЯ строка** или **строка-РАЗДЕЛИТЕЛЬ**, то будут удалены все атрибуты, указанные в **attributename**.

```
# удаляем одно значение атрибута

dn: cn=Robert Ssmith,ou=people,dc=example,dc=com
changetype: modify
# удаляются только атрибуты, значения которых 123-111 и 111
# все остальные атрибуты telephonenumber не меняются
delete: telephonenumber
telephonenumber: 123-111
telephonenumber: 111

# удаляем атрибут с несколькими значениями

dn: cn=Robert Ssmith,ou=people,dc=example,dc=com
changetype: modify
# удаляются все атрибуты telephonenumber
```

```
delete: telephonenumber
```



8.2.2.6 Директива deleteoldrdn

Формат:

```
deleteoldrdn: action
```

Директива **deleteoldrdn** следует за директивой **changetype: modrdn** и определяет действие, которое будет произведено с оригинальным DN. В качестве **action** данная директива принимает **0** (ложь), в этом случае оригинальная запись сохраняется, или **1**, в этом случае оригинальная запись удаляется.

Пример:

```
# устраняет ошибки в DN записи и
# удаляет текущую запись

dn: cn=Robert Ssmith,ou=people,dc=example,dc=com
changetype: modrdn
# создаётся новое имя (RDN)
newrdn: cn=Robert Smith
# удаляется текущая запись (Robert Ssmith)
deleteoldrdn: 1
```



8.2.2.7 Директива dn

Формат:

```
dn: DN
```

Директива **dn** определяет [Уникальное имя \(Distinguished Name, DN\)](#) и используется для указания расположения (или адреса) [записи](#), по которому будут выполняться последующие директивы ([последовательность ЗАПИСИ](#)).

Если это не первая запись в LDIF-файле, то директива **dn** ДОЛЖНА предваряться [ПУСТОЙ строкой](#).

Пример:

```
...
# предыдущая последовательность, затем ПУСТАЯ строка
dn: ou=people,dc=example,dc=com
...
```



8.2.2.8 Директива newrdn

Формат:

```
newrdn: RDN
```

Директива **newrdn** следует за директивой **changetype: modrdn** (или **moddn**) и создаёт копию записи, указанной в предшествующей директиве **dn**, используя [RDN](#), указанный в данной (то есть **newrdn**) директиве.

Обычно за этой директивой следует директива [deleteoldrdn](#).

Совместно с директивой [newsuperior](#) может использоваться для копирования или перемещения записи в DIT.

Примеры:

```
# используем при исправлении ошибок
dn: cn=Robert Ssmith,ou=people,dc=example,dc=com
changetype: modrdn
# корректируется запись
newrdn: cn=Robert Smith
# удаляется старая запись
deleteoldrdn: 1
```



8.2.2.9 Директива [newsuperior](#)

Формат:

```
newsuperior: DN
```

Директива **newsuperior** позволяет перемещать запись в DIT. Данная директива указывает LDAP переместить текущую запись в указанный данной директивой [DN](#).

Данная директива используется совместно с директивами [changetype: modrdn](#), [newrdn](#) и [deleteoldrdn](#).

Примеры:

```
# перемещение записи из people в expeople
dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: modrdn
# rdn не меняется
newrdn: cn=Robert Smith
# старая запись удаляется
deleteoldrdn: 1
# добавляется в иерархию expeople
newsuperior: ou=expeople,dc=example,dc=com

# копирование записи в expeople
dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: modrdn
# rdn не меняется
newrdn: cn=Robert Smith
# текущая запись сохраняется
deleteoldrdn: 0
# добавляется в иерархию expeople
newsuperior: ou=expeople,dc=example,dc=com
```



8.2.2.10 Директива [objectclass](#)

Формат:

```
objectclass: objectclassname
```

Директива **objectclass** позволяет добавить к записи объектный класс.

Примечания:

1. В записи должен быть хотя бы один [структурный \(STRUCTURAL\)](#) объектный класс.
2. Начиная с OpenLDAP 2.0+, нет необходимости определять все объектные классы в [иерархии объектных классов](#), достаточно определить объектный класс самого нижнего уровня в иерархии.

Примеры:

```
# добавление объектного класса к новой записи

dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: add
# inetorgperson - самый нижний уровень в иерархии
objectclass: inetorgperson
cn: Robert smith
cn: Robert J Smith
cn: Bob Smith
telephonenumber: 123-111
...

# добавление объектных классов к новой записи

dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: add
# inetorgperson - самый нижний уровень в иерархии
objectclass: inetorgperson
# здесь должны присутствовать оба объектных класса
# поскольку posixaccount - вспомогательный объектный класс
objectclass: posixaccount
cn: Robert smith
cn: Robert J Smith
cn: Bob Smith
telephonenumber: 123-111
...

# добавление нового вспомогательного объектного класса
# к существующей записи

dn: cn=Robert Smith,ou=people,dc=example,dc=com
# модифицировать запись, указанную dn в предыдущей строке
changetype: modify
# данный объектный класс используется для присоединения
objectclass: inetorgperson
# к нему будет добавлен вспомогательный объектный класс
objectclass: posixaccount
# далее включаются обязательные атрибуты
# нового объектного класса
uidnumber: 200
gidnumber: 207
homedirectory: /home/rsmith
...
```



8.2.2.11 Директива replace

Формат:

```
replace: attributename
```

Директива **replace** следует за директивой **changetype: modify** и определяет название атрибута, который будет заменён.

Если у атрибута несколько значений, то ВСЕ текущие значения будут заменены на один или несколько следующих за данной директивой атрибутов.

Если нужно заменить только одно значение атрибута, имеющего несколько значений, используйте **delete**, а затем **add**. Смотрите пример ниже.

Примеры:

```
# заменить одно значение атрибута

dn: cn=Robert Ssmith,ou=people,dc=example,dc=com
```

```
changetype: modify
replace: uid
uid: bill

# заменить атрибут с несколькими значениями

dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: modify
# заменяем все атрибуты telephonenumber
# на 555-111-1212
replace: telephonenumber
telephonenumber: 555-111-1212

# заменить одно значение у атрибута с несколькими значениями
# удаляем, затем добавляем
# в примере заменяется 555-111-1212 на 555

dn: cn=Robert Smith,ou=people,dc=example,dc=com
changetype: modify
# сначала удаляем требуемый атрибут
delete: telephonenumber
telephonenumber: 555-111-1212
# строка-РАЗДЕЛИТЕЛЬ (необходимо)
-
# добавляем новое значение
add: telephonenumber
telephonenumber: 555
```



8.2.2.12 Директива version

Формат:

```
version: number
```

Директива **version** определяет версию формата директив файла LDIF. Эта директива не является обязательной и, из-за несоответствия реализаций, мы подразумеваем, что она не используется.

Данная директива, если она присутствует, должна быть первой директивой в файле (некоторые реализации воспринимают это буквально и отклоняют файл, если строке version предшествует строка комментария). В настоящее время принимается номер версии, равный только 1.

```
# должно быть первой записью в LDIF-файле
version: 1
```



8.3 Обработка двоичных данных (включая пароли) в LDIF



8.4 Импорт файлов LDIF



8.5 Примеры LDIF



Глава 8. DSML

Уже совсем скоро (One day real soon now ™)



Глава 9. Функциональная модель LDAP

В этом разделе описываются команды протокола, которые клиенты и серверы LDAP используют "за кадром" при соединении друг с другом, и формат LDAP URL, который может использоваться при работе с современными браузерами, а также в сочетании с определёнными объектными классами и функциями (в первую очередь с [динамическими группами](#)).

Если Вам действительно нужно разобраться в этих вещах, неочевидный, но по-настоящему умный способ - использовать превосходный Open Source продукт [wireshark](#) (бывший ethereal), который выполнит для Вас полную разборку трафика - мы еще не сталкивались с таким протоколом, который не поддерживался бы **wireshark**. Замечательная программа.

Наконец, LDAP использует спецификацию ITU BER (Basic Encoding Rules), чтобы сэкономить по три байта при каждом трансфере. Вам понадобится раздобыть (за приличные деньги) стандарт X.690, в котором описан BER.

В качестве альтернативы Вы можете скачать книгу профессора John Larmouth "[ASN.1 Complete](#)" (распространяется свободно, но для её получения необходима регистрация), либо приобрести её бумажный экземпляр от издательства [Morgan Kaufmann Publishers](#) (ISBN: 0-12-233435-3). Можно также посмотреть книгу другого всемирно известного эксперта по ASN.1 Olivier Dubuisson "[ASN.1 - Communication between heterogeneous systems](#)" (распространяется свободно, но для её получения необходима регистрация). Опять же, можно приобрести бумажный экземпляр этой книги от [Morgan Kaufmann Publishers](#) (ISBN: 0-12-6333361-0). Кроме того, замечательную подборку ресурсов по ASN.1 и BER можно найти на [этом сайте](#).

Содержание

- [9.1 Обзор функциональной модели](#)
- [9.2 Общий формат](#)
- [9.3 Примитивы \(операции\)](#)
- [9.4 LDAP URL](#)

9.1 Обзор функциональной модели

Уже совсем скоро (One day real soon now ™).



9.2 Общий формат

Уже совсем скоро (One day real soon now ™).



9.3 Примитивы (операции)

Уже совсем скоро (One day real soon now ™).

9.4 LDAP URL

LDAP URL - вещь полезная, но с элементом излишества. LDAP URL ([RFC 4510](#) и [RFC 4516](#)) определяет метод, посредством которого Вы можете ввести что-то похожее на URL в адресную строку некоторых браузеров (MSIE 5.5+ и любые основанные на [Gecko](#) браузеры поддерживают ldap, Opera 7.x beta и Konqueror - нет) и те выполняют LDAP-запрос только на чтение к LDAP-серверу, используя параметры, указанные в данном URL. И MSIE, и Gecko позволят Вам добавить любые найденные записи в адресную книгу (создаётся впечатление, что эта функция реализована с помощью одного и того же программного кода).

Элемент излишества (мы, конечно, сильно придирчивы) - то, что браузер транслирует такой запрос в стандартный [поисковый примитив](#) LDAP. URL-нотация - всего лишь (полезный) интерфейс для работы через браузер. Однако, LDAP URL используются также для кое-чего более серьёзного, например, [динамических групп](#) (в сочетании с объектным классом [groupOfURLs](#) и его атрибутом [memberURL](#)). Динамические группы - нестандартная (нет соответствующего RFC), но широко реализованная функция LDAP.

Формат LDAP URL

Синтаксис формата таков:

```
scheme "://" [host:port] ["/"[dn ["?" [attributes] ["?" [scope] ["?" [filter]
["?" extensions]]]]]]
```

Здесь:

Параметр	Описание
scheme	Схема. Может быть либо ldap - обычный доступ на стандартном порту 389 (либо нестандартном), либо ldaps - SSL-доступ на стандартном порту 636 (либо нестандартном).
host:port	Хост и порт. Необязательный параметр - если он пропущен, браузер использует значения по умолчанию. В противном случае задаёт URL хоста и, опционально, номер порта на данном хосте, например, ldapserver.example.com или ldapserver.example.com:8777 (используется нестандартный порт 8777).
dn	Базовый DN поиска. Необязательный параметр - если он пропущен, браузер использует значение по умолчанию. В противном случае задаёт DN , который будет использоваться для поиска, например, ou=people,dc=example,dc=com.
attributes	Атрибуты. Необязательный параметр - если он пропущен, будут возвращены все доступные атрибуты для найденной записи. В противном случае задаёт те атрибуты, которые необходимо вернуть, в виде списка, разделённого запятыми, например, cn,mail,telephonenumber.
scope	Диапазон поиска. Необязательный параметр - если он пропущен, подразумевается base . В противном случае задаёт диапазон поиска и принимает одно из следующих значений: base поиск производится только на уровне, заданном в параметре dn. one поиск производится на уровне, заданном в параметре dn и на один уровень ниже по иерархии. sub поиск производится на уровне, заданном в параметре dn и на всех остальных уровнях ниже по иерархии до самого конца дерева (DIT).
filter	Поисковый фильтр. Согласно документации это необязательное поле - если оно пропущено, подразумевается (objectclass=*). Судя по всему, это значение по умолчанию не поддерживается ни MSIE, ни Gecko - Вы должны

	указать что-нибудь, например, (objectclass=*). В противном случае задаёт текстовое представление поискового фильтра .
extensions	Расширения. Текущее RFC по LDAP URL (RFC 4516) не определяет никаких расширений.

Примеры LDAP URL

Подключение с использованием анонимного доступа к host.example.com на порт 389 с базовым DN поиска ou=people,dc=example,dc=com. Возвращаются все доступные атрибуты для найденных записей. Поиск осуществляется по базовому DN и на один уровень ниже. Возвращаются все найденные записи.

```
ldap://host.example.com/ou=people,dc=example,dc=com??one?(objectclass=*)
```

Подключение с использованием анонимного доступа к host.example.com на порт 9000 с базовым DN поиска ou=people,dc=example,dc=com. Возвращаются все доступные атрибуты для найденных записей. Поиск осуществляется по базовому DN и на один уровень ниже. Возвращаются все найденные записи.

```
ldap://host.example.com:9000/ou=people,dc=example,dc=com??one?(objectclass=*)
```

Подключение с использованием анонимного доступа к хосту по выбору браузера (мы так и не смогли сообразить, как это можно настроить) на порт 389 с базовым DN поиска ou=people,dc=example,dc=com. Возвращаются все доступные атрибуты для найденных записей. Поиск осуществляется по базовому DN и на один уровень ниже. Возвращаются все записи с s или S в любом месте атрибута commonName:

```
ldap:///ou=people,dc=example,dc=com??one?(cn=*s*)
```

Подключение с использованием анонимного доступа к host.example.com на порт 389 с базовым DN поиска ou=people,dc=example,dc=com. Возвращаются только атрибуты mail. Поиск осуществляется по базовому DN и на один уровень ниже. Возвращаются все записи, в которых есть один или несколько атрибутов mail:

```
ldap://host.example.com/ou=people,dc=example,dc=com?mail?one?(mail=*)
```

Подключение с использованием анонимного доступа к host.example.com на порт 389 с базовым DN поиска ou=people,dc=example,dc=com. Возвращаются все доступные атрибуты для найденных записей. Поиск осуществляется по базовому DN и по всему дереву ниже базового DN. Возвращаются все записи, атрибут sn которых начинается на a или A.

```
ldap://host.example.com/ou=people,dc=example,dc=com???(sn=a*)
```

Подключение с использованием анонимного доступа к ldap на локальном хосте (localhost) на порт 389 с базовым DN поиска ou=people,dc=example,dc=com. Возвращаются все доступные атрибуты для найденных записей. Поиск осуществляется по базовому DN и по всему дереву ниже базового DN. Возвращаются все записи, атрибут sn которых начинается на a или A.

```
ldap:///ou=people,dc=example,dc=com???(sn=a*)
```

Глава 10. LDAP API

Уже совсем скоро (One day real soon now TM).



Глава 11. LDAP HOWTO

Представленные ниже HOWTO демонстрируют некоторые необходимые или широко используемые функции в OpenLDAP:

[Конфигурирование нескольких DIT в OpenLDAP](#)

[Настройка отсылок в OpenLDAP](#)

[Настройка сцепления при отсылках в OpenLDAP](#)

[Настройка репликации в стиле slurpd в OpenLDAP](#)

[Настройка репликации в стиле syncrepl в OpenLDAP](#)

[Настройка delta-синхронизации \(syncrepl\) в OpenLDAP](#)

[Настройка разнонаправленной репликации с несколькими главными серверами \(N-Way Multi-master\) в OpenLDAP](#)

[Настройка и использование cn=config в OpenLDAP](#)

[LDAP URL](#)

[Конфигурирование групп пользователей](#)

[Конфигурирование динамических групп пользователей в OpenLDAP](#)

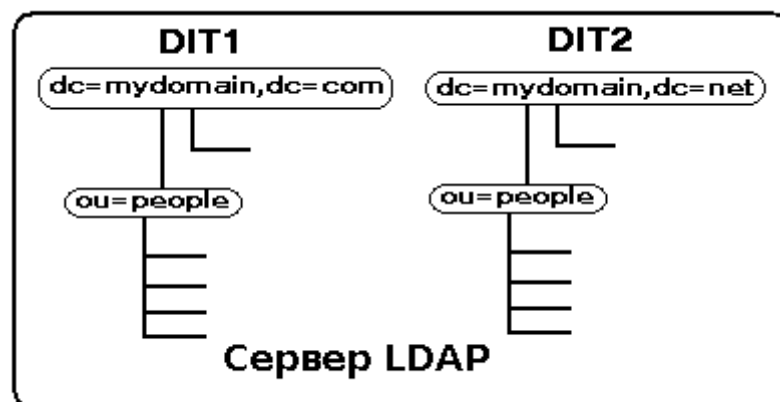
[Конфигурирование TLS в OpenLDAP](#)

[Настройка защищённого TLS сервера в OpenLDAP](#)



Конфигурирование нескольких DIT в OpenLDAP

Каждое DIT описывается в разделе [database](#) файла slapd.conf. При определении нескольких разделов **database** на одном сервере каталогов будет определено несколько DIT. Каждое DIT является отдельным и имеет свой собственный [контекст именования](#) (или пространство имён). Предположим, мы хотим создать на одном LDAP-сервере следующую структуру:



slapd.conf

```
#
##### НЕСКОЛЬКО DIT #####
#
# Примечание: inetorgperson использует атрибуты и
# объектные классы из всех трёх наборов схемы
include /usr/local/etc/openldap/schema/core.schema
include /usr/local/etc/openldap/schema/cosine.schema
include /usr/local/etc/openldap/schema/inetorgperson.schema

# НЕТ СИСТЕМЫ БЕЗОПАСНОСТИ - условия безопасности пропущены
# по умолчанию анонимный доступ на чтение
# только rootdn может осуществлять запись

# НЕТ ОТСЫЛОК

# НЕ заботимся о файле ARGS, пока не обретём уверенности в своих силах
# stop-скрипту slapd следующая строка необходима для работы:
pidfile /var/run/slapd.pid

# Включаем большое количество вывода в журнал - нам это может понадобится
```



```

# но при этом генерируются журналы огромных размеров
loglevel -1

# НЕТ динамических модулей для механизмов манипуляции данными

# НЕТ соединений TLS

# Определение backend не требуется

#####
# Определение ПЕРВОЙ базы данных bdb
# для EXAMPLE.COM
# Замените ниже example и com на подходящее имя домена
#
# Если у Вас нет домена, то, поскольку example.com зарезервирован для
# экспериментов, можете оставить его, либо поменять на my.inc
#
#####

database bdb
suffix "dc=example, dc=com"

# root или администратор
rootdn "cn=jimbo, dc=example, dc=com"
rootpw dirtysecret
# Директория базы данных ДОЛЖНА существовать до того, как Вы запустите slapd
# Не забудьте поменять путь на подходящий Вам
directory /var/db/openldap/example-com

# Устанавливаемые для каталога индексы
# Только полное соответствие для unique id
index uid eq
# Стандартный поиск для commonname, givenname и email
index cn,gn,mail eq,sub
# Несколько вариантов для поиска surname
index sn eq,sub,subinitial,subany,subfinal
# Оптимизируем поиск по подразделениям
index ou eq
# Продемонстрировано использование параметра индексирования default
index default eq,sub
# Настройка индексирования пропущена, по умолчанию используется eq,sub
index telephoneNumber

# Другие параметры базы данных
# Дополнительная информация в разделе справочника по slapd.conf
cachesize 10000
checkpoint 128 15
dbnosync
dirtyread
searchstack 5

#####
# Определение ВТОРОЙ базы данных bdb
# для EXAMPLE.NET
# Замените ниже example и net на подходящее имя домена
#
# Если у Вас нет домена, то, поскольку example.net зарезервирован для
# экспериментов, можете оставить его, либо поменять на my.inc
#
#####

database bdb
suffix "dc=example, dc=net"

# root или администратор
rootdn "cn=jimbo, dc=example, dc=net"
rootpw dirtysecret
# Директория базы данных ДОЛЖНА существовать до того, как Вы запустите slapd
# Не забудьте поменять путь на подходящий Вам. Эта директория должна быть
# отдельной от той, которая задана для первого раздела database
directory /var/db/openldap/example-net

# Устанавливаемые для каталога индексы
# Только полное соответствие для unique id
index uid eq
# Оптимизируем поиск по подразделениям
index ou eq
# Продемонстрировано использование параметра индексирования default
index default eq,sub
# Настройка индексирования пропущена, по умолчанию используется eq,sub
index telephoneNumber

# Другие параметры базы данных
# Дополнительная информация в разделе справочника по slapd.conf
cachesize 10000
checkpoint 128 15
dbnosync

```

Очевидно, Вам придётся остановить и снова запустить сервер LDAP, чтобы этот новый файл был принят. Затем добавьте записи из показанного ниже LDIF с помощью [ldapadd](#).

LDIF

В этом LDIF подразумевается, что EXAMPLE.COM уже существует и нам нужно просто добавить EXAMPLE.NET.

```
# добавление example.net на существующий LDAP-сервер
version: 1

dn: dc=example,dc=net
dc: example
description: Example Network Operations
objectClass: dcObject
objectClass: organization
o: Example, Inc.

dn: ou=people, dc=example,dc=net
ou: people
description: All people in organisation
objectClass: organizationalUnit
```

В этом LDIF подразумевается, что мы добавляем сразу EXAMPLE.COM и EXAMPLE.NET.

```
# добавление example.com и example.net
# ПЕРВЫМ добавляем example.com
version: 1

dn: dc=example,dc=com
dc: example
description: Example Company
objectClass: dcObject
objectClass: organization
o: Example, Inc.

dn: ou=people, dc=example,dc=com
ou: people
description: All people in organisation
objectClass: organizationalUnit

# ВТОРЫМ добавляем example.net
version: 1

dn: dc=example,dc=net
dc: example
description: Example Network Operations
objectClass: dcObject
objectClass: organization
o: Example, Inc.

dn: ou=people, dc=example,dc=net
ou: people
description: All people in organisation
objectClass: organizationalUnit
```



Конфигурирование динамических групп

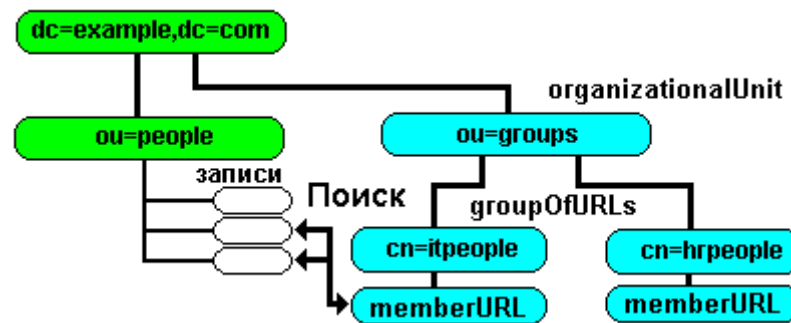
Группы - быстрый путь предоставления пользователям прав доступа к определённым возможностям или функциональности в каталогах LDAP. Для этих целей в директиве [access to](#) (атрибуте `olcAccess` в `cn=config`) есть специфичный для групп вариант условия `<who>`. Реальные права, которые будут предоставлены группе, также определяются в директиве **access to** (атрибуте `olcAccess` в `cn=config`). Группы могут настраиваться статически с использованием объектного класса [groupOfNames](#). Каждый член такой группы должен быть определён индивидуально, а затем должна осуществляться его поддержка при изменении статуса его членства в группе. В LDAP есть функция динамических групп, когда группа формируется динамически на основании результатов поискового запроса LDAP. Такие группы могут быть чрезвычайно эффективны, когда в группе очень много членов, либо они постоянно изменяются. Функция

динамических групп, как насущная потребность, реализована многими производителями служб каталогов, однако до сих пор не стандартизована (нет соответствующего RFC).

Обычно динамические группы используют объектный класс `groupOfURLs`.

Пример:

Приведённые ниже фрагменты LDIF демонстрируют построение группы **itpeople**, которой будут предоставлены привилегии на доступ и изменение паролей или параметров конфигурации в записях пользователей. Члены группы будут заполняться автоматически путём поискового запроса, основанного на URL. В данной конфигурации создаётся отдельная ветка **groups**, в которой будет располагаться группа **itpeople**. Такая организация каталога показана на диаграмме:



В OpenLDAP динамические группы реализованы с помощью наложения `dynlist` (директива `overlay dynlist`). Описание этого наложения и формат параметров смотрите [здесь](#). Настройка директив данного наложения показана в следующем фрагменте `slapd.conf`:

```
# фрагмент slapd.conf по настройке dynlist
...
# требуется подключение набора схемы dyngroup
include /etc/openldap/schema/dyngroup.schema
...
# пример загрузки модуля для наложения dynlist
# при использовании неабсолютного формата пути
# к модулю требуется директива modulepath
modulepath /usr/libexec/openldap
loadmodule dynlist.la

# абсолютный формат пути
# не требуется директива modulepath
loadmodule /usr/libexec/openldap/dynlist.la
...
database bdb
...
# При использовании после директивы database область
# применения ограничивается данным разделом database.
# Может также определяться до директив database -
# область применения будет глобальной
overlay dynlist
dynlist-attrset groupOfURLs memberURL
...
```

В этом определении указано, что если при выполнении поискового запроса сервер встречает запись с объектным классом `groupOfURLs`, он будет выполнять ещё один поисковый запрос по URL, указанному в атрибуте **memberURL** этой записи, и результат этого поискового запроса в виде множества атрибутов будет возвращён в эту же запись (полный список параметров наложения `dynlist` с пояснениями смотрите [здесь](#)).

```
# фрагмент LDIF для создания ветки group в корне DIT
dn: ou=groups,dc=example,dc=com
objectclass: organizationalunit
ou: groups
description: generic groups branch

# создание записи группы itpeople
dn: cn=itpeople,ou=groups,dc=example,dc=com
objectclass: groupOfURLs
```

```
cn: itpeople
memberURL: ldap:///ou=people,dc=example,dc=com?cn,sn?one?(ou=it*)
...
```

В данном случае указанный [поисковый LDAP URL](#) создаст поисковый запрос на локальном сервере (это следует из синтаксиса `///`) с базой поиска `ou=people,dc=example,dc=com` только по записям на один уровень ниже (**one**), возвращающий атрибуты `cn` и `sn` любых записей, в которых атрибут `ou` (`organizationalUnitName`) начинается с "it", либо содержит только "it" (поиск без учёта регистра символов). Члены данной группы (которые будут возвращены в результате поиска) могут затем использоваться для получения каких-либо привилегий при помощи директив [access to](#) (или атрибутов `olcAccess` в случае `cn=config`).

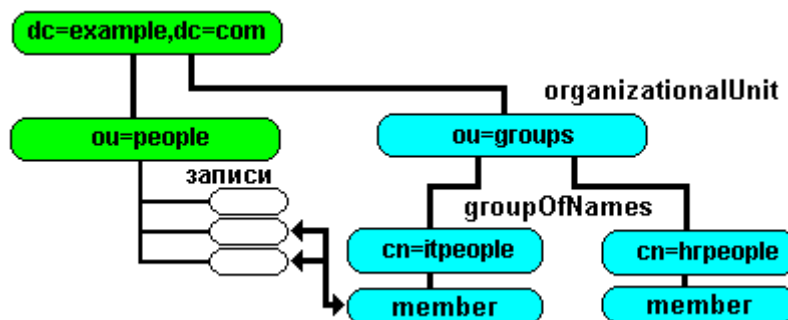
Конфигурирование групп

Группы - быстрый путь предоставления пользователям прав доступа к определённым возможностям или функциональности в каталогах LDAP. Для этих целей в директиве [access to](#) (атрибуте `olcAccess` в `cn=config`) есть специфичный для групп вариант условия `<who>`. Реальные права, которые будут предоставлены группе, также определяются в директиве **access to** (атрибуте `olcAccess` в `cn=config`). В записях групп используется объектный класс [groupOfNames](#).

Примечание: Группы, определённые с помощью объектного класса **groupOfNames** по существу являются статическими и требуют выполнения явных действий для определения каждого члена группы. Когда в группе очень много членов, либо они постоянно изменяются, управление такой группой превращается в серьёзную головную боль. LDAP также предоставляет нестандартную (нет соответствующего RFC), но широко реализованную функцию так называемых [динамических групп](#), использующих объектный класс [groupOfURLs](#).

Пример:

Приведённые ниже фрагменты LDIF демонстрируют построение группы **itpeople**, которой будут предоставлены привилегии на доступ и изменение паролей или параметров конфигурации в записях пользователей. Подразумевается, что индивидуальные записи членов группы уже существуют в каталоге в классической ветке **ou=people**. В данной конфигурации создаётся отдельная ветка **groups**, в которой будет располагаться группа **itpeople**. Такая организация каталога показана на диаграмме:



```
# фрагмент LDIF для создания ветки group в корне DIT
```

```
dn: ou=groups,dc=example,dc=com
objectclass: organizationalunit
ou: groups
description: generic groups branch
```

```
# создание записи группы itpeople
```

```
dn: cn=itpeople,ou=groups,dc=example,dc=com
objectclass: groupofnames
cn: itpeople
description: IT security group
```

```
# добавление членов группы, подразумевается,
# что все они существуют и находятся в ветке people
member: cn=road runner,ou=people,dc=example,dc=com
member: cn=micky mouse,ou=people,dc=example,dc=com
```

...

Таким образом, DN `cn=itpeople,ou=groups,dc=example,dc=com` будет ссылаться на всех **членов (member)** этой группы. Любому человеку, прошедшему аутентификацию как 'micky mouse', могут быть предоставлены права **itpeople** с помощью соответствующей директивы **access to** (атрибута `olcAccess` в `cn=config`). Смотрите рабочий пример использования групп [здесь](#).

Наложения в OpenLDAP, или как заставить это работать

Наложения и загрузка модулей.

Наложения могут быть либо динамическими, либо статическими. Если наложение динамическое, для него требуется загрузить подгружаемый модуль с помощью директивы **moduleload** (атрибута `olcModuleLoad` в `cn=config`). Если наложение статическое, оно встраивается в `slapd`, и потому определение директивы `moduleload` для него не требуется, и действительно, при попытке загрузки такого модуля произойдёт ошибка, поскольку самого файла подгружаемого модуля просто не существует.

Хорошо. А как же узнать какое это наложение: статическое или динамическое?

Вам необходимо выяснить, каким образом был собран `slapd` - в частности, какие директивы скрипта `configure` использовались. Чтобы получить возможность собрать какое-либо наложение, при запуске скрипта `configure` должна быть указана директива `--enable-modules`, что для большинства сборок, по крайней мере известных нам, является значением по умолчанию. Для каждого наложения есть отдельная директива скрипта `configure`, которую можно задать так:

```
# для иллюстрации в этом примере используется наложение accesslog,
# всё то же самое верно и для syncprov и остальных наложений.
--enable-accesslog
# собирается статически - не требуется moduleload
--enable-access=mod
# собирается динамически - требуется moduleload
```

Чтобы определить, как был собран Ваш `slapd`:

```
# для FreeBSD подразумевается сборка из портов:
cd /usr/ports/net/openldap2X-server/work/openldap-X.X.X
vi config.log
# в начале этого файла перечислены все условия,
# задаваемые при запуске скрипта configure

# если Вы использовали 'make install clean', данный файл будет удалён
# чтобы создать его заново, используйте 'make configure'
# а для повторного удаления - 'make clean'

# для дистрибутивов, основанных на RPM
# найдите файл .spec, использовавшийся для построения slapd
# откройте файл и найдите вызов скрипта configure,
# за которым следуют соответствующие директивы
```

Возможно, самый простой путь определения того, является ли наложение динамическим - просто поискать файлы подгружаемых модулей для наложения. Если такие файлы существуют, нужно использовать директиву `moduleload`, в противном случае - нет. Файлы подгружаемых модулей обычно располагаются в `[bsd] /usr/local/libexec/openldap` и `[fc] /usr/libexec/openldap` or `/usr/sbin/openldap`.

Наконец, на большинстве платформ наложения собираются в виде файлов с суффиксами `.la` (библиотека), либо `.la` и `.so` (библиотека разделяемых объектов). Попробуйте сначала использовать `.la`, если загрузка завершится неудачей, используйте `.so`.



Запуск OpenLDAP

Некоторые замечания по запуску различных версий OpenLDAP.

Версия 2.4

В этой статье описан некоторый опыт использования версии 2.4.7 на FreeBSD (5.x и 6.x). Хотя места расположения файлов могут отличаться, основные принципы остаются аналогичными и для Linux. Далее приводится список действий, которые необходимо выполнить для запуска OpenLDAP. В текст также включены описания некоторых средств диагностики и процедур.

1. Ниже приведён список того, что Вы должны сделать, прежде чем запустить OpenLDAP. Большую часть этих действий придётся делать вручную. Некоторые из них очевидны, некоторые нет:
 1. Настройте систему журналирования (/etc/syslog.conf) для OpenLDAP (смотрите примечание в описании директивы [loglevel](#)).
 2. Переместите в соответствующее место ([fc] /etc/openldap [bsd] /usr/local/etc/openldap) или создайте там файл slapd.conf и настройте его должным образом.
 3. Убедитесь, что все директории и файлы (расположение pid-файла, каталог базы данных) существуют и имеют соответствующие разрешения. Скопируйте файл DB_CONFIG.example в директорию размещения DIT (указанный в директиве [directory](#)) и переименуйте его в DB_CONFIG - инструкций, содержащихся в нём, вполне достаточно для начала.
 4. Все последние версии OpenLDAP, следуя существующей практике, используют скрипты, запускающие его от имени пользователя и группы, отличной от root (обычно ldap). Возможно, придётся вручную дать этому пользователю и группе права на все нестандартные места расположения рабочих файлов. Проблем с правами доступа можно избежать, запустив приложение с правами root, но это не лучшая идея. Вместо этого используйте ключ -d для выявления проблем с правами доступа и устранили их.
 5. Особенность FreeBSD: сценарий запуска считает, что PID-файл находится в /var/run/openldap/slapd.pid. Этот сценарий не считывает информацию из файла slapd.conf, так что если Вы используете нестандартное размещение, отредактируйте сценарий или измените директиву [pidfile](#) файла slapd.conf, указав в ней стандартное расположение. В противном случае сценарий не будет работать, поскольку будет пытаться обращаться к неверному расположению pid.
 6. Если OpenLDAP версии 2.4.7 запускается с пустой базой данных, может возникнуть ошибка инициализации BDB при выполнении slaptest для проверки slapd.conf:

```
slaptest
# ошибка по причине отсутствия id2entry.bdb

# попытайтесь проверить только slapd.conf
slaptest -u
# запрещается проверка базы данных

# затем попробуйте нормально загрузить LDAP от имени root
# 1. убедитесь в том, что директория базы данных существует
# 2. убедитесь, что в этой директории находится DB_CONFIG
slapd -d -1

# приведите в порядок права доступа для пользователя ldap и перезапустите
slapd -d -1 -u ldap -g ldap

# когда всё это заработает
# либо используйте нормальный сценарий запуска, либо
slapd -u ldap -g ldap
```

Если эти действия не помогают (всё равно выдаются ошибки), установите версию 2.3 OpenLDAP и повторите всё заново. Как только будут созданы файлы базы данных - снова установите версию 2.4.

7. Если возникает ошибка при запуске slapd, попробуйте запустить его в режиме диагностики из командной строки:

```
[bsd]/usr/local/libexec/slapd -d -1 -u ldap -g ldap
# все ошибки выводятся в терминал
```


Глава 12. Ошибки и устранение неисправностей LDAP

Иногда OpenLDAP подвергается критике за скудность сообщений об ошибках и слабую диагностику. Отчасти это связано с генеральной линией стандартизации сообщений об ошибках, ограничивающей возможности реализаций служб каталогов по выдаче более информативных и творческих сообщений (справедливости ради стоит отметить, что в стандартах предусмотрено наличие в сообщении текстового элемента для более точного освещения проблемы), отчасти с тем, что многие сообщения об ошибках выводятся через клиентские программы, которые могут делать серьёзные искажения исходных диагностических сообщений.

Лучший источник диагностической информации - журналы OpenLDAP необходимого уровня (безусловно, наиболее полный из которых [loglevel -1](#)).

Ниже мы приводим некоторые сведения о том, как правильно читать [журналы OpenLDAP](#), а также [стандартные сообщения об ошибках LDAP](#) с некоторыми подсказками о том, в чём может заключаться причина ошибки.

Стандартные сообщения об ошибках LDAP

Данные сообщения об ошибках определены в [разделе 4.1.9 RFC 4511](#), черновом RFC [LDAP C API \(датированном 2000-м годом\)](#), а также выяснены путём изучения заголовочного файла LDAPResult.h дистрибутива OpenLDAP.

Название ошибки	Номер	Пояснения/причины
LDAP_SUCCESS	0 (x'00)	Успешное завершение запроса.
LDAP_OPERATIONS_ERROR	1 (x'01)	Произошла ошибка операции.
LDAP_PROTOCOL_ERROR	2 (x'02)	Обнаружено нарушение протокола.
LDAP_TIMELIMIT_EXCEEDED	3 (x'03)	Превышено ограничение по времени LDAP.
LDAP_SIZELIMIT_EXCEEDED	4 (x'04)	Превышено ограничение по размеру LDAP.
LDAP_COMPARE_FALSE	5 (x'05)	Операция сравнения вернула "ложь".
LDAP_COMPARE_TRUE	6 (x'06)	Операция сравнения вернула "истину".
LDAP_STRONG_AUTH_NOT_SUPPORTED	7 (x'07)	Сервер LDAP не поддерживает строгую аутентификацию.
LDAP_STRONG_AUTH_REQUIRED	8 (x'08)	Для данной операции требуется прохождение строгой аутентификации.
LDAP_PARTIAL_RESULTS	9 (x'09)	Возвращены только частичные результаты.
LDAP_REFERRAL	10 (x'0A)	Указывает, что в ответе присутствует отсылка LDAP. Данное сообщение будет содержать один или несколько LDAP URL, по которым клиент должен перенаправить последующие операции для получения данного DN.
LDAP_ADMINLIMIT_EXCEEDED	11 (x'0B)	Указывает на то, что какие-либо ограничения, установленные на стороне сервера на количество записей, возвращаемое при поиске, были превышены.
LDAP_UNAVAILABLE_CRITICAL_EXTENSION	12 (x'0C)	Указывает на то, что элемент управления или правило соответствия, запрашиваемые в операции, не поддерживаются данным сервером.
LDAP_CONFIDENTIALITY_REQUIRED	13 (x'0D)	Конфигурация данного сервера требует обеспечения какой-либо формы конфиденциальности (TLS/SSL или SASL) при выполнении подсоединения с предоставляемым DN, например, определённая на глобальном уровне или в разделе database директива security может требовать соблюдения некоторой формы SSF при выполнении simple_bind или операции обновления.
LDAP_SASL_BIND_IN_PROGRESS	14 (x'0E)	Данный сервер в настоящий момент выполняет SASL-подсоединение и в этом контексте запрашиваемая операция является неверной.
	15 (x'0F)	Не используется.

LDAP_NO_SUCH_ATTRIBUTE	16 (x'10)	Указанный в запросе атрибут не присутствует в записи.
LDAP_UNDEFINED_TYPE	17 (x'11)	Указанный в запросе тип атрибута был неверным.
LDAP_INAPPROPRIATE_MATCHING	18 (x'12)	Указывает на то, что правило соответствия с расширяемым фильтром соответствия не поддерживается для указываемого типа атрибута.
LDAP_CONSTRAINT_VIOLATION	19 (x'13)	Указываемое в операции значение атрибута нарушает некоторые ограничения. Возможные причины: 1. Строка слишком большой длины. 2. Неверный тип - строка записывается в числовой атрибут. 2. Неправильное значение, например, атрибут может принимать только определённое значение, либо одно из набора значений.
LDAP_TYPE_OR_VALUE_EXISTS	20 (x'14)	Указываемый тип атрибута или значение атрибута уже присутствует в записи. Возможные причины: 1. При добавлении записи - один или несколько атрибутов в LDIF (или операции добавления/замены) для записи в точности совпадают (дублируются).
LDAP_INVALID_SYNTAX	21 (x'15)	Было указано неверное значение атрибута.
	22 - 31	(x'16 - x'1F). Не используются.
LDAP_NO_SUCH_OBJECT	32 (x'20)	Указанная запись не существует в каталоге (DIT).
LDAP_ALIAS_PROBLEM	33 (x'21)	Псевдоним в DIT указывает на несуществующую запись.
LDAP_INVALID_DN_SYNTAX	34 (x'22)	Был указан синтаксически неверный DN. Может также возникнуть, если Вы используете файл в формате LDIF (dn: cn=xxx и т.д.) с утилитой ldapdelete, которой требуется только указание простого DN.
	35 (x'23)	Зарезервировано и не используется в LDAPv3 (LDAPv2: LDAP_IS_LEAF - указанный объект является листовым, то есть у него нет дочерних объектов).
LDAP_ALIAS_DEREF_PROBLEM	36 (x'24)	Возникла проблема при разыменовании псевдонима. Смотрите также описание ошибки 33.
	37 - 47	(x'25 - x'2F). Не используются.
LDAP_INAPPROPRIATE_AUTH	48 (x'30)	Была указана проверка подлинности, которую невозможно осуществить, например, была указана LDAP_AUTH_SIMPLE, а у записи нет атрибута userPassword.
LDAP_INVALID_CREDENTIALS	49 (x'31)	Были предоставлены неверные учётные данные, например, неправильный пароль. Дополнительный текст: unable to get TLS Client DN (невозможно получить DN клиента TLS). Возможные причины: 1. Не предоставлен сертификат клиента в случае, если директива TLSVerifyClient установлена в 'demand'. 2. Не предоставлен сертификат клиента в случае, если директива TLSVerifyClient установлена в 'never'. В этом случае данное сообщение об ошибке не является фатальным и обслуживание клиента продолжается.
LDAP_INSUFFICIENT_ACCESS	50 (x'32)	У данного пользователя недостаточно прав доступа на осуществление запрашиваемой операции.
LDAP_BUSY	51 (x'33)	Данный сервер (DSA) слишком занят, чтобы выполнить запрашиваемую операцию.
LDAP_UNAVAILABLE	52 (x'34)	DSA недоступен. Он может быть, например, остановлен, поставлен на паузу или находится в процессе инициализации.
LDAP_UNWILLING_TO_PERFORM	53 (x'35)	Данный сервер (DSA) не желает выполнять запрашиваемую

		операцию. Дополнительный текст: no global superior knowledge (нет сведений о глобальном вышестоящем каталоге) - имя записи, которую собираются добавить или модифицировать, не находится ни в одном из контекстов именования и у сервера нет правильной отсылки на вышестоящий каталог. Возможная причина: в slapd.conf не указана директива suffix для данного DIT. Дополнительный текст: Shadow context; no update referral (теневой контекст (реплика); отсылки для выполнения обновлений не указано) - DIT, в которое собираются вносить изменения, является репликой в режиме "только для чтения", и, из-за отсутствия директивы updateref, невозможно возратить отсылку. Возможные причины: 1. Была попытка произвести запись в реплику "только для чтения" (в конфигурации syncrepl потребитель всегда в режиме "только для чтения"). 2. В конфигурации syncrepl multi-master в файле slapd.conf возможно пропущена директива mirrormode true.
LDAP_LOOP_DETECT	54 (x'36)	Выявлено заикливание.
	54 - 59	(x'37 - x'3B). Не используются.
LDAP_SORT_CONTROL_MISSING	60 (x'3C)	В стандартах не используется. Только для Sun LDAP Directory Server. Сервер не получил требуемый элемент управления сортировки на стороне сервера.
LDAP_RANGE_INDEX_ERROR	61 (x'3D)	В стандартах не используется. Только для Sun LDAP Directory Server. Результаты запроса превысили диапазон, указанный в запросе.
	62 - 63	(x'3E - x'3F). Не используются.
LDAP_NAMING_VIOLATION	64 (x'40)	Указывает на то, что данный запрос содержит нарушение именования в отношении текущего DIT.
LDAP_OBJECT_CLASS_VIOLATION	65 (x'41)	Произошло нарушение объектного класса при использовании текущего набора схемы данных, например, при добавлении записи был пропущен обязательный (must) атрибут.
LDAP_NOT_ALLOWED_ON_NONLEAF	66 (x'42)	Операция на нелистовой записи (то есть той, у которой есть дочерние записи) не разрешается.
LDAP_NOT_ALLOWED_ON_RDN	67 (x'43)	Операция над RDN, например, удаление атрибута, использующегося в качестве RDN в DN, не разрешается.
LDAP_ALREADY_EXISTS	68 (x'44)	Данная запись уже существует в этом DIT.
LDAP_NO_OBJECT_CLASS_MODS	69 (x'45)	Не разрешена модификация объектного класса.
LDAP_RESULTS_TOO_LARGE	70 (x'46)	Только С API (черновой RFC). Результаты слишком велики и не могут содержаться в данном сообщении.
LDAP_AFFECTS_MULTIPLE_DSAS	71 (x'47)	Указывает на то, что операцию необходимо выполнить на нескольких серверах (DSA), а это не разрешено.
	72 - 79	(x'48 - x'4F). Не используются.
LDAP_OTHER	80 (x'50)	Произошла неизвестная ошибка. Возможная причина: Попытка удаления атрибута (особенно в sp=config), удаление которого запрещено
LDAP_SERVER_DOWN	81 (x'51)	Только С API (черновой RFC). Библиотека LDAP не может связаться с LDAP-сервером.
LDAP_LOCAL_ERROR	82 (x'52)	Только С API (черновой RFC). Произошла некоторая локальная ошибка. Обычно это неудачная попытка выделения динамической памяти.
LDAP_ENCODING_ERROR	83 (x'53)	Только С API (черновой RFC). Произошла ошибка при кодировании

		параметров, отправляемых на LDAP-сервер.
LDAP_DECODING_ERROR	84 (x'54)	Только C API (черновой RFC). Произошла ошибка при декодировании результатов, полученных от LDAP-сервера.
LDAP_TIMEOUT	85 (x'55)	Только C API (черновой RFC). При ожидании результатов было превышено ограничение по времени.
LDAP_AUTH_UNKNOWN	86 (x'56)	Только C API (черновой RFC). В ldap_bind() был указан неизвестный метод аутентификации.
LDAP_FILTER_ERROR	87 (x'57)	Только C API (черновой RFC). Операции ldap_search() был предоставлен неправильный фильтр (например, количество открывающихся и закрывающихся скобок в фильтре не совпадает).
LDAP_USER_CANCELLED	88 (x'58)	Только C API (черновой RFC). Указывает на то, что пользователь прервал запрошенную операцию.
LDAP_PARAM_ERROR	89 (x'59)	Только C API (черновой RFC). Процедура ldap была вызвана с неверными параметрами.
LDAP_NO_MEMORY	90 (x'5A)	Только C API (черновой RFC). Выделение памяти (например, с помощью malloc(3) или другого механизма динамического выделения памяти) вызвало сбой в процедуре из библиотеки ldap.
LDAP_CONNECT_ERROR	91 (x'5B)	Только C API (черновой RFC). Библиотека/клиент не может соединиться с LDAP-сервером, указанным в URL.
LDAP_NOT_SUPPORTED	92 (x'5C)	Только C API (черновой RFC). Указывает на то, что в запросе используется функция, не поддерживаемая данным сервером.
LDAP_CONTROL_NOT_FOUND	93 (x'5D)	Только C API (черновой RFC). Запрашиваемый элемент управления не найден на данном сервере.
LDAP_NO_RESULTS_RETURNED	94 (x'5E)	Только C API (черновой RFC). Запрашиваемая операция завершилась успешно, но никаких результатов возвращено (получено) не было.
LDAP_MORE_RESULTS_TO_RETURN	95 (x'5F)	Только C API (черновой RFC). Запрашиваемая операция завершилась успешно, но должны быть возвращены дополнительные результаты, которые можно уместить в текущее сообщение.
LDAP_CLIENT_LOOP	96 (x'60)	Только C API (черновой RFC). Клиент выявил заикливание, например, при следовании по отсылкам.
LDAP_REFERRAL_LIMIT_EXCEEDED	97 (x'61)	Только C API (черновой RFC). Сервер или клиент превысил какое-либо установленное ограничение при следовании по отсылкам.



Журналы OpenLDAP

В данном разделе показаны журналы OpenLDAP с нашими пояснениями. Строки, начинающиеся с # - комментарии, добавленные в целях пояснения, в нормальных журналах (логах) их не будет.



Глава 13. Производительность LDAP

Уже совсем скоро (One day real soon now TM).



Глава 14. Инструменты LDAP

Инструменты OpenLDAP - утилиты командной строки
LDAPBrowser/Editor - LDAP браузер, которым пользуемся мы
Инструменты ApacheDS - инструменты и утилиты

Инструменты OpenLDAP

У OpenLDAP есть набор инструментов. Мы описываем их здесь для полноты картины, но эту информацию всегда можно получить из соответствующих **man**-страниц, если Вам посчастливилось использовать системы 'nix.

Жизненно важное правило: в общем случае, если команда начинается с **ldap**, slapd ДОЛЖЕН быть запущен. Если же она начинается с **slap** (а именно slapcat, slapindex, slapadd) slapd НЕ ДОЛЖЕН быть запущен, а если он запущен, Вы можете повредить базу данных ldap. Использовать их совместно - только на свой страх и риск, хотя в последних версиях OpenLDAP утверждается, что при использовании bdb или hdb slapcat можно выполнять при запущенном slapd. Хм.

Единственное исключение из этого правила - это slappasswd, незлобная маленькая утилитка, предназначенная для создания паролей. Печально, что это маленькое исключение существует - настолько важное значение имеет необходимость остановки **slapd** при запуске всех остальных команд. И хотя это и бессмысленно, - останавливать **slapd** только для того, чтобы выполнить **slappasswd**, лучше выработать условный рефлекс на префикс **slap** и всегда останавливать **slapd**, чем получить разрушенную базу данных при запуске, скажем, **slapcat**. Вот насколько важна эта штука. Некоторые из нас скажут: "Я уж точно никогда не ошибусь, я ведь умный". Но когда в панике пытаешься оживить службу, и при этом 57 человек дышат тебе в затылок, - ну и кто тут умный?

Список инструментов OpenLDAP

- ldapadd** - добавляет LDIF-записи в каталог LDAP
- ldapauth** - добавляет LDIF-записи в каталог LDAP
- ldapdelete** - удаляет LDAP-записи
- ldapmodify** - модифицирует существующие записи LDAP
- ldapmodrdn** - модифицирует DN записи LDAP
- ldappasswd** - модифицирует пароль записи
- ldapsearch** - осуществляет поиск записей LDAP
- ldapwhoami** - выполняет LDAP-операцию Who Am I на сервере
- slapacl** - проверяет доступ к атрибутам, изучая конфигурацию DIT
- slapadd** - добавляет записи LDAP в базу данных - СНАЧАЛА ОСТАНОВИТЕ SLAPD
- slapauth** - проверяет данные SASL на соответствие с DIT
- slapcat** - экспортирует LDIF из базы данных LDAP - СНАЧАЛА ОСТАНОВИТЕ SLAPD
- slapd** - автономный демон LDAP
- slapdn** - проверяет DN на соответствие конфигурации DIT
- slapindex** - переиндексирует базу данных LDAP - СНАЧАЛА ОСТАНОВИТЕ SLAPD
- slappasswd** - генерирует пароль
- slaptest** - проверяет файл slapd.conf или директорию cn=config (slapd.d)

ldapadd & ldapmodify

ldapadd и **ldapmodify** (**обновлёно для 2.4**) принимают одни и те же аргументы и по существу могут рассматриваться как синонимы, то есть ldapmodify с аргументом -a - это и есть ldapadd. Для обоих утилит требуется работающий LDAP-сервер и обе ожидают входную информацию (в формате LDIF) либо со стандартного входа (консоли), либо из LDIF-файла при использовании аргумента -f. Обе команды принимают обширный набор аргументов. На практике реальная работа выполняется командами из LDIF-файла.

Примечание: В версиях до 2.3(?), если в LDIF опущено **changetype**, то по умолчанию при использовании **ldapadd** выполняется операция **add**, а при запуске **ldapmodify** выполняется операция **modify**. Начиная с 2.3 никаких предположений не делается, и LDIF должен содержать все необходимые директивы.

```
ldapmodify/ldapadd [-a] [-c] [-d debug_level] [-f file] [-D binddn] [-H ldapuri] [-h  
ldaphost] [-I] [-k] [-K] [-M[M]] [-n] [-O security-properties] [-p ldapport] [-P 2|3] [-Q]  
[-S file] [-R realm] [-U authcid] [-v] [-w] [-w password] [-x] [-X authzid] [-y passwdfile]
```

Аргумент	Описание
-a	Добавить новые записи. <code>ldapmodify</code> по умолчанию модифицирует существующие записи. При использовании <code>ldapmodify</code> в качестве <code>ldapadd</code> , этот флаг должен быть установлен.
-c	Режим продолжения операции. О возникающих ошибках сообщается, но <code>ldapmodify</code> продолжает вносить изменения. По умолчанию после сообщения об ошибке происходит выход.
-d debug	Устанавливает уровень отладки LDAP в debug . Чтобы эта опция имела эффект, <code>ldapmodify/add</code> должны быть скомпилированы с установленным <code>LDAP_DEBUG</code> .
-D binddn	Использует Distinguished Name, предъявленный в аргументе (<code>binddn</code>), при соединении с каталогом LDAP.
-f file	Читает информацию о модификации записи из файла <code>file</code> , а не со стандартного входа.
-F	Принудительное применение всех изменений, независимо от содержимого входных строк, начинающихся с <code>replica:</code> (по умолчанию строки <code>replica:</code> сравниваются с хостом и портом LDAP-сервера, на который вносится изменение, для принятия решения, должна ли действительно применяться запись <code>replog</code>).
-h ldaphost	Указывает хост, на котором запущен ldap-сервер (по умолчанию <code>localhost</code>). Устаревший аргумент, используйте вместо него <code>-H</code> .
-H ldapuri	Заменяет <code>-h</code> и <code>-p</code>. Определяет один или несколько (разделённых пробелами или запятыми) URI, указывающих на ldap-сервер(ы) в форме <code>scheme://host.name:port</code>. Если не указано, значение по умолчанию <code>ldap://localhost:389</code>. Другой вариант: можно указать DN, который будет использоваться для нахождения соответствующего хоста (хостов) с помощью SRV-записей DNS, как определено в RFC 2782. Данный DN должен быть непустой последовательностью AVA, тип атрибута которых "dc" (domain component), и должен быть экранирован в соответствии с RFC 2396. Примеры: <pre># стандартный формат URI (порт по умолчанию - 389) -H ldap://ldap.example.com # поиск SRV-записей DNS в форме # _ldap._tcp._example.com SRV -H "dc=example,dc=com"</pre>
-I	Включает интерактивный режим SASL. Запрос происходит всегда. По умолчанию запрос происходит только в случае необходимости.
-k	Использует аутентификацию Kerberos IV вместо простой аутентификации. Подразумевается, что у Вас уже есть действительное разрешение на получение разрешения (ticket granting ticket). Чтобы данная опция имела эффект, Вы должны компилировать инструменты с поддержкой Kerberos.
-K	То же, что и <code>-k</code> , но выполняется только первый шаг операции соединения Kerberos IV. Это полезно, когда Вы соединяетесь с <code>slapd</code> и на Вашем контроллере(ах) домена Kerberos не зарегистрирован принципал <code>x500dsa.hostname</code> .
-M[M]	Включает элемент управления manage DSA IT. <code>-MM</code> делает этот элемент управления критичным. Требуется для удаления отсылок.
-n	Демонстрирует, что будет сделано, но реальных модификаций записей не производится. Полезно для отладки в сочетании с <code>-v</code> .
-O props	Указывает свойства безопасности SASL.
-Q	Включает тихий режим SASL. Запрос не происходит никогда.
-R realm	Указывает область (realm) аутентификационного ID для подключений SASL. Форма <code>realm</code> зависит от реально используемого механизма SASL.
-S file	Добавления или изменения записей, которые были пропущены из-за возникших ошибок, записываются в файл <code>file</code> и возвращаемые сервером сообщения об ошибках добавляются в качестве комментариев. При использовании с <code>-c</code> создаётся LDIF-файл, который можно откорректировать (отредактировать) и снова использовать с <code>ldapadd / ldapmodify</code> .
-U authcid	Указывает аутентификационный ID для подключений SASL. Форма ID зависит от реально используемого механизма SASL.
-v	Режим подробного вывода - диагностические сообщения выводятся на стандартный вывод.

-w password	Указывает использовать пароль в открытом виде password для простой аутентификации. Для безопасности паролей лучше использовать -W или -у.
-W	Указывает запрашивать пароль для простой аутентификации. Это значительно более безопасно, чем указывать пароль в открытом виде в командной строке (-w).
-x	Указывает использовать простую аутентификацию вместо SASL. Если не указан, по умолчанию используется SASL-аутентификация.
-X authzid	Указывает ID запроса авторизации для подключений SASL. authzid должен быть в одном из следующих форматов: dn:<distinguished name> или u:<username>
-y passfile	Использует полное содержимое passfile в качестве пароля для простой аутентификации.
-Y mech	Указывает используемый для аутентификации механизм SASL. Если не указан, выберет лучший из известных серверу механизмов.
-Z[Z]	Пытается выполнить расширенную операцию StartTLS (Transport Layer Security). При использовании -ZZ, команда требует успешного выполнения операции.

Примеры:

Используется указанный LDIF-файл для модификации указанного LDAP-сервера, аутентификация с использованием **rootdn** и его пароля. Используется простая аутентификация.

Приведённая ниже команда разбита на две строки только из соображений форматирования HTML и должна записываться в одну строку:

```
ldapadd -H ldap://ldaphost.example.com -x -D "cn=jimbob,dc=example,dc=com"
-f /tmp/addgroups.ldif -w dirtysecret
```

Примечания:

1. Начиная с OpenLDAP 2.x механизм безопасности по умолчанию - SASL. Если он не используется, должен быть задан аргумент **-x**.
2. Если сервер LDAP находится на том же самом хосте, аргумент -H можно опустить.
3. Если используется аргумент -W (не путать с **-w**), то утилита запрашивает ввод пароля.



Idapdelete

Idapdelete открывает соединение с сервером LDAP, подключается и удаляет одну или несколько записей. Если команде был предоставлен один или несколько аргументов DN, записи с этими Distinguished Names будут удалены. Каждый DN должен быть оформлен в виде строкового представления LDAPv3, как определено в RFC 2253. Если не было предоставлено аргументов dn, список DN будет считываться со стандартного входа (или из файла, если используется флаг -f).

Примечание: Файл, передаваемый этой команде, не в LDIF-формате. Это текстовый файл, содержащий один или несколько DN (по одному в строке), которые будут обрабатываться командой. Пример:

```
cn=someone,ou=people,dc=example,dc=com
cn=someone else,ou=people,dc=example,dc=com
```

Альтернативный метод удаления записей - использовать **ldapmodify** с LDIF-файлом такого содержания:

```
dn: cn=someone,ou=people,dc=example,dc=com
changetype: delete

dn: cn=someone else,ou=people,dc=example,dc=com
changetype: delete
```

```
ldapdelete [-c] [-d debuglevel] [-D binddn] [-f file] [-h ldaphost] [-H ldapuri] [-I] [-k]
[-K] [-M[M]] [-n] [-O security-properties] [-P 2|3] [-p ldapport] [-Q] [-R realm] [-U
authcid] [-v] [-W] [-w passwd] [-x] [-X authzid] [-y passwdfile] [-Y mech] [-Z[Z]] [dn]...
```

Аргумент	Описание
-c	Режим продолжения операции. О возникающих ошибках сообщается, но ldapdelete продолжает вносить изменения. По умолчанию после сообщения об ошибке происходит выход.
-d debug	Устанавливает уровень отладки LDAP в debug . Чтобы эта опция имела эффект, ldapdelete должна быть скомпилирована с установленным LDAP_DEBUG.
-D binddn	Использует Distinguished Name, предъявленный в аргументе (binddn), при соединении с каталогом LDAP.
-f file	Указывает считывать информацию о модификации записи из файла, а не со стандартного входа. Данный файл НЕ в формате LDIF, это простой текстовый файл с DN записей, которые требуется удалить, - по одному в строке.
-h ldaphost	Указывает хост, на котором запущен ldap-сервер (по умолчанию localhost). Устаревший аргумент, используйте вместо него -H.
-H ldapuri	Заменяет -h и -p. Определяет один или несколько (разделённых пробелами или запятыми) URI, указывающих на ldap-сервер(ы) в форме scheme://host.name:port. Если не указано, значение по умолчанию ldap://localhost:389. Другой вариант: можно указать DN, который будет использоваться для нахождения соответствующего хоста (хостов) с помощью SRV-записей DNS, как определено в RFC 2782. Данный DN должен быть непустой последовательностью AVA, тип атрибута которых "dc" (domain component), и должен быть экранирован в соответствии с RFC 2396. Примеры: <pre># стандартный формат URI (порт по умолчанию - 389) -H ldap://ldap.example.com # поиск SRV-записей DNS в форме # _ldap._tcp._example.com SRV -H "dc=example,dc=com"</pre>
-l	Включает интерактивный режим SASL. Запрос происходит всегда. По умолчанию запрос происходит только в случае необходимости.
-k	Использует аутентификацию Kerberos IV вместо простой аутентификации. Подразумевается, что у Вас уже есть действительное разрешение на получение разрешения (ticket granting ticket). Чтобы данная опция имела эффект, Вы должны компилировать инструменты с поддержкой Kerberos.
-K	То же, что и -k, но выполняется только первый шаг операции соединения Kerberos IV. Это полезно, когда Вы соединяетесь с slapd и на Вашем контроллере(ах) домена Kerberos не зарегистрирован принципал x500dsa.hostname.
-M[M]	Включает элемент управления manage DSA IT. -MM делает этот элемент управления критичным.
-n	Демонстрирует, что будет сделано, но реальных модификаций записей не производится. Полезно для отладки в сочетании с -v.
-O props	Указывает свойства безопасности SASL.
-p ldapport	Указывает альтернативный TCP-порт, на котором ожидает подключения ldap-сервер. Устаревший аргумент, используйте вместо него -H.
-P 2 3	Указывает, какую версию протокола LDAP использовать. По умолчанию - 3 (LDAPv3).
-Q	Включает тихий режим SASL. Запрос не происходит никогда.
-R realm	Указывает область (realm) аутентификационного ID для подключений SASL. Форма realm зависит от реально используемого механизма SASL.
-U authcid	Указывает аутентификационный ID для подключений SASL. Форма ID зависит от реально используемого механизма SASL.
-v	Режим подробного вывода - диагностические сообщения выводятся на стандартный вывод.
-w password	Указывает использовать пароль в открытом виде password для простой аутентификации. Для безопасности паролей лучше использовать -W или -y.
-W	Указывает запрашивать пароль для простой аутентификации. Это значительно более безопасно, чем указывать пароль в открытом виде в командной строке (-w).
-x	Указывает использовать простую аутентификацию вместо SASL. Если не указан, по умолчанию используется SASL-аутентификация.

-X authzid	Указывает ID запроса авторизации для подключений SASL. authzid должен быть в одном из следующих форматов: dn:<distinguished name> или u:<username>
-y passfile	Использует полное содержимое passfile в качестве пароля для простой аутентификации.
-Y mech	Указывает используемый для аутентификации механизм SASL. Если не указан, выберет лучший из известных серверу механизмов.
-Z[Z]	Пытается выполнить расширенную операцию StartTLS (Transport Layer Security). При использовании -ZZ, команда требует успешного выполнения операции.
dn..	Разделённый запятыми список DN, которые нужно удалить (может быть прочитан из файла, если используется аргумент -f).



ldapmodrdn

ldapmodrdn открывает соединение с сервером LDAP, подключается и модифицирует RDN записей. Данные о записи считываются со стандартного входа, из файла в случае использования опции -f, либо из пары dn и rdn в командной строке.

```
ldapmodrdn [-r] [-n] [-v] [-k] [-K] [-c] [-M[M]] [-d debuglevel] [-D binddn] [-W] [-w passwd] [-y passwdfile] [-H ldapuri] [-h ldaphost] [-p ldapport] [-P 2|3] [-O security-properties] [-I] [-Q] [-U authcid] [-R realm] [-x] [-X authzid] [-Y mech] [-Z[Z]] [-f file] [dn rdn]
```

Аргумент	Описание
-r	Удаляет старые значения RDN из записи. По умолчанию старые значения сохраняются.
-n	Демонстрирует, что будет сделано, но реальных модификаций записей не производится. Полезно для отладки в сочетании с -v.
-v	Режим подробного вывода - диагностические сообщения выводятся на стандартный вывод.
-k	Использует аутентификацию Kerberos IV вместо простой аутентификации. Подразумевается, что у Вас уже есть действительное разрешение на получение разрешения (ticket granting ticket). Чтобы данная опция имела эффект, Вы должны компилировать инструменты с поддержкой Kerberos.
-K	То же, что и -k, но выполняется только первый шаг операции соединения Kerberos IV. Это полезно, когда Вы соединяетесь с slapd и на Вашем контроллере(ах) домена Kerberos не зарегистрирован принципал x500dsa.hostname.
-c	Режим продолжения операции. О возникающих ошибках сообщается, но ldapmodrdn продолжает вносить изменения. По умолчанию после сообщения об ошибке происходит выход.
-M[M]	Включает элемент управления manage DSA IT. -MM делает этот элемент управления критичным.
-d debug	Устанавливает уровень отладки LDAP в debug . Чтобы эта опция имела эффект, ldapmodrdn должна быть скомпилирована с установленным LDAP_DEBUG.
-D binddn	Использует Distinguished Name, предъявленный в аргументе (binddn), при соединении с каталогом LDAP.
-h ldaphost	Указывает хост, на котором запущен ldap-сервер (по умолчанию localhost). Устаревший аргумент, используйте вместо него -H.
-H ldapuri	Заменяет -h и -p. Определяет один или несколько (разделённых пробелами или запятыми) URI, указывающих на ldap-сервер(ы) в форме scheme://host.name:port. Если не указано, значение по умолчанию ldap://localhost:389. Другой вариант: можно указать DN, который будет использоваться для нахождения соответствующего хоста (хостов) с помощью SRV-записей DNS, как определено в RFC 2782. Данный DN должен быть непустой последовательностью AVA, тип атрибута которых "dc" (domain component), и должен быть экранирован в соответствии с RFC 2396. Примеры: # стандартный формат URI (порт по умолчанию - 389) -H ldap://ldap.example.com # поиск SRV-записей DNS в форме # _ldap._tcp._example.com SRV -H "dc=example,dc=com"

-p ldapport	Указывает альтернативный TCP-порт, на котором ожидает подключения ldap-сервер. Устаревший аргумент, используйте вместо него -H.
-P 2 3	Указывает, какую версию протокола LDAP использовать. По умолчанию - 3 (LDAPv3).
-w password	Указывает использовать пароль в открытом виде password для простой аутентификации. Для безопасности паролей лучше использовать -W или -у.
-W	Указывает запрашивать пароль для простой аутентификации. Это значительно более безопасно, чем указывать пароль в открытом виде в командной строке (-w).
-x	Указывает использовать простую аутентификацию вместо SASL. Если не указан, по умолчанию используется SASL-аутентификация.
-X authzid	Указывает ID запроса авторизации для подключений SASL. authzid должен быть в одном из следующих форматов: dn:<distinguished name> или u:<username>.
-y passfile	Использует полное содержимое passfile в качестве пароля для простой аутентификации.
-Y mech	Указывает используемый для аутентификации механизм SASL. Если не указан, выберет лучший из известных серверу механизмов.
-O props	Указывает свойства безопасности SASL.
-I	Включает интерактивный режим SASL. Запрос происходит всегда. По умолчанию запрос происходит только в случае необходимости.
-Q	Включает тихий режим SASL. Запрос не происходит никогда.
-U authcid	Указывает аутентификационный ID для подключений SASL. Форма ID зависит от реально используемого механизма SASL.
-R realm	Указывает область (realm) аутентификационного ID для подключений SASL. Форма realm зависит от реально используемого механизма SASL.
-Z[Z]	Пытается выполнить расширенную операцию StartTLS (Transport Layer Security). При использовании -ZZ, команда требует успешного выполнения операции.
-f file	Считывает информацию о модификации записи из файла вместо стандартного входа.
dn rdn	Модифицирует dn , используя предоставленный rdn .



Idappasswd

Idappasswd использует расширенную операцию модификации пароля LDAP (LDAP Password Modify Extended Operation), определённую в [RFC 3062](#) для изменения пароля пользователя, который может располагаться в пределах LDAP DIT (в случае, если пользователь указан с помощью DN), либо браться извне с помощью SASL. Поскольку RFC предполагает, что будет использоваться какая-либо из форм аутентификации, Idappasswd не контролирует явно, какая из них должна использоваться.

```
ldappasswd [-A] [-a oldpasswd] [-t oldpasswdfile] [-D binddn]
[-d debuglevel] [-H ldapuri] [-h ldaphost] [-n] [-p ldapport]
[-S] [-s newpasswd] [-T newpasswdfile] [-v] [-w] [-w passwd]
[-y passwdfile] [-O props] [-I] [-Q] [-U authcid]
[-x] [-X authzid] [-R realm] [-Y mech] [-Z[Z]] [user]
```

Аргумент	Описание
-A	Запрашивает старый (текущий) пароль. Несовместимо с опциями -a или -t.
-a oldpassword	Текущий пароль указывается значением oldpassword . Несовместимо с опциями -A или -t.
-d debug	Устанавливает уровень отладки LDAP в debug. Чтобы эта опция имела эффект, Idappasswd должна быть скомпилирована с установленным LDAP_DEBUG.
-D binddn	Использует Distinguished Name, предъявленный в аргументе (binddn), при соединении с каталогом LDAP.
-h ldaphost	Указывает хост, на котором запущен ldap-сервер (по умолчанию localhost). Устаревший аргумент,

	используйте вместо него -H.
-H ldapuri	Заменяет -h и -p. Определяет один или несколько (разделённых пробелами или запятыми) URI, указывающих на ldap-сервер(ы) в форме scheme://host.name:port. Если не указано, значение по умолчанию ldap://localhost:389. Другой вариант: можно указать DN, который будет использоваться для нахождения соответствующего хоста (хостов) с помощью SRV-записей DNS, как определено в RFC 2782. Данный DN должен быть непустой последовательностью AVA, тип атрибута которых "dc" (domain component), и должен быть экранирован в соответствии с RFC 2396. Примеры: <pre># стандартный формат URI (порт по умолчанию - 389) -H ldap://ldap.example.com # поиск SRV-записей DNS в форме # _ldap._tcp._example.com SRV -H "dc=example,dc=com"</pre>
-I	Включает интерактивный режим SASL. Запрос происходит всегда. По умолчанию запрос происходит только в случае необходимости.
-n	Выполняются все операции соединения и протокола, но пароль НЕ Устанавливается. Полезно для отладки в сочетании с -v и/или -d.
-O props	Указывает свойства безопасности SASL.
-p ldapport	Указывает альтернативный TCP-порт, на котором ожидает подключения ldap-сервер. Устаревший аргумент, используйте вместо него -H.
-P 2 3	Указывает, какую версию протокола LDAP использовать. По умолчанию - 3 (LDAPv3).
-Q	Включает тихий режим SASL. Запрос не происходит никогда.
-R realm	Указывает область (realm) аутентификационного ID для подключений SASL. Форма realm зависит от реально используемого механизма SASL.
-s newpasswd	Использует значение newpasswd . Несовместимо с опциями -S и -T.
-S	Запрашивает новый пароль (дважды). Несовместимо с опциями -s и -T.
-t oldpasswdfile	Считывает старый (текущий) пароль из указанного файла. Несовместимо с опциями -a и -A.
-T path	Считывает новый пароль из указанного файла. Несовместимо с опциями -s и -S.
-U authcid	Указывает аутентификационный ID для подключений SASL. Форма ID зависит от реально используемого механизма SASL.
-v	Режим подробного вывода, диагностические сообщения выводятся на стандартный вывод.
-w password	Указывает использовать пароль в открытом виде password для простой аутентификации. Для безопасности паролей лучше использовать -W или -y.
-W	Указывает запрашивать пароль для простой аутентификации. Это значительно более безопасно, чем указывать пароль в открытом виде в командной строке (-w).
-x	Указывает использовать простую аутентификацию вместо SASL. Если не указан, по умолчанию используется SASL-аутентификация.
-X authzid	Указывает ID запроса авторизации для подключений SASL. authzid должен быть в одном из следующих форматов: dn:<distinguished name> или u:<username>.
-y passfile	Использует полное содержимое passfile в качестве пароля для простой аутентификации.
-Y mech	Указывает используемый для аутентификации механизм SASL. Если не указан, выберет лучший из известных серверу механизмов.
-Z[Z]	Пытается выполнить расширенную операцию StartTLS (Transport Layer Security). При использовании -Z и неудачном старте TLS, выполнение требуемой операции будет продолжено. При использовании -ZZ требуется успешный старт TLS.
user	Пользователь, указанный DN этого пользователя, заключённым в двойные кавычки, например: "cn=slimy toad,ou=people,dc=example,dc=com".

Примеры:

Изменяем userPassword для записи cn=slimy toad,ou=people,dc=example,dc=com используя rootdn в качестве DN для подключения. Запрашивается старый пароль, новый пароль и пароль rootdn.

```
ldappasswd [-H ldap://localhost] -D cn=admin,dc=example,dc=com -W -A
-S "cn=slimy toad,ou=people,dc=example,dc=com"
```



Idapsearch

Idapsearch открывает соединение с сервером LDAP, подключается и производит поиск, используя указанные параметры. Указываемый фильтр должен соответствовать строковому представлению [поисковых фильтров](#), определённого в RFC 4515. Если фильтр не был указан, используется фильтр по умолчанию(objectClass=*).

Если **Idapsearch** находит одну или несколько записей, возвращаются атрибуты, указанные как аргументы attrs. Если в списке атрибутов присутствует *, возвращаются все пользовательские атрибуты. Если присутствует +, возвращаются все **операционные** атрибуты. Если не было указано аргументов attrs, будут возвращены все пользовательские атрибуты (то есть подразумевается *).

Результаты Idapsearch отображаются в формате LDIF (более или менее подробный формат вывода контролируется с помощью -L -LL и -LLL).

```
ldapsearch [-a never|always|search|find] [-A] [-b searchbase] [-c] [-d debuglevel]
[-D binddn] [-e [!]ext[=extparam]] [-E [!]ext[=extparam]] [-f file] [-F prefix]
[-h ldaphost] [-H ldapuri] [-I] [-l time] [-L[L[L]]] [-M[M]] [-n]
[-O security-properties] [-p ldapport] [-P 2|3] [-Q] [-R realm]
[-s base|one|sub|children] [-S attribute] [-t[t]] [-T path] [-u] [-U authcid]
[-v] [-w passwd] [-W] [-x] [-X authzid] [-y passwdfile] [-Y mech] [-z sizelimit]
[-Z[Z]] filter [attrs...]
```

Аргумент	Описание
-a never always search find	Определяет, как будет выполняться разыменование псевдонимов. Возможные варианты: never , always , search или find , которые, соответственно, указывают, что псевдонимы никогда не разыменовываются, всегда разыменовываются, разыменовываются при поиске или разыменовываются только при нахождении базового объекта для поиска. По умолчанию - never .
-A	Извлекать только атрибуты (без значений). Полезно, когда Вам нужно посмотреть только наличие атрибута в записи, а конкретные значения не интересуют.
-b base	Использовать base (DN) в качестве точки начала поиска вместо определённой по умолчанию.
-c	(LDAP 2.4+) продолжать после возникновения ошибки. По умолчанию после возникновения ошибки будет завершаться. Имеет смысл только в сочетании с -f (считывание информации о поисках из файла).
-d debug	Устанавливает уровень отладки LDAP в debug. Чтобы эта опция имела эффект, Idapsearch должна быть скомпилирована с установленным LDAP_DEBUG.
-D binddn	Использует Distinguished Name, предъявленный в аргументе (binddn), при соединении с каталогом LDAP.
-e [!]ext[=extparam]	Смотрите -E ниже.
-E [!]ext[=extparam]	Указывает общие (при -e) и поисковые (при -E) расширения. '!' обозначает высокую критичность расширения. Общие расширения: [!]assert=<filter> (Фильтр RFC 4515) [!]authzid=<authzid> ("dn:<dn>" или "u:<user>") [!]manageDSAit [!]noop ppolicy [!]postread[=<attrs>] (разделённый запятыми список атрибутов) [!]preread[=<attrs>] (разделённый запятыми список атрибутов) abandon, cancel (отказ/сброс посылки SIGINT) Поисковые расширения: [!]domainScope (доменный диапазон) [!]mv=<filter> (фильтр значений, для которых найдено соответствие)

	<pre>[!]pr=<size>[/prompt noprompt] (постраничный вывод результатов/запрос постраничного вывода) [!]subentries[=true false] (подзаписи) [!]sync=ro[/<cookie>] (LDAP Sync refreshOnly) rp[/<cookie>][/<slimit>] (LDAP Sync refreshAndPersist)</pre>
-f file	Считывает ряд строк из файла, выполняя по одному поисковому LDAP-запросу для каждой строки. В этом случае, заданный в командной строке фильтр рассматривается как шаблон, которым заменяется первое и единственное появление %s в строке файла. Любое другое появление символа % в шаблоне будет рассматриваться как ошибка. Если нужно включить в поисковый фильтр символ %, он должен быть закодирован как \25 (смотрите RFC 4515). Если в качестве file был передан единичный символ -, строки будут считываться со стандартного входа. Если не используется -с (только 2.4+), ldapsearch завершит работу при возвращении первого неуспешного результата поиска.
-F prefix	URL-префикс для временных файлов. По умолчанию file:/path/ , где path - /var/tmp/ или указанный в аргументе -T.
-h ldaphost	Указывает хост, на котором запущен ldap-сервер (по умолчанию localhost). Устаревший аргумент, используйте вместо него -H.
-H ldapuri	Заменяет -h и -р. Определяет один или несколько (разделённых пробелами или запятыми) URI, указывающих на ldap-сервер(ы) в форме scheme://host.name:port. Если не указано, значение по умолчанию ldap://localhost:389. Другой вариант: можно указать DN, который будет использоваться для нахождения соответствующего хоста (хостов) с помощью SRV-записей DNS, как определено в RFC 2782. Данный DN должен быть непустой последовательностью AVA, тип атрибута которых "dc" (domain component), и должен быть экранирован в соответствии с RFC 2396. Примеры: <pre># стандартный формат URI (порт по умолчанию - 389) -H ldap://ldap.example.com # поиск SRV-записей DNS в форме # _ldap._tcp._example.com SRV -H "dc=example,dc=com"</pre>
-I	Включает интерактивный режим SASL. Запрос происходит всегда. По умолчанию запрос происходит только в случае необходимости.
-l time	Указывает ожидать до окончания поиска не более time секунд. Если time установлено в 0 (ноль), то ограничения из ldap.conf снимаются. Данное значение не может превышать любое ограничение timelimit (в slapd.conf), за исключением случая подключения от rootdn (с помощью аргумента -D).
-L[L[L]]	Результаты поиска отображаются в формате LDIF . Одиночный -L указывает осуществлять вывод в формате LDIFv1. Второй L (-LL) указывает выводить без комментариев. Третий L (-LLL) отключает печать версии LDIF. По умолчанию (без -L) вывод осуществляется в расширенной версии LDIF.
-M[M]	Включает элемент управления manage DSA IT. -MM делает этот элемент управления критичным.
-n	Демонстрирует, что будет сделано, но реальных модификаций записей не производится. Полезно для отладки в сочетании с -v.
-O props	Указывает свойства безопасности SASL.
-p ldapport	Указывает альтернативный TCP-порт, на котором ожидает подключения ldap-сервер. Устаревший аргумент, используйте вместо него -H.
-P 2 3	Указывает, какую версию протокола LDAP использовать. По умолчанию - 3 (LDAPv3)
-Q	Включает тихий режим SASL. Запрос не происходит никогда.
-R realm	Указывает область (realm) аутентификационного ID для подключений SASL. Форма realm зависит от реально используемого механизма SASL.
-s scope	Указывает диапазон поиска. Может быть base , one , sub или children и указывает, соответственно, поиск в базе поиска, на один уровень ниже или во всём поддереве. По умолчанию - sub . Примечание: диапазон children требует расширения функции подчинения LDAPv3.
-S attribute	Сортировать возвращаемые записи на основании указанного атрибута. По умолчанию возвращаемые записи не сортируются. Если значение атрибута - строка нулевой длины (""), записи сортируются по компонентам их Distinguished Name. Дополнительную информацию

	смотрите в <code>ldap_sort(3)</code> . Имейте в виду, что по умолчанию <code>ldapsearch</code> выводит записи по мере их получения. Использование опции <code>-S</code> изменяет это поведение, приводя к тому, что сначала все записи получаются, затем сортируются, а затем выводятся.
<code>-t</code>	Записывает извлечённые значения в набор временных файлов. Полезно для работы с не-ASCII значениями, такими как <code>jpegPhoto</code> или <code>audio</code> . Второй <code>t</code> (<code>-tt</code>) указывает сохранять все значения во временные файлы. По умолчанию файлы располагаются в <code>/var/tmp</code> .
<code>-T path</code>	Указывает путь к директории, которая будет использована для хранения файлов, создаваемых при использовании аргумента <code>-t</code> (применяется для переопределения значения по умолчанию <code>/var/tmp</code>).
<code>-u</code>	Включает в вывод удобную для пользователя форму Distinguished Name (DN).
<code>-U authcid</code>	Указывает аутентификационный ID для подключений SASL. Форма ID зависит от реально используемого механизма SASL.
<code>-v</code>	Режим подробного вывода - диагностические сообщения выводятся на стандартный вывод.
<code>-w password</code>	Указывает использовать пароль в открытом виде password для простой аутентификации. Для безопасности паролей лучше использовать <code>-W</code> или <code>-y</code> .
<code>-W</code>	Указывает запрашивать пароль для простой аутентификации. Это значительно более безопасно, чем указывать пароль в открытом виде в командной строке (<code>-w</code>).
<code>-x</code>	Указывает использовать простую аутентификацию вместо SASL. Если не указан, по умолчанию используется SASL-аутентификация.
<code>-X authzid</code>	Указывает ID запроса авторизации для подключений SASL. <code>authzid</code> должен быть в одном из следующих форматов: <code>dn:<distinguished name></code> или <code>u:<username></code>
<code>-y passfile</code>	Использует полное содержимое <code>passfile</code> в качестве пароля для простой аутентификации.
<code>-Y mech</code>	Указывает используемый для аутентификации механизм SASL. Если не указан, выберет лучший из известных серверу механизмов.
<code>-z size</code>	получать не более size записей за операцию поиска. Если size установлено в 0 (ноль), то ограничения из <code>ldap.conf</code> снимаются. Данное значение не может превышать любое ограничение <code>sizelimit</code> (в <code>slapd.conf</code>), за исключением случая подключения от <code>rootdn</code> (с помощью аргумента <code>-D</code>).
<code>-Z[Z]</code>	Пытается выполнить расширенную операцию StartTLS (Transport Layer Security). При использовании <code>-ZZ</code> , команда требует успешного выполнения операции.
<code>filter</code>	Определяет поисковый фильтр , который будет использоваться. Он заключается в двойные кавычки и ограничивается скобками.
<code>attrs</code>	Разделённый пробелами список атрибутов, которые требуется вернуть.

Примеры:

В следующем примере будет осуществляться поиск на нескольких уровнях (значение аргумента `-s` по умолчанию `sub`) записей с атрибутом `mail`, почтовый адрес в значении которого содержит `smith` в любом месте, затем отобразить атрибуты `sn`, `cn` и `mail` и вывести всё это в формате LDIFv1 без комментариев. Аргумент `-LL` включен, чтобы выводимое LDIF-содержимое можно было перенаправить в файл и затем обработать его с помощью, скажем, `ldapmodify`.

```
ldapsearch -H ldap://ldap.example.com -LL -b ou=people,dc=example,dc=com
"{mail=*smith*}" sn cn mail

# при использовании стандартного перенаправления результаты запроса могут быть записаны в
# файл
# > /tmp/search1.ldif
```

Idapwhoami

Уже совсем скоро (One Day Real Soon Now™)



slapacl

Обновлено для 2.4. **slapacl** позволяет пользователю проверить возможность доступа от имени определённого DN подключения (-b) к указанным атрибутам, применяя текущие директивы **access** в файле slapd.conf. Данной утилите нужен доступ только на чтение к файлу slapd.conf и DIT, поэтому она может использоваться при запущенном slapd.

```
slapacl -b DN [-d level] [-D authcDN | -U authcID] [-f slapd.conf] [-F confdir] [-o name[=value]] [-u] [-v] [-X authzID | -o authzDN=DN] [attr[/access][:value]] [...]
```

Аргумент	Описание
-b DN	DN пользователя, от имени которого предполагается подключаться к DIT. Запись извлекается из DIT и, следовательно, должна существовать. DN тестируется по соответствующим ACL (глобальным и специфичным для DIT), чтобы определить, будут ли доступны указанные атрибуты согласно назначенным на них правилам доступа. Также смотрите -u.
-d level	Включить отладочные сообщения согласно указанному уровню level.
-D authcDN	Указывает DN, который будет использоваться в качестве идентификационной сущности в ходе тестовой сессии при выборе соответствующего условия <by> в списках контроля доступа.
-f slapd.conf	Определяет местоположение slapd.conf. По умолчанию - [fc] /etc/openssldap/slapd.conf или [bsd] /usr/local/etc/openssldap/slapd.conf.
-F confdir	Определяет конфигурационную директорию cn=config . Если указаны оба аргумента -f и -F, конфигурационный файл (указанный аргументом -f) будет прочтён и переконвертирован в формат cn=config и записан в директорию, указанную аргументом -F. Если не было указано ни одного из аргументов -f и -F, OpenLDAP сначала попытается прочитать конфигурационную директорию по умолчанию ([fc] /etc/openssldap/slapd.d, [bsd] /usr/local/etc/openssldap/slapd.d) и если таковой не существует, вернётся к использованию конфигурационного файла по умолчанию (slapd.conf). Если существует конфигурационная директория в правильном формате, то конфигурационный файл игнорируется. Если поддерживается и задан режим холостого прогона (аргумент -u, поддерживается большинством утилит, но не slapd), переконвертации не произойдёт.
-o name[=value]	Указывает опции slapd и, при необходимости, их значения. Примеры: <pre>syslog=subsystems (`-s' в slapd) syslog-user=user     (`-l' в slapd)</pre> Возможные опции/значения применительно к slapacl: <pre>authzDN domain peername sasl_ssf sockname sockurl ssf tls_ssf transport_ssf</pre>
-u	Не извлекать запись из базы данных. В этом случае, если запись не существует, используется подставная запись с DN, заданным в опции -b, без атрибутов. Как следствие, те правила, которые зависят от содержимого целевого объекта, не будут вести себя так, как с реальным объектом. Заданный в опции -b DN всё же будет использоваться для выбора правил, которые будут применяться, поэтому он должен быть в контексте именования конфигурируемой базы данных. Смотрите также -b.
-v	Включить подробный режим.
-X authzID	Указывает ID, который будет отображаться в DN посредством правил authz-regexp или authz-rewrite; несовместимо с -D.
attr[/access][:value] [...]	Последовательность атрибутов, каждый из которых представляет собой строку, заключённую в кавычки, как показано в примере: <pre># проверяем, возможно ли прочитать # organizationName (o) со значением 'Example, inc.'</pre>

slapadd

Обновлено для 2.4+. **slapadd** используется для добавления записей, указанных в формате LDIF в **базу данных** LDAP. ПРИ ВЫПОЛНЕНИИ ЭТОЙ КОМАНДЫ LDAP НЕ ДОЛЖЕН БЫТЬ ЗАПУЩЕН, В ПРОТИВНОМ СЛУЧАЕ МОЖЕТ ПРОИЗОЙТИ СЕРЬЁЗНОЕ РАЗРУШЕНИЕ БАЗЫ ДАННЫХ. Команда применяет LDIF к базе данных, определяемой по номеру базы данных или суффиксу. Входной LDIF считывается со стандартного входа, либо из указанного файла (аргумент -l).

ПЕРЕД ЗАПУСКОМ ОСТАНОВИТЕ SLAPD. **slapadd** предназначена для принудительного занесения LDIF в базу данных, она не проверяет существование вышестоящих записей перед добавлением записи, не выполняет проверку пользователя и системной схемы данных и не соблюдает операционные атрибуты (такие как createTimeStamp и modifiersName).

```
slapadd [-b suffix] [-c] [-d level] [-f slapd.conf]
        [-F confdir] [-g] [-j lineno] [-l ldif-file]
        [-n dbnum] [-o name[=value]]
        [-q] [-s] [-S SID] [-u dryrun] [-v] [-w]
```

Аргумент	Описание
-b suffix	Использовать указанный суффикс для определения базы данных, в которую будут добавляться записи. -b не может использоваться совместно с опцией -n.
-c	Включить режим продолжения (игнорирования ошибок).
-d level	Включить отладочные сообщения согласно указанному уровню level.
-f slapd.conf	Определяет местоположение slapd.conf. По умолчанию - [fc] /etc/openldap/slapd.conf или [bsd] /usr/local/etc/openldap/slapd.conf.
-F confdir	Определяет конфигурационную директорию cn=config . Если указаны оба аргумента -f и -F, конфигурационный файл (указанный аргументом -f) будет прочтён и переконвертирован в формат cn=config и записан в директорию, указанную аргументом -F. Если не было указано ни одного из аргументов -f и -F, OpenLDAP сначала попытается прочитать конфигурационную директорию по умолчанию ([fc] /etc/opensslap/slapd.d, [bsd] /usr/local/etc/openldap/slapd.d) и если таковой не существует, вернётся к использованию конфигурационного файла по умолчанию (slapd.conf). Если существует конфигурационная директория в правильном формате, то конфигурационный файл игнорируется. Если поддерживается и задан режим холостого прогона (аргумент -u, поддерживается большинством утилит, но не slapd), переконвертации не произойдёт.
-g	Отключить подчинённые связи. Будет обработана только указанная база данных, а связанные с ней подчинённые (если они вообще есть) - нет.
-j lineno	Перейти к строке с указанным lineno (номером строки) в LDIF-файле перед началом обработки записей. Это позволяет продолжить загрузку, прерванную из-за ошибок во входном LDIF, после исправления ошибок.
-l ldif-file	Считывать LDIF из указанного файла вместо стандартного входа.
-n dbnum	Добавлять записи в базу данных с порядковым номером dbnum согласно их перечислению в конфигурационном файле. -n не может использоваться совместно с опцией -b.
-o name[=value]	Указывает опции slapd и, при необходимости, их значения. Примеры: syslog=subsystems (`-s' в slapd) syslog-user=user    (`-l' в slapd)
-q	Включает быстрый режим (с меньшим количеством проверок целостности). Делается меньше проверок целостности во входных данных, и вообще не делается проверок целостности при записи в базу данных. Сокращает время загрузки, но если во время загрузки возникают какие-либо ошибки или сбои, с получившейся базой данных невозможно будет работать.
-s	Отключает проверку схемы данных. Эта опция предназначена для использования при загрузке баз данных, содержащих специальные объекты, такие как раздробленные объекты на неполных репликах. Загрузка же

	нормальных, но не удовлетворяющих схеме данных объектов, может привести к неработоспособности DIT, поэтому не рекомендуется.
-S SID	Задаёт ID сервера, который используется для генерации entryCSN . Также используется для генерации contextCSN при заданном аргументе <code>`-w'</code> . По умолчанию - 0.
-u	Только проверка правильности конфигурации (slapd.conf или slapd.d - смотрите аргументы -f и -F). По умолчанию утилита будет проверять наличие и целостность баз данных, определённых в разделах database конфигурации. Данный флаг подавляет такие проверки. Чтобы проверить только одну конкретную базу данных используйте флаг -n. Кроме того, использование флага -q подавляет переконвертацию файла slapd.conf в cn=config если используются одновременно флаги -f и -F. По существу, при использовании флага -q утилита выполняет только проверочные функции.
-v	Включить подробный режим.
-w	2.3+. Записывать контекстную информацию syncrepl. После добавления всех записей обновляется contextCSN , куда записывается наибольший CSN в базе данных. Подразумевается, что LDIF-файл содержит атрибуты entryCSN (были созданы в последних версиях 2.2 DIT). Использование этой опции позволяет потребителям генерировать SyncCookie , минимизируя тем самым время начального этапа синхронизации при репликации в стиле syncrepl. Смотрите также синхронизацию syncrepl .

slapauth

Уже совсем скоро (One Day Real Soon Now™)

slapcat

Обновлено для 2.4. ПЕРЕД ЗАПУСКОМ ОСТАНОВИТЕ SLAPD - хотя в последних версиях (2.3+) OpenLDAP утверждает, что при использовании механизмов манипуляции данными HDB или BDB slapcat можно безопасно выполнять при запущенном slapd. slapcat используется для генерации LDIF, основанных на содержимом базы данных LDAP. Она открывает базу данных, определяемую по номеру базы данных или суффиксу, и пишет соответствующий LDIF на стандартный вывод или в указанный файл (аргумент -l). При выполнении этой команды OpenLDAP не должен быть запущен.

Сгенерированный данной утилитой LDIF может быть использован **slapadd**. Поскольку записи располагаются в порядке их нахождения в базе данных, а не в порядке их взаимной подчинённости, их нельзя загрузить с помощью **ldapadd** без переупорядочивания.

```
slapcat [-a filter] [-b suffix] [-c] [-d level]
        [-f slapd.conf] [-F confdir] [-g] [-l ldif-file]
        [-n dbnum] [-o name=value] [-s subtree-dn] [-v]
```

Аргумент	Описание
-a filter	Делать дамп только записей, соответствующих указанному фильтру (filter). Например, <pre>slapcat -a \"!(entryDN:dnSubtreeMatch:=ou=People,dc=example,dc=com)\"</pre> сделает дамп всей базы данных "dc=example,dc=com", кроме поддерева "ou=People,dc=example,dc=com".
-b suffix	Использовать указанный суффикс suffix (как определено в директиве suffix slapd.conf) для определения базы данных, для которой требуется сгенерировать вывод. -b не может использоваться совместно с опцией -n.
-c	Включить режим продолжения (игнорирования ошибок).
-d level	Включить отладочные сообщения согласно указанному уровню level.
-f slapd.conf	Определяет местоположение slapd.conf. По умолчанию - [fc] /etc/openldap/slapd.conf или [bsd] /usr/local/etc/openldap/slapd.conf.
-F confdir	Определяет конфигурационную директорию cn=config . Если указаны оба аргумента -f и -F, конфигурационный файл (указанный аргументом -f) будет прочтён и переконвертирован в формат cn=config и записан в

	директорию, указанную аргументом -F. Если не было указано ни одного из аргументов -f и -F, OpenLDAP сначала попытается прочитать конфигурационную директорию по умолчанию ([fc] /etc/openssl/slapd.d, [bsd] /usr/local/etc/openldap/slapd.d) и если таковой не существует, вернётся к использованию конфигурационного файла по умолчанию (slapd.conf). Если существует конфигурационная директория в правильном формате, то конфигурационный файл игнорируется. Если поддерживается и задан режим холостого прогона (аргумент -u, поддерживается большинством утилит, но не slapd), переконвертации не произойдёт.
-g	Отключить подчинённые связи. Будет обработана только указанная база данных, а связанные с ней подчинённые (если они вообще есть) - нет.
-l ldif.file	Записывать LDIF в указанный файл вместо стандартного вывода.
-n dbnum	Генерировать вывод для базы данных с порядковым номером dbnum согласно их перечислению в конфигурационном файле. -n не может использоваться совместно с опцией -b.
-o name[=value]	Указывает опции slapd и, при необходимости, их значения. Примеры: syslog=subsystems ('-s' в slapd) syslog-user=user ('-l' в slapd)
-q	Включает быстрый режим (с меньшим количеством проверок целостности). Делается меньше проверок целостности во входных данных, и вообще не делается проверок целостности при записи в базу данных. Сокращает время загрузки, но если во время загрузки возникают какие-либо ошибки или сбои, с получившейся базой данных невозможно будет работать.
-s subtree-dn	Делать дампы только записей поддерева, указанного данным DN. Работает как '-b subtree-dn', если не задавались опции -b или -n.
-v	Включить подробный режим.



slapd

Обновлено для 2.4+. **slapd** - это автономный демон OpenLDAP. Обычно он запускается с использованием скрипта ([fc] /etc/rc.d/init.d/slapd или [bsd] /usr/local/rc.d/slapd с ключевыми словами stop|start|restart). **slapd** обычно устанавливается в [fc] /usr/lib/slapd или [bsd] /usr/local/libexec/slapd. Ниже приводятся аргументы для управления работой демона.

```
slapd [-4|6] [-c cookie] [-d debug-level] [-f slapd-config-file]
      [-F slapd-config-directory] [-g group] [-h URLs]
      [-l syslog-local-user] [-n service-name] [-s syslog-level]
      [-r directory] [-T {acl|add|auth|cat|dn|index|passwd|test}]
      [-u user]
```

Аргумент	Описание
-4 6	Указывает, что запросы будут приниматься только на интерфейсах IPv4 (4), либо только на IPv6 (6). По умолчанию запросы принимаются на всех сетевых интерфейсах на стандартных портах ldap (389) и ldaps (636) на всех поддерживаемых протоколах.
-c cookie	Опция -c позволяет принудительно запустить репликацию с определённой пользователем точки, путём принудительной отправки определённого значения SyncCookie на открытое соединение синхронизации, в отличие от поведения по умолчанию, когда значение последнего сохранённого SyncCookie считывается при загрузке из основной базы данных. Куки - это разделённый запятыми список пар имя=значение. В качестве имени параметра сейчас поддерживаются rid и csn . rid идентифицирует директиву syncrepl в slapd.conf по соответствующему параметру rid . csn - это порядковый номер изменения (commit sequence number), обычно являющийся последним полученным от поставщика значением.
-d debug-level	Устанавливает уровень отладки в указанный debug-level. debug-level - это число, принимающее значения, аналогичные определённым для директивы loglevel в slapd.conf. Так, -d -1 включает максимальную диагностику. Когда slapd запускается из командной строки, обычно он отделяется от консоли/tty, инициировавшей данную команду. Когда используется аргумент -d, даже если его значение 0, slapd не будет отделяться от вызвавшего его терминала, таким образом все сообщения об ошибках будут выводиться прямо в терминал/консоль/tty, делая данную функцию бесценной для быстрой диагностики проблем с загрузкой. Чтобы выполнять отделение от терминала/консоли/tty, но при этом

	изменять loglevel, используйте аргумент -s.
-f slapd.conf	Определяет местоположение slapd.conf. По умолчанию - [fc] /etc/openldap/slapd.conf или [bsd] /usr/local/etc/openldap/slapd.conf.
-F confdir	Определяет конфигурационную директорию cn=config . Если указаны оба аргумента -f и -F, конфигурационный файл (указанный аргументом -f) будет прочтён и переконвертирован в формат cn=config и записан в директорию, указанную аргументом -F. Если не было указано ни одного из аргументов -f и -F, OpenLDAP сначала попытается прочитать конфигурационную директорию по умолчанию ([fc] /etc/openldap/slapd.d, [bsd] /usr/local/etc/openldap/slapd.d) и если таковой не существует, вернётся к использованию конфигурационного файла по умолчанию (slapd.conf). Если существует конфигурационная директория в правильном формате, то конфигурационный файл игнорируется. Если поддерживается и задан режим холостого прогона (аргумент -u, поддерживается большинством утилит, но не slapd), переконвертации не произойдёт.
-g group	Указывает имя или ID группы, от которой будет запущен OpenLDAP. На большинстве систем по умолчанию это будет группа ldap . OpenLDAP загружается от root для того чтобы зарезервировать свои привилегированные порты (389 и 636), убеждается, что у slapd.conf корректные уровни привилегий (меняя их по мере необходимости), а затем снижает привилегии, переходя на использование заданных учётных записей group и, возможно, user (-u user). Если задан аргумент -r, то файл групп ([fc &] /etc/group) должен находиться в пределах заданной структуры директорий.
-h URLs	Определяет IP-адреса и порты, которые будут использоваться OpenLDAP при инициализации своих сокетов для подсоединений и ожидания запросов. По умолчанию это только порт 389 на всех поддерживаемых сетевых интерфейсах (IPv4 и/или IPv6), что соответствует -h ldap:///. Если система настроена на оба вида сетевых протоколов, то для принудительного приёма запросов только на интерфейсах IPv4 или IPv6 можно использовать аргументы -4 и -6. Аргумент -h заменяет создаваемые по умолчанию соединения согласно ldap URL и может использоваться для принудительного ожидания соединений на портах ldaps. Аргумент -h принимает один или несколько URL, разделённых пробелами. Если передаётся несколько URL, они должны заключаться в двойные кавычки ("). Общий формат URL scheme://[host[:port]]/. Здесь host может быть IP-адресом (IPv4 или IPv6) или именем хоста. Если IP-адрес IPv6, он должен заключаться в квадратные скобки ([]). Если в качестве host указан 0.0.0.0, соединения будут ожидать на всех интерфейсах только с IPv4 адресами, эквивалентная форма для адресов IPv6 - [::]. Если не был указан номер порта, то при URL со схемой ldap будет открыт порт 389, а при URL со схемой ldaps будет открыт порт 636. Примеры: <pre># соединения на всех интерфейсах с адресами только IPv4, порт по умолчанию 389 slapd -h ldap://0.0.0.0/ # функционально эквивалентно slapd -4 # соединения на всех интерфейсах с адресами только IPv6, порт по умолчанию 389 slapd -h ldap://[::]/ # функционально эквивалентно slapd -6 # соединения на IPv4 и IPv6, только 2000 порт slapd -h ldap://:2000/ # соединения на IPv4 и IPv6, порты 389 и 2000 slapd -h "ldap:/// ldap://:2000/" # соединения на IPv4 и IPv6, порт 389, а также на IPv4, только 2000 порт slapd -h "ldap:/// ldap://0.0.0.0:2000/" # соединения на всех интерфейсах с адресами IPv4 и IPv6, по умолчанию только 636 порт slapd -h ldaps:/// # соединения на всех интерфейсах с адресами IPv4 и IPv6, порты по умолчанию 389 и 636 slapd -h "ldaps:/// ldaps:///" # соединения на IPv4 и IPv6, порт по умолчанию 389 # и ldaps на всех интерфейсах с адресами только IPv6, порт 2001 slapd -h "ldaps:/// ldaps://[::]:2001/"</pre>
-l syslog-local-user	По умолчанию OpenLDAP производит журналирование с помощью демона syslogd в пользовательский канал local4. Данный аргумент позволяет использовать другой канал, который должен быть либо user, либо daemon, либо из промежутка local0 - local7. Примеры: <pre>slapd -l local5 # чтобы syslog принимал поток из local5 # отредактируйте /etc/syslog.conf local5.* /var/log/ldap.log # а затем перезапустите syslogd [fc] /etc/init.d/syslog restart [bsd] killall -HUP syslogd</pre>

-n service-name	Указывает имя службы демона openldap для журналирования и других целей. По умолчанию - argv[0], то есть slapd . Если, к примеру, нужно писать в журнал от имени ldap , используйте -n ldap .
-r directory	Указывает запускать slapd в режиме chroot, используя заданную директорию directory как базу chroot. Это делается после открытия портов и проверки прав на конфигурационные файлы или директории, но до считывания конфигурации из файла или директории и инициализации каких-либо механизмов манипуляции данными. Данную опцию следует использовать в сочетании с опциями -u и -g. Все файлы и директории, используемые OpenLDAP в данном режиме, будут добавлять directory к заданным путям перед тем как получить доступ к запрашиваемым файлам, таким как базы данных, журналы и другие операционные файлы. Следует отметить, что запрос chroot выполняется перед тем, как OpenLDAP переключается на заданных user и group , таким образом, директория chroot должна включать копии файлов безопасности, содержащих пользователей и группы (смотрите описания аргументов -u и -g).
-s syslog-level	Устанавливает уровень отладочных сообщений, журналируемых через syslogd. Замещает значение директивы loglevel в slapd.conf (или добавляет, если таковая отсутствовала). В отличие от аргумента -d, -s позволяет демону slapd отделиться от терминала после загрузки.
-T {a c d i p t acl auth}	Запуск в режиме инструмента (утилиты). Параметр аргумента указывает, в качестве какого инструмента запускаться (в порядке, указанном в прототипе аргумента) - slapadd, slapcat, slapdn, slapindex, slappasswd, slaptest, slapacl или slapauth. Данный аргумент должен указываться самым первым, тогда все остальные аргументы будут интерпретироваться как аргументы соответствующих программ slap-инструментов. Данная опция предназначена только для ситуаций, когда символические ссылки не поддерживаются, либо их невозможно использовать.
-u user	Указывает имя или ID пользователя, от которого будет запущен OpenLDAP. На большинстве систем по умолчанию это будет пользователь ldap . OpenLDAP загружается от root для того, чтобы зарезервировать свои привилегированные порты (389 и 636), убеждается, что у slapd.conf корректные уровни привилегий (меняя их по мере необходимости), а затем снижает привилегии, переходя на использование заданных учётных записей user и, возможно, group (-g group). Если задан аргумент -r, то файлы (базы данных) password/shadow ([fc] /etc/passwd, /etc/shadow [bsd] /etc/passwd/, /etc/master.passwd, /etc/pwd.db and /etc/spwd.db) должны находиться в пределах заданной структуры директорий.



slapdn

Уже совсем скоро (One Day Real Soon Now™)



slapindex

Обновлено для 2.4+. ПЕРЕД ЗАПУСКОМ ОСТАНОВИТЕ SLAPD. **slapindex** используется для пересоздания индексов LDAP на основании текущего содержимого базы данных. Она открывает указанную базу данных, определяемую по номеру базы данных или суффиксу, и обновляет индексы для всех значений всех атрибутов всех записей, индексирование которых настроено в **slapd.conf**.

```
slapindex [-b suffix] [-c] [-d level] [-f slapd.conf]
          [-F confdir] [-g] [-n dbnum] [-o name[=value]]
          [-q] [-t] [-v] [attr] [...]
```

Аргумент	Описание
-b suffix	Использовать указанный суффикс для определения базы данных, для которой требуется сгенерировать вывод. -b не может использоваться совместно с опцией -n.
-c	Включить режим продолжения (игнорирования ошибок).
-d level	Включить отладочные сообщения согласно указанному уровню level.
-f slapd.conf	Определяет местоположение slapd.conf. По умолчанию - [fc] /etc/openldap/slapd.conf или [bsd] /usr/local/etc/openldap/slapd.conf.
-F confdir	Определяет конфигурационную директорию cn=config . Если указаны оба аргумента -f и -F, конфигурационный файл (указанный аргументом -f) будет прочтён и переконвертирован в формат cn=config и записан в

	директорию, указанную аргументом -F. Если не было указано ни одного из аргументов -f и -F, OpenLDAP сначала попытается прочитать конфигурационную директорию по умолчанию ([fc] /etc/openssl/slapd.d, [bsd] /usr/local/etc/openssl/slapd.d) и если таковой не существует, вернётся к использованию конфигурационного файла по умолчанию (slapd.conf). Если существует конфигурационная директория в правильном формате, то конфигурационный файл игнорируется. Если поддерживается и задан режим холостого прогона (аргумент -u, поддерживается большинством утилит, но не slapd), переконвертации не произойдёт.
-g	Отключить подчинённые связи. Будет обработана только указанная база данных, а связанные с ней подчинённые (если они вообще есть) - нет.
-n dbnum	Генерировать вывод для базы данных с порядковым номером dbnum согласно их перечислению в конфигурационном файле. -n не может использоваться совместно с опцией -b.
-o name[=value]	Указывает опции slapd и, при необходимости, их значения. Примеры: <pre>syslog=subsystems (`-s' в slapd) syslog-user=user     (`-l' в slapd)</pre>
-q	Включает быстрый режим (с меньшим количеством проверок целостности). Делается меньше проверок целостности во входных данных, и вообще не делается проверок целостности при записи в базу данных. Сокращает время загрузки, но если во время загрузки возникают какие-либо ошибки или сбои, с получившейся базой данных невозможно будет работать.
-t	Включить режим усеечения (truncate mode). Индексная база данных усекается (очищается) перед началом индексирования записей. Может использоваться только совместно с быстрым режимом (-q).
-v	Включить подробный режим.
attr	По умолчанию индексы строятся на основании настроек в файле slapd.conf, но можно задать один или несколько атрибутов в командной строке.

▲

slappasswd

slappasswd используется для генерации парольных строк с использованием различных алгоритмов. Эти парольные строки могут использоваться в файлах, таких как slapd.conf или LDIF (в значениях атрибутов userPassword или authPassword). Данная утилита может использоваться для создания значения rootpw. О том, как добавить пароль в файл, смотрите в примерах ниже.

```
slappasswd [-v] [-u] [-s secret|-T file] [-h hash] [-c salt-format]
```

Аргумент	Описание
-c salt-format	Определяет формат "соли", используемой при генерации паролей {CRYPT} (DES). Передаваемая с аргументом строка должна быть заключена в кавычки, быть в формате sprintf, и может включать одно (и только одно) преобразование %s. Данное преобразование будет замещено строкой случайных символов из набора [A-Za-z0-9./]. Например, "%.2s" требует предоставления двух символов "соли", а "\$1\$%.8s" сообщает некоторым версиям crypt(3) использовать алгоритм MD5 и предоставлять 8 случайных символов "соли". Значение по умолчанию - "%s", требующее предоставлять 31 символ "соли". Дополнительную информацию можно получить в man-странице crypt на Вашей платформе.
-h hash	Если -h не указан, по умолчанию при генерации userPassword (и authPassword) будет использоваться схема {SSHA}. Если указан -h, аргумент может принимать одно из следующих значений схемы (согласно RFC 2307): {CRYPT}, {MD5}, {SMD5}, {SSHA}, {SHA} и {CLEARTEXT}. Примечание: в зависимости от используемой оболочки может понадобиться экранировать фигурные скобки {}, в которые заключено название схемы. {SHA} и {SSHA} используют алгоритм SHA-1 (FIPS 160-1), последний из них с "солью". {MD5} и {SMD5} используют алгоритм MD5 (RFC 1321), последний из них с "солью". {CRYPT} использует библиотеку crypt(3) для генерации строк DES. {CLEARTEXT} означает, что будет использован пароль в открытом виде (кодирования пользовательского пароля не производится - довольно полезная вещь).

-s secret	Секретная последовательность (secret), которая будет захэширована или закодирована с помощью указанного алгоритма хэширования (-h). Опции -s и -T несовместимы. Если не указано ни одной из опций -s или -T, утилита будет дважды запрашивать секретную последовательность для хэширования, что значительно безопаснее, чем указывать секретную последовательность с помощью опции -s, и безопаснее, чем использовать опцию -T.
-T /path/to/file	Использует полное содержимое файла /path/to/file в качестве секретной последовательности, которая будет захэширована или закодирована с помощью указанного алгоритма хэширования (-h). Опции -s и -T несовместимы. Если не указано ни одной из опций -s или -T, утилита будет дважды запрашивать секретную последовательность для хэширования, что значительно безопаснее, чем указывать секретную последовательность с помощью опции -s, и безопаснее, чем использовать опцию -T.
-u	Генерировать значения для атрибута userPassword, использующегося во многих объектных классах, таких как inetOrgPerson, organization, organizationalUnit, в формате, определённом в RFC 2307 (поведение по умолчанию). Будущие версии могут по умолчанию генерировать альтернативные синтаксисы, и данная опция представлена для совместимости с будущими версиями.
-v	Включить подробный режим.

Примеры slappasswd

Генерация пароля SSHA для использования в качестве rootpw (в slapd.conf), либо для использования в LDIF-файле для атрибутов userPassword или authPassword.

```
# не требуется никаких опций
slappasswd

# дважды запрашивается пароль, затем выводится
{SSHA}kjhfhfhehflejhfvlldkl

# сохранение в файл с помощью стандартного перенаправления
slappasswd > /tmp/slappassword

# генерация {SSHA} хэша от пароля secret
slappasswd -s secret

# генерация {MD5} хэша от пароля secret
slappasswd -s secret -h {MD5}
```

Для того, чтобы поместить вывод в файл LDIF или slapd.conf, сохраните его в другой файл, а затем скопируйте и вставьте его в соответствующий файл, если Вы используете GUI-редактор. Если используется vi, нужно перейти к тому месту в файле, куда требуется вставить пароль, а затем используйте :r !slappasswd [opts] - выполнится команда и её вывод будет вставлен в редактируемый файл в место последней позиции курсора. Альтернативный способ: сохранить вывод slappasswd в файл, в vi перейти к тому месту, куда требуется вставить пароль и выполнить :r /path/to/file - содержимое файла будет вставлено в место последней позиции курсора.



slaptest

slaptest может использоваться для проверки конфигурационного файла slapd.conf, переконвертации файла slapd.conf в OLC (cn=config) и файлов .schema в файлы .ldif для их последующего использования в конфигурации OLC (cn=config). Она открывает указанный конфигурационный файл, проверяет синтаксис директив, в том числе специфичных для конкретных механизмов манипуляции данными и наложений, и выводит результаты проверки. По умолчанию утилита проверяет также базы данных, но эти проверки можно подавить с помощью флага -u. Данная утилита является предпочтительным методом переконвертации из slapd.conf в конфигурацию OLC (cn=config) ([смотрите примеры](#)), однако для этой цели может быть использована любая утилита, поддерживающая флаги -f и -F, например [slapdadd](#).

```
slaptest [-d level] [-f slapd.conf] [-F confdir]
         [-n dbname] [-o name=value] [-Q] [-u] [-v]
```

Аргумент	Описание
-d level	Включает отладочные сообщения согласно указанному уровню level .
-f slapd.conf	Определяет местоположение slapd.conf. По умолчанию - [fc] /etc/openldap/slapd.conf или [bsd]

	/usr/local/etc/openldap/slapd.conf.
-F confdir	Определяет конфигурационную директорию cn=config . Если указаны оба аргумента -f и -F, конфигурационный файл (указанный аргументом -f) будет прочтён и переконvertирован в формат cn=config и записан в директорию, указанную аргументом -F. Если не было указано ни одного из аргументов -f и -F, OpenLDAP сначала попытается прочитать конфигурационную директорию по умолчанию ([fc] /etc/openldap/slapd.d, [bsd] /usr/local/etc/openldap/slapd.d) и если таковой не существует, вернётся к использованию конфигурационного файла по умолчанию (slapd.conf). Если существует конфигурационная директория в правильном формате, то конфигурационный файл игнорируется. Если поддерживается и задан режим холостого прогона (аргумент -u, поддерживается большинством утилит, но не slapd), переконвертации не произойдёт.
-n dbnum	Выполняется нормальная проверка синтаксиса всего файла slapd.conf (включая директивы для конкретных типов баз данных и наложений), после чего выполняется тестирование на наличие и целостность только базы данных с номером dbnum . Базы данных нумеруются последовательно, начиная с 0, в том порядке, в котором они определены в файле slapd.conf, таким образом, для тестирования базы данных, определённой во втором разделе database, используйте -n 1 .
-o name[=value]	Указывает опции/флаги slapd и, при необходимости, их значения. Примеры: <pre>syslog=subsystems (эквивалентно использованию '-s' в slapd) syslog-user=user (эквивалентно использованию '-l' в slapd)</pre>
-Q	Тихий режим. Выводится только окончательный код возврата, показывающий успешное (0) или неуспешное выполнение операции.
-u	Только проверка правильности конфигурации (slapd.conf или slapd.d - смотрите аргументы -f и -F). По умолчанию утилита будет проверять наличие и целостность баз данных, определённых в разделах database конфигурации. Данный флаг подавляет такие проверки. Чтобы проверить только одну конкретную базу данных используйте флаг -n. Кроме того, использование флага -q подавляет переконвертацию файла slapd.conf в cn=config если используются одновременно флаги -f и -F. По существу, при использовании флага -u утилита выполняет только проверочные функции.
-v	Подробный режим. Выводится информация о всех ошибках и сбоях.

Примеры

Простая проверка slapd.conf (или директории slapd.d при её наличии) в стандартном месте расположения, подавляются проверки наличия и целостности всех баз данных .

```
slaptest -u
```

Полная проверка slapd.conf (или директории slapd.d при её наличии) и всех баз данных с выводом подробных сообщений:

```
slaptest -v
```

Полная проверка конфигурационного файла (slapd.conf.test) и всех баз данных с выводом подробных сообщений:

```
slaptest -f slapd.conf.test -v
```

Полная проверка slapd.conf или cn=config, а также проверка наличия и целостности только 0-й базы данных (которой будет база данных cn=config, если в конфигурации указывалось её определение) с выводом подробных сообщений:

```
slaptest -n 0 -v
```

Конвертация slapd.conf в cn=config в стандартном месте расположения директории slapd.d:

```
[fc]slaptest -f /etc/openldap/slapd.conf -F /etc/openldap/slapd.d -u
[bsd]slaptest -f /usr/local/etc/openldap/slapd.conf -F /usr/local/etc/openldap/slapd.d -u
```

Конвертация файла .schema в файл .ldif, пригодный для загрузки в конфигурацию OLC (cn=config):

Это грубый, но несложный процесс детально описывается [здесь](#).



LDAPBrowser / Editor

Существует ряд Open Source [LDAP-браузеров](#). Мы выбрали [LDAPBrowser/Editor](#) без всякого веского основания, за исключением того, что он маленький и легкий, основанный на Java и, следовательно, может запускаться на разных платформах, как, впрочем, и многие другие браузеры. LDAPBrowser/Editor не поддерживается уже несколько лет и доступен только в виде исходного кода для образовательных учреждений - но он прост, функционально адекватен и вполне солиден.

Примечание: Имейте в виду, что Argonne Labs (текущее расположение LDAP Browser/Editor) теперь сделал данное программное обеспечение доступным через OpenChannel Foundation, который взимает плату (в настоящий момент \$35) за каждую копию. Мы оставили нижеследующую информацию здесь для всех исторических пользователей, а также для новых пользователей в случае, если данное программное обеспечение вернется в наш мир. Любой из альтернативных [LDAP-браузеров](#) является вполне разумной заменой.

LDAP Browser/Editor может быть немного причудливым, поэтому следующие заметки выполнены лишь с целью раскрыть некоторые из его наиболее эзотерических характеристик, а не быть руководством пользователя - справочная система браузера необычайно хороша, кроме того, есть несколько полезных HTML-страниц документации. Но... нам потребовались эти заметки при:

[печати и сохранении](#),
[создании и использовании шаблонов](#),
[чтении операционных атрибутов при соединении \(rootDSE, subschema, cn=monitor, cn=config\)](#) и
[доступе к cn=config](#).

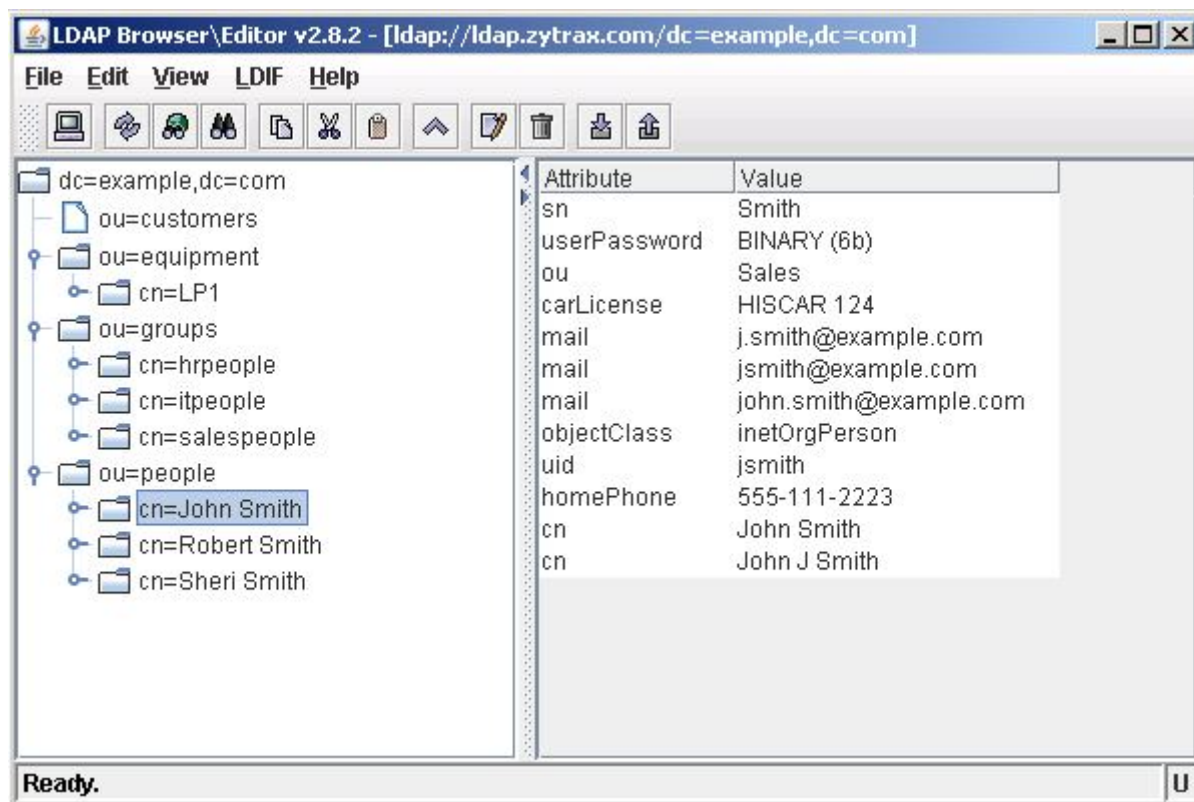
Печать и сохранение

У LDAPBrowser нет функции печати. Если Вам нужно сохранить или распечатать результаты, выберите требуемые записи в правой или левой панели. Кликните по иконке **copy**, а затем вставьте в подходящую утилиту, такую как Notepad на Windows или kedit в KDE, либо подобную текстовую утилиту. Затем редактируйте, сохраняйте или печатайте, как Вам нужно.

Создание шаблонов

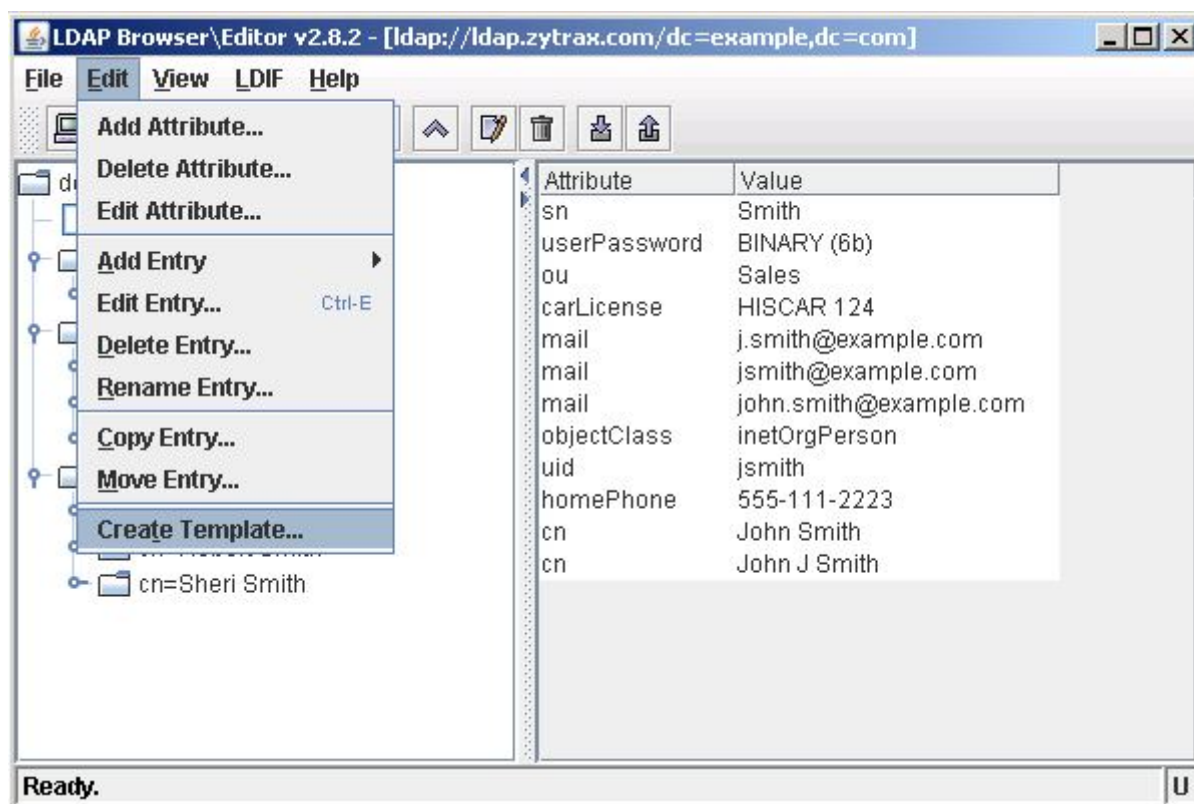
Система шаблонов очень гибкая, но в ней нужно разбираться. Чтобы создать шаблон, выполните следующие действия:

1. Выберите любую запись в текущем DIT, использующую тот же объектный класс, что и шаблон, который Вы хотите создать - в этом примере мы выбрали запись, использующую inetOrgPerson:

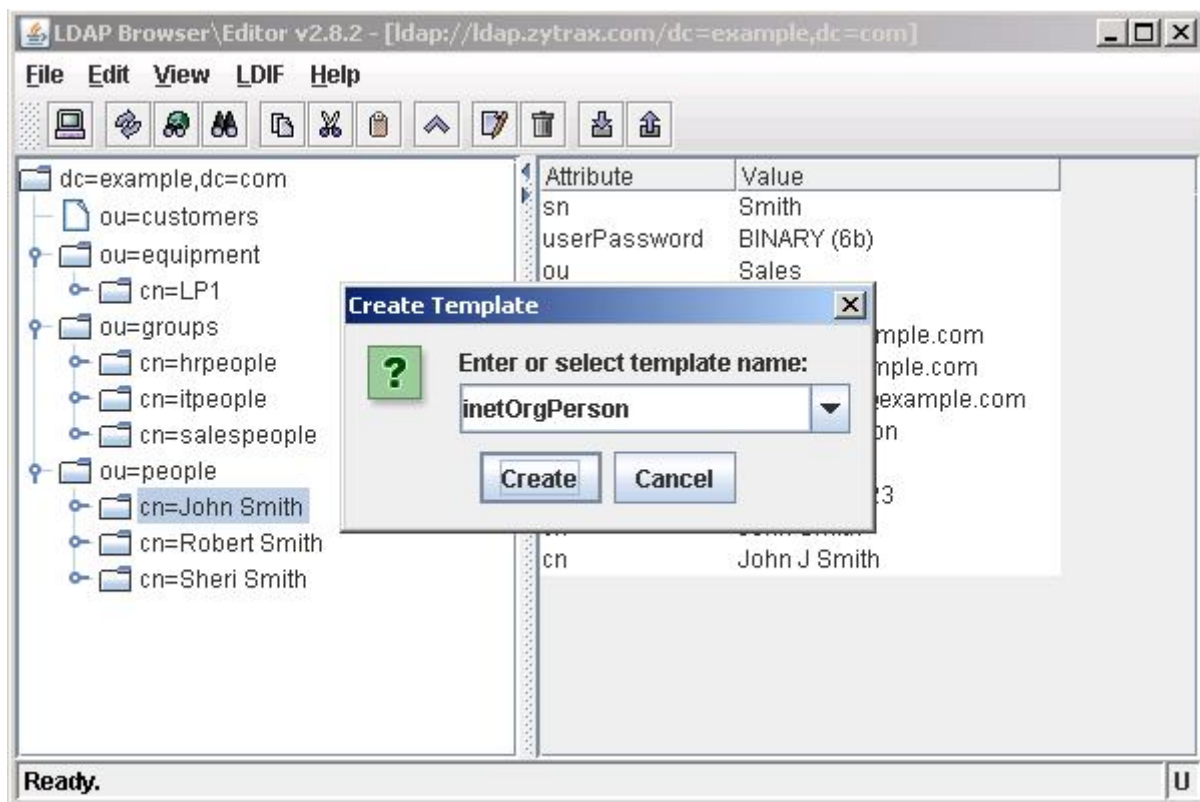


Подсказка: Выбирайте запись с максимальным количеством атрибутов, поскольку шаблон сохраняет только атрибуты той записи, на которой он основан (в шаге 7 дан метод ручного редактирования шаблона).

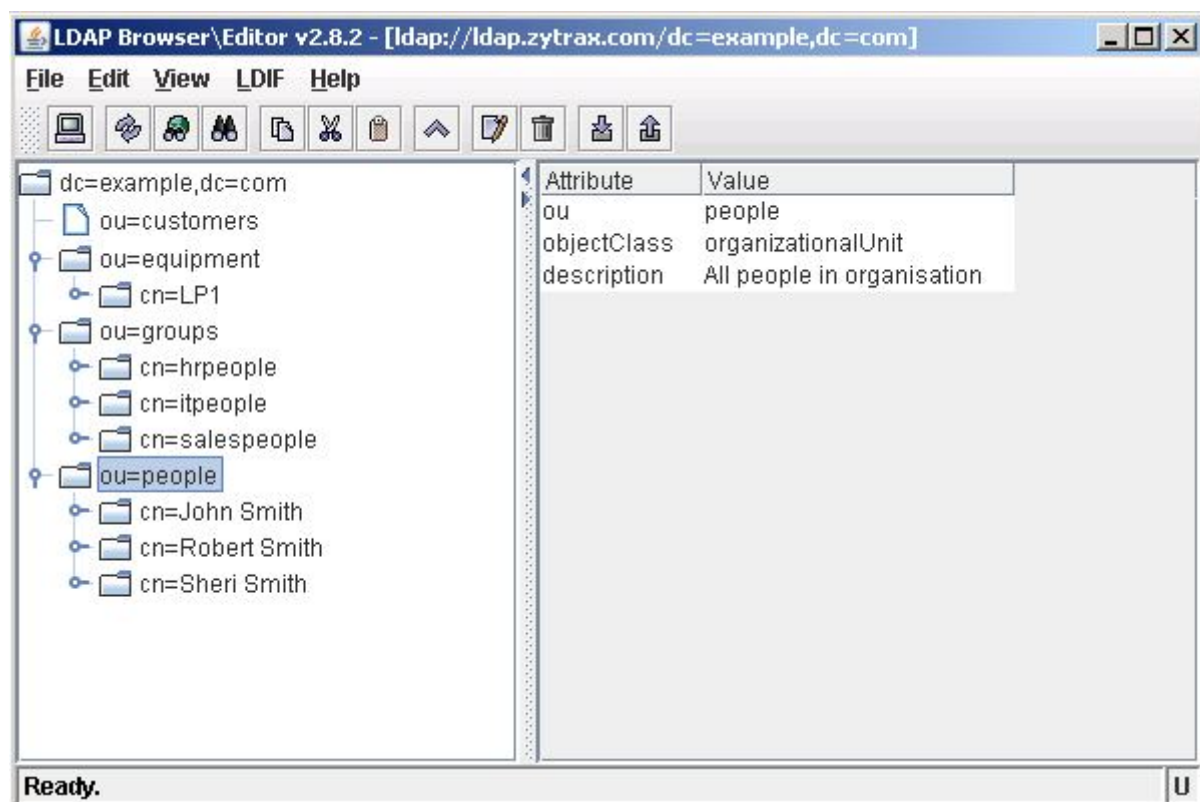
2. Выберите меню **Edit**, пункт **Create Template** :



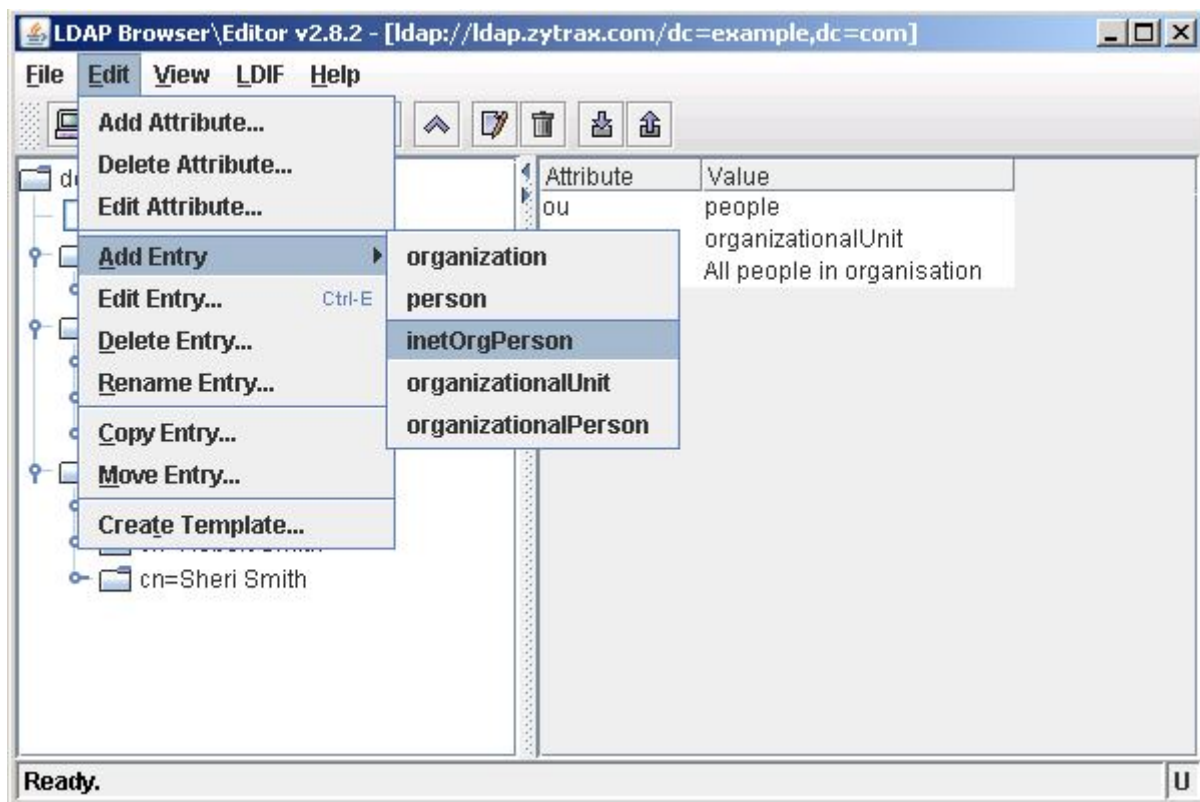
3. LDAPBrowser предложит имя шаблона на основании имени объектного класса - соглашайтесь или отредактируйте его по своему желанию, затем нажмите **Create** :



4. Чтобы использовать шаблон, выберите в DIT запись, в которую Вы хотите добавить новую дочернюю запись, в данном примере мы будем добавлять в ou=people:



5. Выберите меню **Edit**, пункт **Add Entry**, а затем выберите шаблон, на основании которого Вы хотите создать новую запись - в данном примере мы используем только что созданный шаблон inetOrgPerson:



6. Откроется окно, содержащее все атрибуты из сохранённого шаблона. Заполните необходимые поля, затем выберите **Apply** :

7. Шаблоны сохраняют объектный класс и атрибуты записи, на которой они основаны. Если Вам нужно добавить дополнительные атрибуты, можете использовать меню **Edit, Add Attribute...**, либо отредактировать файл шаблона. Файл шаблона для inetOrgPerson показан ниже (сохранён в templates/template-name.template в директории приложения LDAPBrowser). Просто добавьте имена требуемых атрибутов под заголовки Required или Optional (LDAPBrowser не проверяет атрибуты Required):

```
# name : inetOrgPerson
#
# objectClass : inetOrgPerson
#

PREFIX: cn

REQUIRED ATTRIBUTES

OPTIONAL ATTRIBUTES
sn
userPassword
ou
carLicense
mail
uid
```

```
homePhone
cn
description
```

Операционные атрибуты при соединении

При доступе к rootDSE, subSchema, cn=monitor и особенно cn=config, при соединении Вам может понадобиться читать пользовательские и операционные атрибуты - по умолчанию LDAPBrowser отображает только пользовательские атрибуты. Мы не нашли другого способа, кроме использования функции сессии (файл .cfg) для получения операционных данных (конечно, Вы всегда можете использовать [ldapsearch](#)).

Следующие три файла .cfg позволяют отображать rootDSE, subschema (все пользовательские и операционные атрибуты) и subSchema (только коллекция объектных классов) в общем случае для localhost. Они сохраняются в директорию приложения LDAPBrowser и будут отображаться при загрузке приложения или при выборе File->Connect. Можно отредактировать нужные Вам поля, используя обычный текстовый редактор, сохранить файл, а затем использовать его для Вашего сервера, а можно воспользоваться функцией LDAPBrowser [session edit](#). Первый файл .cfg file (показан ниже) просто отобразит значения rootDSE ([просмотреть файл .cfg отдельно](#)):

```
#####
#
# LDAP Browser v2.8 config file #
#
# rootDSE
#
#####

# хост ldap-сервера - измените
# или используйте функцию edit в LDAPBrowser
host=localhost

# порт ldap-сервера
port=389

# ssl-порт ldap-сервера (если есть)
sslport=636

# base dn для rootDSE пуст
basedn=

# настройка номера версии ldap [2|3]
version=3

# dn менеджера каталога (rootDSE is anonymous read)
managerdn=

# пароль менеджера каталога (N/A)
password=

# подключаться от учётной записи менеджера [yes|no]
managerlogin=no

# должно ли соединение устанавливаться
# автоматически [yes|no]
autoconnect=no

# тип поддерживаемого индикатора
# листовых узлов [int | boolean ]
leafindicatorortype=int

# имя индикатора листовых узлов
# корректный атрибут - обычно операционный атрибут
# примечание: работает только с серверами LdapV3!
leafindicator=numsordinates

# максимальное количество результатов
# 0 или не определено - без ограничений
# limit=0

# размер пакета
# 0 или не определено - все записи возвращаются сразу
# batchsize=0

# таймаут операций ldap в миллисекундах
# 0 или не определено - таймаут не установлен
# timeout=0

# управление отсылками [yes|no]
```

```

# при установке в по отсылки возвращаются как нормальные записи
managereferrals=no

# установите в yes если Ваш ldap-сервер
# поддерживает перемещение дерева в DIT
supportsmovetree=no

# опция 'dereference alias' (разыменование псевдонимов)
# [never|alwya|search|find]
derefaliases=never

# сохранять или удалять старый rdn
# при переименовании записи
deleteolddn=yes

# порядок сортировки дерева DIT
# если не определено, сортировка не выполняется
# иначе сортируется по возрастанию или убыванию [ascending|descending]
sorttree=ascending

# используется для исправления положения окон
# на экране. связано с ошибками java
# fixlocation = 10

# размер буфера окна ошибок
# (в байтах), по умолчанию 2048
# logsize=2048

# показывать окно ошибок при возникновении
# ошибки. [yes|no], по умолчанию: no
# poperrorwindow=no

# ldap-фильтр для построения древовидной
# структуры. по умолчанию: (objectclass=*)
# ldap.list.filter=(objectclass=*)

# список атрибутов, получаемых при каждом чтении
# полезно для указания операционных атрибутов
# (разделённых пробелами). данная строка используется
# для получения всех (пользовательских и операционных) атрибутов
ldap.attributes.list=* +

#####
# Настройки БЕЗОПАСНОСТИ #
#####

# настройка защищённых протоколов
# варианты [ssl|gssapi?]
# security.protocol=ssl

# настройка аутентификации
# варианты [simple|external]
# security.authentication=simple

# настройка socket factory
# варианты javax.net.ssl.SSLSocketFactory
# ldapsocketfactory=javax.net.ssl.SSLSocketFactory

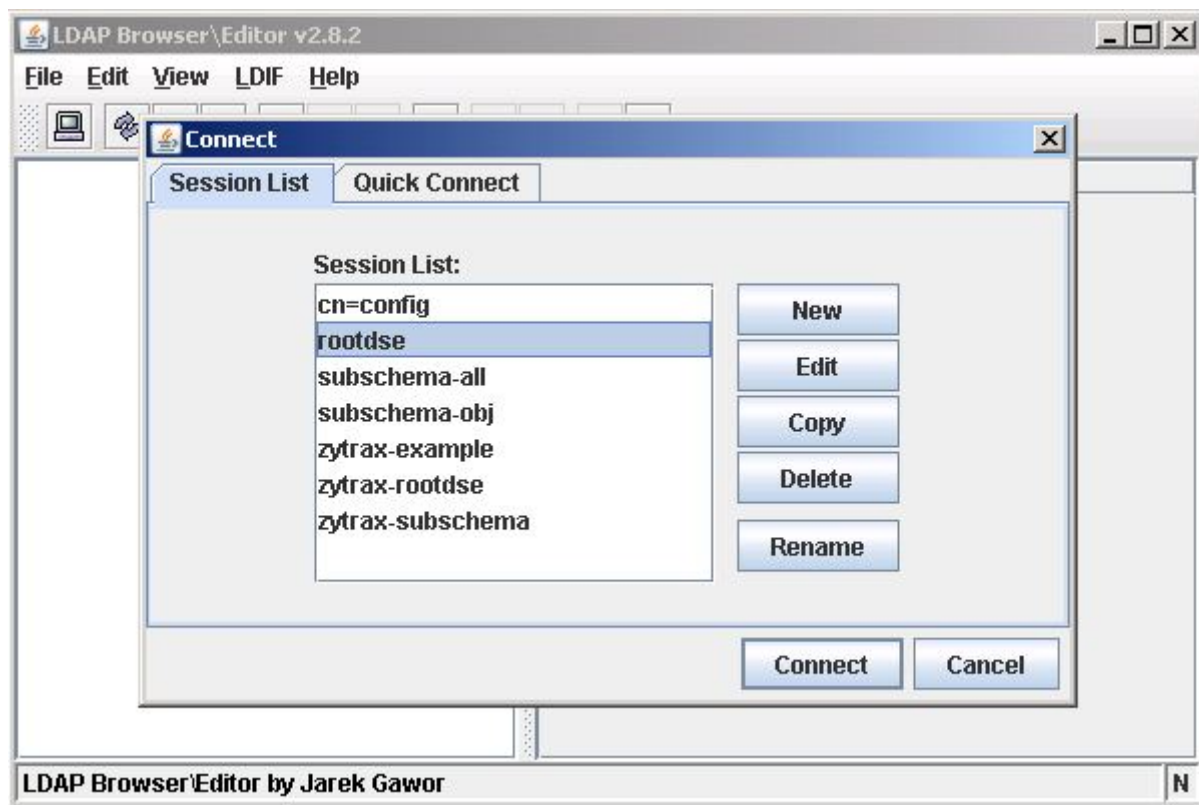
```

Аналогичный процесс используется для отображения всех операционных атрибутов в subschema ([посмотреть файл .cfg](#)). Для иллюстрации использования дополнительных опций мы приводим файл, который настраивает программу на отображение только коллекции объектных классов в subschema ([посмотреть файл .cfg](#)).

Мы используем аналогичную настройку сессии для доступа к операционным атрибутам `cn=config` ([дополнительная информация и настройка cn=config](#)), это просто необходимо, поскольку все интересные данные содержатся в операционных атрибутах. Посмотрите файлы `cn=config.cfg` и `cn=monitor.cfg`, затем можете сохранить их, используя функцию 'save as' Вашего браузера, и отредактировать под свои нужды с помощью любимого текстового редактора.

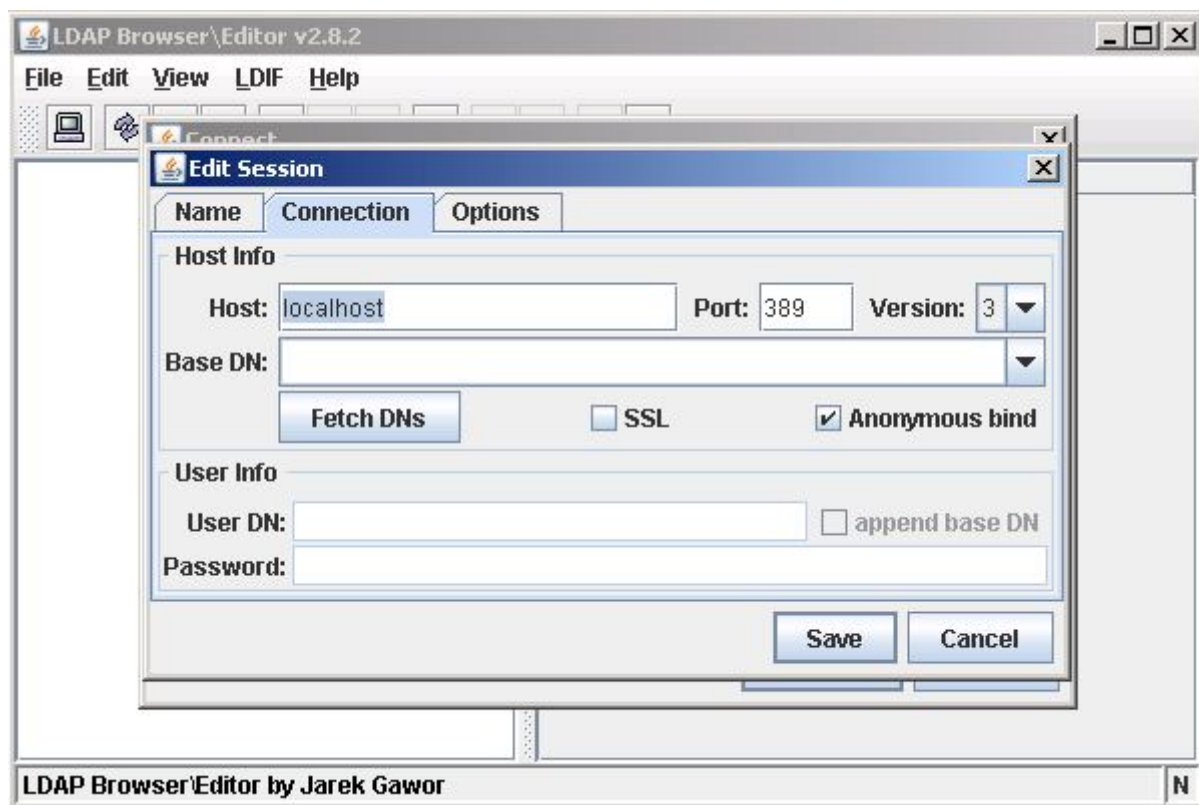
Если Вы сохранили приведенные выше файлы .cfg без редактирования, их можно изменить во время работы, используя следующую процедуру:

1. При загрузке LDAPBrowser/Editor отображаются текущие конфигурационные файлы сессий, как показано:



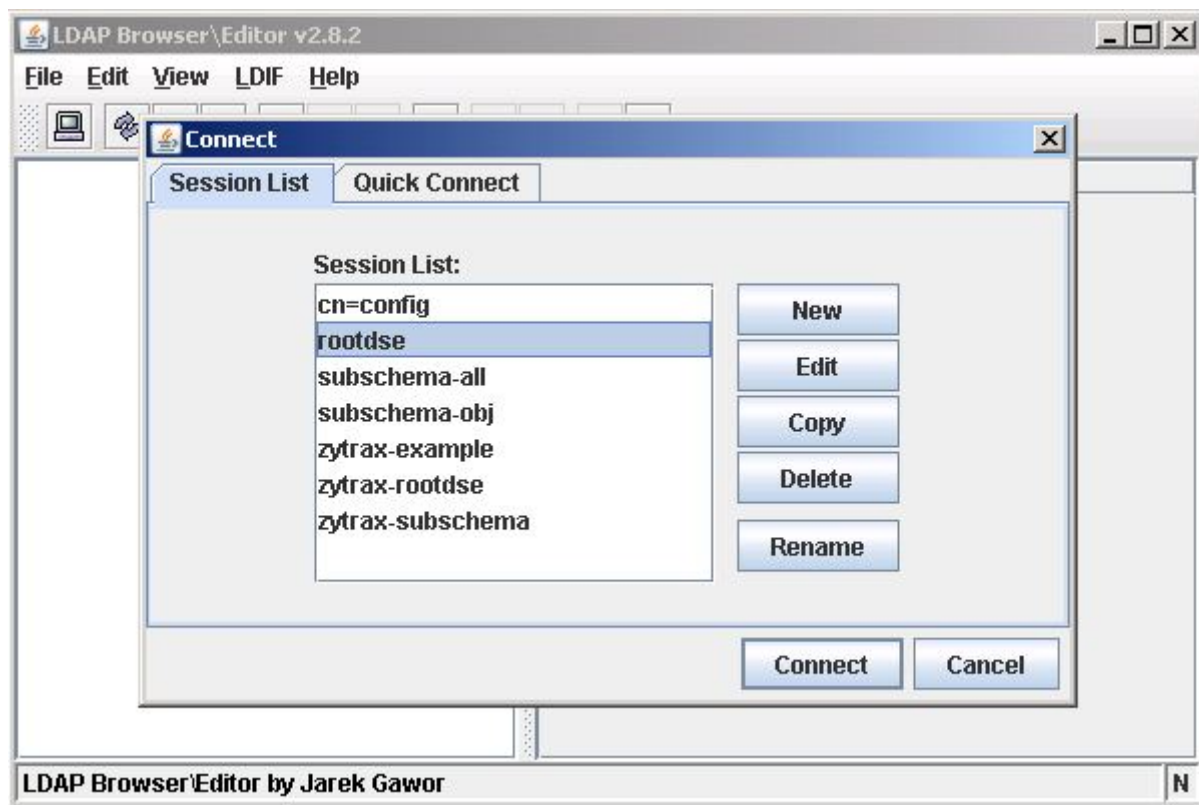
Выберите требуемую сессию и нажмите кнопку **Edit**.

2. Отредактируйте поле **Host** и другие необходимые переменные:

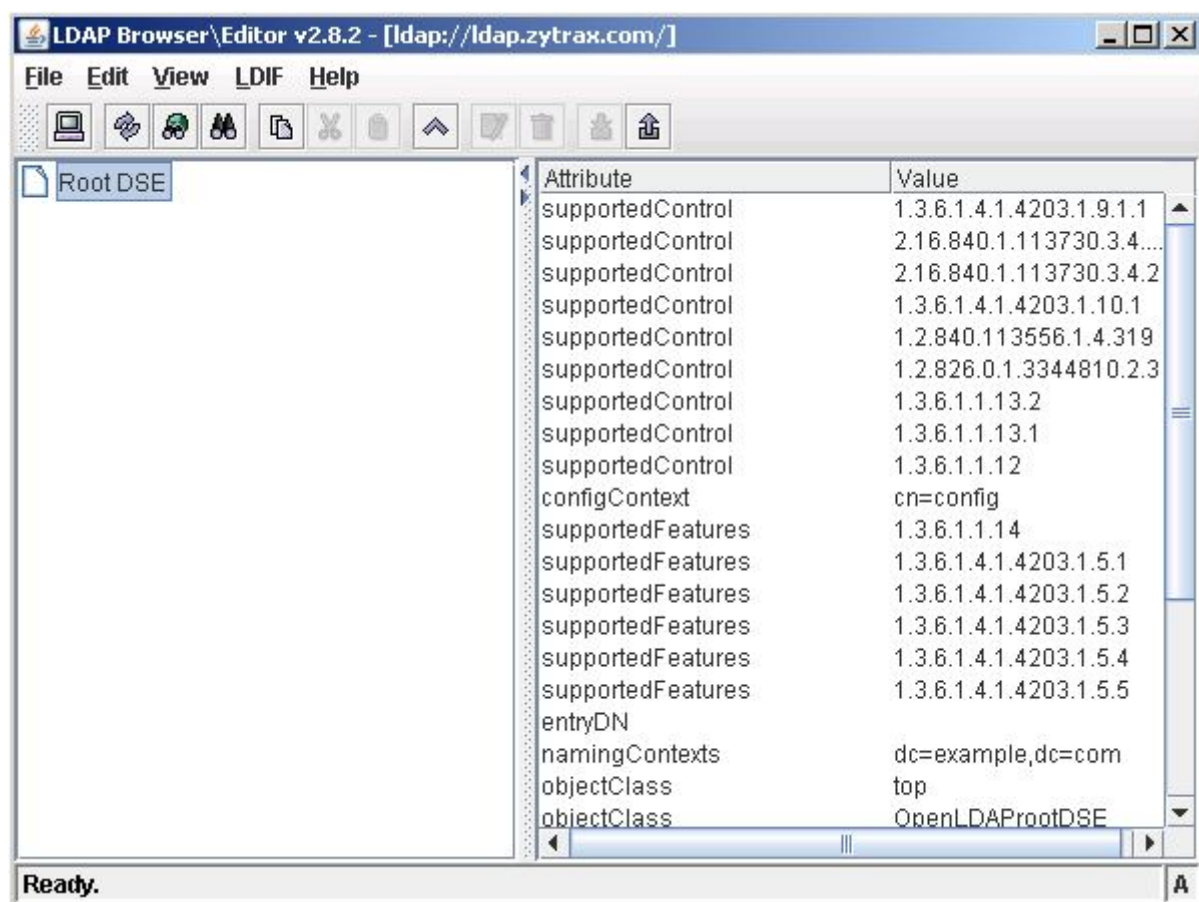


Нажмите кнопку **Save**.

3. Нажмите кнопку **Connect**:



4. При подсоединении LDAPBrowser/Editor отображает RootDSE в левой панели. Дважды кликните по этой записи, чтобы отобразить атрибуты в правой панели, как показано:



Для распечатки и сохранения переменных [используйте описанную выше процедуру](#).



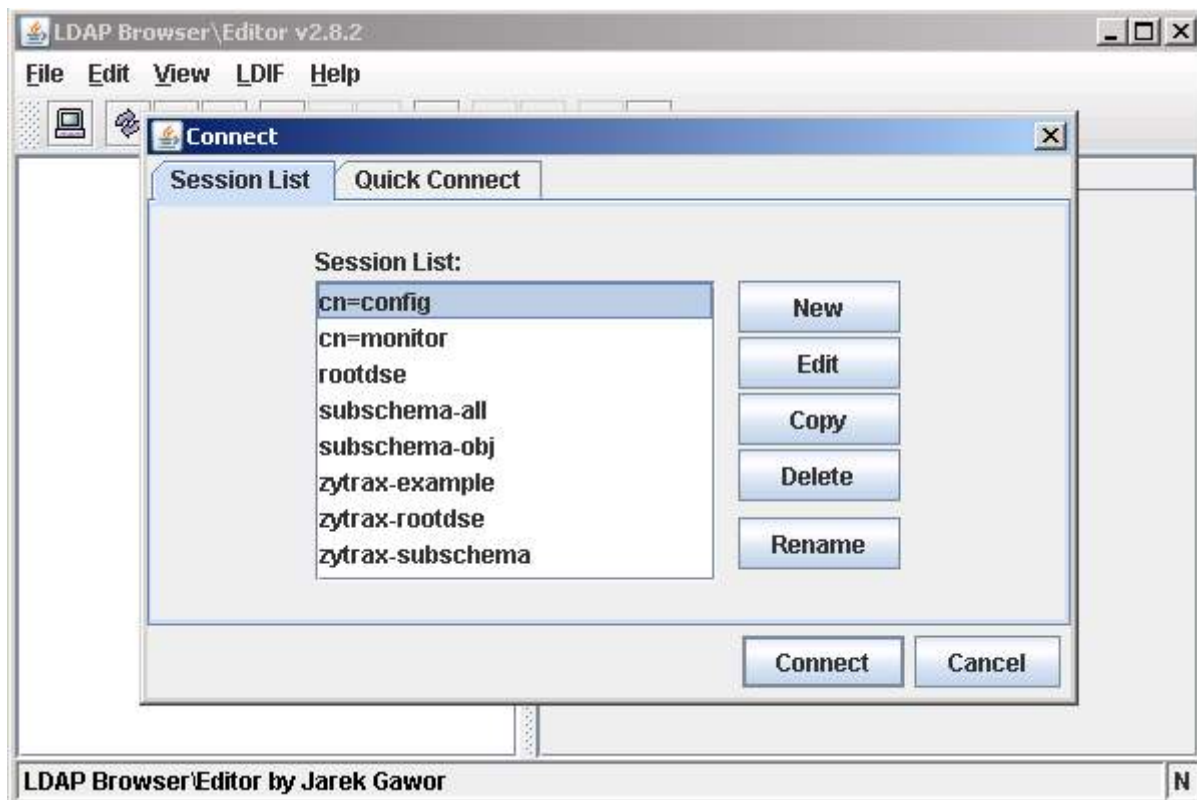
Доступ к cn=config

Предполагается, что Вы создали файл сессии для cn=config с помощью приведённой выше [процедуры](#). Затем,

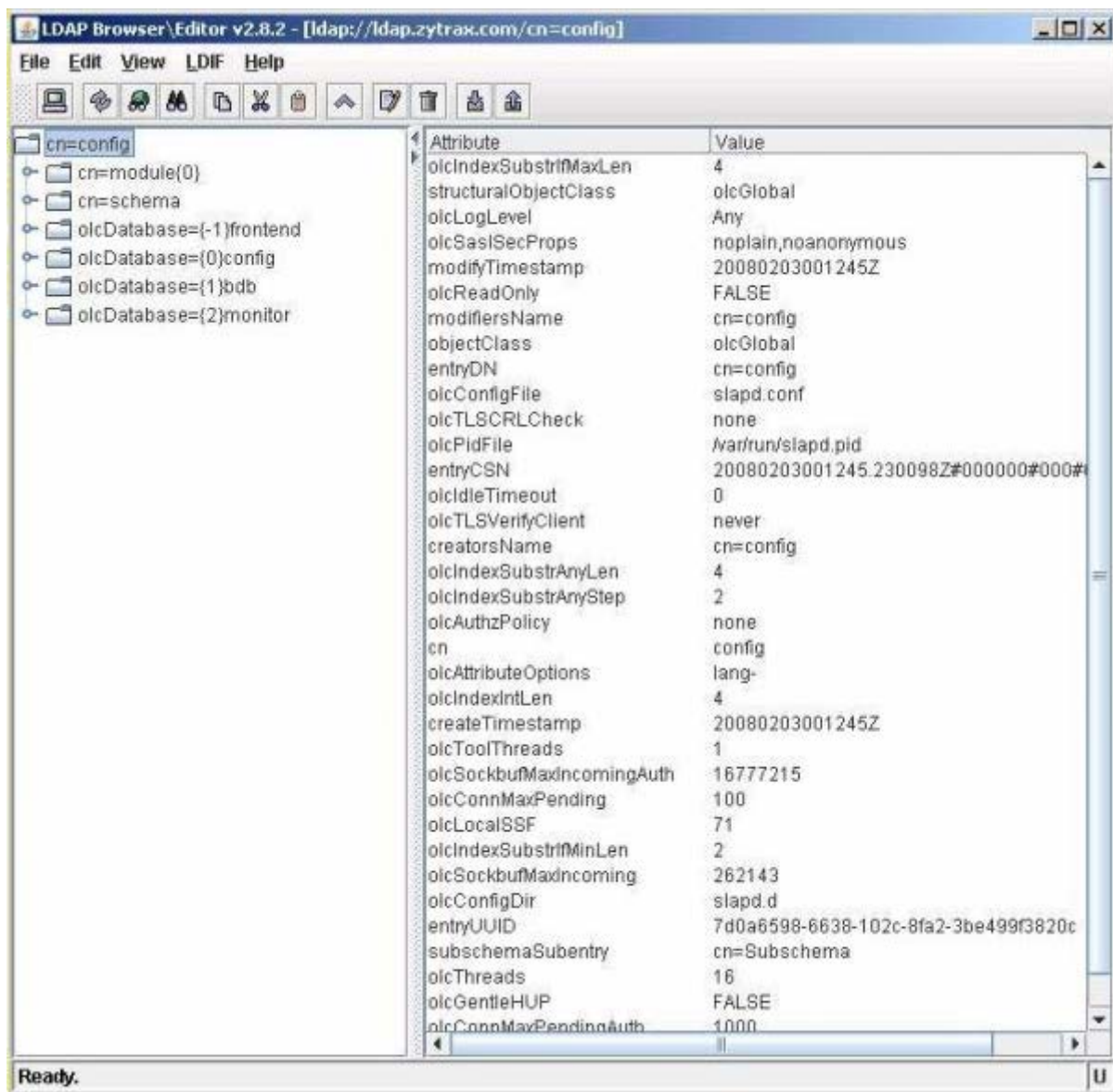
для отображения и использования `cn=config` DIT, выполняйте следующие действия:

Примечание: Мы получили письмо, в котором говорилось, что, при попытке разумного ограничения доступа к базе данных `cn=config` (которая, кроме всего прочего, может быть объектом атак) с помощью комплексных ACL в некоторых "некоробочных" установках LDAP, доступ к `cn=config` утрачивался. Если описанная ниже процедура не работает, Вам либо придётся использовать `ldapsearch`, либо внимательно читать документацию по установке чтобы выяснить, что же произошло.

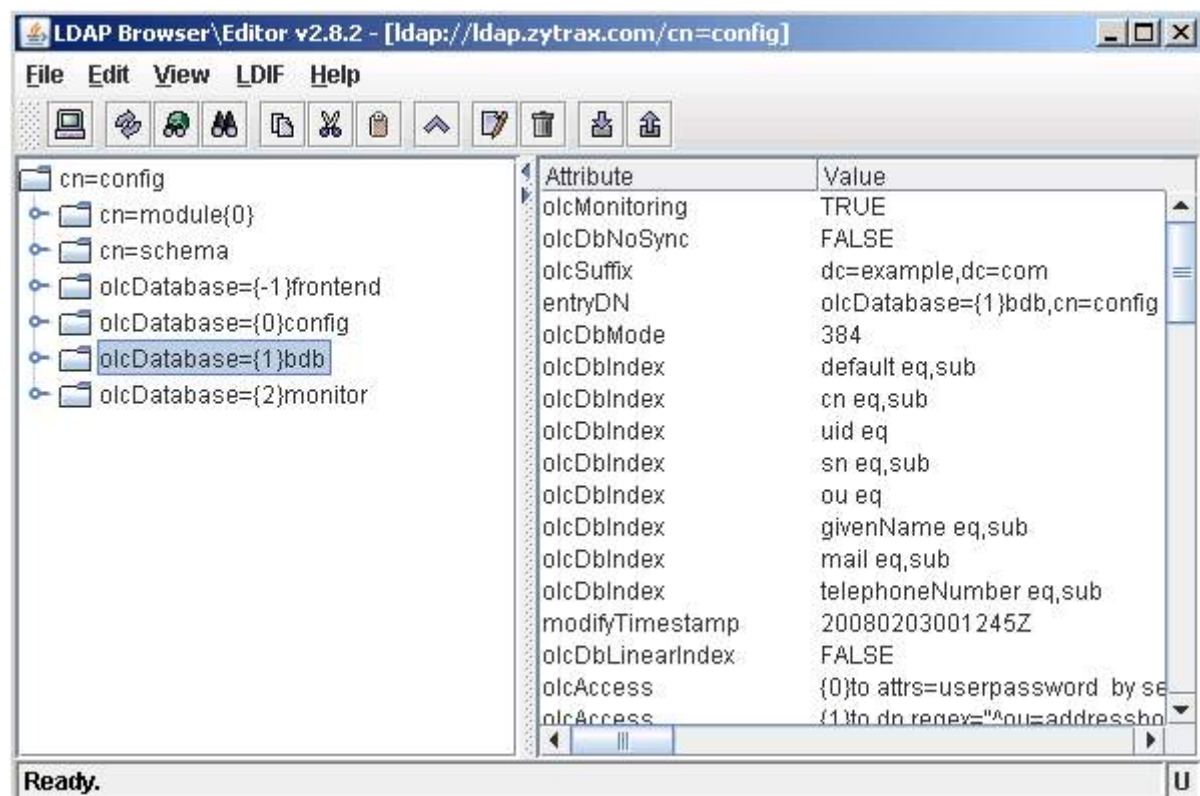
1. Выберите сессию `cn=config` и нажмите **Connect**.



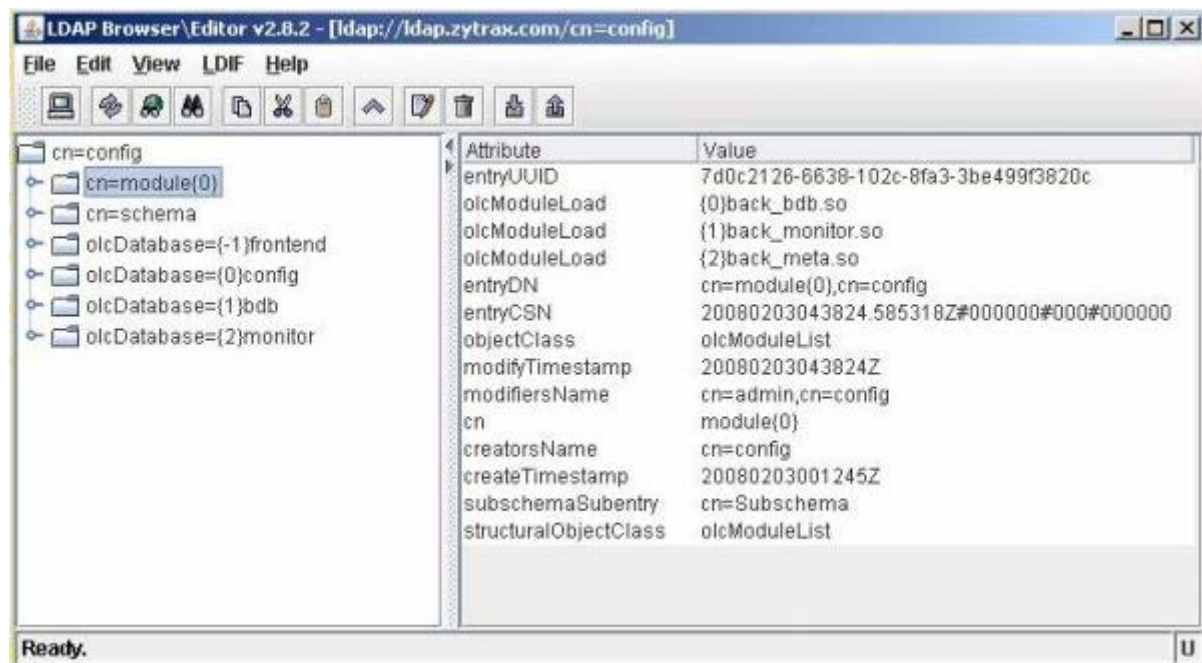
2. Начальная запись `cn=config` отображает секцию глобальных настроек старого файла `slapd.conf`, за исключением `ModuleLoad` (в поддереве `cn=module`) и подключаемых схем (в поддереве `cn=schema`).



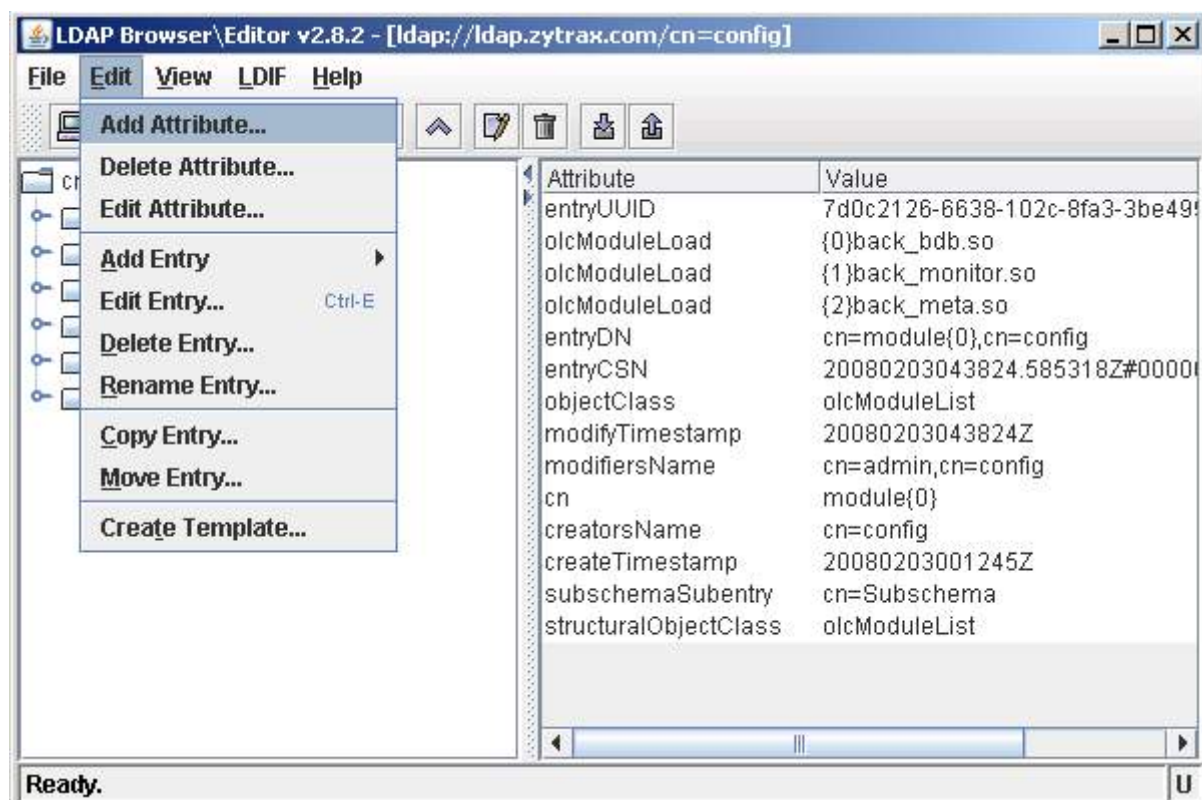
3. Запись `olcDatabase={1}bdb` отображает секцию настроек одного экземпляра базы данных bdb.



4. Запись `cn=module` отображает загруженные в настоящий момент модули - в данном случае `back_bdb`, `back_monitor` и `back_meta`. OpenLDAP автоматически выделяет для каждого механизма манипуляции данными уникальный индекс, нумеруя их от `{0}`, например, `{1}back_monitor.so` в приведённом ниже скриншоте. Для каждого типа механизма манипуляции данными определяется один экземпляр модуля, и, если он включен, то может использоваться для создания нескольких экземпляров баз данных.

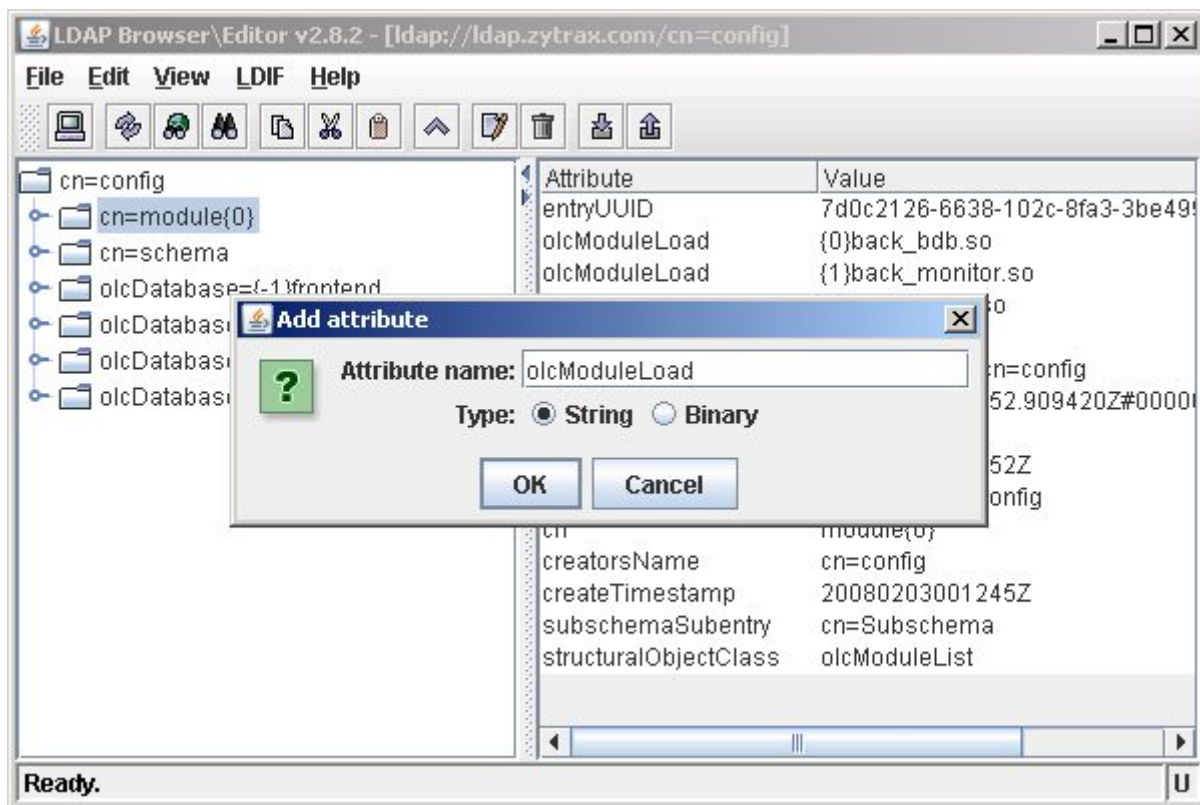


5. Для иллюстрации использования возможностей `cn=config` мы добавим новый механизм манипуляции данными. Выберите **Edit** и **Add Attribute...**

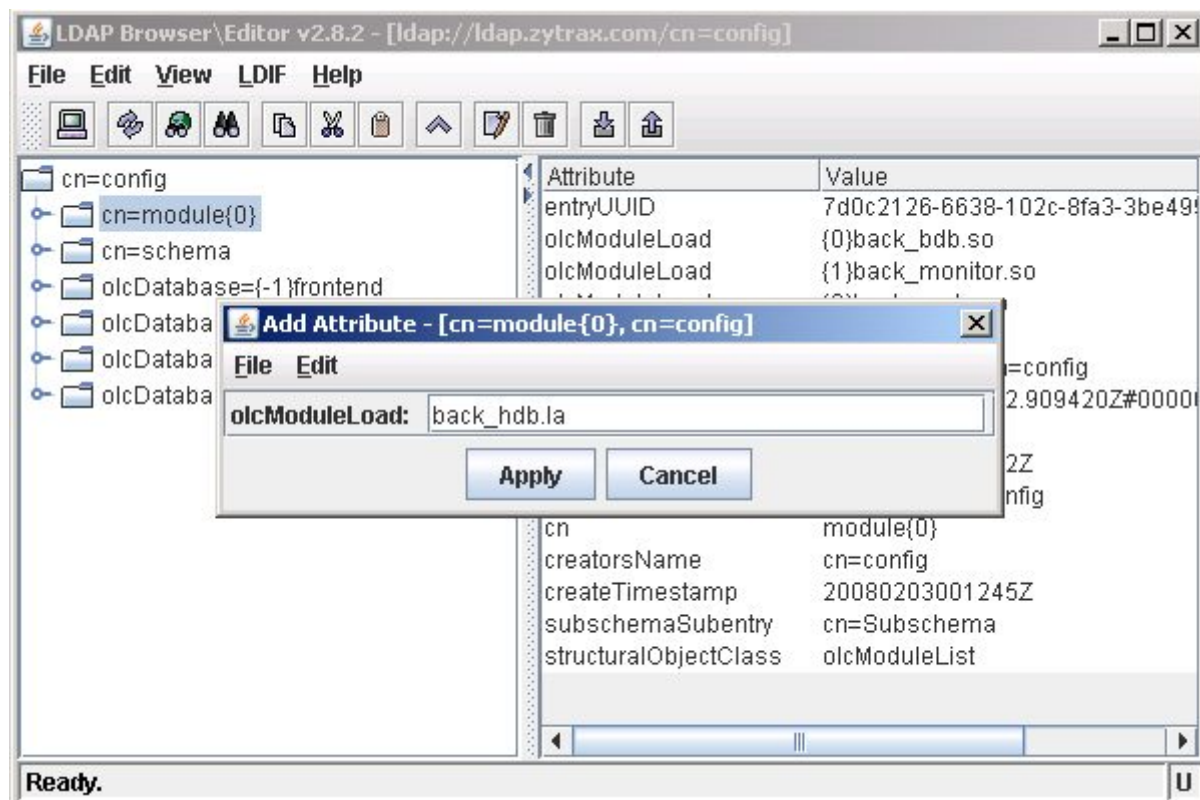


Прежде чем определить базу данных, Вы должны добавить механизм манипуляции данными, которому она будет принадлежать.

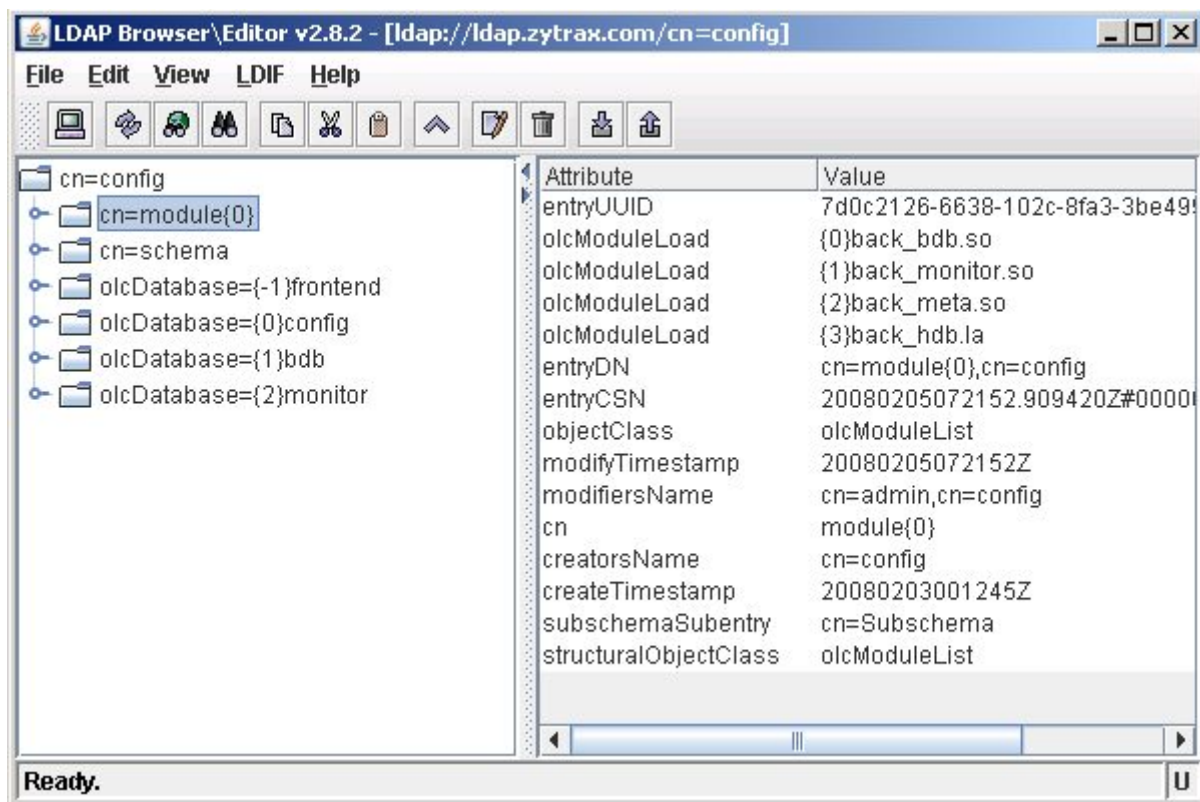
6. Вам будет предложено ввести имя того **атрибута**, который Вы хотите добавить, в данном случае **olcModuleLoad**. Выберите соответствующий тип атрибута, в данном случае **string**. Нажмите **OK**.



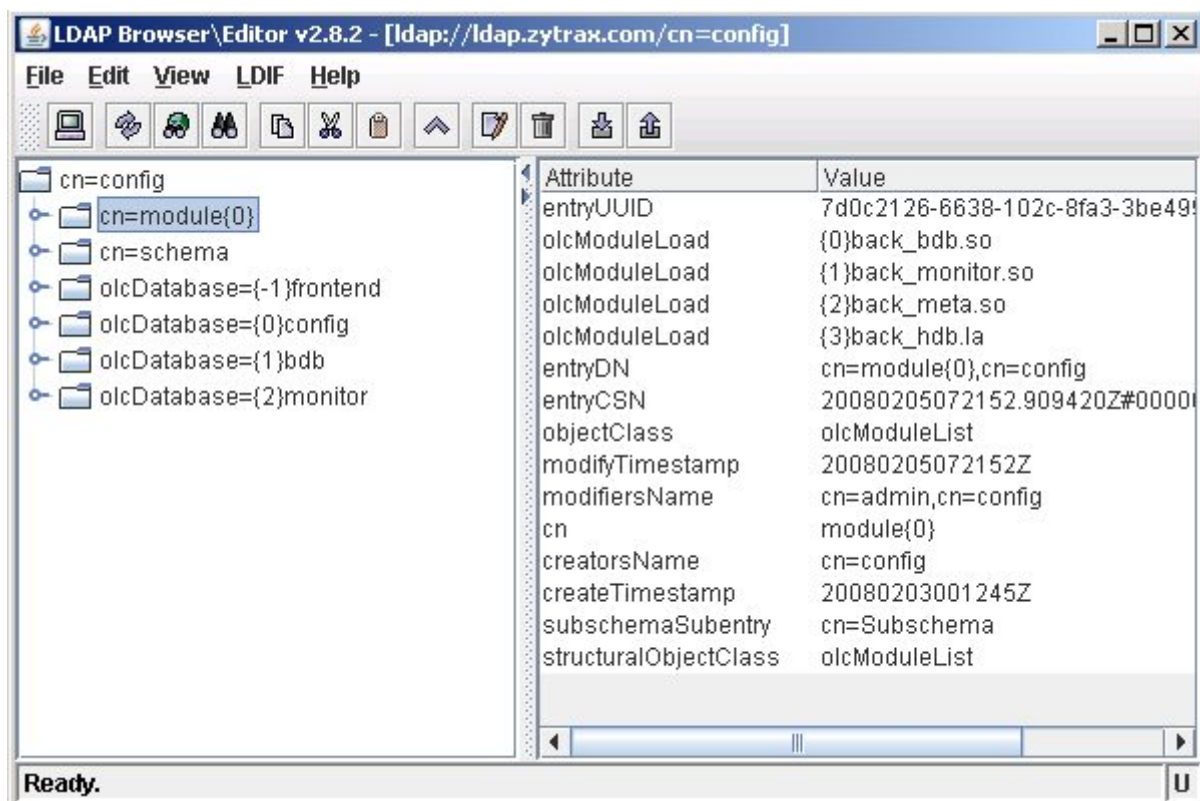
7. Затем будет предложено ввести значение атрибута, имя которого Вы вводили на предыдущем экране. Просто введите название механизма манипуляции данными (в данном случае back_hdb.la) - OpenLDAP сам выделит уникальный индекс при добавлении атрибута. Нажмите **Apply**.



8. На данном скриншоте показано, что атрибут был успешно добавлен. OpenLDAP добавил back_hdb.la в список атрибутов olcModuleLoad и назначил ему следующий свободный уникальный индекс {3}back_hdb.la.



9. Следующая последовательность скриншотов показывает добавление нового экземпляра базы данных (секция database на жаргоне slapd.conf). Создайте [шаблон](#) из существующего экземпляра требуемого типа базы данных - в данном случае мы будем использовать bdb - поэтому мы создаём шаблон из olcDatabase={1}bdb. Требуемые атрибуты olcBdbConfig описаны [здесь](#).



Инструменты ApacheDS



Глава 15. Безопасность LDAP

[Обзор безопасности LDAP](#)
[Настройка SASL в OpenLDAP](#)
[Настройка SASL - TLS в OpenLDAP](#)
[Настройка TLS в OpenLDAP](#)
[Настройка смешанного доступа SSL/TLS в OpenLDAP](#)

Обзор безопасности LDAP

У LDAP, и особенно у OpenLDAP, есть ряд возможностей обеспечения безопасности, которые на первый (да и на второй и третий) взгляд могут показаться немного сложными. Диаграмма 15-1 показывает общую картину проблемы, перед тем как углубиться в детали. Далее демонстрируются различные методы доступа и интерфейсы к LDAP-системе, а затем описываются проблемные вопросы безопасности и доступные методы управления рисками. Цель данного упражнения - либо определить с набором политик безопасности, либо выделить приоритеты реализации.

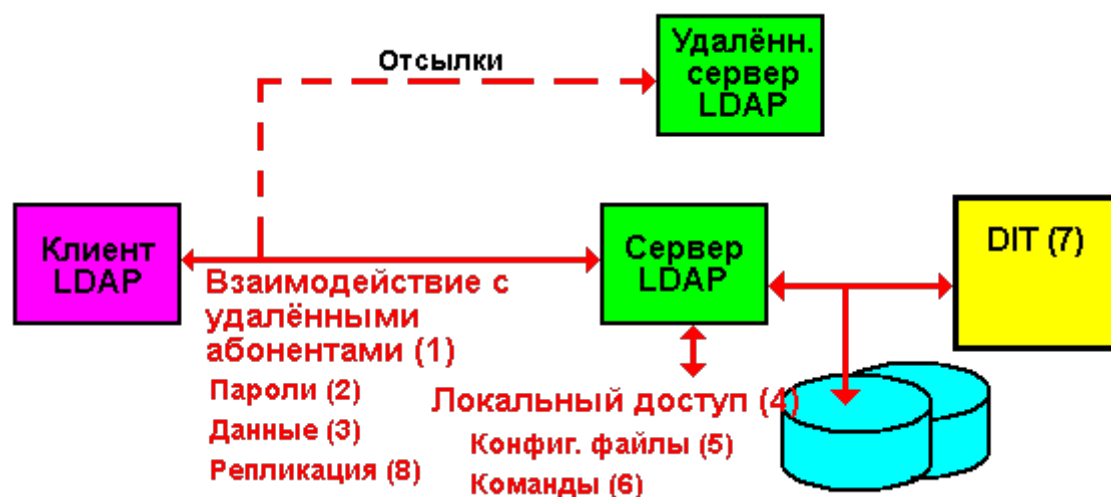


Рисунок 15-1. Общая картина вопросов обеспечения безопасности

Все номера, встречающиеся в описании ниже, ссылаются на рисунок 15-1.

1. **Взаимодействие с удалёнными абонентами (1):** Обеспечение безопасности при взаимодействии с удалёнными абонентами может как быть, так и не быть проблемным вопросом. При предоставлении неограниченного (анонимного) доступа к неконфиденциальной LDAP-информации вопросы обеспечения безопасности не ставятся. **Внимание:** в этих условиях Ваш сервер всё равно становится потенциальным объектом DoS/DDoS-атак посредством нагрузки его вредоносными LDAP-запросами. Таким образом, даже такая на первый взгляд тривиальная реализация может потребовать тщательного проектирования.

Если все взаимодействия с LDAP гарантированно происходят в доверенной сети, то Вы можете остановить свой выбор на работе с простыми паролями в открытом виде без дополнительных заморочек с безопасностью. Однако, даже в таких случаях, в зависимости от топологии доверенной сети, бывает достаточно просто перехватывать трафик и получать либо конфиденциальные данные (3), либо пароли (2), посылаемые в открытом виде.

Когда взаимодействие с LDAP осуществляется через ненадёжные сети, у деструктивных личностей в сети сразу находится, чем заняться: будьте готовы к тому, что перехваты, отслеживания, атаки типа "человек посередине" ("man-in-the-middle") и т.п. будут постоянным явлением. Также имейте в виду, что всё возрастающее количество каталогов с функциями конфигурации времени исполнения (RTC, sp=config) и мониторинга (sp=monitor) может привести к тому, что LDAP-браузеры станут обычным инструментом вроде удалённой консоли для администрирования LDAP-серверов. А этот трафик по своей природе невероятно конфиденциален.

Если решено, что некоторый уровень обеспечения безопасности всё же необходим, то первый вопрос, возникающий в этом случае: нужно ли нам защищать только пароли (2), или нужно защищать и данные (3), и пароли (2)? В зависимости от ответа на этот вопрос будут определяться наши следующие шаги.

2. **Пароли (2):** Не следует путать защиту паролей во время передачи по сети с защитой их в конфигурационных файлах или в DIT. Даже если Вы защитили все пароли в конфигурационных файлах или в DIT с помощью методов хэширования, таких как {SSHA}, при отправке пароля клиентом серверу он посылается в открытом виде, хэшируется на сервере и сравнивается с хранимым значением. Без принятия дополнительных мер он, таким образом, может быть перехвачен (или отслежен, в зависимости от ваших предпочтений к подобным терминам). **Примечание:** Когда клиент запрашивает запись, содержащую, скажем, атрибут **userPassword**, значение которого захэшировано, допустим, по алгоритму {CRYPT}, то этот пароль посылается не в открытом виде, а в своей хэшированной форме (то есть в той, в которой он хранится). Однако, когда осуществляется доступ от имени этой же самой записи, клиент посылает пароль (с помощью которого он проходит аутентификацию) в открытом виде, и, естественно, если аутентификация прошла успешно, перехватывающий может вполне резонно предположить, что посланный в открытом виде пароль был верен!

При пересылке хэшированных паролей в потоке данных, который может подвергнуться перехвату, они становятся уязвимыми к атакам по словарю - получив хэшированный пароль, атакующий прогоняет список паролей (так называемый словарь) через алгоритм хэширования, пока не найдёт совпадения. Использование **соли** (одного или нескольких байт, - в зависимости от реализации, - добавляемых к паролю перед хэшированием и отбрасываемых перед сравнением) значительно повышает безопасность хэшированных паролей, и, если у Вас нет веских причин к обратному, нужно всегда использовать форму того или иного алгоритма хэширования с **солью**. Нужно также использовать **ACL** для предоставления доступа к паролям только определённому кругу авторизованных пользователей. Например, подразумевая, что пароли хранятся в атрибуте **userPassword**, следующий ACL будет разрешать доступ к данному атрибуту только владельцу записи и определённой группе пользователей (admin):

```
access to attrs=userpassword
  by self write
  by anonymous auth
  by group.exact="cn=admin,ou=groups,dc=example,dc=com" write
  by * none
```

Если требуется защитить только пароли, то решением может стать использование SASL с каким-либо алгоритмом (например CRAM-MD5), обеспечивающим выполнение безопасного диалога подключения (с помощью общей секретной последовательности), во время которого пароли никогда не передаются в открытом виде ([примеры конфигурации SASL](#)). Альтернативой может стать использование SSL/TLS (с или без SASL) или Kerberos 5/4. В этом случае можно использовать простые механизмы паролей, поскольку весь обмен по сети шифруется, и потому не может быть подвержен перехвату.

Наконец, [наложение ppolicy](#) позволяет контролировать устаревание, сложность, обязательный сброс, а также другие характеристики используемых паролей.

3. **Данные (3):** Если данные, поступающие от LDAP-сервера, требуется защитить от перехватов, единственным решением становится шифрование всего потока данных с помощью SSL/TLS ([с SASL](#) или [без SASL](#)) или Kerberos (SASL). Обратная сторона такого подхода заключается в том, что шифрование интенсивно использует процессор, и если потребление ресурсов и производительность являются основными критериями, то выбор подходящего метода шифрования из доступных в пакете SSL/TLS становится очень ответственным решением (смотрите [конфигурирование SSL/TLS](#) и [конфигурирование SSL/TLS через SASL](#)). Существует возможность смешивать и сочетать различные методы при передаче информации по сети. Например, приемлемой может оказаться такая ситуация: для обычного удалённого доступа к LDAP используются простые пароли в открытом виде (или даже анонимный доступ), но для некоторых классов пользователей требуются дополнительные меры защиты (примеры конфигурации для смешанного доступа SSL/TLS и SASL смотрите [здесь](#)).

До сих пор мы обсуждали вопросы только простого получения доступа к данным. А как насчёт изменения или модификации этих данных? OpenLDAP предоставляет две возможности для ведения аудита: наложение **auditlog** (для получения дополнительной информации смотрите map-страницу slapo-auditlog) и наложение **accesslog**. У обоих есть функции журналирования изменений, вносимых в рабочее DIT, а у accesslog даже есть возможность помещать в журнал информацию о подключениях, доступах для чтения/поиска, а также предыдущее содержимое записей или атрибутов.

4. **Локальный доступ (4):** Под локальным доступом подразумеваются любые события, происходящие непосредственно на LDAP-сервере или кластере серверов (либо посредством защищённого удалённого доступа к серверу с помощью, например, ssh). Наиболее очевидно, под эту категорию попадают манипуляции с конфигурационными файлами (5) и локально выполняемые команды (6).
5. **Конфигурационные файлы (5):** Здесь мы должны рассмотреть два компонента:
1. **Владельцы и права:** По умолчанию, современные системы LDAP работают от имени учетных записей пользователя/группы с ограниченными привилегиями (обычно ldap:ldap). OpenLDAP проделывает замечательную штуку со своим конфигурационным файлом slapd.conf, который может содержать весьма важные данные. Он загружается с правами root (чтобы открыть привилегированные порты), а затем очень быстро переходит к работе со своими нормальными (низкими) рабочими привилегиями. Перед тем, как понизить привилегии, он убеждается (и, в случае необходимости приводит в соответствие), что slapd.conf принадлежит ldap:ldap (или тем user:group, от имени которых он запущен) и ему назначены права 0600. Таким образом предотвращается получение случайного доступа к файлу.
 2. **Пароли:** Пароли, встречающиеся в конфигурационных файлах и директориях slapd.d (при использовании cn=config), как правило, относятся к категории особо конфиденциальных (например, rootpw). Можно порекомендовать полностью удалить их после того, как DIT было запущено в эксплуатацию, либо, на крайний случай, хранить их в хэшированной форме для предотвращения случайной компрометации. К сожалению, использование подключаемых файлов с очень жёсткими правами (такими, как только чтение для root) для хранения наиболее важных данных, применимое во многих других системах, не дает никаких преимуществ, поскольку OpenLDAP перед обработкой инструкций из файла slapd.conf уже снижает свои привилегии до рабочих. Однако, как было сказано выше, у OpenLDAP есть замечательный механизм модификации прав на конфигурационный файл, поэтому манипуляции с выносом паролей в отдельный файл становятся излишними.
6. **Команды (6):** Исторически LDAP администрировался через локальный интерфейс, в основном из командной строки. Предполагалось, что локальный (внутрисерверный) трафик не подвержен отслеживанию, и потому даже применение простых паролей в открытом виде считалось адекватной мерой защиты для большинства административных сервисов. Однако, как отмечалось выше, увеличение количества реализаций каталогов с конфигурацией времени исполнения (RTC, cn=config) и мониторингом (cn=monitor) может означать, что удалённые LDAP-браузеры становятся нормой в вопросах администрирования LDAP-систем. В этом случае доступ к указанным сервисам порождает передачу по сети информации высокой степени важности, которую необходимо защищать с помощью специальных технологий обеспечения безопасности данных, таких как SSL/TLS.
- Наконец, поскольку при конфигурации времени исполнения все настройки хранятся в DIT (cn=config), стоит рассмотреть использование мощных возможностей [наложения accesslog](#) - инструмента генерации журнала для аудита изменений, вносимых в данное DIT.
7. **DIT (7):** Безопасность DIT определяется моделью безопасности LDAP и реализуется исключительно при помощи директивы [access to](#) файла slapd.conf или атрибута **olcAccess** при использовании [конфигурации времени исполнения \(RTC, cn=config\)](#).
8. **Репликация (8):** Даже если обычный клиентский доступ к LDAP-системе не требует серьёзных мер безопасности, иная ситуация может быть при [репликации](#) того же DIT. Во время цикла репликации на том или ином этапе все данные в DIT (пользовательские и операционные атрибуты) будут передаваться по сети. Вероятно, часть этой информации будет весьма важной. Сетевое взаимодействие при репликации заслуживает отдельного рассмотрения. Вы можете настроить смешанное соединение SSL/TLS и другого защищённого (или даже незащищённого) типа к одному серверу (смотрите примеры конфигурации смешанного доступа SSL/TLS и SASL [здесь](#)).

Настройка SASL в OpenLDAP

Мы закончим этот раздел уже совсем скоро (one day real soon now ™)

Настройка SASL SSL/TLS в OpenLDAP

В руководстве администратора OpenLDAP утверждается, что при использовании TLS с механизмом EXTERNAL SASL и клиенту, и серверу необходимо иметь действительный сертификат X.509. Если это правда, то мы получаем излишне сложную конфигурацию, а уровень безопасности повышается незначительно, и потому в большинстве случаев её использование представляется сомнительным (но есть заметные исключения, такие как репликация). В настоящее время подробно обсуждать это мы не будем.



Настройка SSL/TLS в OpenLDAP

Обзор

TLS (Transport Level Security) - это просто название стандартизированной IETF версии Secure Sockets Layer (SSL) от Netscape. Практически нет никаких различий между SSL(v3) и TLS(v1), и в этой документации данные термины считаются синонимами. TLS/SSL поддерживается OpenLDAP начиная с версии 2.1.

Обычно для работы TLS/SSL требуется сертификат X.509 (более известный как сертификат SSL), который можно приобрести в коммерческом центре сертификации или удостоверяющем центре (Certificate Authority, CA), либо это может быть самоподписанный сертификат. В этой документации дано пошаговое руководство по созданию самоподписанных сертификатов, а также, в случае необходимости, приводятся дополнительные замечания по использованию коммерческих сертификатов.

[Сессия TLS/SSL начинается](#) с переговоров о выборе набора шифров, включающего в себя шифр обмена ключами, шифр для шифрования данных и алгоритм MAC (Message Authentication Code, представляет собой хэш). Шифр для шифрования данных или массового шифрования (который всегда является симметричным шифром, использующим вычисленный ключ сессии) и MAC используются при передаче данных и потребляют значительно меньше ресурсов CPU, чем шифр обмена ключами (асимметричный шифр, например RSA или DSA/DSS), применение которого ограничивается посылкой ключа pre-master (на основании которого вычисляется ключ сессии) в начале обмена.

При использовании с контекстом TLS термин аутентификация указывает только на аутентификацию сервера (с помощью сертификата X.509), его не следует путать с аутентификацией LDAP, используемой для контроля доступа к DIT. При защите сессии с помощью TLS/SSL аутентификация LDAP с использованием **простого (simple)** механизма подсоединения становится полностью безопасной, поскольку, не смотря на то, что пароли отправляются открытым текстом, они инкапсулируются в зашифрованный поток данных, таким образом, при пересылке как по доверенным, так и по недоверенным сетям, перехват (прослушивание) данных становится невозможным.

Приводимые ниже конфигурации не дают исчерпывающего описания всех возможностей использования TLS с LDAP. Скорее, они представляют собой те варианты конфигурации, которые охватывают наиболее часто встречающиеся ситуации, и, надеемся, поддаются относительно простому внедрению даже со скромным уровнем знаний в области TLS/SSL и сертификатов X.509. Однако, если пользовательские требования выдвигают какие-то специфические условия, могут понадобиться более глубокие знания. Тогда Вам может быть полезно [это руководство по выживанию](#), дающее жуткие подробности протокола TLS/SSL, форматов сертификатов X.509 и генерации самоподписанных сертификатов для использования в различных условиях.

Процесс инициации сессии TLS может происходить одним из двух способов:

1. **Автоматически:** Если клиент использует LDAP URL в форме **ldaps://hostname/**, то диалог TLS иницируется автоматически (обе стороны ожидают элементов протокола TLS/SSL) на порту по умолчанию для **ldaps** 636. Порт по умолчанию для **ldaps** может быть изменён на альтернативный путём использования URL в форме **ldaps://hostname:port/**, при этом сервер должен быть настроен на приём соединений на этом альтернативном порту (с помощью параметра **-h** при запуске **slapd**). Чтобы окончательно всех запутать, можно даже указать в качестве альтернативного **ldaps** порта 389-й (то есть нормальный порт **ldap**), при этом даже не придётся перенастраивать фаервол. Даже в этом случае диалог TLS будет по-прежнему иницироваться **автоматически** благодаря использованию схемы **ldaps** в URL.
2. **Согласно определению:** LDAP также может быть настроен на приём соединений TLS/SSL на

стандартном порту LDAP при использовании URL **ldap://hostname** (порт по умолчанию - 389, но это можно изменить). В этом случае клиент должен быть явно запрограммирован (или ему должно быть указано) использовать последовательность StartTLS (сообщение с запросом расширенной операции с OID элемента управления StartTLS 1.3.6.1.4.1.1466.20037). Например, при использовании [syncrepl](#) с TLS и URL в форме **ldap://hostname/** ДОЛЖЕН быть задан параметр **starttls=yes** или **starttls=critical** (если же используется URL в форме **ldaps://hostname/**, задавать его не обязательно).

Хотя OpenLDAP, в соответствии со своей обычной философией, позволяет производить множество разнообразных вариантов настройки, все описанные ниже конфигурации TLS схожи по характеристикам и используют лишь некоторый набор из доступных опций настройки. Отчасти это сделано для того, чтобы уменьшить сложность этих конфигураций, отчасти - чтобы сосредоточиться на наиболее распространенных стратегиях использования. Требования системной политики таковы:

1. Для вызова защищённого соединения в настройках клиента TLS должна всегда использоваться схема **ldaps**. Это упрощает настройку клиента и даёт понять пользователям, что они получают доступ к защищённому серверу, аналогично тому, что при доступе к защищённому web-сайту используется схема **https**.
2. Серверы TLS должны иметь сертификаты X.509. Предполагается, что у клиентов TLS, напротив, НИКОГДА НЕТ сертификатов X.509, то есть директива [TLSVerifyClient](#) всегда имеет значение по умолчанию **never**. Существуют обстоятельства, например, конфигурация потребителя репликации в стиле [syncrepl](#) (или поставщика при репликации с несколькими главными серверами), когда взаимная аутентификация может быть необходима, однако в нашей политике подразумевается, что это лишь исключение, а не общее требование. После того, как конфигурация TLS-сервера полностью отлажена и он находится в рабочем состоянии, относительно просто добавить взаимную TLS-аутентификацию, тем более, что пользователь уже будет хорошо знаком с основными механизмами безопасности. Если же начинать с такой конфигурации, пользователь просто столкнётся с множеством ошибок, разочарований, метаний из стороны в сторону, либо погрязнет в бесконечном количестве новой информации. Лучше наращивать сложность шаг за шагом.
3. Всегда используются только самоподписанные сертификаты (и процесс их генерации подробно документирован). Это сделано по двум причинам: во-первых, самоподписанные сертификаты имеют неограниченное значение в процессе тестирования, так как они ничего не стоят и предоставляют дополнительные знания о использовании и функциональности X.509; во-вторых, преимущество от приобретения и использования с LDAP коммерческих сертификатов **на данном этапе** невелико. В этом отношении есть отличие от web-серверов, клиенты которых (браузеры) изначально оснащены несколькими коммерческими CA (корневыми) сертификатами, а также подсветкой адресной строки (жёлтой для сертификатов SSL или зелёной для сертификатов EV), подтверждающей подлинность сайта, что позволяет пользователю чувствовать себя комфортно, и потому может принести коммерческую выгоду. В случае же с LDAP мы пока не нашли ни одного клиентского дистрибутива с предустановленными коммерческими CA (корневыми) сертификатами. Для установки в LDAP-клиент CA (корневого) сертификата потребуются предпринять некоторые усилия, при этом нет никакой разницы по сложности установки коммерческого или самоподписанного сертификата - различается только их стоимость. Тем не менее, многие пользователи предпочтут установить коммерческие сертификаты X.509, и потому, если имеются различия в установке, мы их обязательно опишем.
4. Для аутентификации и обмена ключами используются только асимметричные алгоритмы RSA (Rivest, Shamir и Adleman). В целом, работать с ними значительно проще. Хотя в некоторых случаях может потребоваться применение алгоритмов DSA (Digital Signature Architecture), например, в правительственных интересах, в настоящее время они используются не так широко, как RSA. Вопросы применения DSA будут рассматриваться в разделе примечаний в конце описания каждой конфигурации.

В первой из представленных конфигураций подразумевается, что LDAP-сервер будет принимать только TLS-соединения с использованием схемы **ldaps**. Все остальные попытки доступа к серверу будут отклоняться. В ещё двух [представленных конфигурациях](#) сервер поддерживает смешанный TLS (**ldaps**) и не-TLS трафик LDAP.

Настройка защищённого (только TLS соединения) сервера (и клиентов) OpenLDAP

Настройка сервера TLS

LDAP-серверу требуется принимать только защищённые TLS-соединения от клиентов с проверкой подлинности LDAP. Поддерживается единственное пользовательское DIT с включённой службой репликации - то есть данный сервер использует наложение [syncprov](#), кроме того, требуется доступ к сервисам cn=monitor и cn=config. Во всех случаях используются самоподписанные сертификаты (с соответствующими замечаниями по использованию коммерческих сертификатов).

1. **Генерация сертификатов LDAP-сервера и CA:** На данном этапе будет сгенерирован самоподписанный сертификат - если будет использоваться коммерческий сертификат X.509 (SSL), данный процесс не требуется и может быть полностью пропущен. Для генерации единого сертификата X.509, который может применяться и для проверки подлинности сервера, и в качестве CA (корневого) сертификата для клиента, используется простой метод в одну команду. Данный метод детально (со всеми диалогами) документирован [здесь](#). Последовательность команд:

```
# создаём новые директории (опционально)
mkdir /certs
mkdir /certs/keys
cd /certs
# создаём сертификат сервера/CA и закрытый ключ без парольной фразы
# действительный в течение 10 лет, использующий текущие рекомендации RSA по размеру ключа
# RSA используется в качестве протокола обмена ключами
openssl req -x509 -nodes -days 3650 -newkey rsa:2048
-keyout keys/ldapskey.pem -out ldapscert.pem

# сертификат может использоваться в качестве сертификата сервера или сертификата CA
# помещаем сертификат в:
/certs/ldapscert.pem
# помещаем закрытый ключ в:
/certs/keys/ldapskey.pem

# устанавливаем права доступа
chown -R ldap:ldap /certs/*
chmod 0400 keys/ldapskey.pem
```

Примечания:

1. Полное объяснение параметров команды openssl смотрите [здесь](#).
2. У файлов ключей обычно есть парольная фраза (пароль) для защиты конфиденциальных данных. Если сертификат генерируется для сервера, подобная парольная фраза никогда не используется (её генерация подавляется аргументом -nodes).
3. В зависимости от требований, предъявляемых к сертификату, могут использоваться другие методы генерации самоподписанных сертификатов, описанные [здесь](#).

2. **Добавление директив TLS в slapd.conf** (замечания по cn=config смотрите в конце раздела).
Необходимые директивы TLS добавляются в раздел глобальных настроек:

```
# slapd.conf
# раздел глобальных настроек
...
# Безопасность - раздел TLS
TLSCertificateFile /certs/ldapscert.pem
TLSCertificateKeyFile /certs/keys/ldapskey.pem
TLSCipherSuite TLSv1+RSA:!NULL
# значение следующей директивы установлено так по умолчанию,
# однако мы приводим её здесь для наглядности
TLSVerifyClient never
...
# пользовательское DIT
database bdb
suffix "d=example,dc=com"
...
# поставщик репликации
overlay syncprov

# cn=config DIT
database config
rootdn="cn=admin,cn=config"
rootpw ....

# cn=monitor DIT
database monitor
rootdn="cn=admin,cn=monitor"
rootpw ....
```


Примечания:

1. Дополнительная информация по TLS cipher suite [здесь](#). Используемые параметры исключают применение NULL-шифров (без шифрования). TLSv1 охватывает SSLv3. Разрешены EXPORT-шифры - допускаются международные подключения. Если вопросы производительности и нагрузки стоят остро, лучше явно задать список шифров с приемлемыми характеристиками производительности и загрузки системы, чем оставлять возможность произвольного выбора шифра во время переговоров TLS/SSL.
 2. **Конфигурация DSA:** скоро будет.
 3. **cn=config:** скоро будет.
3. **Блокировка сервера:** (замечания по cn=config смотрите в конце раздела). Согласно выдвинутым требованиям, данный сервер будет поддерживать только соединения TLS. По этой стратегии сервер не будет ожидать подключений для стандартного трафика с LDAP URL (порт 389), ожидаются только подключения с LDAPS URL. Кроме того, вводится пара простых правил безопасности, требующих обязательного выполнения простого подсоединения (bind) при подключении и предотвращающих анонимное подключение. Косвенный эффект такой политики - запрет анонимных подключений с LDAP URL к rootDSE и некоторые другие интересные вещи. Для высокозащищённых серверов это, вероятно, оправданный шаг. Для достижения аналогичных целей можно было бы использовать директивы [access to](#), однако, поскольку ACL игнорируются при подключении от имени [rootdn](#) (например, в случае с cn=config и cn=monitor), эффект от их применения ограничен. Данный метод применяется в конфигурации смешанного доступа и описан [здесь](#). Наконец, при запуске LDAP-сервера ему указывается принимать соединения только на порту по умолчанию для LDAPS URL 636 (смотрите [примечания ниже](#)). Итак, начнём. Во-первых, внесём дополнительные настройки в slapd.conf (добавляются директивы [disallow](#), [require](#) и [security](#)):

```
# slapd.conf
# раздел глобальных настроек
...
# Безопасность - раздел TLS
TLSCertificateFile /certs/ldaps-cert.pem
TLSCertificateKeyFile /certs/keys/ldaps-key.pem
TLSCipherSuite TLSv1+RSA:!NULL
# значение следующей директивы установлено так по умолчанию,
# однако мы приводим её здесь для наглядности
TLSVerifyClient never

# Безопасность - другие директивы
# предотвращаем анонимные подключения при любом типе соединения
disallow bind_anon
# требуем выполнения операции bind перед доступом к DIT
require bind
# Ожидание подключения только на порту ldaps требует использовать TLS/SSL,
# но не выдвигает требований к минимальной длине ключа.
# Требования к минимальной длине ключа выдвигает следующая директива:
security simple_bind=128

# DIT
database bdb
suffix "d=example,dc=com"
...
# replica provider
overlay syncprov

# cn=config DIT
database config
rootdn="cn=admin,cn=config"
rootpw ....

# cn=monitor DIT
database monitor
rootdn="cn=admin,cn=monitor"
rootpw ....
```

Примечания:

1. **cn=config:** скоро будет.
2. **Ожидаем подключения только для LDAPS URL:** Управление портами, на которых ожидаются подключения, осуществляется с помощью [аргумента -h](#) при запуске slapd. По умолчанию данное значение не задаётся, что (обычно) соответствует -h ldap:///. По требованиям данной конфигурации мы должны разрешить только операции ldaps, поэтому при запуске slapd должна использоваться команда:

```
slapd -h ldaps:/// -u ldap -g ldap
```

Чтобы это выполнялось автоматически, необходимо подправить сценарии запуска в linux (/etc/rc.d/init.c/slapd), а в BSD /etc/rc.conf должен содержать такую строку:

slapd_flags="ldaps:///".

Если Вы переживаете о конфигурации фаервола, то LDAPS может быть настроен на использование порта 389 (LDAPS - это схема URL, говорящая о том, что будет использоваться TLS, а не о том, какой будет использоваться порт). В этом случае команда запуска будет такая:

```
slapd -h ldaps://:389/ -u ldap -g ldap
# при подключении ldaps URL должны быть в форме:
ldaps://ldap.server.name:389
```

3. **Директивы disallow, require и security:** Ожидание подключений только на порту (портах) LDAPS приводит к принудительному использованию TLS/SSL при всех соединениях. Никакие другие типы соединений не будут приниматься в данной конфигурации сервера. Чтобы обязать пользователей проходить LDAP-аутентификацию, используется директива **require bind**, а для предотвращения анонимных подсоединений используется директива **disallow bind_anon**. Таким образом, чтобы получить какой-либо доступ к DIT, при любом подключении должна выполняться операция подсоединения (bind), и это подсоединение не может быть анонимным. Если указать только директиву **security simple_bind=128**, это не приведёт к обеспечению требуемого уровня защиты. Данная директива просто говорит: если выполняется простое подсоединение (simple bind), то должно использоваться TLS/SSL. Она не требует, чтобы подсоединение выполнялось обязательно. Единственная цель использования **security simple_bind=128** в данной конфигурации - определить минимальное значение SSF. Если это не нужно, данную директиву можно не указывать.
4. **Доступ к rootDSE:** Побочный эффект данной политики - запрет анонимного доступа к **rootDSE**, что, как уже отмечалось, может быть вполне оправдано для защищённых серверов.
5. **Директивы ACL:** В данной конфигурации не предусмотрено дополнительных мер безопасности, основанных на применении ACL - безопасность сервера полностью обеспечивается глобальными директивами конфигурации и ограничением подключений только на LDAPS портах. ACL может использоваться для установления необходимого разграничения доступа на основании учётных данных пользователей.

4. **Настройка клиентов:** Из требования, что сервер LDAP будет обслуживать только соединения TLS, следует, что все клиенты при подключении должны использовать URL со схемой LDAPS и выполнять проверку сертификата X.509 сервера, для чего требуется доступ к CA (корневому) сертификату. В нашем контексте под клиентами подразумеваются [утилиты и инструменты ldap](#), а также серверы LDAP, на которых выполняется сервис репликации с использованием [syncrepl](#). Очевидно, что, поскольку утилиты и инструменты ldap являются клиентами, они получают свои настройки из файла [ldap.conf](#). Возможно менее очевидно, что LDAP-серверы, на которых выполняется сервис репликации, также являются TLS-клиентами, и потому также получают информацию по CA (корневым) сертификатам из ldap.conf. Конфигурация ldap.conf:

```
# скопируйте сертификат, сгенерированный на шаге 1,
# в подходящее место на клиентской системе
# подразумевается: /certs/ldaps-cert.pem

# добавьте в ldap.conf
TLS_CACERT /certs/ldaps-cert.pem
```

Конфигурация LDAP-сервера, на котором запущена реплика:

```
# slapd.conf
# раздел глобальных настроек
...
# Безопасность - раздел TLS
# нет никаких настроек

# реплицируемое DIT
database bdb
suffix "d=example,dc=com"
...
# настройки потребителя репликации
syncrepl
rid=000
```

```
type=RefreshandPersist
provider=ldaps://ldap-master.example.com
bindmethod=simple
searchbase="dc=example,dc=com"
retry="5 3 300 +"
attrs="*,+"
binddn="...."
credentials=....
...
```

Примечания:

1. CA (корневой) сертификат, используемый в примере потребителем **syncrepl** (и определяемый директивой **TLS_CACERT** в `ldap.conf`), является тем же самым, что используется и в качестве сертификата сервера поставщиком главного DIT. Это следствие применения метода с единственным сертификатом, который мы использовали для генерации данного сертификата: он функционирует сразу и как сертификат сервера, и как сертификат CA. При использовании других методов генерации самоподписанных сертификатов сертификат сервера и сертификат CA могут быть различными, и, конечно, они всегда различны при использовании коммерческих сертификатов.
2. Если, как обсуждалось выше, сервис `ldaps` обслуживается на порту 389 (например, для устранения вопросов блокировки трафика фаерволом), то определение `syncrepl` будет использовать расширенный формат URL с указанием порта:

```
syncrepl
...
provider=ldaps://ldap-master.example.com:389
...
```

3. **LDAP-серверы в качестве TLS-клиентов:** Когда сервер LDAP выступает в качестве потребителя репликации `syncrepl`, он выступает в роли TLS-клиента, и потому ему требуется осуществлять проверку подлинности серверного сертификата с помощью CA (корневого) сертификата, которым он подписан. Поскольку данный сервер выступает в роли TLS-клиента, он будет использовать директивы TLS (и, соответственно, файлы сертификатов TLS), определяемые для клиентов LDAP - в частности, он будет использовать директиву **TLS_CACERT** в `ldap.conf`. Клиенту TLS требуется сгенерировать сообщение обмена ключами (*key-exchange message*), зашифрованное с помощью **открытого ключа (public key)** сервера, который извлекается из посылаемого сервером сертификата X.509. Также клиент TLS при первоначальной установке соединения по протоколу TLS на этапе **ClientHello** посылает список разрешенных наборов шифров, который контролируется директивой настройки клиента TLS **TLS_CIPHER_SUITE** (по умолчанию посылаются все возможные наборы шифров (ALL), что эквивалентно команде **openssl ciphers -v ALL**). Перед прочтением следующего предложения сделайте глубокий вдох. Если сервер LDAP, являющийся потребителем репликации `syncrepl` и потому клиентом TLS, также предоставляет доступ к другому DIT (например, к `cn=config`) и при этом подразумевается, что для контроля этого доступа также необходима поддержка TLS, то он **одновременно** будет сервером TLS и для него требуется задать все директивы сервера TLS, которые определялись выше на шаге 2 и 3.



Настройка смешанного доступа SSL/TLS в OpenLDAP

Если Вы сразу перешли к этому разделу, минуя описание нормальной конфигурации TLS, пожалуйста, найдите время ознакомиться как минимум с [вводными замечаниями](#), а желательно и со всем разделом, поскольку в противном случае Вы можете очень быстро прийти в замешательство (хотя после прочтения Вы можете прийти в замешательство еще быстрее). Надеемся, однако, что некоторое просветление настанет.

Конфигурация - кто-то использует TLS, кто-то - нет

В данном случае выдвигается требование, что некоторые пользователи будут использовать TLS, а другие - нет. Например, политика может быть следующей:

1. Разрешён анонимный доступ (и незащищённый обмен данными) к ограниченному набору общедоступных записей LDAP.

2. Пользователям, прошедшим LDAP-аутентификацию (простое подключение с незащищённым обменом данными) разрешён доступ на чтение к общедоступным записям LDAP, плюс ограниченные права на обновление только той записи, владельцем которой является данный пользователь.
3. Пользователи с привилегиями на обновление более одной собственной записи должны использовать TLS и проходить аутентификацию (подразумевается, что все они члены группы admins).
4. Все потребители репликации должны использовать TLS/SSL.
5. При удалённом доступе к cn=config должен использоваться TLS/SSL.

Данное решение требует применения ACL и директив настройки сервера, как показано ниже:

1. **Генерация сертификатов LDAP-сервера и CA:** На данном этапе будет сгенерирован самоподписанный сертификат - если будет использоваться коммерческий сертификат X.509 (SSL), данный процесс не требуется и может быть полностью пропущен. Для генерации единого сертификата X.509, который может применяться и для проверки подлинности сервера, и в качестве CA (корневого) сертификата для клиента, используется простой метод в одну команду. Данный метод детально (со всеми диалогами) документирован [здесь](#). Последовательность команд:

```
# создаём новые директории (опционально)
mkdir /certs
mkdir /certs/keys
cd /certs
# создаём сертификат сервера/CA и закрытый ключ без парольной фразы
# действительный в течение 10 лет, использующий текущие рекомендации RSA по размеру ключа
# RSA используется в качестве протокола обмена ключами
openssl req -x509 -nodes -days 3650 -newkey rsa:2048
-keyout keys/ldapskey.pem -out ldapscert.pem

# сертификат может использоваться в качестве сертификата сервера или сертификата CA
# помещаем сертификат в:
/certs/ldapscert.pem
# помещаем закрытый ключ в:
/certs/keys/ldapskey.pem

# устанавливаем права доступа
chown -R ldap:ldap /certs/*
chmod 0400 keys/ldapskey.pem
```

Примечания:

1. Полное объяснение параметров команды openssl смотрите [здесь](#).
 2. У файлов ключей обычно есть парольная фраза (пароль) для защиты конфиденциальных данных. Если сертификат генерируется для сервера, подобная парольная фраза никогда не используется (её генерация подавляется аргументом -nodes).
 3. В зависимости от требований, предъявляемых к сертификату, могут использоваться другие методы генерации самоподписанных сертификатов, описанные [здесь](#).
2. **Добавление директив TLS и ACL в slapd.conf:** (замечания по cn=config смотрите в конце раздела). Необходимые директивы TLS добавляются в раздел глобальных настроек:

```
# slapd.conf
# раздел глобальных настроек
...
# Безопасность - раздел TLS
TLSCertificateFile /certs/ldapscert.pem
TLSCertificateKeyFile /certs/keys/ldapskey.pem
TLSCipherSuite TLSv1+RSA:!NULL
# значение следующей директивы установлено так по умолчанию,
# однако мы приводим её здесь для наглядности
TLSVerifyClient never
...

# пользовательское DIT
database bdb
suffix "d=example,dc=com"
# нет директив rootdn и rootpw
# смотрите примечания по обеспечению безопасности доступа от имени rootdn
...

# ACL
# ACL1 - доступ для потребителя репликации
# (подразумевается подключение от имени cn=replica,dc=example,dc=com)
# потребителю, использующему TLS, разрешён доступ на чтение ко всему DIT
# все остальные переходят к ACL2 (break)
access to *
```

```

by dn.exact="cn=replica,dc=example,dc=com" tls_ssf=128 read
by * break
# ACL2 - доступ к атрибуту userpassword
# анонимные пользователи могут только проходить аутентификацию
# пользователи, прошедшие аутентификацию, могут модифицировать свои собственные пароли
(self)
# члены группы admins, использующие TLS, могут модифицировать пароли
access to attrs=userPassword
by self write
by anonymous auth
by group.exact "cn=admins,ou=groups,dc=example,dc=com" tls_ssf=128 write
# ACL3
# ограниченный анонимный доступ только на чтение к поддереву public
# любые пользователи, прошедшие аутентификацию, могут модифицировать свои
# собственные записи (self) в пределах поддерева
# члены группы admins, использующие TLS, имеют право на запись во всём поддереве public
access to dn.subtree="ou=public,dc=example,dc=com"
by group.exact "cn=admins,ou=groups,dc=example,dc=com" tls_ssf=128 write
by self write
by anonymous read
by users read
# ACL4
# члены группы admins, использующие TLS, имеют право на запись в остальном DIT
access to *
by group.exact "cn=admins,ou=groups,dc=example,dc=com" tls_ssf=128 write

# последний ACL4 слишком прямолинеен - если требуются дополнительные правила,
# его нужно заменить приведённым ниже, который будет отфильтровывать всех пользователей,
# не использующих TLS. В данном случае break означает, что если пользователь использует
TLS,
# он переходит к следующему ACL, в противном случае - обработка останавливается.
# ACL4
access to *
by group.exact "cn=admins,ou=groups,dc=example,dc=com" tls_ssf=128 break
# ACL5 etc.
access ....

# поставщик репликации
overlay syncprov

# cn=config DIT
database config
rootdn "cn=admin,cn=config"
rootpw {SSHA} hfkfhfldkhlkhh
# SSF большой или равный 128 используется для обеспечения безопасного доступа к cn=config
security simple_bind=128

```

Примечания:

1. Дополнительная информация по TLS cipher suite [здесь](#). Используемые параметры исключают применение NULL-шифров (без шифрования). TLSv1 охватывает SSLv3. Разрешены EXPORT-шифры - допускаются международные подключения. Если вопросы производительности и нагрузки стоят остро, лучше явно задать список шифров с приемлемыми характеристиками производительности и загрузки системы, чем оставлять возможность произвольного выбора шифра во время переговоров TLS/SSL.
2. **Конфигурация DSA:** скоро будет.
3. **cn=config:** скоро будет.
4. **Примечания по ACL:**
 1. **ACL1:** Данный ACL, который можно с тем же успехом поместить в раздел глобальных настроек, разработан с целью вычленив на раннем этапе подключения потребителей репликации (DN cn=replica,dc=example,dc=com должен присутствовать в DIT с соответствующим паролем). **by dn.exact="cn=replica,dc=example,dc=com" tls_ssf=128 read** содержит два условия, которые работают (хотя явно не документированы), и совпадения по которым будут найдены в том случае, если потребитель успешно прошёл аутентификацию И использовал SSF (TLS) не ниже 128. Для того чтобы потребитель репликации (да и все остальные) смог пройти аутентификацию (или просто получить доступ к DIT), необходимо наличие условия **by * break**, позволяющего перейти к ACL2 и продолжить анализ подключения.
 2. **ACL2:** Позволяет любому пользователю, прошедшему аутентификацию, модифицировать свой пароль. Анонимный доступ предоставляется в целях аутентификации (**auth**). Любой член группы cn=admins,ou=groups,dc=example,dc=com, использующий при подключении

TLS, может осуществлять изменения любого пароля.

3. **ACL3:** В указанном порядке - любому члену группы `cn=admins,ou=groups,dc=example,dc=com`, использующему при подключении TLS, разрешено производить запись в любой атрибут любой записи поддерева `public` базы данных. Анонимным пользователям предоставляется доступ на чтение к поддереву **public** DIT (за исключением атрибутов `userPassword`). Пользователю, прошедшему аутентификацию, разрешено изменять любую часть своей записи. Наконец, пользователю, прошедшему аутентификацию (подключающемуся с использованием или без использования TLS), разрешён доступ только для чтения к поддереву `public`, за исключением атрибутов `userPassword` (а также за исключением своих собственных записей, контроль доступа к которым был указан в условии `self`).
4. **ACL4:** Только членам группы `cn=admins,ou=groups,dc=example,dc=com`, использующим при подключении TLS, разрешён доступ на осуществление записи во все остальные части DIT. Всем остальным пользователям запрещён любой доступ (неявное условие **by * none**).
5. **Ожидаем подключения для LDAPS и LDAP URL:** Управление портами, на которых ожидаются подключения, осуществляется с помощью аргумента `-h` при запуске `slapd`. По умолчанию данное значение не задаётся, что (обычно) соответствует `-h ldap:///`. По требованиям данной конфигурации мы должны разрешить как операции **ldap**, так и **ldaps**, поэтому при запуске `slapd` должна использоваться команда:

```
slapd -h "ldap:/// ldaps:///" -u ldap -g ldap
```

Чтобы это выполнялось автоматически, необходимо подправить сценарии запуска в `linux` (`/etc/rc.d/init.c/slapd`), а в `BSD` `/etc/rc.conf` должен содержать такую строку:
slapd_flags="ldap:/// ldaps:///".

6. **Безопасность доступа от имени rootdn:** После того, как DIT было первоначально загружено, какие функции выполняет **rootdn**? Во всех штатных случаях привилегии **типа rootdn** для DIT могут быть предоставлены какому-либо конкретному пользователю (назовём его **псевдо-суперпользователь**) с помощью ACL, и потому директиву **rootdn** можно смело удалять. В хорошо продуманной системе контроля можно предоставить новому **псевдо-суперпользователю** необходимые полномочия с помощью стандартных ACL - собственно, для этого они и предназначаются. Единственный подводный камень данного подхода - то, что ошибка в ACL может привести к ограничению необходимых полномочий. В худшем случае, **rootdn** может быть временно восстановлен, а потом удалён вновь, когда проблема будет решена. Лучшим решением обеспечения безопасности доступа от имени **rootdn** к обычному DIT будет удаление самого **rootdn**. И точка. Как в приведённой выше конфигурации.

В тех случаях, когда **rootdn** является единственным методом доступа, таких как `cn=config` в приведённом выше примере, для принудительного включения механизмов защиты в конкретном разделе `database` используется директива `security simple_bind=128`.

LDAP. Приложения

Приложение А: Заметки и пояснения к LDAP

Приложение В: Ресурсы LDAP

Приложение С: LDAP RFC и документация

Приложение D: Глоссарий LDAP

Приложение Е: Распространённые наборы схемы данных, объектные классы и атрибуты LDAP

Приложение А - Заметки и пояснения к LDAP

В приведённых ниже заметках раскрываются подробности о некоторых аспектах LDAP и OpenLDAP:

1. LDAP: Головная боль с именованием корневой записи (базовой записи или суффикса)
2. LDAP: Определение простой корневой записи или суффикса

3. [LDAP: Определение корневой записи или суффикса в формате X.500](#)
4. [LDAP: Определение корневой записи или суффикса в формате RFC 2247](#)
5. [LDAP: Поиск по URL](#)
6. [LDAP: DN и RDN](#)
7. [LDAP: DN для аутентификации \(DN подключения \(Bind DN\) и DN принципала \(Principal DN\)\)](#)
8. [LDAP: Текстовые поисковые фильтры](#)
9. [LDAP: Поисковые фильтры соответствия компонентов](#)
10. [OpenLDAP: Несколько DIT](#)
11. [LDAP: OID \(Object Identifiers, идентификаторы объектов\)](#)
12. [LDAP: Руководство по выживанию в ASN.1](#)
13. [LDAP: Нужно ли определять иерархию объектных классов?](#)
14. [LDAP: Наследование - или как может случиться так, что у записи два структурных объектных класса](#)
15. [LDAP: Типы данных](#)
16. [OpenLDAP: Индексирование записей](#)
17. [LDAP: Информация о сравнении и поиске](#)
18. [OpenLDAP: Добавление записей - slapadd vs ldapadd](#)
19. [LDAP: Некоторые размышления по структурированию каталогов](#)
20. [OpenLDAP: Что можно, а что нельзя сделать, без того, чтобы рушить каталог и запускать его заново](#)
21. [OpenLDAP: Аргументы командной строки slapd](#)
22. [OpenLDAP: Заметки о наложениях в OpenLDAP \(или как заставить это работать\)](#)
23. [LDAP: Определение поисковых фильтров](#)
24. [OpenLDAP: Заметки о запуске/инициализации OpenLDAP](#)



Приложение A - OpenLDAP: slapadd vs ldapadd

В этой заметке обсуждаются два метода создания каталога или добавления записей в OpenLDAP - с помощью **slapadd** и **ldapadd**.

Уже совсем скоро (One day real soon now™).



Приложение A: LDAP - Руководство по выживанию в ASN.1

В этом разделе содержится краткий и поверхностный обзор синтаксиса ASN.1, достаточный для понимания выражений [соответствия компонентов](#). Вообще-то, Вам редко когда придётся спускаться до такого уровня детализации, если, конечно, Вы генетически к этому не предрасположены. Существует множество прекрасных и свободных [ASN.1-ресурсов](#), которые посвящённые читатели найдут бесконечно увлекательными и очень веселыми. Менее посвящённые читатели, возможно, упадут в обморок от одного только случайного взгляда на эти вещи.

Если обратиться к определению ASN.1 [OID](#), то можно увидеть, что оно состоит из текстового описания и некоторого количества синтаксиса ASN.1 (или тарабарщины), формально определяющей данный объект. Синтаксис ASN.1 губительно сложен, и в данном руководстве по выживанию внимание сфокусировано на 5 типах, имеющих важное значение при поиске соответствия компонентов: SEQUENCE OF, SEQUENCE, SET OF, SET и CHOICE. Короткие рабочие определения этих элементов:

SEQUENCE OF	Упорядоченный список элементов (вхождений) одинакового типа. Порядок фиксированный и содержит только один тип (или атрибут в контексте LDAP).
SET OF	Неупорядоченный список элементов (вхождений) одинакового типа. Порядок нефиксированный (фиксируется только во время выполнения) и содержит только один тип (или атрибут в контексте LDAP).
SEQUENCE	Упорядоченный список элементов (вхождений) разных типов. Порядок фиксированный и содержит несколько типов (или атрибутов в контексте LDAP).
SET	Неупорядоченный список элементов (вхождений) разных типов. Порядок нефиксированный (фиксируется

только во время выполнения) и содержит несколько типов (или атрибутов в контексте LDAP).

CHOICE	Определяет один элемент (вхождение). Этот элемент может быть одного из тех типов, которые определены в упорядоченном списке.

В этом руководстве соответствие компонентов разделено на две категории: простое и комплексное. Простое соответствие компонентов, как сказано в данном руководстве, используется только с атрибутами, определёнными с помощью конструкций SEQUENCE OF и SET OF - по существу, эти конструкции определяют несколько элементов (вхождений), из которых все одного и того же типа. Например, **DistinguishedName** определён с помощью SEQUENCE OF и имеет, согласно X.501, следующее определение ASN.1:

```
# Определение из X.501
DistinguishedName ::= RDNSequence
# DistinguishedName является псевдонимом для RDNSequence
RDNSequence ::= SEQUENCE OF RelativeDistinguishedName
# Определяет RDNSequence как упорядоченный список (SEQUENCE OF)
# элементов (вхождений) RelativeDistinguishedName

RelativeDistinguishedName ::= SET SIZE (1..MAX) OF
    AttributeTypeAndValue
# Определяет состав каждого RelativeDistinguishedName
# как SET (неупорядоченный список) AttributeTypeAndValue
# каждый из которых состоит из SEQUENCE (упорядоченного списка)
# типов и значений записей, как определено ниже
AttributeTypeAndValue ::= SEQUENCE {
    type      AttributeType ({SupportedAttributes}),
    value      AttributeValue ({SupportedAttributes}@type)} }

AttributeType ::= ATTRIBUTE.&id
AttributeValue ::= ATTRIBUTE.&Type
```

Примечание: В определениях ASN.1 регистр символов имеет *визуально* существенное значение, так, например, **distinguishedName** - это имя атрибута, а **DistinguishedName** - это имя определения атрибута.

Поскольку все элементы (вхождения) имеют один и тот же тип, в любом выражении соответствия компонентов достаточно идентифицировать количество элементов. Например, при использовании положительных чисел при извлечении элементов в выражении соответствия компонентов, "1" будет соответствовать первому вхождению в списке элементов (в случае с DN и положительными числами номера элементам назначаются СПРАВА НАЛЕВО). Примеры атрибутов, использующих DistinguishedName: [DistinguishedName \(2.5.4.49\)](#), [member \(2.5.4.31 \)](#), [seeAlso \(2.5.4.34\)](#), [owner \(2.5.4.32\)](#), [roleOccupant \(2.5.4.33\)](#).

Комплексное соответствие компонентов применяется к атрибутам, определённым с помощью конструкций SEQUENCE и SET. Эти конструкции определяют несколько значений, каждое из которых может быть разного типа. Например, [uniqueMember \(2.5.4.50\)](#) имеет следующую нотацию ASN.1:

```
# Определение из X.520
uniqueMember ATTRIBUTE ::= {
    WITH SYNTAX NameAndOptionalUID
    EQUALITY MATCHING RULE uniqueMemberMatch
    ID id-at-uniqueMember }

# указывает на ASN.1-определение NameAndOptionalUID,
# являющееся SEQUENCE
NameAndOptionalUID ::= SEQUENCE {
    dn DistinguishedName,
    uid UniqueIdentifier OPTIONAL }

# в свою очередь UniqueIdentifier определён в X.520 как
uniqueIdentifier ATTRIBUTE ::= {
    WITH SYNTAX UniqueIdentifier
    EQUALITY MATCHING RULE bitStringMatch
    ID id-at-uniqueIdentifier }
UniqueIdentifier ::= BIT STRING

# DistinguishedName определено в предыдущем примере
```

Поскольку все элементы (вхождения) в атрибуте различны, а в конструкции SET они могут ещё и появляться в каком угодно порядке, любое выражение извлечения компонента должно каким-то образом ссылаться на конкретный элемент, например, по имени его типа атрибута или какой-то другой характеристике.

В DN описывается содержимое **атрибутов** в дереве (так называемый путь навигации), требуемое для доступа к конкретной записи ИЛИ базовой (стартовой) записи поиска.

DN состоит из серии RDN (Relative Distinguished Names, относительных уникальных имён), определяемых путём перемещения ВВЕРХ по дереву (**DIT**) в направлении его **корневой записи (суффикса или базовой записи)**, и записываемых СЛЕВА НАПРАВО, в отличие, например, от файловой системы, где пути записываются СПРАВА НАЛЕВО.

DN записывается СЛЕВА НАПРАВО.

При добавлении в DIT новой записи с помощью DN серверу сообщается, каким образом структурировать (или разместить) данную запись. Пример добавления новых записей с использованием файла LDIF (точно также их можно добавить с помощью LDAP-браузера или специализированного клиентского инструмента LDAP).

```
version: 1

## нет строгого требования указывать версию (version),
## но это является хорошим тоном для совместимости с будущими версиями

## ОПРЕДЕЛЯЕМ DIT ROOT/BASE/SUFFIX #####
## используется формат RFC 2377

## dcObject - это ВСПОМОГАТЕЛЬНЫЙ объектный класс и, кроме него, запись
## ДОЛЖНА иметь СТРУКТУРНЫЙ объектный класс (в данном случае, organization)
# это последовательность ЗАПИСИ и ей предшествует ПУСТАЯ СТРОКА

dn: dc=example,dc=com
dc: example
description: The best company in the whole world
objectClass: dcObject
objectClass: organization
o: Example, Inc.

## ПЕРВЫЙ уровень иерархии - люди (people)
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой

dn: ou=people, dc=example,dc=com
ou: people
description: All people in organisation
objectClass: organizationalUnit

## ВТОРОЙ уровень иерархии - записи людей
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой

dn: cn=Joe Schmo,ou=people,dc=example,dc=com
objectclass: inetOrgPerson
cn: Joe Schmo
sn: Schmo
uid: jschmo
mail: joe@example.com
mail: j.schmo@example.com
ou: sales
```

Как правило, нет разницы, значение какого атрибута используется при добавлении записи, лишь бы соблюдалось условие уникальности 'dn:'. В последней записи приведённого примера для этой цели было решено использовать cn=Joe Schmo. Точно также можно было бы использовать uid=jschmo. В процессе поиска LDAP может быть использован любой атрибут или комбинация атрибутов, так что записи могут быть найдены независимо от того, какое значение 'dn:' использовалось при их создании. Во избежание излишних операций поиска, в большинстве случаев имеет смысл использовать значение 'dn:', представляющее собой тот DN, который чаще всего будет использоваться для доступа к записи. Таким образом, в предыдущем примере подразумевается, что чаще всего при обращении к каталогу будет использоваться атрибут cn=.

Однако, если какую-то запись планируется использовать для аутентификации пользователя, значение её 'dn:' создания становится чрезвычайно важным и определяет единственно возможный DN входа в систему. Причина этого в том, что проверка подлинности осуществляется с помощью LDAP-операции Bind, которая требует предоставления DN подключения (Bind DN) и, опционально, пароля, при этом никакие операции поиска НЕ разрешены. Таким образом, данный DN подключения **МОЖЕТ БЫТЬ ТОЛЬКО** тем DN, который использовался при добавлении (создании) записи.

В стандартах LDAP нет специального термина, указывающего на DN, используемый при первоначальном создании записи. Однако иногда, особенно в контексте LDAP, применяемого в Microsoft AD, данный DN

'создания' называется [DN принципа \(Principal DN\)](#), главным образом вследствие использования его в качестве принципа (принципа безопасности) в [Kerberos](#).

Приложение А - OpenLDAP: Чтобы не начинать сначала

Несколько замечаний, полезных при работе с OpenLDAP. В них описано, что можно, а что нельзя сделать, без того, чтобы, признав свою ошибку, создать резервные копии и попытаться запустить каталог заново.

Что можно сделать

Все перечисленные ниже вещи можно сделать, когда каталог уже запущен в работу. Не нужно рушить каталог и запускать его снова, то есть необязательно иметь все эти вещи с самого начала:

1. Вы можете добавить новые записи - в самом деле! Используйте [ldapadd](#) с работающим каталогом, либо [slapadd](#), когда **OpenLDAP (slapd)** остановлен.
2. Вы можете добавить **вспомогательный (AUXILLIARY) объектный класс** к существующим записям. Например, Вы можете добавить [posixaccount](#) к существующей записи с объектным классом [inetorgperson](#).
3. Вы можете добавить **структурный (STRUCTURAL) объектный класс** к существующим записям, но ТОЛЬКО в том случае, если запись уже содержит объектный класс, родительский (**SUP**) по отношению к тому, который Вы хотите добавить. Например, Вы можете добавить [inetorgperson](#) к существующей записи, в которой присутствует объектный класс [person](#). Но, при тех же условиях, Вы не сможете добавить объектный класс [account](#), поскольку для него родительским классом является **top** и, таким образом, будет создана вторая иерархия структурных объектных классов, что в настоящий момент строго запрещено.
- 4.

Чего сделать нельзя

Вы НЕ сможете сделать перечисленные ниже вещи, если каталог уже запущен в работу. Ошибка, допущенная в одной из этих вещей, скорее всего окажется болезненной, зато Вы приобретёте бесценный опыт:

- 1.

Начинаем всё сначала

Весь мир погрузился во тьму. В структуре Вашего каталога обнаружились серьёзные проблемы и придётся начинать всё заново. Во-первых, не паникуйте! Во-вторых, немного подумайте.

1. Экпортируйте каталог целиком как файл LDIF. Поскольку файлы LDIF - это обыкновенные текстовые файлы, можно написать простенькие скрипты для изменения текста записей каталога в файле, тем самым производя манипуляции с каталогом в целом.
2. Остановите **OpenLDAP (slapd)**. Перейдите в директорию, указанную в разделе [database](#) файла **slapd.conf**, и удалите всё из этой директории.
3. Запустите **OpenLDAP (slapd)**. Используйте [ldapadd](#) для импорта Вашего модифицированного файла LDIF обратно в каталог.

Приложение А - LDAP: Поисковые фильтры соответствия компонентов

Нормальная текстовая форма поискового фильтра определена в [RFC 4515](#), немного улучшена в [RFC 4510](#) и описывается [здесь](#). Поисковые фильтры были значительно расширены с помощью так называемого соответствия компонентов (component matching, [RFC3687](#)) и общих правил кодирования строк (Generic String

Encoding Rules, GSER, [RFC4792](#)).

Примечание: Как уже было сказано, соответствие компонентов - часть основной спецификации LDAPv3, и потому не требует [SupportedExtension](#) OID. С другой стороны, соответствие компонентов было определено значительно позже, чем оригинальные спецификации LDAPv3. Поэтому не совсем понятно, каким образом, кроме как попробовав (или произведя поиск `componentMatchingRule` в ветке `matchingRules` в `rootDSE`), можно определить, поддерживает ли какая-то конкретная реализация LDAP, утверждающая, что она совместима с LDAPv3, эту возможность. Можно предположить, что серверы LDAP, в которых декларируется совместимость с RFC серии 45XX, должны поддерживать соответствие компонентов, а те из них, которые совместимы с более ранними RFC серии 225X, скорее всего - нет. В общем, бардак.

Что касается OpenLDAP - здесь всё находится в развитии. Утверждалось, что соответствие компонентов будет реализовано в OpenLDAP 2.4.8, но это не так.

Соответствие компонентов (Component Matching, RFC 3687)

Соответствие компонентов позволяет выполнять поиск по компонентам сложного (составного) атрибута, такого как **member**, путём явной ссылки на них (или их извлечения). Первопричина разработки данного синтаксиса правил соответствия предположительно связана с необходимостью дополнительной обработки сертификатов X.509. Для этой цели данный синтаксис позволяет пользователю спуститься до уровня ASN.1-деталей требуемого атрибута и сослаться на содержимое [грамматик SEQUENCE OF, SEQUENCE, SET OF и SET](#). Такой уровень детализации может потребовать перекрёстных ссылок на несколько документов серии X.5xx (как минимум, X.501 и X.520).

Однако, соответствие компонентов также предоставляет более читабельный (хотя и более длинный) синтаксис, особенно для нормальных комбинированных выражений, который может оказаться полезным для пользователя. Чтобы упростить описание данной возможности (как для нас самих, так и для всех читателей), мы решили отдельно описать [простые](#) и [комплексные](#) выражения соответствия компонентов.

В функции соответствия компонентов представлены два новых правила соответствия: `rdnMatch` (1.2.36.79672281.1.13.3), позволяющее обработку RDN в DN, и `presentMatch` (1.2.36.79672281.1.13.5), возвращающее TRUE, если целевой атрибут извлекаемого компонента существует, эквивалентно фильтру наличия `attr=*` в нормальных поисковых фильтрах.

Простое соответствие компонентов (RFC 3687)

Простое соответствие компонентов определяется либо как альтернативный синтаксис для поиска простого атрибута (значение которого представляет собой один элемент), либо для поиска компонента, извлечённого из комплексного атрибута (значение которого представляет собой несколько элементов), определённого с помощью грамматик [ASN.1 SEQUENCE OF](#) или [SET OF](#) - в основном, для атрибутов, содержащих несколько компонентов одного и того же типа.

```
(attr:componentFilterMatch:=[boolexp:{  
  item:{{component comexp} [useDefaultValues TRUE|FALSE] rule  
  matchingRule value valexp } [next-item ... }])
```

Поисковый фильтр заключён в круглые скобки, а элементы фильтра соответствия компонентов заключены в фигурные скобки `{}`.

В следующих подразделах описывается синтаксис соответствия компонентов при использовании в простых выражениях. При использовании в [комплексных выражениях](#) возможны дополнительные детали. Синтаксис может быть невообразимо сложным, - не отчаивайтесь, - лучше посмотрите сначала [простые примеры](#), в которых показаны синтаксисы нормальных фильтров и правил соответствия компонентов, а затем возвращайтесь к формальным определениям.

Attr

Имя атрибута, по которому будет производиться поиск. Это может быть простой или сложный (составной) атрибут, или операционный атрибут.

componentMatchingRule

Правило соответствия для выполнения функции соответствия компонентов.

[boolexp{]

Необязательное логическое выражение, возможные варианты 'and:', 'or:' или 'not:'. Логические выражения заключаются в фигурные скобки {}, могут быть вложенными и предварять очень сложные выражения.

item:{

Ключевое слово **item** означает начало component-выражения, заключённого в фигурные скобки {}.

component

Необязательное ключевое слово **component**, за которым следует выражение извлечения (или изоляции) компонентов. Если это ключевое слово пропущено, подразумевается, что поисковый фильтр будет ссылаться на атрибут **attr** целиком, то есть, компоненты извлекаться не будут и такой фильтр превращается просто в альтернативный синтаксис определения поиска. Если component-выражение присутствует, то тип извлекаемого им компонента определяется при применении правила **rule**.

comexp

Выражение извлечения компонента. Может быть простым значением, заключённым в кавычки, либо вложенным выражением **item**. Простые, заключённые в кавычки значения, могут быть следующими:

число - определяет номер экземпляра компонента согласно грамматике SET OF или SEQUENCE OF в определении атрибута. В случае, когда атрибут содержит DN, экземплярами, определёнными в SEQUENCE OF, будут RDN. Самому правому RDN присваивается номер 1, следующему - номер 2 и т.д. Так в DN ou=me,dc=my,dc=us RDN dc=us будет иметь номер экземпляра "1", dc=my - "2" и ou=me - "3". Экземпляры именуются справа налево.

отрицательное
число - аналогично предыдущему, только "-1" соответствует самому левому экземпляру.
Так в DN ou=me,dc=my,dc=us RDN ou=me будет иметь номер экземпляра "-1",
dc=my - "-2" и dc=us - "-3". Экземпляры именуются слева направо.

количество - значение "0" указывает, что последующие тесты будут применяться по отношению к количеству экземпляров компонента в атрибуте, а не к значениям этих экземпляров.

все - значение "*" указывает, что будут использованы все экземпляры компонентов атрибута.

useDefaultValues TRUE|FALSE

Необязательное значение (по умолчанию TRUE). Определяет, будут ли значения по умолчанию для атрибута (если таковые определены) использоваться (в случае TRUE) при сравнении в component-выражении, или нет (в случае FALSE).

rule

За ключевым словом **rule** следует правило соответствия (matchingRule), которое будет применено к извлечённому компоненту.

matchingRule

Правило соответствия, которое будет применено к извлечённому компоненту. Может определяться по имени или OID. Правило соответствия должно четко соответствовать извлеченным компонентам.

value

За ключевым словом **value** следует то value-выражение, которое будет использоваться при сравнении в данном поисковом фильтре.

Заключённое в кавычки value-выражение, которое будет использоваться при сравнении в данном поисковом фильтре, либо значение NULL, применяемое только с правилом соответствия **presentMatch**.

Примеры простого соответствия компонентов

Соответствие компонентов в следующих примерах используется как альтернативный синтаксис для уже показанных ранее нормальных поисковых фильтров.

```
# нормальный поисковый фильтр
(mail=*) # возвращает все записи, у которых есть атрибут mail
# синтаксис соответствия компонентов
(mail:componentFilterMatch:=item:{rule presentMatch, value NULL})
# presentMatch возвращает TRUE если компонент есть в наличии, иначе FALSE
# отрицательная форма (возвращает все записи, у которых НЕТ атрибута mail)
(mail:componentFilterMatch:=not:{item:{rule presentMatch, value NULL}})

# нормальный поисковый фильтр
(mail=*@*) # возвращает записи, значение атрибута mail которых соответствует правильному
            почтовому адресу формата RFC822
# синтаксис соответствия компонентов
(mail:componentFilterMatch:=item:{rule caseIgnoreMatch, value "*@*"})

# нормальный поисковый фильтр
(&(mail=*)(cn=*r)(sn=s*)) # есть атрибут mail И cn заканчивается на R И sn начинается с s
# синтаксис соответствия компонентов
()

# нормальный поисковый фильтр
(|(sn=a*)(sn=b*)(sb=c*)) # sn начинается с a ИЛИ с b ИЛИ с c
# синтаксис соответствия компонентов
(sn:componentFilterMatch:=or:{item:{rule caseIgnoreMatch, value "a*"},
  item:{rule caseIgnoreMatch, value "b*"},
  item:{rule caseIgnoreMatch, value "c*"}})

# нормальный поисковый фильтр
(!(sn=a*)) # записи с sn НЕ начинающимся с a
# синтаксис поиска компонентов
(sn:componentFilterMatch:=not:{item:{rule caseIgnoreMatch, value "a*"}})

# нормальный поисковый фильтр
(&(!(sn=a*))(!(sn=b*))) # записи с sn НЕ начинающимся с a И НЕ начинающимся с b
# синтаксис поиска компонентов
(sn:componentFilterMatch:=and:{item:{rule caseIgnoreMatch, value "a*"},
  not:{item:{rule caseIgnoreMatch, value "b*"}}})

# нормальный поисковый фильтр
# переопределим соответствие EQUALITY, чтобы оно зависело от регистра символов
sn:caseExactMatch:=Smith
# то же самое с помощью OID
sn:2.5.13.5:=Smith
# синтаксис поиска компонентов
(sn:componentFilterMatch:=item:{rule caseExactMatch, value "Smith"})
# или
(sn:componentFilterMatch:=item:{rule 2.5.13.5, value "Smith"})
```

В следующих примерах извлекаются компоненты из составного атрибута - в данном случае, DN-атрибут, такой как **member**:

```
# Во всех примерах подразумевается, что значение member в формате
# cn=groupname,ou=groups,dc=example,dc=com
# где groupname - переменная

# находим записи с атрибутом member, в котором cn = sales
# в качестве выражения извлечения используем число
# (компоненты нумеруются справа налево)
(cn:componentFilterMatch:=item{component "4",rule rdnMatch, value "cn=sales"})

# находим записи с атрибутом member, в котором cn = sales
# в качестве выражения извлечения используем отрицательное число
# (компоненты нумеруются слева направо)
(cn:componentFilterMatch:=item{component "-1",rule rdnMatch, value "cn=sales"})

# находим записи с атрибутом member, в котором ou = groups
# в любом месте dn
# в качестве выражения извлечения используем синтаксис все
(cn:componentFilterMatch:=item{component "*",rule rdnMatch, value "ou=groups"})
```

```
# находим в записи, у которых в атрибуте member 5 компонентов
# в качестве выражения извлечения используем синтаксис count
(cn:componentFilterMatch:=item{component "0",rule rdnMatch, value "5"})

# находим в записи, у которых в атрибуте member 5 и более, но менее 8-ми компонентов
# в качестве выражения извлечения используем синтаксис count
(cn:componentFilterMatch:=or:{item{component "0",rule rdnMatch, value "5"},
item{component "0",rule rdnMatch, value "6"},
item{component "0",rule rdnMatch, value "7"}})

# находим в записи, у которых в атрибуте member есть и cn=sales, и dc=example, и dc=com
# используем составное выражение
(cn:componentFilterMatch:=and:{item{component "4",rule rdnMatch, value "cn=sales"},
item{component "2",rule rdnMatch, value "dc=example"},
item{component "1",rule rdnMatch, value "dc=com"}})
```

Комплексное соответствие компонентов (RFC 3687)

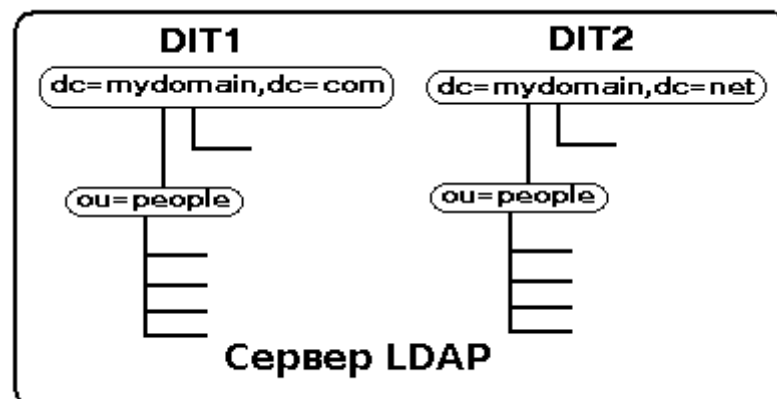
Мы дадим Вам знать, когда сами с этим разберёмся. Но не обещаем, что это будет скоро.

Appendix A - OpenLDAP: Несколько DIT

Один сервер LDAP может поддерживать одно или **несколько DIT**.

Каждое DIT является полностью отдельным и имеет свой собственный **контекст именования** (или пространство имён).

Вот иллюстрация такой конфигурации:



В случае с OpenLDAP каждое DIT и его **суффикс** (он же **корневая** или **базовая** запись), например `dc=mydomain,dc=com` и `dc=mydomain,dc=net`, определяется в отдельном разделе **database** файла `slapd.conf`.

HOWTO по конфигурации нескольких DIT [здесь](#).

Приложение A - LDAP: DN и RDN

DN (Distinguished Name, уникальное имя) должно быть уникальным (или хотя бы приблизительно уникальным, смотрите комментарии ниже) в пределах дерева (**DIT**). В DN описывается содержимое **атрибутов** в дереве (так называемый путь навигации), требуемое для доступа к конкретной записи ИЛИ базовой (стартовой) записи поиска.

DN состоит из серии RDN (Relative Distinguished Names, относительных уникальных имён), определяемых путём перемещения ВВЕРХ по дереву (**DIT**) в направлении его **корневой записи (суффикса или базовой записи)**, и записываемых СЛЕВА НАПРАВО, в отличие, например, от файловой системы, где пути записываются СПРАВА НАЛЕВО. Если проводить аналогии, то это больше напоминает полностью определённые доменные имена (fully qualified domain name, FQDN). Если Вы не знаете, что такое полностью определённые доменные имена - забудьте об аналогиях!

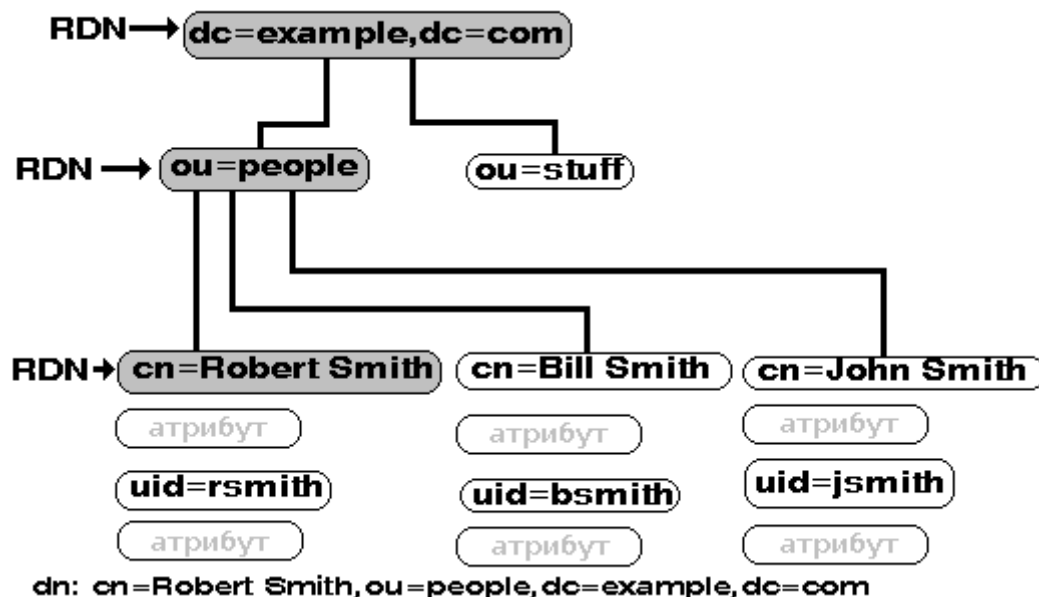
DN записывается СЛЕВА НАПРАВО.

Прежде чем двигаться дальше стоит потратить несколько минут, чтобы определиться с тем, что мы пытаемся делать при получении доступа к DIT, и что может означать термин "уникальный".

1. Если мы производим поиск, термин "уникальный" может означать не совсем то, что нам требуется - нам может понадобиться лишь приблизительно уникальный DN.
2. Если мы производим запись (модификацию), тогда DN должен быть уникальным - ведь мы хотим изменять только конкретную запись.

DN состоит из серии RDN (Relative Distinguished Names, относительных уникальных имён), которые представляют собой уникальные (или хотя бы приблизительно уникальные) атрибуты на каждом уровне иерархии DIT.

На рисунке показано построение DN из RDN:



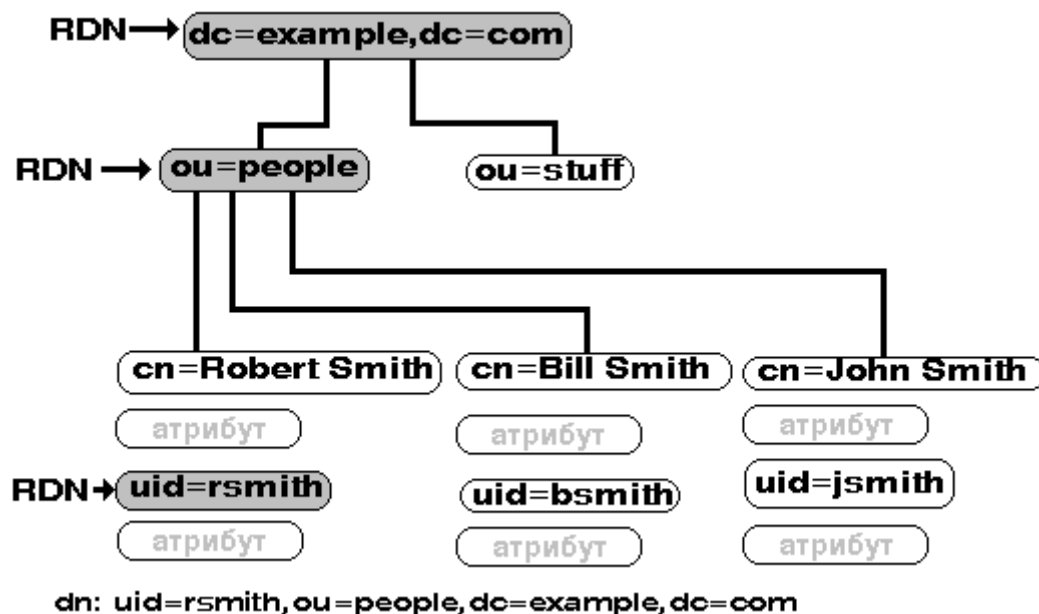
В этом примере в качестве нашего RDN мы выбрали атрибут **cn (commonName)**, поскольку он уникален на данном уровне иерархии в каталоге. Это дало такой полный DN:

DN: cn=Robert Smith,ou=people,dc=example,dc=com

НО мы с тем же успехом могли выбрать в качестве нашего RDN атрибут **uid (userID)**, поскольку он также уникален (смотрите ниже). В обоих случаях в результате получаются вполне допустимые DN.

В атрибутах, используемых в качестве RDN нет ничего особенного, за исключением того, что их значения уникальны (для записи), и должны всегда оставаться уникальными. Но останется ли уникальным наш атрибут **cn**, когда на работу в организацию устроится второй Robert Smith? Если он не будет уникальным, то эта ситуация останется приемлемой для поиска (чтения), но не для записи - в этом случае нужно будет использовать какой-то другой атрибут или комбинацию атрибутов.

Примечание: Когда запись добавляется (или создаётся), ей всегда назначается DN и, как правило, именно этот DN чаще всего используется для получения доступа к записи. Если данная запись будет использоваться в целях прохождения аутентификации, то на неё накладываются некоторые [специальные условия](#).



Наконец, существует возможность объединить два или более атрибута и сформировать **многозначный RDN**, который затем будет использоваться в DN. В нашем примере можно сформировать RDN, объединив **cn+uid**, для создания такого DN:

```
DN: cn=Robert Smith+uid=rsmith,ou=people,dc=example,dc=com
```

Примечание: Как правило, если Вы будете часто производить поиск по какому-либо атрибуту, он должен быть проиндексирован.

Приложение А - OpenLDAP: Индексирование записей

index

Лучше определять директивы **index** файла slapd.conf (или olcDbIndex в конфигурации cn=config) перед первоначальной загрузкой каталога (с помощью [ldapadd](#)), тогда при первоначальной загрузке соответствующие индексы будут созданы автоматически. При внесении изменений в настройки индексов после первоначальной загрузки требуется переиндексирование (повторное создание индексов) каталога с помощью [slapindex](#) (**предупреждение:** перед этим slapd должен быть остановлен).

Формат:

```
index attrlist | default indices
# indices = [pres [,approx] [,eq] [,sub] [,special]]
```

Директивы **index** определяют, какие индексы будут обслуживаться OpenLDAP. Может быть включено любое количество параметров индексирования. Даже если атрибут не был указан в директивах index, его всё равно можно использовать в [поисковых фильтрах](#) - если это будет происходить часто, то, конечно, производительность будет страдать, а если раз в жизни - ничего страшного.

attrlist может быть единичным атрибутом, или списком атрибутов, разделённых запятыми.

Необязательный параметр **default** содержит те индексы, которые должны поддерживаться по умолчанию. Они применяются к атрибутам в последующих директивах index, в которых пропущен список индексов. Директива **index** со значением **default** должна быть определена до появления директив **index** без списка индексов. Если определено несколько директив **index** со значением **default**, каждая из них будет влиять на директивы **index**, следующие непосредственно за ней, до появления очередной директивы **index** со значением **default**.

Индекс **pres** следует использовать, если предполагается, что будут выполняться [поисковые запросы](#) в

форме 'objectclass=person' или 'attribute=mail'.

Индекс **approx** НЕОБХОДИМО использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'sn~=person' (поиск 'созвучных' значений).

Индекс **eq** следует использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'sn=smith', то есть без применения поисковых шаблонов (используется только правило EQUALITY).

Индекс **sub** следует использовать, если предполагается, что будут выполняться [поисковые запросы](#) в форме 'sn=sm*', то есть с применением поисковых шаблонов (используется правило SUBSTR). Данный индекс можно задать более конкретно, указывая **subinitial** (оптимизация под 'sn=s*'), **subany** (оптимизация под 'sn=*n*') или **subfinal** (оптимизация под 'sn=*th'). Для одного атрибута можно указать несколько индексов типа **sub**.

Параметр **special** связан с [подтипами \(subtype\)](#) и может быть либо **nolang**, либо **nosubtypes**.

Будьте очень внимательны при выборе того, какие индексы будут поддерживаться каталогом. Лучше делать это на основании **требований приложений**, если их не учитывать, это может серьёзно повлиять на производительность каталога. И наоборот, нет никакого смысла индексировать те поля, поиск по которым никогда не осуществляется. Если в поисковых запросах используются только правила EQUALITY, нет смысла устанавливать индексирование **sub**. Индексы потребляют память (чем больше индексов, тем больше они занимают оперативную память). Кроме того, операции записи и модификации при большом количестве индексов будут выполняться дольше, поскольку требуется время и ресурсы на обновление индексов.

Примеры:

```
# Простое использование значения default
index default pres,eq
index cn,sn,uid

# Определяем индексы наличия и соответствия
# для атрибутов cn, sn и uid
# две приведённые выше директивы index выполняют
# абсолютно то же, что и три, приведённые ниже
index cn pres,eq
index sn pres,eq
index uid pres,eq

index cn eq,sub,subinitial
# Создаём индексы для атрибута cn (commonname)
# для поисков EQUALITY, SUBSTR, а также дальнейшая оптимизация
# для поисков типа sc=a*

index sn eq,approx,sub
# Создаём индексы для атрибута sn (surname)
# для поисков EQUALITY и SUBSTR
# Примечание: Индекс approx - пустая трата времени,
# поскольку для sn не определено правило ORDERING,
# требуемое для поисков approx. Приводится лишь для
# иллюстрации возможности использования

index mail pres,eq,sub
# Создаём индексы для атрибута mail on
# для поисков на наличие, EQUALITY и SUBSTR

index objectclass eq
# Оптимизируем поиски по форме objectclass=person
```

Обзор других настроек производительности можно найти в соответствующей [главе](#).

Приложение А - LDAP: Наследование объектных классов

Для LDAP определено множество правил, но одно из самых основных - то, что у записи может быть только один структурный (STRUCTURAL) объектный класс. В ситуации, показанной ниже, может создаться впечатление (неверное), что это правило нарушается.

Во многих документах утверждается, что в LDIF-файлах, описывающих **DIT** и его **записи**, должна указываться полная иерархия объектных классов, как показано в этом фрагменте LDIF:

```
dn:cn=Jim Bob,ou=people,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Jim Bob
sn: Bob
mail: jimbob@example.com
ou: sales
....
```

Все объектные классы **person**, **organizationalPerson** и **inetOrgPerson** являются структурными. Кажется, правило о том, что запись может содержать только один объектный класс структурного типа, нарушено.

Но это не так.

Объектные классы могут быть частью иерархии, что указывается в определении объектного класса с помощью параметра SUP с именем вышестоящего объектного класса, отличного от **top**. Когда объектный класс определяется таким способом, он ДОЛЖЕН быть того же типа (STRUCTURAL или AUXILIARY), что и его вышестоящий (родительский) объектный класс. Кроме того, он наследует все атрибуты своего родителя (родителей). Так что в данном случае у нас просто иерархия структурных объектных классов. Для иллюстрации, **organizationalPerson** содержит атрибут **title**; когда мы определяем в записи объектный класс **inetOrgPerson** (даже если указывается только он один), мы можем использовать атрибут **title**. Мы унаследовали **title** от родительского объектного класса.

Множественное наследование

LDAP не может поддерживать множественное наследование в полном смысле этого слова. Однако, в особых условиях, объектные классы могут иметь в определениях несколько имён вышестоящих (SUP) объектных классов и могут наследовать свойства от нескольких родителей:

```
# Из RFC 1274
objectClasses: ( 0.9.2342.19200300.100.4.20 NAME 'pilotOrganization'
  SUP ( organization $ organizationalUnit ) STRUCTURAL
  MAY buildingName )
```

В определении **pilotOrganization** указано два вышестоящих объектных класса: **organization** и **organizationalUnit**, и оба они являются структурными. Теперь у нас действительно нарушается правило единственного структурного объектного класса в записи? Не совсем. Посмотрим немного поглубже на то, как определяются объектные классы. Вот ASN.1-определение **pilotOrganization**:

```
pilotOrganization OBJECT-CLASS
  SUBCLASS OF organization, organizationalUnit
  MAY CONTAIN {
    buildingName}
  ::= {pilotObjectClass 20}
```

Данное **ASN.1**-определение говорит о том, что **pilotOrganization** является подклассом (SUBCLASS OF) сразу и **organization**, и **organizationalUnit**. Теперь посмотрим на ASN.1-определения **organization** и **organizationalUnit**:

```
organizationalUnit OBJECT-CLASS
  SUBCLASS OF top
  MUST CONTAIN {
    organizationalUnitName}
  MAY CONTAIN {
    organizationalAttributeSet}
  ::= {objectClass 5}

organization OBJECT-CLASS
  SUBCLASS OF top
  MUST CONTAIN {
    organizationName}
  MAY CONTAIN {
    organizationalAttributeSet}
```



```
::= {objectClass 4}
```

Оба они являются подклассом (SUBCLASS OF) top, то есть они представляют собой самый верхний уровень своих иерархий. Таким образом, если в определении OBJECT-CLASS указано несколько родителей, то допустимо использовать множественное наследование, в ином случае - нет. Почти не существует определений OBJECT-CLASS, предоставляющих такую возможность - кроме **pilotOrganization** мы не нашли ни одного.

Примечание: RFC 1274, определяющее **pilotOrganization**, стало устаревшим с выходом RFC 4524, и приведённых выше определений ASN.1 в этом RFC больше нет. Однако, файл cosine.schema (OpenLDAP 2.4.11) всё ещё содержит **pilotOrganization**.

На практике, OpenLDAP версий 2.x+ способен обработать иерархию объектных классов, таким образом, следующий фрагмент LDIF будет прекрасно работать (и, возможно, немного меньше смущать):

```
dn:cn=Jim Bob,ou=people,dc=example,dc=com
objectclass: inetOrgPerson
cn: Jim Bob
sn: Bob
mail: jimbob@example.com
ou: sales
....
```

Внимание: Если Вы экспортируете последнее определение из OpenLDAP, на выходе будет именно то, что Вы указали в LDIF, с помощью которого записи были созданы. Не все серверы LDAP способны автоматически обработать иерархию объектных классов. Если затем Вы попытаетесь загрузить экспортированный LDIF в LDAP-сервер, не способный обработать иерархию объектных классов, скорее всего Вы не получите желаемого результата. Если Вы используете только серверы с поддержкой обработки иерархии (например, только OpenLDAP), то можно сэкономить время, не печатая лишнего.

Приложение А - LDAP: Головная боль с именованием корневой записи

Запись самого верхнего уровня в LDAP DIT (Directory Information Tree, информационном дереве каталога) в мире LDAP неоднозначно называется **корневой (root)**, **базовой (base)** или **суффиксом (suffix)**, в зависимости от документации, её автора, дня недели или каких-либо других переменных величин, доподлинно нам не известных.

Термин **Root DSE** определяет своего рода супер-корневую запись, содержащую все DIT, поддерживаемые сервером LDAP, а также **операционные объекты**.

В документации присутствует два разных подхода к предмету **корневой записи** (суффикса или базовой записи). Согласно первому, это нечто **волшебное**, находящееся за пределами понимания простых смертных, воздействовать на которое можно лишь посредством ритуалов, передающихся из поколения в поколение, а потому нет никакого смысла даже пытаться объяснить что-либо. Последователи второго подхода, наоборот, воспевают данный предмет в невероятно длинных песнопениях, обычно сопровождающихся плачем и скрежетом зубов, а также заклинаниями двуликому богу ITU и IETF.

Наш совет - если Вам не нужно предоставлять Ваш каталог в **глобальный** доступ, определите **корневую запись** как **ou** с именем **gobbledegook** и больше не забивайте этим голову. О том, как определить простую корневую запись или суффикс, читайте [здесь](#).

Ну а теперь - шутки в сторону.

Источник головной боли - изначальная и похвальная цель X.500 (очевидно, имеющая продолжение в IETF) создать глобальную иерархию каталогов, схожую с системой DNS.

В этом контексте **корневая запись** (суффикс или базовая запись), выставленная в **глобальный** доступ, играет существенную роль и должна быть уникальной. Возвращаясь к нашему совету, - если не требуется выставлять каталог в глобальный доступ в будущем, забудьте о правилах именования и дайте ему любое имя, - значимое или забавное, - по Вашему желанию. О том, как определить простую корневую запись или суффикс, читайте [здесь](#).

Если же у Вас есть (или могут появиться в будущем) **глобальные** амбиции - читайте далее.

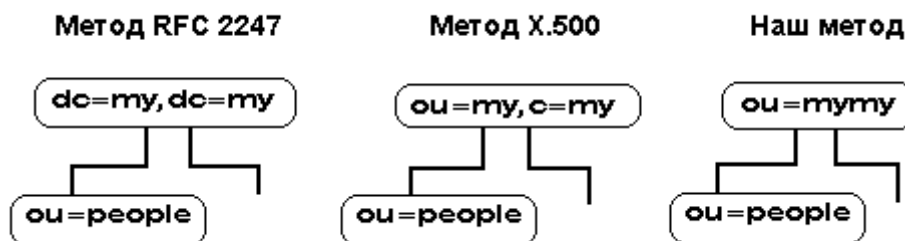
В X.500 в качестве стандарта **корневой записи** выбран **ou=organisation name,c=country code**, например, ou=Example Inc., c=u. Какие на то были побудительные мотивы, - непонятно; возможно, в тот момент подобная идея показалась хорошей, а может это связано с фазами Луны. [Кое-что об определениях ou](#). О том, как определить корневую запись или суффикс в формате X.500, читайте [здесь](#).

IETF (в [RFC 2247](#), имеющем, впрочем, только статус информационного) решила использовать для именования корневой записи структуру доменных имён, например, dc=example, dc=com. Отчасти это связано с тем, что подобное именование основано на доменных именах DNS, являющихся глобально уникальными, отчасти - по некоторым более специфическим причинам, как то: зная либо [SRV-запись DNS](#), либо имя сервера LDAP, можно сделать предположение о суффиксе каталога. То есть, если имя сервера LDAP (полученное напрямую или через SRV-запись) - ldap.example.com, то разумно будет предположить, что суффикс каталога будет dc=example,dc=com. На наш взгляд, не слишком убедительно. О том, как определить корневую запись или суффикс в формате RFC 2247, читайте [здесь](#).

Если у Вас нет доменного имени - что ж, Вам просто не повезло. Если у Вас несколько доменных имён, Вы вольны выбрать любое (Вы можете даже использовать [их все](#)), так как каждое доменное имя уникально.

В итоге, выберите тот формат, который для Вас наиболее удобен. Если очень хочется, можно (с помощью [нескольких DIT на одном сервере](#)) определить DIT с разными форматами корневой записи, а затем связать их с помощью [отсылок](#).

Все три [корневые записи](#) на рисунке ниже являются верными. Две из них могут быть глобально уникальными, одна, сами понимаете, нет. Теперь всё!



Приложение А - LDAP: Определение иерархии объектных классов

Во многих документах утверждается, что в LDIF-файлах, описывающих [DIT](#) и его [записи](#), должна указываться полная иерархия объектных классов, как показано в этом фрагменте LDIF:

```
dn:cn=Jim Bob,ou=people,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Jim Bob
sn: Bob
mail: jimbob@example.com
ou: sales
....
```

Определение объектного класса включает в себя иерархию объектных классов, то есть вышестоящие (SUP) по отношению к нему объектные классы. Все **наборы схемы**, в которых содержатся ВСЕ объектные классы, составляющие иерархию, должны быть известны серверу LDAP (в OpenLDAP это делается с помощью директив [include](#) файла **slapd.conf**). Таким образом, в теории, сервер LDAP способен самостоятельно поднять иерархию объектных классов, и сделать это намного быстрее, чем Вы её напечатаете. Приведенное выше определение может также внести путаницу видимостью того, что в записи присутствует более одного структурного объектного класса - смотрите дополнительные замечания относительно [наследования объектных классов](#).

На практике, OpenLDAP версий 2.x+ способен обработать иерархию объектных классов, таким образом оба фрагмента LDIF (выше и ниже) будут прекрасно работать, в том числе будет возможность доступа к

объектным классам и **атрибутам** из вышестоящей иерархии. Например, будет доступен **атрибут** [street](#) **объектного класса** [organizationalPerson](#).

```
dn:cn=Jim Bob,ou=people,dc=example,dc=com
objectclass: inetOrgPerson
cn: Jim Bob
sn: Bob
mail: jimbob@example.com
ou: sales
....
```

Внимание: Если Вы экспортируете последнее определение из OpenLDAP, на выходе будет именно то, что Вы указали в LDIF, с помощью которого записи были созданы. Не все серверы LDAP способны автоматически обработать иерархию объектных классов. Если Вы затем попытаетесь загрузить экспортированный LDIF в LDAP-сервер, не способный обработать иерархию объектных классов, скорее всего Вы не получите желаемого результата. Если Вы используете только серверы с поддержкой обработки иерархии (например, только OpenLDAP), то можно сэкономить время, не печатая лишнего.

Приложение А - LDAP: OID

OID (Object Identifier, идентификатор объекта) - это **глобально уникальный** номер, идентифицирующий объекты. Глобально-уникальный означает, что во всей вселенной существует единственная организация, отвечающая за определение объекта и его функциональность. Такой организацией может быть международная группа стандартизации, национальная организация или частное предприятие (смотрите обсуждение ниже). Следующее за номером определение OID содержит информацию, состоящую из двух частей: текстового описания и некоторого кода в синтаксисе [ASN.1](#), представляющего собой формальное определение объекта.

OID определяются в Abstract Syntax Notation One ([ASN.1](#)), разработанной ITU-T.

В LDAP (X.500) **OID** используются для идентификации **объектных классов, атрибутов, синтаксисов (типов данных), правил соответствия, механизмов протокола, элементов управления (controls), расширенных операций (extended operations) и поддерживаемых возможностей (supported features)**.

OID представляет собой структурированный в виде дерева ряд чисел, разделённых точкой, читаемый СЛЕВА НАПРАВО. Примеры OID:

```
2.5.6 # OID объектных классов x.500
2.5.6.2 # OID объектного класса country
1.3.6.1.4.1.1446 # Critical Angle - используется во многих определениях LDAP
1.3.6.1.4.1.311 # OID корпорации microsoft
1.2.840.113556 # OID американского подразделения корпорации microsoft
2.16.840.1.113730 # Netscape - используется во многих определениях LDAP
```

Дерево OID организовано СЛЕВА НАПРАВО, таким образом самое левое число означает самый верхний уровень в иерархии и указывает на международную организацию, ответственную за делегирование полномочий по назначению следующих в этом ряду чисел. Самый верхний уровень может быть одним из следующих чисел:

Число	Организация
0	itu (ITU-T)
1	iso ISO
2	объединение itu-iso

Целиком процесс назначения номеров можно проследить с помощью [этого сайта](#). В данной же статье выполнен обзор только наиболее значимых из используемых в LDAP OID, а также порядка их назначения.

OID 2.5.x

Базовый OID **2.5** был назначен **itu-iso** (смотрите таблицу выше) группе изучения X.500; таким образом, номера, начинающиеся с 2.5, например, 2.5.6.x или 2.5.4.x, выделяются (и определяются) этой группой стандартизации.

OID 1.3.6.1.4.1.x

Базовый OID 1.3.6.1.4.1 - это последовательность номеров частного предприятия Интернет, назначаемых [IANA](#). Любая организация может подать заявку на номер предприятия. Значения СПРАВА от данного полученного номера могут затем использоваться по своему усмотрению. Эти номера могут быть записаны в виде iso.org.dod.internet.private.enterprise, где числа заменены именами, имеющими больше смысла. Трансляция чисел в имена определена в RFC 2578 - 2580.

OID 1.3.6.1.4.1.4203 выделен для OpenLDAP. Многие из используемых OpenLDAP OID имеют форму 1.3.6.1.4.1.1446, которую некоторые считают исторической и взятой из оригинальных спецификаций LDAP, ещё до создания организации OpenLDAP. Просто какой-то детективный роман.

Если есть потребность создать новые **объектные классы** или **атрибуты**, обычно прибегают к получению номера частного предприятия от IANA. Повторно использовать уже существующие OID или инвертировать номера - очень скверная штука (Very Bad Thing™), однажды это может Вам аукнуться.

OID используются рядом протоколов IETF, включая SNMP. Не существует установленных правил ассоциации OID с какими-либо пространствами имён; но мы предполагаем, что первая цифра после номера предприятия (**arc**) используется для идентификации протокола, а следующие ассоциируются с объектами этого протокола, например:

```
# в строках ниже X - это выданный IANA номер частного предприятия
1.3.6.1.4.1.X.1 - назначается объектам SNMP
1.3.6.1.4.1.X.2 - назначается объектам LDAP
1.3.6.1.4.1.X.2.1 - назначается синтаксисам LDAP
1.3.6.1.4.1.X.2.2 - назначается правилам соответствия LDAP
1.3.6.1.4.1.X.2.3 - назначается атрибутам LDAP
1.3.6.1.4.1.X.2.4 - назначается объектным классам LDAP
1.3.6.1.4.1.X.2.5 - назначается поддерживаемым возможностям LDAP
1.3.6.1.4.1.X.2.9 - назначается механизмам протокола LDAP
1.3.6.1.4.1.X.2.10 - назначается элементам управления LDAP
1.3.6.1.4.1.X.2.11 - назначается расширенным операциям LDAP
```

И хотя многие из вышеперечисленных категорий могут никогда не быть использованы - думайте масштабно!

OID 1.2.840.x и 2.16.840.x

В базовом OID 1.2.840 **iso** (1) выделила номер 2 разделу стран-членов (member-country), а следующий за ним номер 840 - США. В этом пространстве можно выделять номера для национальных организаций этой страны (в данном случае США).

Базовый OID 2.16.840 - это вариант выделения пространства для стран от объединения iso-itu (2). Здесь 16 - это страны, а 840 - США.

Порядок назначения OID и их ассоциацию с определениями объектов можно идентифицировать с помощью [этого замечательного сайта](#). Примечание: во многих OID на этом сайте есть ссылка на дополнительную информацию на сайте oid.elibel.tm.fr, который, судя по всему, был заброшен, а затем перевоплотился в [www.oid-info.com](#). Чтобы получить правильную ссылку, просто замените в URL, начинающимся с oid.elibel.tm.fr, данную строку на www.oid-info.com/get, и удалите .html в конце этого URL. Другой способ, - возможно, более быстрый, - выполнить поиск на странице базового поиска (по приведённой выше ссылке).

LDAP: Определение корневой записи или суффикса в формате RFC 2247

Запись самого верхнего уровня в LDAP **DIT** (Directory Information Tree, информационном дереве каталога)

в мире LDAP неоднозначно называется [корневой \(root\)](#), [базовой \(base\)](#) или [суффиксом \(suffix\)](#), в зависимости от документации, её автора, дня недели или каких-либо других переменных величин, доподлинно нам не известных.

Термин [Root DSE](#) определяет своего рода супер-корневую запись (суффикс), содержащую определения всех DIT, поддерживаемых сервером LDAP (в операционном атрибуте **namingContexts**), а также другие [операционные объекты](#).

Существует несколько [методов](#) определения корневой записи, или суффикса.

На этой странице описывается метод, основанный на RFC 2247 (доменных именах). В дальнейших рассуждениях подразумевается, что доменное имя рассматриваемой организации - example.net, трансформирующееся в DN dc=example, dc= net.

В RFC определено два разных подхода реализации данного метода. Один из них использует структурный объектный класс **domain**. Однако, гораздо более широкое распространение получил подход с использованием вспомогательного объектного класса **dcObject** совместно со структурным объектным классом **organization**, имеющим всего один обязательный атрибут o (organizationName), хотя с тем же успехом можно использовать структурный объектный класс organizationalUnit с обязательным атрибутом ou. Фрагмент LDIF для добавления корневой записи или суффикса:

```
## ОПРЕДЕЛЯЕМ DIT ROOT/BASE/SUFFIX ####
## используется формат RFC 2377
## замените встречающиеся ниже example и net на требуемые
## или, для экспериментов, оставьте как есть

## dcObject - это ВСПОМОГАТЕЛЬНЫЙ объектный класс и, кроме него, запись
## ДОЛЖНА иметь СТРУКТУРНЫЙ объектный класс (в данном случае, organization)
# это последовательность ЗАПИСИ и ей предшествует ПУСТАЯ СТРОКА

dn: dc=example,dc=net
dc: example
description: Необязательное описание. Определение корневой записи или суффикса RFC 2247. В
эту строку можно поместить столько текста, сколько хотите
в этой строке продолжение информации из предыдущей строки вплоть до 32Kb
строка оканчивается либо на <CR>, либо на <CR><LF>, то есть отрабатывается ENTER
с систем как Windows, так и *nix - новая строка ДОЛЖНА начинаться с ОДНОГО ПРОБЕЛА
objectClass: dcObject
objectClass: organization
o: Example, Inc.
```

Примечания:

1. Создаётся впечатление, что LDAP требуется, чтобы в DIT был реально добавлен только самый левый атрибут корневой записи/суффикса. Почему это так, мы не знаем, изученные нами спецификации ничего по этому поводу не говорят.
2. Если в доменном имени несколько меток ссTLD, как, например, в example.net.br, соответственно нужно использовать такой фрагмент:

```
## ОПРЕДЕЛЯЕМ DIT ROOT/BASE/SUFFIX ####
## используется формат RFC 2377
## замените встречающиеся ниже example и net на требуемые
## или, для экспериментов, оставьте как есть

## dcObject - это ВСПОМОГАТЕЛЬНЫЙ объектный класс и, кроме него, запись
## ДОЛЖНА иметь СТРУКТУРНЫЙ объектный класс (в данном случае, organization)
# это последовательность ЗАПИСИ и ей предшествует ПУСТАЯ СТРОКА

dn: dc=example,dc=net, dc=br
dc: example
description: Необязательное описание. Определение корневой записи или суффикса RFC
2247. В эту строку можно поместить столько текста, сколько хотите
в этой строке продолжение информации из предыдущей строки вплоть до 32Kb
строка оканчивается либо на <CR>, либо на <CR><LF>, то есть отрабатывается ENTER
с систем как Windows, так и *nix - новая строка ДОЛЖНА начинаться с ОДНОГО ПРОБЕЛА
objectClass: dcObject
objectClass: organization
o: Example, Inc.
```

Соответствующие настройки в slapd.conf (или olcSuffix) OpenLDAP и server.xml ApacheDS будут **suffix "dc=example,dc=net,dc=br"**.

В файле slapd.conf OpenLDAP нужно указать **suffix "dc=example,dc=net"** в разделе **database** (или olcSuffix при использовании cn=config).

В файле server.xml ApacheDS нужно указать **suffix="dc=example,dc=net"** в разделе <partitions> <jdbmPartition ...>.

Добавление последующих записей в dc=example,dc=net показано в данном фрагменте LDIF:

```
## ПЕРВЫЙ уровень иерархии - люди (people)
## для объектных классов используется смешанная форма записи в верхнем и нижнем регистре
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой

dn: ou=people, dc=example,dc=net
ou: people
description: All people in organisation
objectclass: organizationalunit

## ВТОРОЙ уровень иерархии
## ДОБАВЛЯЕМ одну запись в ПЕРВЫЙ уровень (people)
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой
# ou: Human Resources - это название подразделения

dn: cn=Robert Smith,ou=people, dc=example,dc=net
objectclass: inetOrgPerson
cn: Robert Smith
cn: Robert J Smith
cn: bob smith
sn: smith
uid: rjsmith
userpassword: rJsmith
carlicense: HISCAR 123
```

Приложение А - LDAP: Текстовые поисковые фильтры

Текстовая форма поискового фильтра определена в [RFC 4515](#), немного улучшена в [RFC 4510](#) и значительно расширена с помощью так называемого [соответствия компонентов](#) (component matching, [RFC3687](#)) и общих правил кодирования строк (Generic String Encoding Rules, GSER, [RFC4792](#)).

Примечание: С одной стороны, соответствие компонентов является составной частью основной спецификации LDAPv3 и поэтому не требует OID расширения LDAP (в rootDSE). С другой стороны, соответствие компонентов было определено значительно позже, чем оригинальные спецификации LDAPv3. Поэтому не совсем понятно, насколько широко распространена их реализация и каким образом, кроме как попробовав, можно определить, поддерживает ли какая-то конкретная реализация LDAP эту возможность. componentMatching было анонсировано в OpenLDAP 2.4.8, и включается в сборку только когда установлена переменная окружения LDAP_DEVEL, что в большинстве сборок не является значением по умолчанию. Даже после компиляции с этой переменной не все matchingRules будут доступны - особенно presentMatch. Видимо, соответствие компонентов до сих пор имеет статус "в разработке" (Заметка №5429).

Очевидно, поиск требуется, только если запись не может быть найдена напрямую. То есть, это не то же самое, что 'базовый' DN операций чтения или модификации. Когда запись первоначально добавляется в DIT (как с помощью LDIF-файла, так и через LDAP-клиент), ей ассоциируется DN. В общем случае, во избежание излишних операций поиска, данный DN 'создания' должен быть таким, который чаще всего используется при доступе к записи. Однако, если данная запись используется для аутентификации, то на неё накладываются [дополнительные условия](#).

Основной формат

```
( filter )
```

Фильтры всегда заключаются в круглые скобки.

Простые фильтры с одним выражением

```
attr=value # соответствие (может содержать поисковые шаблоны)
attr~=value # примерное соответствие
attr>=value # больше чем
attr<=value # меньше чем
ИЛИ
```


objectclass=class

Примечание: Тип проводимого сравнения, например с учётом или без учёта регистра символов, определяется свойствами используемого в сравнении атрибута и формой поиска (может быть [EQUALITY](#), [ORDERING](#) или [SUBSTR](#)). В некоторых случаях используемая при поиске строка называется **substring**, это верно лишь в том случае, если она содержит один или несколько поисковых **шаблонов**. [Определение атрибута](#).

В приведённом выше листинге:

соответствие (=) считается истинным, если найдено либо совпадение EQUALITY (без **поисковых шаблонов** в **value**), либо совпадение SUBSTR (при наличии одного или нескольких **поисковых шаблонов** в **value**).

примерное соответствие (~=) считается истинным, если найдено совпадение с помощью одного из двух алгоритмов поиска 'созвучных слов'. Требуется [индекс](#) типа **approx**.

больше чем (>=) считается истинным, когда при лексикографическом сравнении value с содержимым указанного атрибута последнее будет лексикографически равно или больше (то есть возвращаются все строки, в которых значение указанного атрибута лексикографически равно или больше value). Данная форма поиска работает только в том случае, если у атрибута есть правило [ORDERING](#), то есть с очень немногими атрибутами.

меньше чем (<=) считается истинным, когда при лексикографическом сравнении value с содержимым указанного атрибута последнее будет лексикографически равно или меньше (то есть возвращаются все строки, в которых значение указанного атрибута лексикографически равно или меньше value). Данная форма поиска работает только в том случае, если у атрибута есть правило [ORDERING](#), то есть с очень немногими атрибутами.

Поисковые шаблоны (wildcard)

Поисковый шаблон * может использоваться отдельно (самостоятельно) как индикатор **наличия** (то есть в данной записи существует такой атрибут, или в данной записи существует такой объектный класс), либо как классическое **итерационное** значение, в этом случае его присутствие означает "в позиции * могут находиться 0 или более любых символов". В форме objectclass=obj поисковые шаблоны могут использоваться только в качестве индикатора **наличия**.

Примеры простых поисковых выражений

```
(mail=*) # возвращает все записи, у которых есть атрибут mail
(objectclass=*) # возвращает все записи
(mail=*@*) # возвращает записи, значение атрибута mail которых соответствует правильному
почтовому адресу формата RFC822
(sn=smith) # возвращает точное соответствие Smith, но НЕ Smit
(sn=s*) # возвращает записи с фамилиями, начинающимися на s или S
(cn=*a*i*) # возвращает записи, в общепринятых именах которых присутствуют сразу a и i в любом
месте
(telephonenumber=*555) # возвращает записи с телефонными номерами, оканчивающимися на 555
(objectclass=person) # возвращает записи, в которых используется объектный класс person
```

Примечания:

1. При поиске с фильтром objectclass=person также возвращаются записи с любыми объектными классами, использующими в своей иерархии person, например, записи с объектными классами inetOrgPerson или residentialPerson и т.д.
2. Форма выполнения тестирования также зависит от определения атрибутов (определения [ASN.1](#)). Например, **sn** и **cn** являются частью семейства атрибутов **name**, при определении которого задано [правило соответствия caseIgnoreMatch](#), означающее, что соответствие будет искаться БЕЗ учёта регистра символов, то есть при поиске с фильтром sn=a будут найдены и a, и A.

Простые фильтры с комбинированными выражениями

Два или более выражения могут быть объединены (или вложены) с помощью & (логическое И), ! (логическое НЕ) и | (логическое ИЛИ):

```
(&(exp1)(exp2)(exp3)) # exp1 И exp2 И exp3
(|(exp1)(exp2)(exp3)) # exp1 ИЛИ exp2 ИЛИ exp3
(! (exp1)) # НЕ exp1
(&(! (exp1)) (! (exp2))) # НЕ exp1 И НЕ exp2
```

НЕ (!) смотрится несколько проблематично, зато логично (может быть), и работает только в приведённой выше форме. Смотрите также примеры ниже:

Примеры простых фильтров с комбинированными выражениями

```
(&(mail=*)(cn=*r)(sn=s*)) # есть атрибут mail И cn заканчивается на R И sn начинается с s
(|(sn=a*)(sn=b*)(sn=c*)) # sn начинается с a ИЛИ с b ИЛИ с c
(! (sn=a*)) # записи с sn НЕ начинающимся с a
(&(! (sn=a*)) (! (sn=b*))) # записи с sn НЕ начинающимся с a И НЕ начинающимся с b
(|(sn=*a) (! (sn=s*))) # записи с sn, заканчивающимся на a И НЕ начинающимся с s
# классическая простая ошибка:
(&(sn=a*)(sn=b*)(sb=c*)) # условие невыполнимо, никогда ничего не возвращается
```

Поиск специальных символов

Если требуется поиск по шаблону, включающему специальные символы (*) (\ или NULL), эти символы должны быть **экранированы** с использованием формата '\code', где code - два шестнадцатеричных символа, представляющие ASCII-код символа. Аналогично поиск по любому двоичному значению может быть осуществлён с помощью его шестнадцатеричного представления.

```
\2a заменяет или экранирует *
\28 заменяет или экранирует (
\29 заменяет или экранирует )
\5c заменяет или экранирует \
\00 заменяет или экранирует NUL
\xx поиск шестнадцатеричного значения
где xx лежит в диапазоне 00 - FF
```

Примеры поиска с использованием экранирования

```
(cn=*\2a*) # поиск символа *, расположенного в любом месте в cn
(file=d:\5cmyfile.html) # поиск d:\myfile
(description=*\28*\29) # поиск ( и ), расположенных в любом месте, но в этой последовательности
(bin=\5b\04) # поиск двоичного значения 5b04
```

Расширение возможностей поиска соответствия

Поведение по умолчанию при поиске по любому атрибуту определяется [правилами соответствия](#) данного атрибута отдельно для разных типов поиска (EQUALITY, SUBSTR или ORDERING). Это поведение может быть переопределено путём указания заменяющего правила соответствия (либо имени правила, либо его OID).

```
# для sn поведение по умолчанию при сравнении EQUALITY
# caseIgnoreMatch (2.5.13.2)
sn=smith

# переопределим соответствие EQUALITY, чтобы оно зависело от регистра символов
sn:caseExactMatch:=Smith
# то же самое с помощью OID
sn:2.5.13.5:=Smith

# если при поиске встречаются поисковые шаблоны,
# применяется правило соответствия SUBSTR

# для sn поведение по умолчанию при сравнении SUBSTR
# caseIgnoreSubstringMatch
sn=*s* # находит Smith или smith

# переопределим соответствие SUBSTR, чтобы оно зависело от регистра символов
sn:caseExactSubstringMatch:=*S* # находит только Smith
# то же самое с помощью OID
sn:2.5.13.7:=*S*
```

Используя данный процесс переопределения, можно задавать поисковые критерии, включающие возможности, не определённые в самом атрибуте, такие как ORDERING (которые очень редко встречаются в определениях атрибутов).

Компоненты DN

Существует возможность указать, что любая часть данных из значений атрибутов базового DN может быть также включена в поиск. Это можно сделать с помощью ключевого слова **dn** внутри поискового выражения, как показано ниже:

```
# указывает, что значение dc, соответствующее com, может присутствовать в DN
# или конечная целевая запись определяется базовым DN и диапазоном
dc:dn:=com
```

Соответствие компонентов (Component Matching) определяется примерно по тому же принципу, что и расширенные фильтры. Оно описано [отдельно](#).

LDAP: Информация о сравнении и поиске

Следующие замечания могут оказаться полезными при проектировании и разработке LDAP-систем или клиентских интерфейсов.

Сравнения строк

При сравнении строк типа [DirectoryString](#), [IA5String](#), [PrintableString](#) используются следующие правила:

1. Начальные и конечные пробельные символы удаляются.
2. При наличии нескольких пробельных символов подряд они заменяются одним пробелом.
3. При [поиске](#) с использованием формы `~=` у атрибута, по которому происходит сравнение, требуется наличие [индекса](#) `approx`.
4. При [поиске](#) с использованием форм `<=` или `>=` у атрибута, по которому происходит сравнение, требуется наличие правила [ORDERING](#). У большинства атрибутов оно не определено.

Приложение А - LDAP: Определение простой корневой записи или суффикса

Запись самого верхнего уровня в LDAP [DIT](#) (Directory Information Tree, информационном дереве каталога) в мире LDAP неоднозначно называется [корневой \(root\)](#), [базовой \(base\)](#) или [суффиксом \(suffix\)](#), в зависимости от документации, её автора, дня недели или каких-либо других переменных величин, доподлинно нам не известных.

Термин [Root DSE](#) определяет своего рода супер-корневую запись (суффикс), содержащую определения всех DIT, поддерживаемых сервером LDAP (в операционном атрибуте **namingContexts**), а также другие [операционные объекты](#).

Существует несколько [методов](#) определения корневой записи или суффикса.

На этой странице описывается метод, основанный на **ou**. Он самый простой (и, обычно, самый короткий), и, в то же время, удовлетворяющий всем требуемым критериям.

Данное определение будет использовать структурный объектный класс **organizationalUnit**, у которого всего один обязательный атрибут `ou` (`organizationalUnitName`). Фрагмент LDIF для добавления корневой записи или суффикса:

```
## ОПРЕДЕЛЯЕМ DIT ROOT/BASE/SUFFIX ####
## используется простой формат ou
## замените имя root на любой подходящий текст
## типа my, gobbledegook, suffix, что-либо ещё

## organizationalUnit - это СТРУКТУРНЫЙ объектный класс
# требующий только наличия атрибута ou (organizationalUnitName)
# это последовательность ЗАПИСИ и ей предшествует ПУСТАЯ СТРОКА

dn: ou=root
ou: root
```

```
objectclass: organizationalUnit
description: Необязательное описание. Определение тривиальной корневой записи или суффикса.
В эту строку можно поместить столько текста, сколько хотите
в этой строке продолжение информации из предыдущей строки вплоть до 32Kb
строка оканчивается либо на <CR>, либо на <CR><LF>, то есть отрабатывается ENTER
с систем как Windows, так и *nix - новая строка ДОЛЖНА начинаться с ОДНОГО ПРОБЕЛА
```

В файле slapd.conf OpenLDAP нужно указать **suffix "ou=root"** в разделе **database** (или olcSuffix при использовании cn=config).

В файле server.xml ApacheDS нужно указать **suffix="root"** в разделе <partitions><jdbmPartition ...>.

Добавление последующих записей показано в данном фрагменте LDIF:

```
## ПЕРВЫЙ уровень иерархии - люди (people)
## для объектных классов используется смешанная форма записи в верхнем и нижнем регистре
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой

dn: ou=people, ou=root
ou: people
description: All people in organisation
objectclass: organizationalUnit

## ВТОРОЙ уровень иерархии
## ДОБАВЛЯЕМ одну запись в ПЕРВЫЙ уровень (people)
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой
# ou: Human Resources - это название подразделения

dn: cn=Robert Smith,ou=people,ou=root
objectclass: inetOrgPerson
cn: Robert Smith
cn: Robert J Smith
cn: bob smith
sn: smith
uid: rjsmith
userpassword: rJsmith
carlicense: HISCAR 123
```

Приложение А - LDAP: Некоторые размышления по структурированию каталогов

В этой заметке в общих чертах обсуждается структурирование каталогов LDAP. Структура каталога - очень спорный предмет, по этой теме написаны целые тома. Следующие замечания по этому вопросу могут Вам либо помочь, либо нет.

Рассмотрим пример простой адресной книги для выделения нескольких общих принципов, которые можно применить и к каталогу оборудования, и вообще к чему угодно.

Плоскость приветствуется

Каталоги в целом обычно имеют очень плоскую структуру, состоящую из двух или трёх уровней иерархии - большее количество встречается редко. Хотя, на первый взгляд, это противоречит интуитивному подходу людей, привыкших работать с классическими базами данных. Вспомните, что LDAP **гораздо** серьезней оптимизирован на работу с данными, находящимися НА ОДНОМ УРОВНЕ иерархии, нежели на поиск данных ВВЕРХ и ВНИЗ по иерархии. В этом и кроется суть мощных методов [индексирования](#).

Для иллюстрации достаточно простого примера: когда смотришь на компанию и разрабатываешь структуру каталога, довольно очевидным представляется первоначальное деление по подразделениям компании. НО БУДЕТ ЛИ ЭТО ОПТИМАЛЬНЫМ? Посмотрим на два способа структурирования каталога:



В DIT 1 **подразделение** представляет собой запись **ou**, в DIT 2 сведения о **подразделении** содержатся в атрибуте **ou** в записях, дочерних к **ou=people**. Разница кажется тривиальной.

Теперь давайте посмотрим на то, что происходит при некоторых типичных поисковых запросах:

1. Найти всех людей в sales:

1. DIT 1 - поиск с базовым DN `ou=sales,dc=mycompany,dc=com`, диапазон - one (один уровень), фильтр `cn=*`
2. DIT 2 - поиск с базовым DN `ou=people,dc=mycompany,dc=com`, диапазон - one (один уровень), фильтр `ou=sales`

Практически, одно и то же.

2. Найти всех людей в компании:

1. DIT 1 - поиск с базовым DN `dc=mycompany,dc=com`, диапазон - sub (все уровни), фильтр `cn=*`
2. DIT 2 - поиск с базовым DN `ou=people,dc=mycompany,dc=com`, диапазон - one (один уровень), фильтр `cn=*`

Структура 2 выигрывает в скорости и простоте.

Лень приветствуется

Придерживаясь принятых нами структур, выполним простую задачу: переведём Билла из sales в marketing (разные подразделения корпорации).

1. DIT 1 - экспортируем запись Билла в файл LDIF, удалим её из sales, отредактируем файл LDIF, добавим запись в marketing. И будем надеяться, что он не будет переводиться слишком часто.
2. DIT 2 - изменим атрибут **ou** записи Билла с sales на marketing.

Интересно, кто выиграл этот раунд?

Так в чём же заключается принцип?

Приложение А - LDAP: Типы данных

LDAP поддерживает подмножество (и довольно приличное подмножество) типов данных X.500. Каждый тип данных определяется своим синтаксисом (полный список синтаксисов, поддерживаемых OpenLDAP [здесь](#)). Наиболее интересные из них мы рассмотрим подробнее.

При сравнении и поиске атрибутов LDAP выполняет кое-какие классные штуки. Несколько слов об этом [здесь](#).

Содержание

Строки

Числа (целые числа)

Формат времени
Телефонные номера
Логический тип
Двоичный тип
Distinguished Name
Битовые строки
Поддерживаемые синтаксисы LDAP

Строки LDAP

Строки - наиболее распространённый тип данных. Существует целый ряд строковых типов.

1. **IA5String** (OID=1.3.6.1.4.1.1466.115.121.1.26). Набор символов **IA5 (почти ASCII)**, **7 бит**. НЕ разрешены расширенные символы, такие как é, Ø, å и т.п.
2. **DirectoryString** (OID=1.3.6.1.4.1.1466.115.121.1.15). ISO-10646 (Unicode) кодировка UTF-8 - кодировочная система с нефиксированным количеством бит на символ. Включает в себя IA5/ASCII в качестве подмножества - **кое-что об этом**. Поддерживает расширенные символы, такие как é, Ø, å и т.п. Поддерживает **правила соответствия caseIgnoreMatch** и **caseIgnoreSubstringsMatch**.
3. **PrintableString** (OID=1.3.6.1.4.1.1466.115.121.1.44). Строка в кодировке **IA5 (почти ASCII)**, с набором символов, ограниченным **представлением р раздела 4.1 RFC 2252**. НЕ разрешены расширенные символы, такие как é, Ø, å и т.п. Поддерживает **правила соответствия caseIgnoreMatch** и **caseIgnoreSubstringsMatch**.
4. **OctetString** (OID=1.3.6.1.4.1.1466.115.121.1.40). Рассматривается как набор **8-битных** байт в чистом виде. Может быть, а может и не быть пригодна для печати и чтения людьми. Обычно используется для паролей. Поддерживает **правила соответствия octetStringMatch** и **octetStringOrderingMatch**.
5. **PostalAddress** (OID=1.3.6.1.4.1.1466.115.121.1.41). Специальный формат, использующий ISO-10646 (Unicode) кодировку UTF-8 с разделителями '\$'. Используется для генерации пригодных для печати меток и другого вывода. НЕ разрешены расширенные символы, такие как é, Ø, å и т.п. Поддерживает **правила соответствия caseIgnoreListMatch** и **caseIgnoreListSubstringsMatch**.
6. **CountryString** (OID=1.3.6.1.4.1.1466.115.121.1.11). Специальное поле, использующее набор символов **IA5 (почти ASCII)**, длина которого ограничена ровно двумя символами, представляющими собой код страны согласно ISO 3166. Если требуется полное название страны, используйте **friendlyCountryName**. Поддерживает **правила соответствия caseIgnoreMatch** и **caseIgnoreSubstringsMatch**.
7. **NumericString** (OID=1.3.6.1.4.1.1466.115.121.1.36) использует числовое подмножество (**представление d раздела 4.1 RFC 2252**) набора символов **IA5 (почти ASCII)**. Поддерживает **правила соответствия numericStringMatch** и **numericStringSubstringsMatch**.



Числа LDAP

Числовые значения реально хранятся в виде строк, но могут, если в определении **атрибута** указано правило соответствия ORDERING, позволять выполнять числовые сравнения, то есть 100 > 2. Тип **Integer** определяет 32-битное значение с учётом знака, и, таким образом, может принимать значения в диапазоне от $2^{31}-1$ (2,147,483,647) до -2^{31} (2,147,483,648).

1. **Integer** (OID=1.3.6.1.4.1.1466.115.121.1.27). Поддерживает **правила соответствия integerMatch** (EQUALITY) и **integerOrderingMatch** (ORDERING).

Стандартные наборы схемы содержат мало **целочисленных** атрибутов. Большинство числовых полей используют один из приведённых выше строковых типов, особенно тип **NumericString**.

Формат времени LDAP

1. **GeneralizedTime** (OID=1.3.6.1.4.1.1466.115.121.1.24). Поддерживает [правила соответствия generalizedTimeMatch](#) и [generalizedTimeOrderingMatch](#).

Телефонные номера LDAP

1. **TelephoneNumber** (OID=1.3.6.1.4.1.1466.115.121.1.50). Поддерживает [правила соответствия telephoneNumberMatch](#) и [telephoneNumberSubstringsMatch](#).

Логический тип LDAP

1. **Boolean** (OID=1.3.6.1.4.1.1466.115.121.1.7). Поддерживает [правило соответствия booleanMatch](#).

Двоичный тип LDAP

1. **Binary** (OID=1.3.6.1.4.1.1466.115.121.1.5). Для двоичных атрибутов нет [правил соответствия](#).

Distinguished Name LDAP

1. **DN** (OID=1.3.6.1.4.1.1466.115.121.1.12). Поддерживает [правило соответствия distinguishedNameMatch](#).

Битовые строки LDAP

1. **BitString** (OID=1.3.6.1.4.1.1466.115.121.1.6). Поддерживает [правило соответствия bitStringMatch](#).

Поддерживаемые OpenLDAP синтаксисы

В приведённом ниже списке перечислены все синтаксисы, поддерживаемые OpenLDAP. Этот список получен с помощью [rootDSE subschema](#). Для того, чтобы найти определения [OID](#), используйте этот [сайт](#), спасший уже не одну жизнь.

Приведённый ниже список может быть получен с помощью следующей команды:

```
ldapsearch -H ldap://ldap.mydomain.com -x -s base -b "cn=subschema"
ldapsyntaxes
# Список атрибутов, которые могут быть перечислены:
# matchingruleuse ldapsyntaxes matchingrules attributetypes
# (всё это - коллекции), а также
# createtimestamp modifytimestamp
# Если Вы используете единственный знак +, Вы получите огромный
# список всего, что знает LDAP-сервер.
```

```
# Subschema
dn: cn=Subschema
ldapsyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.4
  DESC 'Audio' X-NOT-HUMAN-READABLE 'TRUE' )
ldapsyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.5
  DESC 'Binary' X-NOT-HUMAN-READABLE 'TRUE' )
ldapsyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.6
  DESC 'Bit String' )
ldapsyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.7
  DESC 'Boolean' )
ldapsyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.8
  DESC 'Certificate' X-BINARY-TRANSFER-REQUIRED
  'TRUE' X-NOT-HUMAN-READABLE 'TRUE' )
ldapsyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.9
  DESC 'Certificate List' X-BINARY-TRANSFER-REQUIRED
  'TRUE' X-NOT-HUMAN-READABLE 'TRUE' )
ldapsyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.10
  DESC 'Certificate Pair' X-BINARY-TRANSFER-REQUIRED
  'TRUE' X-NOT-HUMAN-READABLE 'TRUE' )
ldapsyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.11
```

```

DESC 'Country String' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.12
DESC 'Distinguished Name' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.15
DESC 'Directory String' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.22
DESC 'Facsimile Telephone Number' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.24
DESC 'Generalized Time' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.26
DESC 'IA5 String' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.27
DESC 'Integer' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.28
DESC 'JPEG' X-NOT-HUMAN-READABLE 'TRUE' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.34
DESC 'Name And Optional UID' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.36
DESC 'Numeric String' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.38
DESC 'OID' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.39
DESC 'Other Mailbox' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.40
DESC 'Octet String' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.41
DESC 'Postal Address' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.44
DESC 'Printable String' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.49
DESC 'Supported Algorithm' X-BINARY-TRANSFER-REQUIRED
'TRUE' X-NOT-HUMAN-READABLE 'TRUE' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.50
DESC 'Telephone Number' )
ldapSyntaxes: ( 1.3.6.1.4.1.1466.115.121.1.52
DESC 'Telex Number' )
ldapSyntaxes: ( 1.3.6.1.1.1.0.0 DESC 'RFC2307
NIS Netgroup Triple' )
ldapSyntaxes: ( 1.3.6.1.1.1.0.1
DESC 'RFC2307 Boot Parameter' )
ldapSyntaxes: ( 1.2.826.0.1.3344810.7.1
DESC 'Serial Number and Issuer' )

```



LDAP: Определение корневой записи или суффикса в формате X.500

Запись самого верхнего уровня в LDAP **DIT** (Directory Information Tree, информационном дереве каталога) в мире LDAP неоднозначно называется **корневой (root)**, **базовой (base)** или **суффиксом (suffix)**, в зависимости от документации, её автора, дня недели или каких-либо других переменных величин, доподлинно нам не известных.

Термин **Root DSE** определяет своего рода супер-корневую запись (суффикс), содержащую определения всех DIT, поддерживаемых сервером LDAP (в операционном атрибуте **namingContexts**), а также другие **операционные объекты**.

Существует несколько **методов** определения корневой записи или суффикса.

На этой странице описывается метод, основанный на подходе X.500. В нём используется **ou=name**, **c=country** (хотя никаких формальных стандартов определения корневой записи или суффикса X.500 не существует).

Как нам кажется, самый простой способ - использовать структурный объектный класс **organizationalUnit**, у которого всего один обязательный атрибут **ou** (organizationalUnitName), и специальный объектный класс **extensibleObject** для добавления атрибута **c** (country). Фрагмент LDIF для добавления корневой записи или суффикса:

```
## ОПРЕДЕЛЯЕМ DIT ROOT/BASE/SUFFIX #####
## используется формат X.500
## замените example inc. и us любым подходящим текстом

## organizationalUnit - это СТРУКТУРНЫЙ объектный класс
# требующий только наличия атрибута ou (organizationalUnitName)
# но не содержащий с (country), который добавляется
# с помощью extensibleObject
# это последовательность ЗАПИСИ и ей предшествует ПУСТАЯ СТРОКА

dn: ou=Example Inc.,c=us
ou: Example Inc.
objectclass: organizationalUnit
description: Необязательное описание. Определение корневой записи или суффикса X.500. В эту строку можно поместить столько текста, сколько хотите
в этой строке продолжение информации из предыдущей строки вплоть до 32Kb
строка оканчивается либо на <CR>, либо на <CR><LF>, то есть отрабатывается ENTER
с систем как Windows, так и *nix - новая строка ДОЛЖНА начинаться с ОДНОГО ПРОБЕЛА
c: us
objectClass: extensibleObject
```

Примечания:

1. Строки **c: us** и **objectClass: extensibleObject** могут быть исключены из этого фрагмента LDIF, и LDAP продолжит работу как ни в чём не бывало. Создаётся впечатление, что LDAP требуется, чтобы в DIT был реально добавлен только самый левый атрибут корневой записи/суффикса. Почему это так, мы не знаем, изученные нами спецификации ничего по этому поводу не говорят.
2. Официально, чтобы соответствовать реальному названию организации, Example Inc. должен записываться как Example, Inc. Если используется такой формат, то запятая, к какому бы месту DN она не встретилась, должна экранироваться следующим образом: Example\, Inc.,c=us.

В файле slapd.conf OpenLDAP нужно указать **suffix "ou=Example Inc.,c=us"** в разделе **database** (или **olcSuffix** при использовании **cn=config**).

В файле server.xml ApacheDS нужно указать **suffix="ou=Example Inc.,c=us"** в разделе **<partitions><jdbmPartition ...>**.

Добавление последующих записей показано в данном фрагменте LDIF:

```
## ПЕРВЫЙ уровень иерархии - люди (people)
## для объектных классов используется смешанная форма записи в верхнем и нижнем регистре
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой

dn: ou=people, ou=Example Inc.,c=us
ou: people
description: All people in organisation
objectclass: organizationalUnit

## ВТОРОЙ уровень иерархии
## ДОБАВЛЯЕМ одну запись в ПЕРВЫЙ уровень (people)
# это последовательность ЗАПИСИ, она должна предваряться ПУСТОЙ строкой
# ou: Human Resources - это название подразделения

dn: cn=Robert Smith,ou=people, ou=Example Inc.,c=us
objectclass: inetOrgPerson
cn: Robert Smith
cn: Robert J Smith
cn: bob smith
sn: smith
uid: rjsmith
userpassword: rJsmith
carlicense: HISCAR 123
```

Приложение В: Ресурсы LDAP

[Open Source программное обеспечение LDAP](#)
[Реализации LDAP API](#)
[Open Source инструменты и браузеры LDAP](#)
[Web-сайты LDAP](#)
[Сайты ASN.1 и OID](#)
[Связанные сайты \(например, базы данных\)](#)

Open Source программное обеспечение LDAP

По следующим ссылкам можно найти дополнительную информацию о Open Source программном обеспечении LDAP.

[Apache Directory Project](#)

Проект Apache Directory - набор приложений, написанных на Java, состоящий из LDAPv3 сервера (с триггерами, хранимыми процедурами, представлениями и другими возможностями, обычно ассоциирующимися с миром транзакционных баз данных), а также со встроенной поддержкой Kerberos5, протокола изменения пароля (Password Change protocol) и скромных возможностей DNS и DHCP. Поставляется с фиксированным (в настоящее время) механизмом манипуляции данными (JDBM). В проект также входит Apache Directory Studio - Java-клиент (на фреймворке eclipse), оптимизированный для использования с Apache Directory Server. Запускается на различных платформах. Лицензия Apache V 2.0.

[389 Directory Server \(ранее Fedora Directory Server\)](#)

Проект Fedora Directory Server - часть спонсируемого Red Hat проекта разработки Fedora. Написан на C и использует основу кода 10-летней давности (родословная восходит к Michigan University -> Netscape Directory Server -> Redhat Directory Server-> Fedora Directory Server), и только недавно стал Open Source. Очень амбициозный набор функций - особенно операционные возможности. Использует механизм манипуляции данными BDB (Berkeley Database). В настоящее время только для платформ *nix. GPL.

[OpenDS](#)

Проект OpenDS написан на Java. Относительно новый сервер службы каталогов, разработанный под jav.net, и, кажется, в значительной степени под руководством изначально Sun Microsystems (а теперь под нежной опекой Oracle). Очень амбициозный набор функций. Запускается на различных платформах. Common Development and Distribution License (CDDL).

[OpenLDAP](#)

Высококачественная эталонная разработка LDAP, изначально основанная на работе Мичиганского университета. Активно разрабатывается и широко внедрена. GPL.

[BIND API и драйверы](#)

По следующим ссылкам можно найти информацию о реализациях LDAP API и библиотеках для различных языков.

[BIND DLZ](#)

Библиотека динамической загрузки зон BIND.

[Open Source инструменты и браузеры LDAP](#)

[Apache LDAP Studio](#)

Часть проекта Apache Directory Service. LDAP Studio интегрирован с фреймворком Eclipse, для скачивания которого, в зависимости от требуемого набора функций, может уйти пару дней. Как можно догадаться по пакету Eclipse, LDAP Studio написан на Java и поставляется почти со всем, что Вам только может понадобиться, LDAP-браузер - лишь небольшая часть всего пакета Studio. Apache Software License 2.0 (OS-лицензия, одобренная [Open Source Initiative](#)).

[LDAPExplorerTool](#)

LDAP-браузер - добавление, редактирование, удаление или просмотр записей. Версии для Windows и Linux. Бета-версия (середина 2004), но вполне работоспособен. Нет функции поиска. Лицензия BSD. Хорош для доступа к rootDSE, subschema и т.п.

FusionDirectory

LDAP-браузер - версия только для Linux. Пользовательский интерфейс напоминает проводник/браузер. Похоже, оптимизирован для управления учётными записями пользователей, включая Samba PDC. Написан на PHP. Лицензия GPL V2. Изначально основан на ответвлении от проекта GOsa.

JXPlore

LDAP-браузер, основанный на Java - добавление, редактирование, удаление или просмотр записей. Подарок от CA. Лицензия OSI.

LDAP Browser / Editor

LDAP-браузер, основанный на Java - добавление, редактирование, удаление или просмотр записей. Поддержка LDIF. Версии для Windows и Linux. Не обновлялся с 2001 года, но до сих пор вполне работоспособен. Не Open Source - на ноябрь 2009 г. Argonne Labs (текущее местоположение LDAP Browser/Editor) сделал его доступным через [Open Channel Foundation](#) и требует небольшой платы (на сегодняшний день 35 долларов) за копию. Прекрасные поисковые возможности.

phpLDAPAdmin

LDAP-браузер, основанный на web - добавление, редактирование, удаление или просмотр записей. Написан на PHP - зрелый и активно развивается. Лицензия GNU. Очень мощный.

Web-сайты LDAP

По следующим ссылкам можно найти информацию о LDAP или X.500:

www.bind9.net - ресурсы LDAP и DNS

www.openldap.org - ресурсы LDAP

www.idealx.org - ресурсы LDAP и SAMBA (на французском и английском)

<http://sec.cs.kent.ac.uk/x500book/> On-line книга - "Understanding X.500", ("Понимание X.500"), D.W. Chadwick

Сайты ASN.1 и OID

По следующим ссылкам можно найти информацию о ASN.1 и OIDs:

www.oid-info.com - изумительный сайт по ASN.1 информации. Новый сайт со значительно улучшенным ASN.1 OID интерфейсом.

<http://www.oss.com/asn1/dubuisson.html> - ссылка на книгу Olivier Dubuisson "ASN.1 - Communication between heterogeneous systems", ("ASN.1 - взаимодействие гетерогенных систем"), английская версия в свободном доступе в формате PDF.

<http://www.oss.com/asn1/larmouth.html> - ссылка на книгу профессора John Larmouth "ASN.1 - ASN.1 Complete" ("Полное руководство ASN.1"), английская версия в свободном доступе в формате PDF.

www.alvestrand.no/objectid/ - спасительный сайт, определяющий огромное число номеров OID. Наберите в Google 'oid x.x.x', и, в большинстве случаев, Вы попадёте на этот сайт. Фантастическая работа.

[ASN.1 & BER](#) - "A Layman's guide to a Subset of ASN.1, BER and DER", ("Руководство для неспециалиста по подмножеству ASN.1, BER и DER"), техническое описание защиты информации RSA, Burton S. Kaliski, Jr. В основном сфокусированное на безопасности, милосердно краткое (36 страниц) по сравнению с книгами Dubuisson и Larmouth (каждая около 500 страниц) пособие, - обладающее, однако, большей полнотой охвата.

Связанные сайты

По следующим ссылкам можно найти информацию по темам, связанным с LDAP:

BDB - документация по BDB (sleepy cat - сейчас принадлежит Oracle). Вам может это понадобиться в случае выполнения серьёзной настройки производительности.

Приложение C: LDAP RFC и спецификации X.500

Представленные ниже RFC описывают LDAP или что-либо с ним связанное. Основной репозиторий RFC поддерживается [IETF](http://www.ietf.org), текстовые версии можно найти по адресу www.ietf.org/rfc/rfcXXXX.txt (где XXXX — состоящий из 4 цифр номер RFC, при необходимости дополненный слева нулями). RFC в текстовом формате также можно посмотреть по адресу <http://datatracker.ietf.org/doc/rfcXXXX/> (где XXXX — состоящий из 4 цифр номер RFC, при необходимости дополненный слева нулями), здесь же содержится различная статусная информация по RFC, а также ссылки на просмотр в альтернативных форматах, таких как PDF и HTML. Наконец, существует [страница поиска по RFC](#).

У ITU-T (тяжеловеса мира стандартов) достаточно много спецификаций [X.500](#) (каждая из которых обойдётся Вам в достаточно много денег). Справедливости ради необходимо отметить, что можно получить несколько свободных спецификаций, но нет полной уверенности, что такая политика не изменится. Есть, однако, тенденция к свободной публикации их определений ASN.1 (например, [X.519](#)).

<укоренившаяся привычка> В приведённом ниже списке RFC гиперссылки указывают на текстовые версии, которые мы скопировали на наш сайт, когда эти RFC были выпущены. Мы начинали эту работу очень давно, когда мир был ещё молод и RFC содержались в каких-то странных местах, иногда меняющих своё местоположение, а производительность и доступность этих репозиторий оставляла желать лучшего. Сегодня всё это далеко не так. Тем не менее, мы упорно продолжаем следовать нашей укоренившейся привычке без всякой видимой причины (трюки новые, а пёс-то старый...). Если Вы хотите/предпочитаете/нуждаетесь в чём-то более продвинутом, нежели простой текст, мы рекомендуем воспользоваться приведёнными выше ссылками, если же Вам всего лишь нужно почитать эти несчастные RFC, просто смело жмите на наши ссылки. **</укоренившаяся привычка>**

Примечание переводчика: в русском варианте страницы ссылки указывают на HTML-версии RFC на сайте <http://tools.ietf.org/html/>.

RFC

- [RFC 1274](#) The COSINE and Internet X.500 Schema.
Наборы схемы данных X.500 COSINE и Internet.
P. Barker, S. Kille. Ноябрь 1991 г. Формат: TXT, 92827 байт. Считается устаревшим с выходом RFC4524. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 1823](#) The LDAP Application Program Interface.
Интерфейс программирования приложений LDAP.
T. Howes, M. Smith. Август 1995 г. Формат: TXT, 41081 байт. Статус: INFORMATIONAL (информационный)
- [RFC 2247](#) Using Domains in LDAP/X.500 Distinguished Names.
Использование доменов в уникальных именах LDAP/X.500.
S. Kille, M. Wahl, A. Grimstad, R. Huber, S. Sataluri. Январь 1998 г. Формат: TXT, 12411 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 2251](#) Lightweight Directory Access Protocol (v3).
Облегчённый протокол доступа к службам каталогов, версия 3.
M. Wahl, T. Howes, S. Kille. Декабрь 1997 г. Формат: TXT, 114488 байт. Считается устаревшим с выходом RFC4510, RFC4511, RFC4513, RFC4512. Обновлено с выходом RFC3377, RFC3771. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 2252](#) Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions.
LDAPv3: Определение синтаксиса атрибутов.
M. Wahl, A. Coulbeck, T. Howes, S. Kille. Декабрь 1997 г. Формат: TXT, 60204 байт. Считается устаревшим с выходом RFC4510, RFC4517, RFC4523, RFC4512. Обновлено с выходом RFC3377. Статус: PROPOSED STANDARD (предложенный стандарт)

- RFC 2253** Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names.
LDAPv3: UTF-8 Строковое представление уникальных имён.
M. Wahl, S. Kille, T. Howes. Декабрь 1997 г. Формат: TXT, 18226 байт. Obsoletes RFC1779.
Считается устаревшим с выходом RFC4510, RFC4514. Обновлено с выходом RFC3377. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2254** The String Representation of LDAP Search Filters.
Строковое представление поисковых фильтров LDAP.
T. Howes. Декабрь 1997 г. Формат: TXT, 13511 байт. Obsoletes RFC1960. Считается устаревшим с выходом RFC4510, RFC4515. Обновлено с выходом RFC3377. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2255** The LDAP URL Format.
Формат LDAP URL.
T. Howes, M. Smith. Декабрь 1997 г. Формат: TXT, 20685 байт. Obsoletes RFC1959. Считается устаревшим с выходом RFC4510, RFC4516. Обновлено с выходом RFC3377. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2256** A Summary of the X.500(96) User Schema for use with LDAPv3.
Обзор пользовательской схемы данных X.500(96) для использования с LDAPv3.
M. Wahl. Декабрь 1997 г. Формат: TXT, 32377 байт. Считается устаревшим с выходом RFC4517, RFC4519, RFC4523, RFC4512, RFC4510. Обновлено с выходом RFC3377. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2307** An Approach for Using LDAP as a Network Information Service.
Подход к использованию LDAP в качестве Сетевой Информационной Службы.
L. Howard. Март 1998 г. Формат: TXT, 41396 байт. Статус: EXPERIMENTAL (экспериментальный)
- RFC 2377** Naming Plan for Internet Directory-Enabled Applications.
План именования для сетевых приложений с поддержкой каталогов.
A. Grimstad, R. Huber, S. Sataluri, M. Wahl. Сентябрь 1998 г. Формат: TXT, 38274 байт. Статус: INFORMATIONAL (информационный)
- RFC 2425** A MIME Content-Type for Directory Information.
MIME Content-Type для информации каталога.
T. Howes, M. Smith, F. Dawson. Сентябрь 1998 г. Формат: TXT, 64478 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2426** vCard MIME Directory Profile.
vCard MIME-профиль для каталога.
F. Dawson, T. Howes. Сентябрь 1998 г. Формат: TXT, 74646 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2596** Use of Language Codes in LDAP.
Использование языковых кодов в LDAP.
M. Wahl, T. Howes. Май 1999 г. Формат: TXT, 17413 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2696** LDAP Control Extension for Simple Paged Results Manipulation.
Расширение элементов управления LDAP для манипуляций с простыми постраничными результатами.
C. Weider, A. Herron, A. Anantha, T. Howes. Сентябрь 1999 г. Формат: TXT, 12809 байт. Статус: INFORMATIONAL (информационный)
- RFC 2713** Schema for Representing Java(tm) Objects in an LDAP Directory.
Набор схемы для представления объектов Java(tm) в каталогах LDAP.
V. Ryan, S. Seligman, R. Lee. Октябрь 1999 г. Формат: TXT, 40745 байт. Статус: INFORMATIONAL (информационный)
- RFC 2714** Schema for Representing CORBA Object References in an LDAP Directory.
Набор схемы для представления ссылок на объекты CORBA в каталогах LDAP.
V. Ryan, R. Lee, S. Seligman. Октябрь 1999 г. Формат: TXT, 14709 байт. Статус: INFORMATIONAL (информационный)
- RFC 2739** Calendar Attributes for vCard and LDAP.
Атрибуты календаря для vCard и LDAP.
T. Small, D. Hennessy, F. Dawson. Январь 2000 г. Формат: TXT, 25892 байт. Статус: PROPOSED STANDARD (предложенный стандарт)

- RFC 2798** Definition of the inetOrgPerson LDAP Object Class.
Определение объектного класса LDAP inetOrgPerson.
M. Smith. Апрель 2000 г. Формат: TXT, 32929 байт. Статус: INFORMATIONAL (информационный)
- RFC 2829** Authentication Methods for LDAP.
Методы аутентификации для LDAP.
M. Wahl, H. Alvestrand, J. Hodges, R. Morgan. Май 2000 г. Формат: TXT, 33471 байт. Обновлено с выходом RFC3377. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2849(рус.)** The LDAP Data Interchange Format (LDIF) — Technical Specification.
Техническая спецификация Формата обмена данными LDAP (LDIF).
RFC 2849(ориг.) G. Good. Июнь 2000 г. Формат: TXT, 26017 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2891** LDAP Control Extension for Server Side Sorting of Search Results.
Расширение элементов управления LDAP для сортировки результатов поиска на стороне сервера.
T. Howes, M. Wahl, A. Anantha. Август 2000 г. Формат: TXT, 15833 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 2927** MIME Directory Profile for LDAP Schema.
MIME-профиль каталога для схемы данных LDAP.
M. Wahl. Сентябрь 2000 г. Формат: TXT, 16122 байт. Статус: INFORMATIONAL (информационный)
- RFC 3045** Storing Vendor Information in the LDAP root DSE.
Хранение информации о владельце каталога в корневой записи LDAP.
M. Meredith. Январь 2001 г. Формат: TXT, 10518 байт. Статус: INFORMATIONAL (информационный)
- RFC 3062** LDAP Password Modify Extended Operation.
Расширенная операция изменения паролей LDAP.
K. Zeilenga. Февраль 2001 г. Формат: TXT, 11807 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3296** Named Subordinate References in Lightweight Directory Access Protocol (LDAP) Directories.
Именованные ссылки подчинённости в каталогах LDAP.
K. Zeilenga. Июль 2002 г. Формат: TXT, 27389 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3377** Lightweight Directory Access Protocol (v3): Technical Specification.
Техническая спецификация LDAPv3.
J. Hodges, R. Morgan. Сентябрь 2002 г. Формат: TXT, 9981 байт. Обновляет RFC2251, RFC2252, RFC2253, RFC2254, RFC2255, RFC2256, RFC2829, RFC2830. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3383** Internet Assigned Numbers Authority (IANA) Considerations for the Lightweight Directory Access Protocol (LDAP).
Соображения IANA относительно LDAP
K. Zeilenga. Сентябрь 2002 г. Формат: TXT, 45893 байт. Также смотрите BCP0064. Статус: BEST CURRENT PRACTICE (лучший современный опыт)
- RFC 3384** Lightweight Directory Access Protocol (version 3) Replication Requirements.
Требования репликации Lightweight Directory Access Protocol (version 3).
E. Stokes, R. Weiser, R. Moats, R. Huber. Октябрь 2002 г. Формат: TXT, 66871 байт. Статус: INFORMATIONAL (информационный)
- RFC 3641** Generic String Encoding Rules (GSER) for ASN.1 Types.
Общие правила кодирования строк (GSER) для типов ASN.1.
S. Legg. Октябрь 2003 г. Формат: TXT, 34207 байт. Обновлено с выходом RFC4792. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3671** Collective Attributes in the Lightweight Directory Access Protocol (LDAP).
Коллективные атрибуты в LDAP.
K. Zeilenga. Декабрь 2003 г. Формат: TXT, 17912 байт. Статус: PROPOSED STANDARD (предложенный стандарт)

- RFC 3672** Subentries in the Lightweight Directory Access Protocol (LDAP).
Подзаписи в LDAP.
K. Zeilenga. Декабрь 2003 г. Формат: TXT, 24447 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3673** Lightweight Directory Access Protocol version 3 (LDAPv3): All Operational Attributes.
LDAPv3: все операционные атрибуты.
K. Zeilenga. Декабрь 2003 г. Формат: TXT, 10003 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3674** Feature Discovery in Lightweight Directory Access Protocol (LDAP).
Функция обнаружения в LDAP.
K. Zeilenga. Декабрь 2003 г. Формат: TXT, 10282 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3687** Lightweight Directory Access Protocol (LDAP) and X.500 Component Matching Rules.
Правила соответствия компонентов LDAP и X.500.
S. Legg. Февраль 2004 г. Формат: TXT, 96256 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3698** Lightweight Directory Access Protocol (LDAP): Additional Matching Rules.
LDAP: дополнительные правила соответствия.
K. Zeilenga, Ed.. Февраль 2004 г. Формат: TXT, 17562 байт. Вносит изменения в RFC2798.
Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3703** Policy Core Lightweight Directory Access Protocol (LDAP) Schema.
Набор схемы LDAP для представления информационной модели Policy Core.
J. Strassner, B. Moore, R. Moats, E. Ellesson. Февраль 2004 г. Формат: TXT, 142983 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3712** Lightweight Directory Access Protocol (LDAP): Schema for Printer Services.
Набор схемы LDAP для служб принтеров.
P. Fleming, I. McDonald. Февраль 2004 г. Формат: TXT, 62301 байт. Статус: INFORMATIONAL (информационный)
- RFC 3727** ASN.1 Module Definition for the LDAP and X.500 Component Matching Rules.
Определение модуля ASN.1 для правил соответствия компонентов LDAP и X.500.
S. Legg. Февраль 2004 г. Формат: TXT, 8297 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3771** The Lightweight Directory Access Protocol (LDAP) Intermediate Response Message.
Промежуточные ответные сообщения LDAP.
R. Harrison, K. Zeilenga. Апрель 2004 г. Формат: TXT, 17114 байт. Вносит изменения в RFC2251.
Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3829** Lightweight Directory Access Protocol (LDAP) Authorization Identity Request and Response Controls.
Механизмы управления запросами авторизационных идентификационных сущностей и ответами на них в LDAP.
R. Weltman, M. Smith, M. Wahl. Июль 2004 г. Формат: TXT, 11986 байт. Статус: INFORMATIONAL (информационный)
- RFC 3866** Language Tags and Ranges in the Lightweight Directory Access Protocol (LDAP).
Языковые тэги и диапазоны в LDAP.
K. Zeilenga, Ed.. Июль 2004 г. Формат: TXT, 31501 байт. Отменяет RFC2596. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3876** Returning Matched Values with the Lightweight Directory Access Protocol version 3 (LDAPv3).
Возврат совпавших значений с помощью LDAPv3.
D. Chadwick, S. Mullan. Сентябрь 2004 г. Формат: TXT, 24233 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3909** Lightweight Directory Access Protocol (LDAP) Cancel Operation.
Операция отмены в LDAP.
K. Zeilenga. Октябрь 2004 г. Формат: TXT, 13423 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 3928** Lightweight Directory Access Protocol (LDAP) Client Update Protocol (LCUP).
Протокол обновления клиента LDAP.

R. Meggison, Ed., M. Smith, O. Natkovich, J. Parham. Октябрь 2004 г. Формат: TXT, 36892 байт. Статус: PROPOSED STANDARD (предложенный стандарт)

- [RFC 4370](#) Lightweight Directory Access Protocol (LDAP) Proxied Authorization Control.
LDAP, проксируемый контроль авторизации.
R. Weltman. Февраль 2006 г. Формат: TXT, 10624 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 4373](#) Lightweight Directory Access Protocol (LDAP) Bulk Update/Replication Protocol (LBURP).
Протокол массового обновления/репликации LDAP (LBURP).
R. Harrison, J. Sermersheim, Y. Dong. Январь 2006 г. Формат: TXT, 31091 байт. Статус: INFORMATIONAL (информационный)
- [RFC 4403](#) Lightweight Directory Access Protocol (LDAP) Schema for Universal Description, Discovery, and Integration version 3 (UDDIv3).
Набор схемы LDAP для UDDIv3.
B. Bergeson, K. Boogert, V. Nanjundaswamy. Февраль 2006 г. Формат: TXT, 78747 байт. Статус: INFORMATIONAL (информационный)
- [RFC 4510](#) Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map.
Техническая спецификация LDAP.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 12354 байт. Отменяет RFC2251, RFC2252, RFC2253, RFC2254, RFC2255, RFC2256, RFC2829, RFC2830, RFC3377, RFC3771. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 4511](#) Lightweight Directory Access Protocol (LDAP): The Protocol.
(рус.) Определение протокола LDAP.
- [RFC 4511](#) Под редакцией J. Sermersheim. Июнь 2006 г. Формат: TXT, 150116 байт. Отменяет RFC2251, RFC2830, RFC3771. Статус: PROPOSED STANDARD (предложенный стандарт)
(ориг.)
- [RFC 4512](#) Lightweight Directory Access Protocol (LDAP): Directory Information Models.
LDAP: Информационные модели каталога.
K. Zeilenga, Ed.. Июнь 2006 г. Формат: TXT, 108377 байт. Отменяет RFC2251, RFC2252, RFC2256, RFC3674. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 4513](#) Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms.
LDAP: Методы аутентификации и механизмы безопасности.
R. Harrison, Ed.. Июнь 2006 г. Формат: TXT, 80546 байт. Отменяет RFC2251, RFC2829, RFC2830. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 4514](#) Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names.
LDAP: Строковое представление уникальных имён.
K. Zeilenga, Ed.. Июнь 2006 г. Формат: TXT, 31859 байт. Отменяет RFC2253. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 4515](#) Lightweight Directory Access Protocol (LDAP): String Representation of Search Filters.
LDAP: Строковое представление поисковых фильтров.
M. Smith, Ed., T. Howes. Июнь 2006 г. Формат: TXT, 23885 байт. Отменяет RFC2254. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 4516](#) Lightweight Directory Access Protocol (LDAP): Uniform Resource Locator.
LDAP: URL.
M. Smith, Ed., T. Howes. Июнь 2006 г. Формат: TXT, 3056 байт. Отменяет RFC2255. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 4517](#) Lightweight Directory Access Protocol (LDAP): Syntaxes and Matching Rules.
LDAP: Синтаксисы и правила соответствия.
S. Legg, Ed.. Июнь 2006 г. Формат: TXT, 114285 байт. Отменяет RFC2252, RFC2256. Вносит изменения в RFC3698. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 4518](#) Lightweight Directory Access Protocol (LDAP): Internationalized String Preparation.
LDAP: Подготовка интернационализированных строк.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 28166 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- [RFC 4519](#) Lightweight Directory Access Protocol (LDAP): Schema for User Applications.
LDAP: Набор схемы для пользовательских приложений.
A. Sciberras, Ed.. Июнь 2006 г. Формат: TXT, 64996 байт. Отменяет RFC2256. Вносит изменения

в RFC2247, RFC2798, RFC2377. Статус: PROPOSED STANDARD (предложенный стандарт)

- RFC 4520** Internet Assigned Numbers Authority (IANA) Considerations for the Lightweight Directory Access Protocol (LDAP).
Соглашения IANA относительно LDAP.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 34298 байт. Отменяет RFC3383. Он же BCP0064. Статус: BEST CURRENT PRACTICE (лучший современный опыт)
- RFC 4521** Considerations for Lightweight Directory Access Protocol (LDAP) Extensions.
Соглашения по расширениям LDAP.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 34585 байт. Также смотрите BCP0118. Статус: BEST CURRENT PRACTICE (лучший современный опыт)
- RFC 4522** Lightweight Directory Access Protocol (LDAP): The Binary Encoding Option.
LDAP: Возможность двоичного кодирования.
S. Legg. Июнь 2006 г. Формат: TXT, 16276 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 4523** Lightweight Directory Access Protocol (LDAP) Schema Definitions for X.509 Certificates.
LDAP: Определения набора схемы для сертификатов X.509.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 43753 байт. Отменяет RFC2252, RFC2256, RFC2587.
Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 4524** COSINE LDAP/X.500 Schema.
Набор схемы COSINE для LDAP/X.500.
K. Zeilenga, Ed.. Июнь 2006 г. Формат: TXT, 11245 байт. Отменяет RFC1274. Вносит изменения в RFC2247, RFC2798. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 4525** Lightweight Directory Access Protocol (LDAP) Modify-Increment Extension.
Расширение инкрементных изменений LDAP.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 11251 байт. Статус: INFORMATIONAL (информационный)
- RFC 4526** Lightweight Directory Access Protocol (LDAP) Absolute True and False Filters.
Фильтры абсолютной правды и лжи в LDAP.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 10097 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 4527** Lightweight Directory Access Protocol (LDAP) Read Entry Controls.
Управление чтением записей в LDAP.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 15987 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 4528** Lightweight Directory Access Protocol (LDAP) Assertion Control.
Контроль утверждений в LDAP.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 12539 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 4529** Requesting Attributes by Object Class in the Lightweight Directory Access Protocol.
Запрос атрибутов по объектному классу в LDAP.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 11927 байт. Статус: INFORMATIONAL (информационный)
- RFC 4530** Lightweight Directory Access Protocol (LDAP) entryUUID Operational Attribute.
Операционный атрибут LDAP entryUUID.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 15191 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 4531** Lightweight Directory Access Protocol (LDAP) Turn Operation.
Операция изменения роли LDAP.
K. Zeilenga. Июнь 2006 г. Формат: TXT, 18986 байт. Статус: EXPERIMENTAL (экспериментальный)
- RFC 4532** Lightweight Directory Access Protocol (LDAP) "Who am I?" Operation.
Операция LDAP "Who am I?"
K. Zeilenga. Июнь 2006 г. Формат: TXT, 14247 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 4533** The Lightweight Directory Access Protocol (LDAP) Content Synchronization Operation.
(рус.) Операция синхронизации содержимого LDAP.
- RFC 4533** K. Zeilenga, J.H. Choi. Июнь 2006 г. Формат: TXT, 73895 байт. Статус: EXPERIMENTAL

- (ориг.) (экспериментальный)
- RFC 4792** Encoding Instructions for the Generic String Encoding Rules (GSER).
Инструкции кодирования для общих правил кодирования строк (GSER).
S. Legg. Январь 2007 г. Формат: TXT, 17637 байт. Вносит изменения в RFC3641. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 5020** The Lightweight Directory Access Protocol (LDAP) entryDN Operational Attribute.
Операционный атрибут LDAP entryDN.
K. Zeilenga. Август 2007 г. Формат: TXT, 8607 байт. Статус: PROPOSED STANDARD (предложенный стандарт)
- RFC 6171** The Lightweight Directory Access Protocol (LDAP) Don't Use Copy Control.
Элемент управления LDAP Don't Use Copy.
K. Zeilenga. Март 2011 г. Статус: PROPOSED STANDARD (предложенный стандарт)

Серия X.500 ITU-T

Вы можете купить спецификации X.500 у [ITU](#) (за немалую сумму). Недавно у них было предложение на две бесплатных спецификации в год, но политика этой организации изменяется постоянно.

Номер	Название	ISO
X.500	Information technology — Open Systems Interconnection — The Directory: Overview of concepts, models and services Информационная технология — Взаимодействие открытых систем — Каталог: Обзор концепций, моделей и сервисов	-
X.501	Information technology — Open Systems Interconnection — The Directory: Models Информационная технология — Взаимодействие открытых систем — Каталог: Модели	-
X.509	Information technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks Информационная технология — Взаимодействие открытых систем — Каталог: Структуры сертификатов открытого ключа и атрибута	-
X.511	Information technology — Open Systems Interconnection — The Directory: Abstract service definition Информационная технология — Взаимодействие открытых систем — Каталог: Абстрактное определение службы	-
X.518	Information technology — Open Systems Interconnection — The Directory: Procedures for distributed operation Информационная технология — Взаимодействие открытых систем — Каталог: Процедуры и распределённые операции	-
X.519	Information technology — Open Systems Interconnection — The Directory: Protocol specifications Информационная технология — Взаимодействие открытых систем — Каталог: Спецификации протокола	ISO/IEC 9594-5:1998
X.520	Information technology — Open Systems Interconnection — The Directory: Selected attribute types Информационная технология — Взаимодействие открытых систем — Каталог: Избранные типы атрибутов	-
X.521	Information technology — Open Systems Interconnection — The Directory: Selected object classes Информационная технология — Взаимодействие открытых систем — Каталог: Избранные объектные классы	-
X.525	Information technology — Open Systems Interconnection — The Directory: Replication Информационная технология — Взаимодействие открытых систем — Каталог: Репликация	-
X.530	Information technology — Open Systems Interconnection — The Directory: Use of systems management for administration of the Directory Информационная технология — Взаимодействие открытых систем — Каталог: Использование средств управления системой для администрирования каталога	-
X.583	Information technology — Open Systems Interconnection — The Directory: Protocol Implementation	-
X.584	Conformance Statement (PICS) proforma for the X.500 series	-
X.585	Информационная технология — Взаимодействие открытых систем — Каталог: Образцы свидетельств о соответствии реализации протоколу для серии X.500	-
X.586		-

Приложение D - Глоссарий LDAP

В данном глоссарии даются краткие определения терминов, в которых могут содержаться ссылки на дополнительную информацию как с этого, так и с других сайтов.

Термин	Перевод	Пояснение
anonymous	анонимное соединение	Сессия описывается как анонимная , если при её инициализации (посылке bind) не передаётся ни DN, ни пароля (данных для проверки подлинности). Если при посылке bind передаётся только DN и не передаётся пароль, LDAP определяет состояние, которое называется неавторизованное соединение (unauthorized) . В OpenLDAP реальный эффект такого подсоединения - создание анонимной сессии.
ASN.1		Abstract Syntax Notation One, абстрактная нотация синтаксиса, разработанная ITU-T (стандарты серии X.208/X.680), представляет собой язык описания и кодирования правил для представления данных. ASN.1 используется для кодирования блоков данных протокола (protocol data unit (PDU), известных также как сообщения (message), блоки (block) или фреймы (frame)) с помощью разных систем кодирования, включая BER (Basic Encoding Rules X.690), CER (Canonical Encoding Rules), DER (Distinguished Encoding Rules), XER (XML Encoding Rules) и PER (Packed Encoding Rules X.691). В случае LDAP используется только простейший BER, а не ужасно сложный PER. ASN.1-описание OID . Обзор СИНТАКСИСА ASN.1 .
Attribute	атрибут	В записи данные содержатся в парах атрибут-значение. Каждый атрибут имеет имя (а также, иногда, короткую форму имени) и принадлежит объектному классу . Характеристики атрибутов полностью описываются в определениях ASN.1 . Один или несколько объектных классов могут быть объединены в набор схемы . В зависимости от определений ASN.1 атрибута, в записи возможна одна (SINGLE-VALUE) или более одной (по умолчанию) пары атрибут-значение с одним и тем же именем атрибута. Один (или несколько) атрибутов, называемых атрибутами именования или RDN , должны всегда уникально идентифицировать запись . Атрибуты делятся на пользовательские (userApplication) и операционные (dSAOperation).
authenticated	соединение с проверкой подлинности	Сессия описывается как соединение с проверкой подлинности , если при её инициализации (посылке bind) передаётся DN пользователя и секретная последовательность.
AVA		Attribute Value Assertion, утверждение значения атрибута. Пара имя_атрибута=значение_атрибута.
Base	базовая запись	базовая запись (известная также как корневая запись , суффикс) - один из многочисленных терминов, используемых для описания записи самого верхнего уровня в DIT или контексте именования . Похоже, что термин base происходит от того, что в поисковых диапазонах LDAP URL и других видов поиска обычно применяется значение base . Root DSE является своего рода супер-корневой записью. Об именовании корневой записи .
BER		Basic Encoding Rules (основные правила кодирования) двоичный формат ITU-T (определён в X.690) для форматирования полей ASN.1 для передачи в рамках протокола. В ряде случаев, особенно в поисковых фильтрах , LDAP использует простые строки, а не двоичную кодировку (BER).
bind	подсоединение	При установке соединения с сервером LDAP первая операция в последовательности операций называется подсоединением (bind) . Операция подсоединения посылает dn записи, которая будет использоваться для аутентификации, и пароль, который будет использоваться (обычно хранящийся в атрибуте userPassword). В случае анонимного соединения оба этих значения будут NULL. При операции bind не разрешён поиск, поэтому используемый для аутентификации DN должен совпадать с DN, с которым запись создавалась изначально. Дополнительная информация по этой теме .
Chaining	сцепление	Если во время операции поиска LDAP-серверу встречается запись с объектным классом Referral , она может быть возвращена клиенту, запрашивающему операцию поиска, в качестве отсылки, либо сервер может быть настроен (с помощью директивы overlay chain) на разрешение отсылки (следование по отсылке) с

использованием процесса, называемого **сцеплением**. По умолчанию OpenLDAP возвращает отсылки, но может быть сконфигурирован на выполнение сцепления с соответствующим повышением нагрузки на сервер. [Дополнительная информация](#).

Client	клиент	Клиент или LDAP-клиент - программный код, с помощью которого производится доступ к LDAP-серверу. Большинство стандартных веб-браузеров (MSIE и Gecko) предоставляют ограниченные возможности клиента LDAP при использовании LDAP URL. LDAP-браузеры и веб-интерфейсы - типичные примеры LDAP-клиентов. Список Open Source клиентов .
Component Matching	соответствие компонентов	Соответствие компонентов (RFC 3687) предоставляет сразу и альтернативный (но более громоздкий) синтаксис поискового фильтра для простых атрибутов, и метод, с помощью которого компоненты (части или экземпляры) составных атрибутов могут быть найдены и извлечены. Дополнительная информация .
contextCSN		Порядковый номер изменения (Change Sequence Number, CSN) контекста (наибольший entryCSN из используемых в контексте или в диапазоне синхронизационного поиска). CSN (и entryCSN , и contextCSN) широко используются в операциях репликации в стиле sync repl OpenLDAP. contextCSN входит в SyncCookie. Назначение CSN определяется только в просроченном черновом RFC draft-chu-ldap-csn-xx.txt и, для версии 2.4, в этой статье FAQ , в состав его входит метка времени (timestamp) - включая микросекунды, счётчик operation-within-second (6 байт), 3 байта идентификатора реплики (определённого в ServerID) и 6 байт операции counte-within-modify.
consumer	потребитель	Так называется сервер, который использует (потребляет) сервисы, предоставляемые сервером- поставщиком . Пример потребителя - используемый в репликации Sync-клиент (RFC 4533 (рус.) , RFC 4533 (ориг.)).
CSN		Порядковый номер изменения (Change Sequence Number, CSN) используется в OpenLDAP для идентификации изменений в конфигурациях с репликацией. Назначение CSN определяется только в просроченном черновом RFC draft-chu-ldap-csn-xx.txt и, для версии 2.4, в этой статье FAQ , в состав его входит метка времени (timestamp) - включая микросекунды, счётчик operation-within-second (6 байт), 3 байта идентификатора реплики (определённого в ServerID) и 6 байт операции counte-within-modify.
DAP		Directory Access Protocol, протокол доступа к каталогу. Термин X.500 для основанного на OSI сетевого протокола, который предназначен для осуществления доступа к DSA , и в котором подразумевается модель данных каталога.
DIT		Directory Information Tree, информационное дерево каталога (также известное как naming-context). DIT - это иерархия объектов, составляющих структуру локального каталога. Одним LDAP-сервером может поддерживаться более одного DIT. Эта информация предоставляется Root DSE . Дополнительная информация здесь .
DN		Distinguished Name, уникальное имя. DN состоит из серии RDN , уникально описывающих атрибуты именования по пути ВВЕРХ по DIT от требуемой записи до корневой записи каталога. DN записывается СЛЕВА НАПРАВО и выглядит примерно так: DN: uid=bill,ou=people,dc=smokeyjoe,dc=com Дополнительная информация .
DSA		Directory System Agent, системный агент каталога. Термин X.500, означающий любую службу каталогов, предоставляющая доступ через DAP или LDAP , например LDAP-сервер.
DSE		DSA Specific Entry (DSE) , специфичная для DSA запись. Контрольная запись в локальном сервере каталогов.
entry	запись	Так называется объект, содержащийся в каталоге LDAP. У каждой записи есть одна родительская запись (объект) и ноль или более дочерних записей (объектов). Информационное содержимое записи состоит из одного или нескольких атрибутов , один (или несколько) из которых используется в качестве атрибута именования (или, правильнее, RDN) для уникальной идентификации данного объекта в DIT .

entryCSN		Порядковый номер изменения (Change Sequence Number, CSN) записи. CSN (и entryCSN, и contextCSN) широко используются в операциях репликации в стиле syncrepl OpenLDAP. Назначение CSN определяется только в просроченном черновом RFC draft-chu-ldap-csn-xx.txt и, для версии 2.4, в этой статье FAQ , в состав его входит метка времени (timestamp) - включая микросекунды, счётчик operation-within-second (6 байт), 3 байта идентификатора реплики (определённого в ServerID) и 6 байт операции counte-within-modify.
entryUUID		Атрибут, содержащий универсальный уникальный идентификатор (Universally Unique ID, UUID) записи в DIT. В настоящее время серверы должны создавать атрибут entryUUID при добавлении новой записи в любой DIT. UUID определён в RFC 4122, а его LDAP-реализация и синтаксис entryUUID - в RFC 4530 . В этом RFC определён синтаксис entryUUID, а также правила соответствия uuidMatch и uuidOrderingMatch.
EQUALITY		EQUALITY определяет правило сравнения атрибута , когда тот используется в поисковом фильтре , НЕ содержащем шаблоны сравнения ; то есть, и содержимое, и длина значения атрибута должны точно совпадать. Когда в поисковом фильтре используются шаблоны сравнения , это называется substring (подстрока) и применяется правило SUBSTR . Определение атрибута .
filter	фильтр	Поиск LDAP осуществляется путём определения базового DN , диапазона и поискового фильтра .
LDAP		Lightweight Directory Access Protocol, облегчённый протокол доступа к службам каталогов. Термин IETF, означающий основанный на TCP/IP сетевой протокол для осуществления доступа к DSA . Отличается от спецификации X.500 DAP некоторым уменьшением функциональности.
LDIF		LDAP Data Interchange Format, формат обмена данными LDAP. Термин IETF, означающий текстовый формат для загрузки (импорта) и сохранения (экспорта) записей в LDAP-каталоги. LDIF определён в RFC 2849(рус.) (RFC 2849(ориг.)).
matchingRule	правило соответствия	Метод, с помощью которого в операциях поиска сравниваются атрибуты . matchingRule - определение ASN.1, содержащее OID , (обычно) имя, например caseIgnoreMatch (OID = 2.5.13.2), и тип данных , с которыми он оперирует, например DirectoryString . Дополнительная информация .
name space	пространство имён	Термин, используемый для описания всех DN , лежащих в (или находящихся внутри, или ограниченных) заданным DIT ; то есть, если корневая запись DIT - dc=example,dc=com, то говорят, что cn=people,dc=example,dc=com находится внутри этого пространства имён, а ou=people,dc=example,dc=net - нет, поскольку находится внутри пространства имён dc=example,dc=net.
naming attribute	атрибут именования	Один из атрибутов , атрибут именования (известный также как RDN), используется для уникальной идентификации каждой записи в DIT .
naming context	контекст именования	Известен также как namingContexts или DIT . Определяет уникальное пространство имён , начинающееся с корневого DN (DN суффикса) и включающее его в себя. Контексты именования, поддерживаемые LDAP-сервером, перечислены в операционном атрибуте namingContexts записи rootDSE .
objectClass	объектный класс	Объектные классы - это коллекции атрибутов. Они определяют: 1. для каждого атрибута - будет ли он обязательным (MUST) или необязательным (MAY); 2. иерархию объектных классов и, как следствие, наследование. Каждый объектный класс уникально идентифицируется с помощью OID.
OID	идентификатор объекта	Object Identifier (OID) представляет собой разделённое точками значение, например 2.5.6.2 (OID объектного класса country), которое уникально определяет объект и того, кто несёт ответственность за определение этого объекта. OID, используемые в LDAP .
Operational	операционные объекты	Операционные объекты используются сервером LDAP для предоставления информации об этом сервере, а также для управления поведением сервера. Они располагаются в Root DSE , и их можно опросить , чтобы вывести все секреты вселенной.

ORDERING		ORDERING определяет правило сравнения атрибута , когда тот используется в поисковом фильтре , содержащем операторы >= или <= . Определение атрибута .
Organizational Unit	подразделение организации	organizationalUnit (ou) определяет произвольное подразделение организации и может использоваться на нескольких уровнях иерархии. Его значение, как правило, соотносится с контекстом, в котором оно используется. Так, в контексте определения формата имени корневой записи ITU (формат ou,c), скорее всего, это будет название компании или организации; в контексте более низких уровней иерархии это может быть 'people', или 'manufacturing', или 'usa', или 'usa-manufacturing', или что-нибудь другое, что имеет смысл в отношении тех объектов, которые будут там располагаться.
Ports	порты	LDAP использует протокол TCP/IP и подключается к порту 389 (ldap) для доступа без SSL и к порту 636 при использовании SSL (ldaps).
primary	первичное имя	Первичное имя - это первое имя из тех, которые встречаются в определении типа атрибута. При индексировании атрибута с помощью директивы index в slapd.conf ДОЛЖНО быть использовано первичное имя. Пример: <pre>attributetype (2.5.4.3 NAME ('cn' 'commonName') SUP name)</pre> В этом примере cn является первичным именем этого атрибута.
Principal DN	DN принципа	При создании (добавлении) записи в DIT, используя LDIF (или LDAP-клиент), она ассоциируется с начальным DN. У этого DN 'создания' нет специального ассоциированного ему термина в стандартах. Если запись будет использоваться для аутентификации, то DN создания должен быть таким же, какие используются при подсоединениях (Bind DN). При использовании подобным образом, DN создания иногда называется DN принципа (Principal DN), особенно в контексте Microsoft AD. Дополнительная информация по этой теме .
provider	поставщик	Так называется сервер, предоставляющий сервисы, которые потребляются (или используются) одним или несколькими другими серверами или клиентами. Пример поставщика - используемый в репликации Sync-сервер (RFC 4533 (рус.) , RFC 4533 (ориг.)). Если брать грубо, то можно утверждать примерное равенство поставщик = главный сервер, потребитель = подчинённый сервер, на объяснение же всех тонкостей разницы могут уйти часы.
RDN		Relative Distinguished Name , относительное уникальное имя (часто также встречается неверная запись этого термина как Relatively Distinguished Name). Это имя даётся атрибуту (атрибутам), которые уникальны на данном уровне иерархии. RDN могут быть с одним значением или с несколькими значениями , в последнем случае для создания RDN два или более атрибута объединяются с помощью знака '+' (плюс), например cn+uid. Термин RDN имеет значение только в случае использования в качестве части DN для уникального описания атрибутов по пути BBEPX по DIT от выбранной записи (или места начала поиска) до корневой записи каталога (или, более корректно, Root DSE). Дополнительная информация .
referral	отсылка	Отсылкой называется возвращаемое от LDAP-сервера LDAP-клиенту имя (обычно LDAP URL) другого LDAP-сервера, который может содержать (или не содержать) запрашиваемую информацию. О конфигурировании отсылок . LDAP-серверы могут быть настроены на автоматическое разрешение отсылок (следование по отсылкам) с использованием процесса, называемого сцеплением .
root	корневая запись	Корневая запись (известная также как базовая запись , суффикс) - один из многочисленных терминов, используемых для описания записи самого верхнего уровня в DIT . Root DSE является своего рода супер-корневой записью. Об именовании корневой записи .
rootdn		rootdn - это запутанно названная директива в файле slapd.conf , определяющая DN администратора для каждого DIT , который может обходить обычные правила доступа к каталогу. rootdn не обязательно должен присутствовать в каталоге или даже каким-либо образом быть связанным со структурой DIT, то есть это может быть DN из любого пространства имён.
Root DSE		Концептуально - запись самого верхнего уровня в иерархии LDAP, можно сказать

super root (супер-корневая запись). Обычно невидима, то есть к ней нельзя получить доступ с помощью нормальных операций. Иногда её путают с **root** (корневой записью), она же **base** (базовая запись), она же **suffix** (суффикс). DSE расшифровывается как **DSA Specific Entry** (специфичная для DSA запись), в свою очередь DSA расшифровывается как **Directory System Agent** (системный агент каталога, то есть любая служба каталогов, предоставляющая доступ через **DAP** или **LDAP**). Информация по **rootDSE** в OpenLDAP может быть получена путём запроса **объектного класса** OpenLDAProotDSE или, в любом LDAP-сервере (включая OpenLDAP), путём запроса установки анонимного подсоединения с пустым базовым DN (""). Эта информация будет содержать данные о поддерживаемых версиях протокола, сервисах и **контекстах именования** или **DIT**.

scope	диапазон	Используется в двух значениях: 1. диапазон поиска: может быть base (база поиска), в этом случае поиск ведётся только в указанном DN, one (один уровень), в этом случае процесс поиска опускается на один уровень от указанного DN, или sub (поддерево), в этом случае процесс поиска опускается вниз по иерархии от указанного DN до самого нижнего уровня в дереве (DIT). 2. диапазон имён: этот термин иногда используется в тех случаях, когда на одном сервере присутствуют несколько DIT. В этом контексте диапазон имён задаётся корневой записью DIT (она же базовая запись или суффикс), например, dn, оканчивающееся на dc=example,dc=net, будет в одном диапазоне имён, а dn оканчивающееся на dc=example,dc=com, будет в другом (отличном) диапазоне имён.
Schema	схема данных, набор схемы данных	Пакет атрибутов и объектных классов, которые могут быть (номинально) связаны друг с другом. Набор (наборы) схемы данных с упакованными в них объектными классами и атрибутами, которые будут использовать (на которые будут ссылаться) приложения, должны быть переданы LDAP-серверу и определяться им, поэтому он должен уметь читать и разбирать все эти милые ASN.1-вещицы. В OpenLDAP это делается с помощью файла slapd.conf.
search	поиск	Поиск LDAP осуществляется путём определения базового DN, диапазона и поискового фильтра.
session	сессия	Сессия возникает между LDAP-клиентом и сервером когда клиент посылает команду bind. Сессия может быть либо анонимной, либо с проверкой подлинности.
slapd		slapd - один из двух демонов, выполняющихся в службе OpenLDAP (другой - slurpd). slapd предоставляет локальную службу LDAP и настраивается с помощью файла slapd.conf.
slapd.conf		slapd.conf - статический конфигурационный файл, используемый slapd и, если настроено, slurpd. В OpenLDAP версии 2.2+ представлена альтернативная возможность конфигурации времени исполнения slapd.d (cn=config).
slurpd		slurpd - один из двух демонов, выполняющихся в службе OpenLDAP (другой - slapd). slurpd предоставляет сервис репликации LDAP, если это необходимо и настроено с помощью файлов slapd.conf и ldap.conf. Дополнительная информация по репликации. Репликация в стиле slurpd устарела и в OpenLDAP версии 2.4+ была заменена на репликацию в стиле syncrepl.
subordinate	нижестоящий	В документации OpenLDAP термин нижестоящая база данных (DIT) используется для определения DIT, на которую происходит отсылка в объекте referral. Поскольку с помощью объектов referral DIT, на который происходит отсылка, делегируется осуществление поиска остальной части DN, такие DIT могут быть представлены в виде иерархии, и DIT, на которую происходит отсылка, будет нижестоящей в такой иерархии. К сожалению, в OpenLDAP используется также термин вышестоящий, определение которого является гораздо более сомнительным. Конфигурация и использование отсылок.
SUBSTR		SUBSTR определяет правило сравнения атрибута, когда тот используется в поисковом фильтре, содержащем шаблоны сравнения. Когда в поисковом фильтре используется целая строка, применяется правило EQUALITY. Определение атрибута.
substring	подстрока	Подстроками являются любые строковые значения, используемые в поисковых

[фильтрах](#) и содержащие **шаблоны сравнения**. Форма сравнения, например, с учётом или без учёта регистра символов, задаётся правилом **SUBSTR** в [определении атрибута](#).

subtype	подтип	В LDAPv3 есть возможность определять подтипы . В настоящий момент уже определено два: binary (в RFC 2251) и lang (в RFC 2596). Подтипы могут использоваться, когда указывается атрибут и квалификация поиска, например <code>cn;lang-en-us=smith</code> требует выполнения поиска с использованием американского английского (US english). Этот подтип не влияет на кодировку, поскольку UTF-8 (применяемый для cn) позволяет использовать все языковые типы. Подтипы lang не чувствительны к регистру символов.
suffix	суффикс	Суффикс (известный также как корневая запись , базовая запись) - один из многочисленных терминов, используемых для описания записи самого верхнего уровня в DIT . Обычно этот термин используется, поскольку данная запись указывается в директиве suffix файла <code>slapd.conf</code> OpenLDAP. Root DSE является своего рода супер-корневой записью. Об именовании корневой записи .
superior	вышестоящий	В документации OpenLDAP термин вышестоящая база данных (DIT) используется для определения DIT, от которой в объекте referral происходит отсылка на другое DIT. Поскольку с помощью объектов referral DIT, на который происходит отсылка, делегируется осуществление поиска остальной части DN, такие DIT могут быть представлены в виде иерархии, и DIT, содержащий объект referral, будет вышестоящим по отношению к DIT, на который происходит отсылка. DIT, являющийся назначением отсылки, называется нижестоящим . Конфигурация и использование отсылок .
supportedExtension	поддерживаемые расширения	Спецификации LDAP (RFC) допускают дополнительные возможности и расширения. Если LDAP-сервер поддерживает какое-то конкретное расширение, его OID будет опубликован в rootDSE . В RFC3674 определён атрибут supportedFeatures записи rootDSE , в котором и содержится список поддерживаемых возможностей и расширений.
SyncCookie		Куки, посылаемое поставщиком потребителю во время репликации в стиле sysncrepl . Подробное описание и настройка репликации .
syncrepl		Метод репликации, основанный на протоколе синхронизации содержимого LDAP (LDAP Content Synchronization protocol, RFC 4533 (рус.) , RFC 4533 (ориг.)) и реализованный в OpenLDAP версии 2.2. Начиная с OpenLDAP версии 2.4, все остальные методы репликации являются устаревшими. Название термина происходит от конфигурационной директивы syncrepl . Дополнительная информация по репликации .
types	типы данных	Термин типы (они же типы данных) обычно используется для указания на ASN.1 СИНТАКСИС атрибута . Типы данных LDAP .
unauthorized	неавторизованное соединение	Сессия описывается как неавторизованная , если при её инициализации (посылке bind) передаётся DN без пароля (данных для проверки подлинности). В OpenLDAP реальный эффект такого подсоединения - создание анонимной сессии (это должно быть явно разрешено с помощью allow bind_anon_dn).

Приложение Е: Объектные классы и атрибуты LDAP

Существует множество готовых **атрибутов** и **объектных классов**, часть из них стандартизована, а часть просто появилась по доброте душевной их авторов. Многие из них упакованы в наборы схемы данных, распространяемые с дистрибутивом OpenLDAP. Некоторые, наиболее часто используемые, приведены ниже. Данный список неисчерпывающий. Там, где это возможно, предпочтительно использовать готовые (уже существующие) **атрибуты** и **объектные классы**, [но Вы можете создать свои собственные](#) - если, конечно, Ваш организм выдержит прививку ASN.1.

Найдите **атрибут**, который Вы хотите, а затем проверьте его **объектный класс**, чтобы выяснить, что ещё потянется за этим атрибутом. Иерархия **объектных классов** показана в виде ссылки `[->objectclassname]` под названием объектного класса в графе **Название** (в большинстве случаев ссылается на определение набора схемы). Таким образом, если Вы используете, скажем, **объектный класс** [residentialPerson](#), который является потомком [person](#), то обязательные (MUST) **атрибуты** будут браться из обоих **объектных классов**

(на жаргоне это называется **наследоваться**), что в данном случае означает обязательное наличие атрибутов **cn**, **sn** и **l**.

Примечание: Имена атрибутов нечувствительны к регистру символов, однако в большинстве случаев они приводятся в ужасной псевдовенгерской нотации, когда заглавные буквы расставлены где попало.

Содержание

- Часто используемые атрибуты
- Объектные классы
- [config.ldif](#) - используется функцией OLC (cn=config) OpenLDAP - оснащён гиперссылками
 - [corba.schema](#) - в дистрибутиве OpenLDAP - оснащён гиперссылками
 - [core.schema](#) - в дистрибутиве OpenLDAP - оснащён гиперссылками
 - [cosine.schema](#) - в дистрибутиве OpenLDAP - оснащён гиперссылками
 - [dyngroup.schema](#) - используется функцией динамических групп - оснащён гиперссылками
 - [inetorgperson.schema](#) - в дистрибутиве OpenLDAP - оснащён гиперссылками
 - [java.schema](#) - в дистрибутиве OpenLDAP - не оснащён гиперссылками
 - [misc.schema](#) - в дистрибутиве OpenLDAP - не оснащён гиперссылками
 - [nis.schema](#) - в дистрибутиве OpenLDAP - оснащён гиперссылками
 - [openldap.schema](#) - в дистрибутиве OpenLDAP - не оснащён гиперссылками
 - [qmail.schema](#) - в дистрибутиве Qmail - оснащён гиперссылками
 - [samba3.schema](#) - (отредактировано) в дистрибутиве OpenLDAP - оснащён гиперссылками
 - [authldap.schema \(courier-imap\)](#) - в дистрибутиве Courier - оснащён гиперссылками
 - [ppolicy.schema](#) - используется [наложением ppolicy](#) OpenLDAP - не оснащён гиперссылками

Часто используемые атрибуты

Этот список неисчерпывающий, но в нём собраны некоторые часто используемые атрибуты и перекрёстные ссылки от них на некоторые объектные классы, в которых они используются. Перейдя по ссылке **набор схем**, можно посмотреть определение атрибута, перейдя по ссылке **objectClass**, можно увидеть, как данный атрибут используется в этом объектном классе.

Название	Псевдоним	Объектный класс	Описание	Набор схемы
c	countryName	country	Двухсимвольный код страны, определённый в ISO 3166	core.schema
cn	commonName	person organizationalPerson organizationalRole groupOfNames applicationProcess applicationEntity posixAccount device		core.schema
dc	domainComponent	dcObject	Любая часть доменного имени, например, domain.com, domain или com	core.schema
-	facsimileTelephoneNumber	residentialPerson organizationalRole organizationalPerson		core.schema
co	friendlyCountryName	friendlyCountry	Полное название страны	cosine.schema
gn	givenName	inetOrgPerson	Имя	core.schema
homePhone	homeTelephoneNumber	inetOrgPerson		cosine.schema
-	jpegPhoto	inetOrgPerson	Фотография в формате jpg	inetorgperson.schema
l	localityName	locality organizationalPerson		core.schema
mail	rfc822Mailbox	inetOrgPerson	Адрес электронной почты, например joe@smokeyjoe.com	core.schema

mobile	mobileTelephoneNumber	inetOrgPerson	Номер мобильного или сотового телефона	cosine.schema
o	organizationName	organization	Название организации	core.schema
ou	organisationalUnitName	organizationUnit	Обычно либо подразделение организации, либо подзапись более глобальной записи	core.schema
-	owner	groupOfNames device groupOfUniqueNames		core.schema
pager	pagerTelephoneNumber	inetOrgPerson		cosine.schema
-	postalAddress	organizationalPerson		core.schema
postalCode	postalCode	organizationalPerson	Почтовый индекс	core.schema
sn	surname	person	Фамилия	core.schema
st	stateOrProvinceName	organizationalPerson		core.schema
street	streetAddress	organizationalPerson		core.schema
-	telephoneNumber	organizationalPerson		core.schema
userPassword	-	organization organizationalUnit person dmd simpleSecurityObject domain posixAccount	Пароль пользователя для некоторых форм контроля доступа	core.schema
uid	userid	account inetOrgPerson posixAccount	Различное назначение, обычно имя пользователя или другое уникальное значение	core.schema



Объектные классы

Этот список неисчерпывающий, но в нем показаны обязательные (MUST) и необязательные (MAY) атрибуты в некоторых часто используемых объектных классах. Перейдя по ссылке **набор схемы**, можно посмотреть определение объектного класса. Хотя в определении многих объектных классов показано, что у них нет обязательных (MUST) атрибутов, Вы должны проследовать по иерархии (показанной в виде ссылки [->...]), чтобы определить, как реально обстоят дела. Так, если Вы попытаетесь создать запись с объектным классом **inetOrgPerson**, не указав как минимум по одному экземпляру атрибутов **cn** и **sn**, такая попытка завершится неудачей. Дополнительная информация о иерархии объектных классов и атрибутов [здесь](#).

Название	MUST	MAY	Набор схемы
account	userid	description \$ seeAlso \$ localityName \$ organizationName \$ organizationalUnitName \$ host	cosine.schema
country	c	searchGuide \$ description	core.schema
dcObject	dc	-	core.schema
device	cn	serialNumber \$ seeAlso \$ owner \$ ou \$ o \$ l \$ description	core.schema
friendlyCountry [-> country]	friendlyCountyName	-	cosine.schema
groupOfNames	member \$ cn	businessCategory \$ seeAlso \$ owner \$ ou \$ o \$ description	core.schema
groupOfUniqueNames	uniqueMember \$ cn	businessCategory \$ seeAlso \$ owner \$ ou \$ o \$ description	core.schema
inetOrgPerson [-> organizationalPerson]	-	audio \$ businessCategory \$ carLicense \$ departmentNumber \$ displayName \$ employeeNumber \$ employeeType \$ givenName \$ homePhone \$ homePostalAddress \$ initials \$ jpegPhoto \$	inetorgperson.schema

		labeledURI \$ mail \$ manager \$ mobile \$ o \$ pager \$ photo \$ roomNumber \$ secretary \$ uid \$ userCertificate \$ x500uniqueIdentifier \$ preferredLanguage \$ userSMIMECertificate \$ userPKCS12	
locality	-	street \$ seeAlso \$ searchGuide \$ st \$ l \$ description	core.schema
organizationalPerson [-> person]	-	title \$ x121Address \$ registeredAddress \$ destinationIndicator \$ preferredDeliveryMethod \$ telexNumber \$ teletexTerminalIdentifier \$ telephoneNumber \$ internationaliSDNNumber \$ facsimileTelephoneNumber \$ street \$ postOfficeBox \$ postalCode \$ postalAddress \$ physicalDeliveryOfficeName \$ ou \$ st \$ l	core.schema
organization	o	userPassword \$ searchGuide \$ seeAlso \$ businessCategory \$ x121Address \$ registeredAddress \$ destinationIndicator \$ preferredDeliveryMethod \$ telexNumber \$ teletexTerminalIdentifier \$ telephoneNumber \$ internationaliSDNNumber \$ facsimileTelephoneNumber \$ street \$ postOfficeBox \$ postalCode \$ postalAddress \$ physicalDeliveryOfficeName \$ st \$ l \$ description	core.schema
organizationalRole	cn	x121Address \$ registeredAddress \$ destinationIndicator \$ preferredDeliveryMethod \$ telexNumber \$ teletexTerminalIdentifier \$ telephoneNumber \$ internationaliSDNNumber \$ facsimileTelephoneNumber \$ seeAlso \$ roleOccupant \$ preferredDeliveryMethod \$ street \$ postOfficeBox \$ postalCode \$ postalAddress \$ physicalDeliveryOfficeName \$ ou \$ st \$ l \$ description	core.schema
organizationalUnit	ou	userPassword \$ searchGuide \$ seeAlso \$ businessCategory \$ x121Address \$ registeredAddress \$ destinationIndicator \$ preferredDeliveryMethod \$ telexNumber \$ teletexTerminalIdentifier \$ telephoneNumber \$ internationaliSDNNumber \$ facsimileTelephoneNumber \$ street \$ postOfficeBox \$ postalCode \$ postalAddress \$ physicalDeliveryOfficeName \$ st \$ l \$ description	core.schema
person	sn \$ cn	userPassword \$ telephoneNumber \$ seeAlso \$ description	core.schema
posixAccount	cn \$ uid \$ uidNumber \$ gidNumber \$ homeDirectory	userPassword \$ loginShell \$ gecos \$ description	nis.schema
residentialPerson [-> person]	l	businessCategory \$ x121Address \$ registeredAddress \$ destinationIndicator \$ preferredDeliveryMethod \$ telexNumber \$ teletexTerminalIdentifier \$ telephoneNumber \$ internationaliSDNNumber \$ facsimileTelephoneNumber \$ preferredDeliveryMethod \$ street \$ postOfficeBox \$ postalCode \$ postalAddress \$ physicalDeliveryOfficeName \$ st \$ l	core.schema



Приложение E: OpenLDAP config.ldif

Эта оптимизированная для просмотра (ну хорошо, она будет оптимизирована уже совсем скоро (Real Soon Now™)) и откомментированная (ну хорошо, она обязательно будет откомментирована) версия LDIF-файла, используемого для загрузки схемы данных, которая применяется в функции `cn=config` (конфигурации времени исполнения) OpenLDAP 2.3+, а также для некоторых других вещей. Он находится в `/etc/openldap/slapd.d/cn=config/cn=schema.ldif` (или `[bsd]` `/usr/local/etc/openldap/slapd.d/cn=config/cn=schema.ldif`). Вы можете получить его из записи `DIT cn=schema,cn=config` с помощью `ldapsearch` или LDAP-браузера. В нём представлен набор объектных классов - контейнеров атрибутов `cn=config`, которые в основном соответствуют названиям [директив](#) `slapd.conf` с

префиксом **olc**, то есть директива `slapd.conf` [access](#) становится **olcAccess**. Ряд важных объектных классов, используемых в OLC (`cn=config`) (смотрите описание макета/организации записей в OLC (`cn=config`) [здесь](#)):

olcGlobal - содержит все атрибуты, которые можно использовать в записи глобальных настроек.

olcSchemaConfig - содержит все атрибуты, которые можно использовать в записи схемы данных.

olcBackendConfig - содержит все атрибуты, которые можно использовать в записях механизмов манипуляции данными.

olcDatabaseConfig - содержит все атрибуты, которые можно использовать в записях баз данных.

olcOverlayConfig - содержит все атрибуты, которые можно использовать в записях наложений.

[Дополнительная информация по `cn=config`.](#)

Представленный ниже файл снабжён комментариями и гиперссылками - исходную версию, поставляемую с OpenLDAP 2.4, можно взять [здесь](#) (сохраните как `.ldif`, используя функцию "сохранить как" Вашего браузера).

```
dn: cn=schema
objectClass: olcSchemaConfig
cn: schema
olcObjectIdentifier: OLcfg 1.3.6.1.4.1.14203.666.11.1
olcObjectIdentifier: OLcfgAt OLcfg:3
olcObjectIdentifier: OLcfgGLAt OLcfgAt:0
olcObjectIdentifier: OLcfgBKAt OLcfgAt:1
olcObjectIdentifier: OLcfgDbAt OLcfgAt:2
olcObjectIdentifier: OLcfgOvAt OLcfgAt:3
olcObjectIdentifier: OLcfgCtAt OLcfgAt:4
olcObjectIdentifier: OLcfgOc OLcfg:4
olcObjectIdentifier: OLcfgGLoc OLcfgOc:0
olcObjectIdentifier: OLcfgBKoc OLcfgOc:1
olcObjectIdentifier: OLcfgDboc OLcfgOc:2
olcObjectIdentifier: OLcfgOvoc OLcfgOc:3
olcObjectIdentifier: OLcfgCtoc OLcfgOc:4
olcObjectIdentifier: OMsyn 1.3.6.1.4.1.1466.115.121.1
olcObjectIdentifier: OMsBoolean OMsyn:7
olcObjectIdentifier: OMSDN OMsyn:12
olcObjectIdentifier: OMSDirectoryString OMsyn:15
olcObjectIdentifier: OMSIA5String OMsyn:26
olcObjectIdentifier: OMSInteger OMsyn:27
olcObjectIdentifier: OMSOID OMsyn:38
olcObjectIdentifier: OMSOctetString OMsyn:40

olcAttributeTypes: ( 2.5.4.0 NAME 'objectClass'
  DESC 'RFC4512: object classes of the entity' EQUALITY objectIdentifierMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

olcAttributeTypes: ( 2.5.21.9 NAME 'structuralObjectClass'
  DESC 'RFC4512: structural object class of entry' EQUALITY objectIdentifierMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 SINGLE-VALUE NO-USER-MODIFICATION
  USAGE directoryOperation )

olcAttributeTypes: ( 2.5.18.1 NAME 'createTimestamp'
  DESC 'RFC4512: time which object was created' EQUALITY generalizedTimeMatch
  ORDERING generalizedTimeOrderingMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
  SINGLE-VALUE NO-USER-MODIFICATION USAGE directoryOperation )

olcAttributeTypes: ( 2.5.18.2 NAME 'modifyTimestamp'
  DESC 'RFC4512: time which object was last modified' EQUALITY generalizedTimeMatch
  ORDERING generalizedTimeOrderingMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
  SINGLE-VALUE NO-USER-MODIFICATION USAGE directoryOperation )

olcAttributeTypes: ( 2.5.18.3 NAME 'creatorsName'
  DESC 'RFC4512: name of creator' EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE NO-USER-MODIFICATION
  USAGE directoryOperation )

olcAttributeTypes: ( 2.5.18.4 NAME 'modifiersName'
  DESC 'RFC4512: name of last modifier' EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE NO-USER-MODIFICATION
  USAGE directoryOperation )

olcAttributeTypes: ( 2.5.18.9 NAME 'hasSubordinates'
  DESC 'X.501: entry has children' EQUALITY booleanMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE NO-USER-MODIFICATION
  USAGE directoryOperation )

olcAttributeTypes: ( 2.5.18.10 NAME 'subschemaSubentry'
  DESC 'RFC4512: name of controlling subschema entry' EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE NO-USER-MODIFICATION
  USAGE directoryOperation )

olcAttributeTypes: ( 1.3.6.1.1.20 NAME 'entryDN'
```

```

DESC 'DN of the entry' EQUALITY distinguishedNameMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )

olcAttributeTypes: ( 1.3.6.1.1.16.4 NAME 'entryUUID'
DESC 'UUID of the entry' EQUALITY UUIDMatch ORDERING UUIDOrderingMatch
SYNTAX 1.3.6.1.1.16.1 SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.1.7
NAME 'entryCSN' DESC 'change sequence number of the entry content'
EQUALITY CSNMatch ORDERING CSNOrderingMatch
SYNTAX 1.3.6.1.4.1.4203.666.11.2.1{64} SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.1.13 NAME 'namingCSN'
DESC 'change sequence number of the entry naming (RDN)'
EQUALITY CSNMatch ORDERING CSNOrderingMatch
SYNTAX 1.3.6.1.4.1.4203.666.11.2.1{64} SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.1.23 NAME 'sync replCookie'
DESC 'sync repl Cookie for shadow copy' EQUALITY octetStringMatch
ORDERING octetStringOrderingMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.1.25 NAME 'contextCSN'
DESC 'the largest committed CSN of a context' EQUALITY CSNMatch
ORDERING CSNOrderingMatch SYNTAX 1.3.6.1.4.1.4203.666.11.2.1{64}
NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.1466.101.120.6 NAME 'altServer'
DESC 'RFC4512: alternative servers' SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.1466.101.120.5 NAME 'namingContexts'
DESC 'RFC4512: naming contexts' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.1466.101.120.13 NAME 'supportedControl'
DESC 'RFC4512: supported controls' SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.1466.101.120.7 NAME 'supportedExtension'
DESC 'RFC4512: supported extended operations'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.1466.101.120.15 NAME 'supportedLDAPVersion'
DESC 'RFC4512: supported LDAP versions' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.1466.101.120.14 NAME 'supportedSASLMechanisms'
DESC 'RFC4512: supported SASL mechanisms' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.1.3.5 NAME 'supportedFeatures'
DESC 'RFC4512: features supported by the server'
EQUALITY objectIdentifierMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.1.10 NAME 'monitorContext'
DESC 'monitor context' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE
NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.1.1 NAME 'configContext'
DESC 'config context' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE
NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.1.4 NAME 'vendorName'
DESC 'RFC3045: name of implementation vendor' EQUALITY caseExactMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE NO-USER-MODIFICATION
USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.1.5 NAME 'vendorVersion'
DESC 'RFC3045: version of implementation' EQUALITY caseExactMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE NO-USER-MODIFICATION
USAGE dSAOperation )

olcAttributeTypes: ( 2.5.18.5 NAME 'administrativeRole'
DESC 'RFC3672: administrative role' EQUALITY objectIdentifierMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 USAGE directoryOperation )

olcAttributeTypes: ( 2.5.18.6 NAME 'subtreeSpecification'
DESC 'RFC3672: subtree specification' SYNTAX 1.3.6.1.4.1.1466.115.121.1.45
SINGLE-VALUE USAGE directoryOperation )

olcAttributeTypes: ( 2.5.21.1 NAME 'dITStructureRules'

```



```

DESC 'RFC4512: DIT structure rules' EQUALITY integerFirstComponentMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.17 USAGE directoryOperation )

olcAttributeTypes: ( 2.5.21.2 NAME 'ditContentRules'
DESC 'RFC4512: DIT content rules' EQUALITY objectIdentifierFirstComponentMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.16 USAGE directoryOperation )

olcAttributeTypes: ( 2.5.21.4 NAME 'matchingRules'
DESC 'RFC4512: matching rules' EQUALITY objectIdentifierFirstComponentMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.30 USAGE directoryOperation )

olcAttributeTypes: ( 2.5.21.5 NAME 'attributeTypes'
DESC 'RFC4512: attribute types' EQUALITY objectIdentifierFirstComponentMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.3 USAGE directoryOperation )

olcAttributeTypes: ( 2.5.21.6 NAME 'objectClasses'
DESC 'RFC4512: object classes' EQUALITY objectIdentifierFirstComponentMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.37 USAGE directoryOperation )

olcAttributeTypes: ( 2.5.21.7 NAME 'nameForms'
DESC 'RFC4512: name forms' EQUALITY objectIdentifierFirstComponentMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.35 USAGE directoryOperation )

olcAttributeTypes: ( 2.5.21.8 NAME 'matchingRuleUse'
DESC 'RFC4512: matching rule uses' EQUALITY objectIdentifierFirstComponentMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.31 USAGE directoryOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.1466.101.120.16 NAME 'ldapSyntaxes'
DESC 'RFC4512: LDAP syntaxes' EQUALITY objectIdentifierFirstComponentMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.54 USAGE directoryOperation )

olcAttributeTypes: ( 2.5.4.1 NAME ( 'aliasedObjectName' 'aliasedEntryName' )
DESC 'RFC4512: name of aliased object' EQUALITY distinguishedNameMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( 2.16.840.1.113730.3.1.34 NAME 'ref'
DESC 'RFC3296: subordinate referral URL' EQUALITY caseExactMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE distributedOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.1.3.1 NAME 'entry'
DESC 'OpenLDAP ACL entry pseudo-attribute' SYNTAX 1.3.6.1.4.1.4203.1.1.1
SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.1.3.2 NAME 'children'
DESC 'OpenLDAP ACL children pseudo-attribute' SYNTAX 1.3.6.1.4.1.4203.1.1.1
SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.1.8 NAME ( 'authzTo' 'saslAuthzTo' )
DESC 'proxy authorization targets' EQUALITY authzMatch
SYNTAX 1.3.6.1.4.1.4203.666.2.7 USAGE distributedOperation X-ORDERED 'VALUES' )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.1.9
NAME ( 'authzFrom' 'saslAuthzFrom' )
DESC 'proxy authorization sources' EQUALITY authzMatch
SYNTAX 1.3.6.1.4.1.4203.666.2.7 USAGE distributedOperation X-ORDERED 'VALUES' )

olcAttributeTypes: ( 1.3.6.1.4.1.1466.101.119.3 NAME 'entryTtl'
DESC 'RFC2589: entry time-to-live' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.1466.101.119.4 NAME 'dynamicSubtrees'
DESC 'RFC2589: dynamic subtrees' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( 2.5.4.49 NAME 'distinguishedName'
DESC 'RFC4519: common supertype of DN attributes'
EQUALITY distinguishedNameMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

olcAttributeTypes: ( 2.5.4.41 NAME 'name'
DESC 'RFC4519: common supertype of name attributes' EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

olcAttributeTypes: ( 2.5.4.3 NAME ( 'cn' 'commonName' )
DESC 'RFC4519: common name(s) for which the entity is known by' SUP name )

olcAttributeTypes: ( 0.9.2342.19200300.100.1.1 NAME ( 'uid' 'userid' )
DESC 'RFC4519: user identifier' EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

olcAttributeTypes: ( 1.3.6.1.1.1.1.0 NAME 'uidNumber'
DESC 'RFC2307: An integer uniquely identifying a user in an administrative domain'
EQUALITY integerMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.1.1.1.1 NAME 'gidNumber'
DESC 'RFC2307: An integer uniquely identifying a group in an administrative domain'
EQUALITY integerMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

```



```

olcAttributeTypes: ( 2.5.4.35 NAME 'userPassword'
  DESC 'RFC4519/2307: password of user' EQUALITY octetStringMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.40{128} )

olcAttributeTypes: ( 1.3.6.1.4.1.250.1.57 NAME 'labeledURI'
  DESC 'RFC2079: Uniform Resource Identifier with optional label'
  EQUALITY caseExactMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( 2.5.4.13 NAME 'description'
  DESC 'RFC4519: descriptive information' EQUALITY caseIgnoreMatch
  SUBSTR caseIgnoreSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{1024} )

olcAttributeTypes: ( 2.5.4.34 NAME 'seeAlso'
  DESC 'RFC4519: DN of related object' SUP distinguishedName )

#####
#
# Start of config directives olc + slapd.conf directive name
#
#####
olcAttributeTypes: ( OlcCfgGlat:78 NAME 'olcConfigFile'
  DESC 'File for slapd configuration directives' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgGlat:79 NAME 'olcConfigDir'
  DESC 'Directory for slapd configuration backend' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgGlat:1 NAME 'olcAccess'
  DESC 'Access Control List' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )

olcAttributeTypes: ( OlcCfgGlat:2 NAME 'olcAllows'
  DESC 'Allowed set of deprecated features' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OlcCfgGlat:3 NAME 'olcArgsFile'
  DESC 'File for slapd command line options' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgGlat:5 NAME 'olcAttributeOptions'
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OlcCfgGlat:4 NAME 'olcAttributeTypes'
  DESC 'OpenLDAP attributeTypes' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )

olcAttributeTypes: ( OlcCfgGlat:6 NAME 'olcAuthIDRewrite'
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )

olcAttributeTypes: ( OlcCfgGlat:7 NAME 'olcAuthzPolicy'
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgGlat:8 NAME 'olcAuthzRegexp'
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )

olcAttributeTypes: ( OlcCfgGlat:9 NAME 'olcBackend'
  DESC 'A type of backend' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE X-ORDERED 'SIBLINGS' )

olcAttributeTypes: ( OlcCfgGlat:10 NAME 'olcConcurrency'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgGlat:11 NAME 'olcConnMaxPending'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgGlat:12 NAME 'olcConnMaxPendingAuth'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgGlat:13 NAME 'olcDatabase'
  DESC 'The backend type for a database instance' SUP olcBackend
  SINGLE-VALUE X-ORDERED 'SIBLINGS' )

olcAttributeTypes: ( OlcCfgGlat:14 NAME 'olcDefaultSearchBase'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgGlat:15 NAME 'olcDisallows'
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OlcCfgGlat:16 NAME 'olcDitContentRules'
  DESC 'OpenLDAP DIT content rules' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )

olcAttributeTypes: ( OlcCfgGlat:17 NAME 'olcGentleHUP'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

```

```
olcAttributeTypes: ( OLcfgDbAt:0.17 NAME 'olcHidden'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:18 NAME 'olcIdleTimeout'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:19 NAME 'olcInclude' SUP labeledURI )  
  
olcAttributeTypes: ( OLcfgGlat:20 NAME 'olcIndexSubstrIfMinLen'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:21 NAME 'olcIndexSubstrIfMaxLen'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:22 NAME 'olcIndexSubstrAnyLen'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:23 NAME 'olcIndexSubstrAnyStep'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:84 NAME 'olcIndexIntLen'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgDbAt:0.4 NAME 'olcLastMod'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgDbAt:0.5 NAME 'olcLimits'  
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )  
  
olcAttributeTypes: ( OLcfgGlat:26 NAME 'olcLocalSSF'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:27 NAME 'olcLogFile'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:28 NAME 'olcLogLevel'  
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )  
  
olcAttributeTypes: ( OLcfgDbAt:0.6 NAME 'olcMaxDerefDepth'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgDbAt:0.16 NAME 'olcMirrorMode'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:30 NAME 'olcModuleLoad'  
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )  
  
olcAttributeTypes: ( OLcfgGlat:31 NAME 'olcModulePath'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgDbAt:0.18 NAME 'olcMonitoring'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:32 NAME 'olcObjectClasses'  
  DESC 'OpenLDAP object classes' EQUALITY caseIgnoreMatch  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )  
  
olcAttributeTypes: ( OLcfgGlat:33 NAME 'olcObjectIdentifier'  
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )  
  
olcAttributeTypes: ( OLcfgGlat:34 NAME 'olcOverlay' SUP olcDatabase  
  SINGLE-VALUE X-ORDERED 'SIBLINGS' )  
  
olcAttributeTypes: ( OLcfgGlat:35 NAME 'olcPasswordCryptSaltFormat'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:36 NAME 'olcPasswordHash'  
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )  
  
olcAttributeTypes: ( OLcfgGlat:37 NAME 'olcPidFile'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:38 NAME 'olcPlugin'  
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )  
  
olcAttributeTypes: ( OLcfgGlat:39 NAME 'olcPluginLogFile'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:40 NAME 'olcReadOnly'  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgGlat:41 NAME 'olcReferral' SUP labeledURI  
  SINGLE-VALUE )  
  
olcAttributeTypes: ( OLcfgDbAt:0.7 NAME 'olcReplica' SUP labeledURI  
  EQUALITY caseIgnoreMatch X-ORDERED 'VALUES' )
```

```
olcAttributeTypes: ( OLcfgGlat:43 NAME 'olcReplicaArgsFile'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:44 NAME 'olcReplicaPidFile'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:45 NAME 'olcReplicationInterval'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:46 NAME 'olcReplLogFile'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:47 NAME 'olcRequires' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgGlat:48 NAME 'olcRestrict' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgGlat:49 NAME 'olcReverseLookup'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:0.8 NAME 'olcRootDN'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:51 NAME 'olcRootDSE' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgDbAt:0.9 NAME 'olcRootPW'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:53 NAME 'olcSaslHost'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:54 NAME 'olcSaslRealm'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:56 NAME 'olcSaslSecProps'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:58 NAME 'olcSchemaDN'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:59 NAME 'olcSecurity'
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgGlat:81 NAME 'olcServerID'
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgGlat:60 NAME 'olcSizeLimit'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:61 NAME 'olcSockbufMaxIncoming'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:62 NAME 'olcSockbufMaxIncomingAuth'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:83 NAME 'olcSortVals'
  DESC 'Attributes whose values will always be sorted' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgDbAt:0.15 NAME 'olcSubordinate'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:0.10 NAME 'olcSuffix'
  EQUALITY distinguishedNameMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

olcAttributeTypes: ( OLcfgDbAt:0.11 NAME 'olcSyncrepl'
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )

olcAttributeTypes: ( OLcfgGlat:66 NAME 'olcThreads'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:67 NAME 'olcTimeLimit'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgGlat:68 NAME 'olcTLSCACertificateFile'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:69 NAME 'olcTLSCACertificatePath'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:70 NAME 'olcTLSCertificateFile'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:71 NAME 'olcTLSCertificateKeyFile'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
```

```

olcAttributeTypes: ( OLcfgGlat:72 NAME 'olcTLSCipherSuite'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:73 NAME 'olcTLSCRLCheck'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:82 NAME 'olcTLSCRLFile'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:74 NAME 'olcTLSSRandFile'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:75 NAME 'olcTLSVerifyClient'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:77 NAME 'olcTLSDHParamFile'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgGlat:80 NAME 'olcToolThreads'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:0.12 NAME 'olcUpdateDN'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:0.13 NAME 'olcUpdateRef' SUP labeledURI
  EQUALITY caseIgnoreMatch )

olcAttributeTypes: ( OLcfgDbAt:0.1 NAME 'olcDbDirectory'
  DESC 'Directory for database content' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgDbAt:5.1 NAME 'olcRelay'
  DESC 'Relay DN' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:4.1 NAME 'olcAccessLogDB'
  DESC 'Suffix of database for log content' SUP distinguishedName SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:4.2 NAME 'olcAccessLogOps'
  DESC 'Operation types to log' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgOvAt:4.3 NAME 'olcAccessLogPurge'
  DESC 'Log cleanup parameters' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:4.4 NAME 'olcAccessLogSuccess'
  DESC 'Log successful ops only' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:4.5 NAME 'olcAccessLogOld'
  DESC 'Log old values when modifying entries matching the filter'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:4.6 NAME 'olcAccessLogOldAttr'
  DESC 'Log old values of these attributes even if unmodified'
  EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

#####
#
# end of slapd.conf directives
#
#####

#####
#
# req attributes
#
#####
olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.1 NAME 'reqDN'
  DESC 'Target DN of request' EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.2 NAME 'reqStart'
  DESC 'Start time of request' EQUALITY generalizedTimeMatch
  ORDERING generalizedTimeOrderingMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.3 NAME 'reqEnd'
  DESC 'End time of request' EQUALITY generalizedTimeMatch
  ORDERING generalizedTimeOrderingMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.4 NAME 'reqType'
  DESC 'Type of request' EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.5 NAME 'reqSession'
  DESC 'Session ID of request' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

```



```

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.6 NAME 'reqAuthzID'
  DESC 'Authorization ID of requestor' EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.7 NAME 'reqResult'
  DESC 'Result code of request' EQUALITY integerMatch ORDERING integerOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.8 NAME 'reqMessage'
  DESC 'Error text of request' EQUALITY caseIgnoreMatch SUBSTR caseIgnoreSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.9 NAME 'reqReferral'
  DESC 'Referrals returned for request' SUP labeledURI )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.10 NAME 'reqControls'
  DESC 'Request controls' EQUALITY objectIdentifierFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.4203.666.11.5.3.1 X-ORDERED 'VALUES' )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.11 NAME 'reqRespControls'
  DESC 'Response controls of request' EQUALITY objectIdentifierFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.4203.666.11.5.3.1 X-ORDERED 'VALUES' )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.12 NAME 'reqId'
  DESC 'ID of Request to Abandon' EQUALITY integerMatch ORDERING integerOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.13 NAME 'reqVersion'
  DESC 'Protocol version of Bind request' EQUALITY integerMatch
  ORDERING integerOrderingMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.14 NAME 'reqMethod'
  DESC 'Bind method of request' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.15 NAME 'reqAssertion'
  DESC 'Compare Assertion of request' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.16 NAME 'reqMod'
  DESC 'Modifications of request' EQUALITY octetStringMatch
  SUBSTR octetStringSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.17 NAME 'reqOld'
  DESC 'Old values of entry before request completed' EQUALITY octetStringMatch
  SUBSTR octetStringSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.18 NAME 'reqNewRDN'
  DESC 'New RDN of request' EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.19 NAME 'reqDeleteOldRDN'
  DESC 'Delete old RDN' EQUALITY booleanMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
  SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.20 NAME 'reqNewSuperior'
  DESC 'New superior DN of request' EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.21 NAME 'reqScope'
  DESC 'Scope of request' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.22 NAME 'reqDerefAliases'
  DESC 'Disposition of Aliases in request' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.23 NAME 'reqAttrsOnly'
  DESC 'Attributes and values of request' EQUALITY booleanMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.24 NAME 'reqFilter'
  DESC 'Filter of request' EQUALITY caseIgnoreMatch
  SUBSTR caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.25 NAME 'reqAttr'
  DESC 'Attributes of request' EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.26 NAME 'reqSizeLimit'
  DESC 'Size limit of request' EQUALITY integerMatch ORDERING integerOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.27 NAME 'reqTimeLimit'
  DESC 'Time limit of request' EQUALITY integerMatch ORDERING integerOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

```

```

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.28 NAME 'reqEntries'
DESC 'Number of entries returned' EQUALITY integerMatch
ORDERING integerOrderingMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.29 NAME 'reqData'
DESC 'Data of extended request' EQUALITY octetStringMatch
SUBSTR octetStringSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
SINGLE-VALUE )

#####
#
# audit attributes
#
#####
olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.5.1.30 NAME 'auditContext'
DESC 'DN of auditContainer' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( OlcCfgOvAt:15.1 NAME 'olcAuditLogFile'
DESC 'Filename for auditlogging' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.1.57 NAME 'entryExpireTimestamp'
DESC 'RFC2589 OpenLDAP extension: expire time of a dynamic object, computed as n
ow + entryTtl' EQUALITY generalizedTimeMatch
ORDERING generalizedTimeOrderingMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

olcAttributeTypes: ( OlcCfgOvAt:9.1 NAME 'olcDDSstate'
DESC 'RFC2589 Dynamic directory services state'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgOvAt:9.2 NAME 'olcDDSmxTtl'
DESC 'RFC2589 Dynamic directory services max TTL'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgOvAt:9.3 NAME 'olcDDSmnTtl'
DESC 'RFC2589 Dynamic directory services min TTL'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgOvAt:9.4 NAME 'olcDDSdefaultTtl'
DESC 'RFC2589 Dynamic directory services default TTL'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgOvAt:9.5 NAME 'olcDDSinterval'
DESC 'RFC2589 Dynamic directory services expiration task run interval'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgOvAt:9.6 NAME 'olcDDStolerance'
DESC 'RFC2589 Dynamic directory services additional TTL in expiration scheduling'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgOvAt:9.7 NAME 'olcDDSmxDynamicObjects'
DESC 'RFC2589 Dynamic directory services max number of dynamic objects'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgOvAt:17.1 NAME 'olcDGAttrPair'
DESC 'Member and MemberURL attribute pair' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OlcCfgOvAt:8.1 NAME 'olcDLattrSet'
DESC 'Dynamic list: <group objectClass>, <URL attributeDescription>,
&t;member attributeDescription>'
EQUALITY caseIgnoreMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
X-ORDERED 'VALUES' )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.9.1.1 NAME 'queryId'
DESC 'ID of query the entry belongs to, formatted as a UUID'
EQUALITY octetStringMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.40{64}
NO-USER-MODIFICATION USAGE directoryOperation )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.9.1.2 NAME 'cachedQueryURL'
DESC 'URI describing a cached query' EQUALITY caseExactMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 NO-USER-MODIFICATION USAGE directoryOperation )

olcAttributeTypes: ( OlcCfgOvAt:2.1 NAME 'olcProxyCache'
DESC 'ProxyCache basicparameters' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OlcCfgOvAt:2.2 NAME 'olcProxyAttrset'
DESC 'A set of attributes to cache' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OlcCfgOvAt:2.3 NAME 'olcProxyTemplate'
DESC 'Filter template, attrset, cache TTL, optional negative TTL, optional sizelimit TTL'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OlcCfgOvAt:2.4 NAME 'olcProxyResponseCB'
DESC 'Response callback position in overlay stack' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

```



```

olcAttributeTypes: ( OLcfgOvAt:2.5 NAME 'olcProxyCacheQueries'
DESC 'Maximum number of queries to cache' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

olcAttributeTypes: ( OLcfgOvAt:2.6 NAME 'olcProxySaveQueries'
DESC 'Save cached queries for hot restart' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 )

olcAttributeTypes: ( OLcfgOvAt:11.1 NAME 'olcRefintAttribute'
DESC 'Attributes for referential integrity' EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgOvAt:11.2 NAME 'olcRefintNothing'
DESC 'Replacement DN to supply when needed' SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.4.1.1 NAME 'errCode'
DESC 'LDAP error code' EQUALITY integerMatch ORDERING integerOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.4.1.2 NAME 'errOp'
DESC 'Operations the errObject applies to' EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.4.1.3 NAME 'errText'
DESC 'LDAP error textual description' EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.4.1.4 NAME 'errSleepTime'
DESC 'Time to wait before returning the error' EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.4.1.5 NAME 'errMatchedDN'
DESC 'Value to be returned as matched DN' EQUALITY distinguishedNameMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.4.1.6 NAME 'errUnsolicitedOID'
DESC 'OID to be returned within unsolicited response'
EQUALITY objectIdentifierMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.4.1.7 NAME 'errUnsolicitedData'
DESC 'Data to be returned within unsolicited response'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 SINGLE-VALUE )

olcAttributeTypes: ( 1.3.6.1.4.1.4203.666.11.4.1.8 NAME 'errDisconnect'
DESC 'Disconnect without notice' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:16.1 NAME 'olcRwmRewrite'
DESC 'Rewrites strings' EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORDERED 'VALUES' )

olcAttributeTypes: ( OLcfgOvAt:16.2 NAME 'olcRwmTFSupport'
DESC 'Absolute filters support' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:16.3 NAME 'olcRwmMap'
DESC 'maps attributes/objectClasses' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
X-ORDERED 'VALUES' )

olcAttributeTypes: ( OLcfgOvAt:16.4 NAME 'olcRwmNormalizeMapped'
DESC 'Normalize mapped attributes/objectClasses' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:1.1 NAME 'olcSpCheckpoint'
DESC 'ContextCSN checkpoint interval in ops and minutes'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:1.2 NAME 'olcSpSessionlog'
DESC 'Session log size in ops' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:1.3 NAME 'olcSpNoPresent'
DESC 'Omit Present phase processing' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:1.4 NAME 'olcSpReloadHint'
DESC 'Observe Reload Hint in Request control' SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:14.1 NAME 'olcTranslucentStrict'
DESC 'Reveal a ttribute deletion constraint violations'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:14.2 NAME 'olcTranslucentNoGlue'
DESC 'Disable automatic glue records for ADD and MODRDN'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:10.1 NAME 'olcUniqueBase'
DESC 'Subtree for uniqueness searches' EQUALITY distinguishedNameMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

```

```

olcAttributeTypes: ( OLcfgOvAt:10.2 NAME 'olcUniqueIgnore'
DESC 'Attributes for which uniqueness shall not be enforced'
EQUALITY caseIgnoreMatch ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgOvAt:10.3 NAME 'olcUniqueAttribute'
DESC 'Attributes for which uniqueness shall be enforced'
EQUALITY caseIgnoreMatch ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgOvAt:10.4 NAME 'olcUniqueStrict'
DESC 'Enforce uniqueness of null values' EQUALITY booleanMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgOvAt:10.5 NAME 'olcUniqueURI'
DESC 'List of keywords and LDAP URIs for a uniqueness domain'
EQUALITY caseExactMatch ORDERING caseExactOrderingMatch
SUBSTR caseExactSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgOvAt:5.1 NAME 'olcValSortAttr'
DESC 'Sorting rule for attribute under given DN' EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgDbAt:1.11 NAME 'olcDbCacheFree'
DESC 'Number of extra entries to free when max is reached'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:1.1 NAME 'olcDbCacheSize'
DESC 'Entry cache size in entries' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:1.2 NAME 'olcDbCheckpoint'
DESC 'Database checkpoint interval in kbytes and minutes'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:1.3 NAME 'olcDbConfig'
DESC 'BerkeleyDB DB_CONFIG configuration directives'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 X-ORDERED 'VALUES' )

olcAttributeTypes: ( OLcfgDbAt:1.4 NAME 'olcDbNoSync'
DESC 'Disable synchronous database writes'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:1.5 NAME 'olcDbDirtyRead'
DESC 'Allow reads of uncommitted data'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:1.12 NAME 'olcDbDnCacheSize'
DESC 'DN cache size' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:1.6 NAME 'olcDbIDLCacheSize'
DESC 'IDL cache size in IDLs' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:0.2 NAME 'olcDbIndex'
DESC 'Attribute index parameters' EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

olcAttributeTypes: ( OLcfgDbAt:1.7 NAME 'olcDbLinearIndex'
DESC 'Index attributes one at a time'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:1.8 NAME 'olcDbLockDetect'
DESC 'Deadlock detection algorithm'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:0.3 NAME 'olcDbMode'
DESC 'Unix permissions of database files'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

olcAttributeTypes: ( OLcfgDbAt:1.9 NAME 'olcDbSearchStack'
DESC 'Depth of search stack in IDLs'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
olcAttributeTypes: ( OLcfgDbAt:1.10 NAME 'olcDbShmKey'
DESC 'Key for shared memory region' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

#####
#
# objectClasses
#
#####
olcObjectClasses: ( 2.5.6.0 NAME 'top' DESC 'top of the superclass chain'
ABSTRACT MUST objectClass )

olcObjectClasses: ( 1.3.6.1.4.1.1466.101.120.111 NAME 'extensibleObject'
DESC 'RFC4512: extensible object' SUP top AUXILIARY )

olcObjectClasses: ( 2.5.6.1 NAME 'alias'
DESC 'RFC4512: an alias' SUP top STRUCTURAL MUST aliasedObjectName )

```

```

olcObjectClasses: ( 2.16.840.1.113730.3.2.6 NAME 'referral'
  DESC 'namedref: named subordinate referral' SUP top STRUCTURAL MUST ref )

olcObjectClasses: ( 1.3.6.1.4.1.4203.1.4.1
  NAME ( 'OpenLDAProotDSE' 'LDAProotDSE' )
  DESC 'OpenLDAP Root DSE object' SUP top STRUCTURAL MAY cn )

olcObjectClasses: ( 2.5.17.0 NAME 'subentry'
  DESC 'RFC3672: subentry' SUP top STRUCTURAL MUST ( cn $ subtreeSpecification ) )

olcObjectClasses: ( 2.5.20.1 NAME 'subschema'
  DESC 'RFC4512: controlling subschema (sub)entry' AUXILIARY
  MAY ( dITStructureRules $ nameForms $ dITContentRules $ objectClasses
    $ attributeTypes $ matchingRules $ matchingRuleUse ) )

olcObjectClasses: ( 1.3.6.1.4.1.1466.101.119.2 NAME 'dynamicObject'
  DESC 'RFC2589: Dynamic Object' SUP top AUXILIARY )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.3.4 NAME 'glue'
  DESC 'Glue Entry' SUP top STRUCTURAL )

olcObjectClasses: ( OLcfgGLOc:0 NAME 'olcConfig'
  DESC 'OpenLDAP configuration object' SUP top ABSTRACT )

#####
#
# olcGlobal objectClass - lists the available global directives supported
#
# defines all attributes that can be used in the global section
# equivalent to the old directives in the slapd.conf global section
# NOTE: no mandatory attributes
#
#####

olcObjectClasses: ( OLcfgGLOc:1 NAME 'olcGlobal' DESC 'OpenLDAP Global configura
  tion options' SUP olcConfig STRUCTURAL
  MAY ( cn $ olcConfigFile $ olcConfigDir $ olcAllows $ olcArgsFile $ olcAttributeOptions
    $ olcAuthIDRewrite $ olcAuthzPolicy $ olcAuthzRegexp $ olcConcurrency
    $ olcConnMaxPending $ olcConnMaxPendingAuth $ olcDisallows $ olcGentleHUP
    $ olcIdleTimeout $ olcIndexSubstrIfMaxLen $ olcIndexSubstrIfMinLen
    $ olcIndexSubstrAnyLen $ olcIndexSubstrAnyStep $ olcIndexIntLen $ olcLocalSSF
    $ olcLogFile $ olcLogLevel $ olcPasswordCryptSaltFormat $ olcPasswordHash $ olcPidFile
    $ olcPluginLogFile $ olcReadOnly $ olcReferral $ olcRepllogFile $ olcRequires
    $ olcRestrict $ olcReverseLookup $ olcRootDSE $ olcSaslHost $ olcSaslRealm
    $ olcSaslSecProps $ olcSecurity $ olcServerID $ olcSizeLimit
    $ olcSockbufMaxIncoming $ olcSockbufMaxIncomingAuth $ olcThreads
    $ olcTimeLimit $ olcTLSCACertificateFile $ olcTLSCACertificatePath
    $ olcTLSCertificateFile $ olcTLSCertificateKeyFile $ olcTLSCipherSuite
    $ olcTLSCRLCheck $ olcTLSSRandFile $ olcTLSVerifyClient $ olcTLSDHParamFile
    $ olcTLSCRLFile $ olcToolThreads $ olcObjectIdentifier $ olcAttributeTypes
    $ olcObjectClasses $ olcDitContentRules ) )

olcObjectClasses: ( OLcfgGLOc:2 NAME 'olcSchemaConfig'
  DESC 'OpenLDAP schema object' SUP olcConfig STRUCTURAL
  MAY ( cn $ olcObjectIdentifier $ olcAttributeTypes $ olcObjectClasses $ olcDitContentRules ) )

olcObjectClasses: ( OLcfgGLOc:3 NAME 'olcBackendConfig'
  DESC 'OpenLDAP Backend-specific options' SUP olcConfig STRUCTURAL
  MUST olcBackend )

#####
#
# olcDatabaseConfig objectClass -
#
#####

olcObjectClasses: ( OLcfgGLOc:4 NAME 'olcDatabaseConfig'
  DESC 'OpenLDAP Database-specific options' SUP olcConfig STRUCTURAL
  MUST olcDatabase
  MAY ( olcHidden $ olcSuffix $ olcSubordinate $ olcAccess $ olcLastMod $ olcLimits
    $ olcMaxDerefDepth $ olcPlugin $ olcReadOnly $ olcReplica $ olcReplicaArgsFile
    $ olcReplicaPidFile $ olcReplicationInterval $ olcRepllogFile $ olcRequires
    $ olcRestrict $ olcRootDN $ olcRootPW $ olcSchemaDN $ olcSecurity $ olcSizeLimit
    $ olcSyncrepl $ olcTimeLimit $ olcUpdatedDN $ olcUpdateRef $ olcMirrorMode
    $ olcMonitoring ) )

olcObjectClasses: ( OLcfgGLOc:6 NAME 'olcIncludeFile'
  DESC 'OpenLDAP configuration include file' SUP olcConfig STRUCTURAL
  MUST olcInclude MAY ( cn $ olcRootDSE ) )

olcObjectClasses: ( OLcfgGLOc:7 NAME 'olcFrontendConfig'
  DESC 'OpenLDAP frontend configuration' AUXILIARY MAY
  ( olcDefaultSearchBase $ olcPasswordHash $ olcSortVals ) )

olcObjectClasses: ( OLcfgGLOc:8 NAME 'olcModuleList'

```

```

DESC 'OpenLDAP dynamic module info' SUP olcConfig STRUCTURAL
MAY ( cn $ olcModulePath $ olcModuleLoad ) )

olcObjectClasses: ( OLcfgDbOc:2.1 NAME 'olcLdifConfig'
DESC 'LDIF backend configuration' SUP olcDatabaseConfig STRUCTURAL MUST olcDbDirectory )

olcObjectClasses: ( OLcfgDbOc:5.1 NAME 'olcRelayConfig'
DESC 'Relay backend configuration' SUP olcDatabaseConfig STRUCTURAL MAY olcRelay )
#####
#
# olcOverlayConfig objectClass -
#
#####
olcObjectClasses: ( OLcfgGlOc:5 NAME 'olcOverlayConfig'
DESC 'OpenLDAP Overlay-specific options' SUP olcConfig STRUCTURAL
MUST olcOverlay )

#####
#
# AccessLog objectClasses
#
#####
olcObjectClasses: ( OLcfgOvOc:4.1 NAME 'olcAccessLogConfig'
DESC 'Access log configuration' SUP olcOverlayConfig STRUCTURAL
MUST olcAccessLogDB
MAY ( olcAccessLogOps $ olcAccessLogPurge $ olcAccessLogSuccess $ olcAccessLogOld
$ olcAccessLogOldAttr ) )

#####
#
# Audit objectClasses
#
#####
olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.0 NAME 'auditContainer'
DESC 'AuditLog container' SUP top STRUCTURAL MAY ( cn $ reqStart $ reqEnd ) )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.1 NAME 'auditObject'
DESC 'OpenLDAP request auditing' SUP top STRUCTURAL
MUST ( reqStart $ reqType $ reqSession )
MAY ( reqID $ reqAuthzID $ reqControls $ reqRespControls $ reqEnd
$ reqResult $ reqMessage $ reqReferral ) )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.2
NAME 'auditReadObject'
DESC 'OpenLDAP read request record' SUP auditObject STRUCTURAL )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.3 NAME 'auditWriteObject'
DESC 'OpenLDAP write request record' SUP auditObject STRUCTURAL )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.4 NAME 'auditAbandon'
DESC 'Abandon operation' SUP auditObject STRUCTURAL MUST reqId )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.5 NAME 'auditAdd'
DESC 'Add operation' SUP auditWriteObject STRUCTURAL MUST reqMod )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.6 NAME 'auditBind'
DESC 'Bind operation' SUP auditObject STRUCTURAL MUST ( reqVersion $ reqMethod ) )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.7 NAME 'auditCompare'
DESC 'Compare operation' SUP auditReadObject STRUCTURAL MUST reqAssertion )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.8 NAME 'auditDelete'
DESC 'Delete operation' SUP auditWriteObject STRUCTURAL MAY reqOld )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.9 NAME 'auditModify'
DESC 'Modify operation' SUP auditWriteObject STRUCTURAL MUST reqMod MAY reqOld )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.10 NAME 'auditModRDN'
DESC 'ModRDN operation' SUP auditWriteObject STRUCTURAL
MUST ( reqNewRDN $ reqDeleteOldRDN )
MAY ( reqNewSuperior $ reqMod $ reqOld ) )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.11 NAME 'auditSearch'
DESC 'Search operation' SUP auditreadObject STRUCTURAL
MUST ( reqScope $ reqDerefAliases $ reqAttrsonly )
MAY ( reqFilter $ reqAttr $ reqEntries $ reqSizeLimit $ reqTimeLimit ) )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.5.2.12 NAME 'auditExtended'
DESC 'Extended operation' SUP auditObject STRUCTURAL MAY reqData )

olcObjectClasses: ( OLcfgOvOc:15.1 NAME 'olcAuditlogConfig'
DESC 'Auditlog configuration' SUP olcOverlayConfig STRUCTURAL MAY olcAuditlogFile )

olcObjectClasses: ( OLcfgOvOc:9.1 NAME 'olcDDSSConfig'
DESC 'RFC2589 Dynamic directory services configuration' SUP olcOverlayConfig
STRUCTURAL MAY ( olcDDSstate $ olcDDSmxTtl $ olcDDSminTtl $ olcDDSdefaultTtl
$ olcDDSinterval $ olcDDStolerance $ olcDDSmxDynamicObjects ) )

```



```

olcObjectClasses: ( OLcfgOvOc:17.1 NAME 'olcDGConfig'
  DESC 'Dynamic Group configuration' SUP olcOverlayConfig STRUCTURAL MAY olcDGAttrPair )

olcObjectClasses: ( OLcfgOvOc:8.1 NAME 'olcDynamicList'
  DESC 'Dynamic list configuration' SUP olcOverlayConfig STRUCTURAL MAY olcDLattrSet )
#####
#
# olcPcache objectClasses
#
#####
olcObjectClasses: ( OLcfgOvOc:2.1 NAME 'olcPcacheConfig'
  DESC 'ProxyCache configuration' SUP olcOverlayConfig STRUCTURAL
  MUST ( olcProxyCache $ olcProxyAttrset $ olcProxyTemplate )
  MAY ( olcProxyResponseCB $ olcProxyCacheQueries $ olcProxySaveQueries ) )

olcObjectClasses: ( OLcfgOvOc:2.2 NAME 'olcPcacheDatabase'
  DESC 'Cache database configuration' AUXILIARY )

olcObjectClasses: ( OLcfgOvOc:11.1 NAME 'olcRefintConfig'
  DESC 'Referential integrity configuration' SUP olcOverlayConfig STRUCTURAL
  MAY ( olcRefintAttribute $ olcRefintNothing ) )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.4.3.0 NAME 'errAbsObject'
  SUP top ABSTRACT MUST errCode
  MAY ( cn $ description $ errOp $ errText $ errSleepTime $ errMatchedDN
  $ errUnsolicitedOID $ errUnsolicitedData $ errDisconnect ) )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.4.3.1 NAME 'errObject'
  SUP errAbsObject STRUCTURAL )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.11.4.3.2 NAME 'errAuxObject'
  SUP errAbsObject AUXILIARY )

olcObjectClasses: ( OLcfgOvOc:16.1 NAME 'olcRwmConfig'
  DESC 'Rewrite/remap configuration' SUP olcOverlayConfig STRUCTURAL
  MAY ( olcRwmRewrite $ olcRwmTFSupport $ olcRwmMap $ olcRwmNormalizeMapped ) )

#####
#
# olcSyncProvConfig objectClasses
#
#####
olcObjectClasses: ( OLcfgOvOc:1.1 NAME 'olcSyncProvConfig'
  DESC 'SyncRepl Provider configuration' SUP olcOverlayConfig STRUCTURAL
  MAY ( olcSpCheckpoint $ olcSpSessionlog $ olcSpNoPresent $ olcSpReloadHint ) )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.3.5 NAME 'syncConsumerSubentry'
  DESC 'Persistent Info for SyncRepl Consumer' AUXILIARY MAY syncreplCookie )

olcObjectClasses: ( 1.3.6.1.4.1.4203.666.3.6 NAME 'syncProviderSubentry'
  DESC 'Persistent Info for SyncRepl Producer' AUXILIARY MAY contextCSN )

olcObjectClasses: ( OLcfgOvOc:14.1 NAME 'olcTranslucentConfig'
  DESC 'Translucent configuration' SUP olcOverlayConfig STRUCTURAL
  MAY ( olcTranslucentStrict $ olcTranslucentNoGlue ) )

olcObjectClasses: ( OLcfgOvOc:14.2 NAME 'olcTranslucentDatabase'
  DESC 'Translucent target database configuration' AUXILIARY )

olcObjectClasses: ( OLcfgOvOc:10.1 NAME 'olcUniqueConfig'
  DESC 'Attribute value uniqueness configuration' SUP olcOverlayConfig STRUCTURAL
  MAY ( olcUniqueBase $ olcUniqueIgnore $ olcUniqueAttribute $ olcUniqueStrict
  $ olcUniqueURI ) )

olcObjectClasses: ( OLcfgOvOc:5.1 NAME 'olcValSortConfig'
  DESC 'Value Sorting configuration' SUP olcOverlayConfig STRUCTURAL
  MUST olcValSortAttr )

olcObjectClasses: ( OLcfgDbOc:1.1 NAME 'olcBdbConfig'
  DESC 'BDB backend configuration' SUP olcDatabaseConfig STRUCTURAL
  MUST olcDbDirectory
  MAY ( olcDbCacheSize $ olcDbCheckpoint $ olcDbConfig
  $ olcDbNoSync $ olcDbDirtyRead $ olcDbIDLcacheSize $ olcDbIndex
  $ olcDbLinearIndex $ olcDbLockDetect $ olcDbMode $ olcDbSearchStack
  $ olcDbShmKey $ olcDbCacheFree $ olcDbDnCacheSize ) )

```

Приложение E: OpenLDAP corba.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом OpenLDAP.

```

# Corba Object Schema
# $OpenLDAP: pkg/ldap/servers/slapd/schema/corba.schema,

```

```

# v 1.1.2.1 2000/09/13 00:42:36 kurt Exp $
# depends upon core.schema

# Network Working Group                                V. Ryan
# Request for Comments: 2714                            R. Lee
# Category: Informational                               S. Seligman
#                                                    Sun Microsystems, Inc.
#                                                    October 1999

#
# Schema for Representing CORBA Object References in an LDAP Directory
#
# Status of this Memo
#
# This memo provides information for the Internet community. It does
# not specify an Internet standard of any kind. Distribution of this
# memo is unlimited.
#
# Copyright Notice
#
# Copyright (C) The Internet Society (1999). All Rights Reserved.
#
# Abstract
#
# CORBA [CORBA] is the Common Object Request Broker Architecture
# defined by the Object Management Group. This document defines the
# schema for representing CORBA object references in an LDAP directory
# [LDAPv3].
#
# [trimmed]
#
# 3. Attribute Type Definitions
#
# The following attribute types are defined in this document:
#
#     corbaIor
#     corbaRepositoryId
#
# 3.1 corbaIor
#
# This attribute stores the string representation of the interoperable
# object reference (IOR) for a CORBA object. An IOR is an opaque handle
# for the object which contains the information necessary to locate the
# object, even if the object is in another ORB.
#
# This attribute's syntax is 'IA5 String' and its case is
# insignificant.
#
# ( 1.3.6.1.4.1.42.2.27.4.1.14
#   NAME 'corbaIor'
#   DESC 'Stringified interoperable object reference of a CORBA object'
#   EQUALITY caseIgnoreIA5Match
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
#   SINGLE-VALUE
# )
#
# attributetype ( 1.3.6.1.4.1.42.2.27.4.1.14
#   NAME 'corbaIor'
#   DESC 'Stringified interoperable object reference of a CORBA object'
#   EQUALITY caseIgnoreIA5Match
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
#   SINGLE-VALUE )
#
# 3.2 corbaRepositoryId
#
# Each CORBA interface has a unique "repository id" (also called "type
# id") that identifies the interface. A CORBA object has one or more
# repository ids, one for each interface that it implements.
#
# The format of a repository id can be any string, but the OMG
# specifies four standard formats:
#
#     a. IDL-style
#
#         IDL:Prefix/ModuleName/InterfaceName:VersionNumber
#
# For example, the repository id for the "NamingContext" in OMG's COS
# Naming module is: "IDL:omg.org/CosNaming/NamingContext:1.0".
#
#     b. RMI-style
#
#         RMI:ClassName:HashCode[:SUID]
#
# This format is used by RMI-IIOP remote objects [RMI-IIOP].
# "ClassName" is the fully qualified name of the class (for example,
# "java.lang.String"). "HashCode" is the object's hash code (that is,
# that obtained by invoking the "hashCode()" method). "SUID" is the

```



```

# "stream unique identifier", which is a 64-bit number that uniquely
# identifies the serialization version of the class; SUID is optional
# in the repository id.
#
# c. DCE-style
#
# DCE:UUID
#
# This format is used for DCE/CORBA interoperability [CORBA-DCE].
# "UUID" represents a DCE UUID.
#
# d. "local"
#
# This format is defined by the local Object Request Broker (ORB).
#
# The corbaRepositoryId attribute is a multivalued attribute; each
# value records a single repository id of an interface implemented by
# the CORBA object. This attribute need not contain a complete list of
# the interfaces implemented by the CORBA object.
#
# This attribute's syntax is 'Directory String' and its case is
# significant. The values of this attribute are encoded using UTF-8.
# Some values may require translation from their native representation
# in order to be correctly encoded using UTF-8.
#
# ( 1.3.6.1.4.1.42.2.27.4.1.15
#   NAME 'corbaRepositoryId'
#   DESC 'Repository ids of interfaces implemented by a CORBA object'
#   EQUALITY caseExactMatch
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
# )
#
#
# attributetype ( 1.3.6.1.4.1.42.2.27.4.1.15
#                 NAME 'corbaRepositoryId'
#                 DESC 'Repository ids of interfaces implemented by a CORBA object'
#                 EQUALITY caseExactMatch
#                 SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
#
# 4. Object Class Definitions
#
# The following object classes are defined in this document:
#
# corbaContainer
# corbaObject
# corbaObjectReference
#
# 4.1 corbaContainer
#
# This structural object class represents a container for a CORBA
# object.
#
# ( 1.3.6.1.4.1.42.2.27.4.2.10
#   NAME 'corbaContainer'
#   DESC 'Container for a CORBA object'
#   SUP top
#   STRUCTURAL
#   MUST ( cn )
# )
#
# objectclass ( 1.3.6.1.4.1.42.2.27.4.2.10
#               NAME 'corbaContainer'
#               DESC 'Container for a CORBA object'
#               SUP top
#               STRUCTURAL
#               MUST cn )
#
# 4.2 corbaObject
#
# This abstract object class is the root class for representing a CORBA
# object.
#
# ( 1.3.6.1.4.1.42.2.27.4.2.9
#   NAME 'corbaObject'
#   DESC 'CORBA object representation'
#   SUP top
#   ABSTRACT
#   MAY ( corbaRepositoryId $ description )
# )
#
# objectclass ( 1.3.6.1.4.1.42.2.27.4.2.9
#               NAME 'corbaObject'
#               DESC 'CORBA object representation'
#               SUP top
#               ABSTRACT
#               MAY ( corbaRepositoryId $ description ) )

```

```
# 4.3 corbaObjectReference
#
# This auxiliary object class represents a CORBA object reference. It
# must be mixed in with a structural object class.
#
# ( 1.3.6.1.4.1.42.2.27.4.2.11
#   NAME 'corbaObjectReference'
#   DESC 'CORBA interoperable object reference'
#   SUP corbaObject
#   AUXILIARY
#   MUST ( corbaIor )
# )
#
objectclass ( 1.3.6.1.4.1.42.2.27.4.2.11
              NAME 'corbaObjectReference'
              DESC 'CORBA interoperable object reference'
              SUP corbaObject
              AUXILIARY
              MUST corbaIor )
```

Приложение E: core.schema

Этот стандартный набор схемы данных распространяется с большинством дистрибутивов LDAP.

```
# $OpenLDAP: pkg/ldap/servers/slapd/schema/core.schema,
# v 1.7.2.19 2002/06/06 00:25:24 kurt Exp $
#
# OpenLDAP Core schema
#
# Includes LDAPv3 schema items from:
#   RFC2251-RFC2256 (LDAPv3)
#
# select standard track schema items:
#   RFC2079 (URI)
#   RFC1274 (uid/dc)
#   RFC2247 (dc/dcObject)
#   RFC2589 (Dynamic Directory Services)
#
# select informational schema items:
#   RFC2377 (uidObject)
#
# select IETF 'work in progress' LDAPext/LDUP items
#   ldapSubentry
#   ldapRootDSE
#   named referrals
#   alias draft
#
#####
#
## THESE ATTRIBUTES ARE OPERATIONAL FROM OpenLDAP 2.1+ AND NOT INCLUDED #
## IN CORE.SCHEMA #
# INCLUDED HERE FOR CLARITY ONLY #
#####

# Standard X.501(93) Operational Attribute Types from RFC2252
attributetype ( 2.5.4.41 NAME 'name'
                EQUALITY caseIgnoreMatch
                SUBSTR caseIgnoreSubstringsMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

attributetype ( 2.5.4.49 NAME 'distinguishedName'
                EQUALITY distinguishedNameMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

# Standard attribute types from RFC2256
attributetype ( 2.5.4.0 NAME 'objectClass'
                EQUALITY objectIdentifierMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

attributetype ( 2.5.4.1 NAME 'aliasedObjectName'
                EQUALITY distinguishedNameMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

##### END OF OPERATIONAL ATTRIBUTES #####

attributetype ( 2.5.4.2 NAME 'knowledgeInformation'
                EQUALITY caseIgnoreMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

attributetype ( 2.5.4.3 NAME ( 'cn' 'commonName' ) SUP name )
```

```

attributetype ( 2.5.4.4 NAME ( 'sn' 'surname' ) SUP name )
attributetype ( 2.5.4.5 NAME 'serialNumber' EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{64} )
# (2-letter code from ISO 3166)
attributetype ( 2.5.4.6 NAME ( 'c' 'countryName' ) SUP name SINGLE-VALUE )
attributetype ( 2.5.4.7 NAME ( 'l' 'localityName' ) SUP name )
attributetype ( 2.5.4.8 NAME ( 'st' 'stateOrProvinceName' ) SUP name )
attributetype ( 2.5.4.9 NAME ( 'street' 'streetAddress' )
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} )
attributetype ( 2.5.4.10 NAME ( 'o' 'organizationName' ) SUP name )
attributetype ( 2.5.4.11 NAME ( 'ou' 'organizationalUnitName' ) SUP name )
attributetype ( 2.5.4.12 NAME 'title' SUP name )
attributetype ( 2.5.4.13 NAME 'description'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{1024} )
# Obsoleted by enhancedSearchGuide
attributetype ( 2.5.4.14 NAME 'searchGuide'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.25 )
attributetype ( 2.5.4.15 NAME 'businessCategory'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} )
attributetype ( 2.5.4.16 NAME 'postalAddress'
    EQUALITY caseIgnoreListMatch
    SUBSTR caseIgnoreListSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.41 )
attributetype ( 2.5.4.17 NAME 'postalCode'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{40} )
attributetype ( 2.5.4.18 NAME 'postOfficeBox'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{40} )
attributetype ( 2.5.4.19 NAME 'physicalDeliveryOfficeName'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} )
attributetype ( 2.5.4.20 NAME 'telephoneNumber'
    EQUALITY telephoneNumberMatch
    SUBSTR telephoneNumberSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.50{32} )
attributetype ( 2.5.4.21 NAME 'telexNumber'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.52 )
attributetype ( 2.5.4.22 NAME 'teletexTerminalIdentifier'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.51 )
attributetype ( 2.5.4.23 NAME ( 'facsimileTelephoneNumber' 'fax' )
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.22 )
attributetype ( 2.5.4.24 NAME 'x121Address'
    EQUALITY numericStringMatch
    SUBSTR numericStringSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.36{15} )
attributetype ( 2.5.4.25 NAME 'internationaliSDNNumber'
    EQUALITY numericStringMatch
    SUBSTR numericStringSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.36{16} )
attributetype ( 2.5.4.26 NAME 'registeredAddress' SUP postalAddress
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.41 )
attributetype ( 2.5.4.27 NAME 'destinationIndicator'

```

```

EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{128} )

attributetype ( 2.5.4.28 NAME 'preferredDeliveryMethod'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.14
SINGLE-VALUE )

attributetype ( 2.5.4.29 NAME 'presentationAddress'
EQUALITY presentationAddressMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.43
SINGLE-VALUE )

attributetype ( 2.5.4.30 NAME 'supportedApplicationContext'
EQUALITY objectIdentifierMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

attributetype ( 2.5.4.31 NAME 'member' SUP distinguishedName )

attributetype ( 2.5.4.32 NAME 'owner' SUP distinguishedName )

attributetype ( 2.5.4.33 NAME 'roleOccupant' SUP distinguishedName )

attributetype ( 2.5.4.34 NAME 'seeAlso' SUP distinguishedName )

attributetype ( 2.5.4.35 NAME 'userPassword'
EQUALITY octetStringMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.40{128} )

# Must be transferred using ;binary
attributetype ( 2.5.4.36 NAME 'userCertificate'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.8 )

# Must be transferred using ;binary
attributetype ( 2.5.4.37 NAME 'cACertificate'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.8 )

# Must be transferred using ;binary
attributetype ( 2.5.4.38 NAME 'authorityRevocationList'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.9 )

# Must be transferred using ;binary
attributetype ( 2.5.4.39 NAME 'certificateRevocationList'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.9 )

# Must be stored and requested in the binary form
attributetype ( 2.5.4.40 NAME 'crossCertificatePair'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.10 )

attributetype ( 2.5.4.42 NAME ( 'givenName' 'gn' ) SUP name )

attributetype ( 2.5.4.43 NAME 'initials' SUP name
DESC 'The initials attribute type contains the initials of some
or all of an individuals names, but not the surname(s).' )

attributetype ( 2.5.4.44 NAME 'generationQualifier'
DESC 'e.g. Jr or II.'
SUP name )

attributetype ( 2.5.4.45 NAME 'x500UniqueIdentifier'
EQUALITY bitStringMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.6 )

attributetype ( 2.5.4.46 NAME 'dnQualifier'
EQUALITY caseIgnoreMatch
ORDERING caseIgnoreOrderingMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.44 )

attributetype ( 2.5.4.47 NAME 'enhancedSearchGuide'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.21 )

attributetype ( 2.5.4.48 NAME 'protocolInformation'
EQUALITY protocolInformationMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.42 )

# 2.5.4.49 is defined above as it's used for subtyping
#attributetype ( 2.5.4.49 NAME 'distinguishedName'
# EQUALITY distinguishedNameMatch
# SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

attributetype ( 2.5.4.50 NAME 'uniqueMember'
EQUALITY uniqueMemberMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.34 )

attributetype ( 2.5.4.51 NAME 'houseIdentifier'
EQUALITY caseIgnoreMatch

```

```

SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

# Must be transferred using ;binary
attributetype ( 2.5.4.52 NAME 'supportedAlgorithms'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.49 )

# Must be transferred using ;binary
attributetype ( 2.5.4.53 NAME 'deltaRevocationList'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.9 )

attributetype ( 2.5.4.54 NAME 'dmdName' SUP name )

# Standard object classes from RFC2256

objectclass ( 2.5.6.0 NAME 'top' ABSTRACT
    MUST objectClass )

objectclass ( 2.5.6.1 NAME 'alias' SUP top STRUCTURAL
    MUST aliasedObjectName )

objectclass ( 2.5.6.2 NAME 'country' SUP top STRUCTURAL
    MUST c
    MAY ( searchGuide $ description ) )

objectclass ( 2.5.6.3 NAME 'locality' SUP top STRUCTURAL
    MAY ( street $ seeAlso $ searchGuide $ st $ l $ description ) )

objectclass ( 2.5.6.4 NAME 'organization' SUP top STRUCTURAL
    MUST o
    MAY ( userPassword $ searchGuide $ seeAlso $ businessCategory $
        x121Address $ registeredAddress $ destinationIndicator $
        preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
        telephoneNumber $ internationaliSDNNumber $
        facsimileTelephoneNumber $ street $ postOfficeBox $ postalCode $
        postalAddress $ physicalDeliveryOfficeName $ st $ l $ description ) )

objectclass ( 2.5.6.5 NAME 'organizationalUnit' SUP top STRUCTURAL
    MUST ou
    MAY ( userPassword $ searchGuide $ seeAlso $ businessCategory $
        x121Address $ registeredAddress $ destinationIndicator $
        preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
        telephoneNumber $ internationaliSDNNumber $
        facsimileTelephoneNumber $ street $ postOfficeBox $ postalCode $
        postalAddress $ physicalDeliveryOfficeName $ st $ l $ description ) )

objectclass ( 2.5.6.6 NAME 'person' SUP top STRUCTURAL
    MUST ( sn $ cn )
    MAY ( userPassword $ telephoneNumber $ seeAlso $ description ) )

objectclass ( 2.5.6.7 NAME 'organizationalPerson' SUP person STRUCTURAL
    MAY ( title $ x121Address $ registeredAddress $ destinationIndicator $
        preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
        telephoneNumber $ internationaliSDNNumber $
        facsimileTelephoneNumber $ street $ postOfficeBox $ postalCode $
        postalAddress $ physicalDeliveryOfficeName $ ou $ st $ l ) )

objectclass ( 2.5.6.8 NAME 'organizationalRole' SUP top STRUCTURAL
    MUST cn
    MAY ( x121Address $ registeredAddress $ destinationIndicator $
        preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
        telephoneNumber $ internationaliSDNNumber $ facsimileTelephoneNumber $
        seeAlso $ roleOccupant $ preferredDeliveryMethod $ street $
        postOfficeBox $ postalCode $ postalAddress $
        physicalDeliveryOfficeName $ ou $ st $ l $ description ) )

objectclass ( 2.5.6.9 NAME 'groupOfNames' SUP top STRUCTURAL
    MUST ( member $ cn )
    MAY ( businessCategory $ seeAlso $ owner $ ou $ o $ description ) )

objectclass ( 2.5.6.10 NAME 'residentialPerson' SUP person STRUCTURAL
    MUST l
    MAY ( businessCategory $ x121Address $ registeredAddress $
        destinationIndicator $ preferredDeliveryMethod $ telexNumber $
        teletexTerminalIdentifier $ telephoneNumber $ internationaliSDNNumber $
        facsimileTelephoneNumber $ street $
        postOfficeBox $ postalCode $ postalAddress $
        physicalDeliveryOfficeName $ st $ l ) )

objectclass ( 2.5.6.11 NAME 'applicationProcess' SUP top STRUCTURAL
    MUST cn
    MAY ( seeAlso $ ou $ l $ description ) )

objectclass ( 2.5.6.12 NAME 'applicationEntity' SUP top STRUCTURAL
    MUST ( presentationAddress $ cn )
    MAY ( supportedApplicationContext $ seeAlso $ ou $ o $ l $
        description ) )

```

```

objectclass ( 2.5.6.13 NAME 'dSA' SUP applicationEntity STRUCTURAL
    MAY knowledgeInformation )

objectclass ( 2.5.6.14 NAME 'device' SUP top STRUCTURAL
    MUST cn
    MAY ( serialNumber $ seeAlso $ owner $ ou $ o $ l $ description ) )

objectclass ( 2.5.6.15 NAME 'strongAuthenticationUser' SUP top AUXILIARY
    MUST userCertificate )

objectclass ( 2.5.6.16 NAME 'certificationAuthority' SUP top AUXILIARY
    MUST ( authorityRevocationList $ certificateRevocationList $
        cACertificate )
    MAY crossCertificatePair )

objectclass ( 2.5.6.17 NAME 'groupOfUniqueNames' SUP top STRUCTURAL
    MUST ( uniqueMember $ cn )
    MAY ( businessCategory $ seeAlso $ owner $ ou $ o $ description ) )

objectclass ( 2.5.6.18 NAME 'userSecurityInformation' SUP top AUXILIARY
    MAY ( supportedAlgorithms ) )

objectclass ( 2.5.6.16.2 NAME 'certificationAuthority-V2'
    SUP certificationAuthority
    AUXILIARY MAY ( deltaRevocationList ) )

objectclass ( 2.5.6.19 NAME 'cRLDistributionPoint' SUP top STRUCTURAL
    MUST ( cn )
    MAY ( certificateRevocationList $ authorityRevocationList $
        deltaRevocationList ) )

objectclass ( 2.5.6.20 NAME 'dmd' SUP top STRUCTURAL
    MUST ( dmdName )
    MAY ( userPassword $ searchGuide $ seeAlso $ businessCategory $
        x121Address $ registeredAddress $ destinationIndicator $
        preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
        telephoneNumber $ internationalISDNNumber $ facsimileTelephoneNumber $
        street $ postOfficeBox $ postalCode $ postalAddress $
        physicalDeliveryOfficeName $ st $ l $ description ) )

#
# adding extensibleClass allows the addition of any attribute to an entry
#
objectclass ( 1.3.6.1.4.1.1466.101.120.111 NAME 'extensibleObject'
    DESC 'RFC2252: extensible object'
    SUP top AUXILIARY )

#
# Standard Track URI label schema from RFC2079
#
attributetype ( 1.3.6.1.4.1.250.1.57 NAME 'labeledURI'
    DESC 'RFC2079: Uniform Resource Identifier with optional label'
    EQUALITY caseExactMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

objectclass ( 1.3.6.1.4.1.250.3.15 NAME 'labeledURIObject'
    DESC 'RFC2079: object that contains the URI attribute type'
    MAY ( labeledURI )
    SUP top AUXILIARY )

#
# Standard Track Dynamic Directory Services from RFC2589
#
objectclass ( 1.3.6.1.4.1.1466.101.119.2 NAME 'dynamicObject'
    DESC 'RFC2589: Dynamic Object'
    SUP top AUXILIARY )

attributetype ( 1.3.6.1.4.1.1466.101.119.3 NAME 'entryTtl'
    DESC 'RFC2589: entry time-to-live'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE
    NO-USER-MODIFICATION USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.1466.101.119.4 NAME 'dynamicSubtrees'
    DESC 'RFC2589: dynamic subtrees'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 NO-USER-MODIFICATION
    USAGE dSAOperation )

#
# Derived from RFC1274, but with new "short names"
#
attributetype ( 0.9.2342.19200300.100.1.1
    NAME ( 'uid' 'userid' )
    DESC 'RFC1274: user identifier'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

```



```

attributetype ( 0.9.2342.19200300.100.1.3
    NAME ( 'mail' 'rfc822Mailbox' )
    DESC 'RFC1274: RFC822 Mailbox'
    EQUALITY caseIgnoreIA5Match
    SUBSTR caseIgnoreIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )

objectclass ( 0.9.2342.19200300.100.4.19 NAME 'simpleSecurityObject'
    DESC 'RFC1274: simple security object'
    SUP top AUXILIARY
    MUST userPassword )

# RFC1274 + RFC2247
attributetype ( 0.9.2342.19200300.100.1.25
    NAME ( 'dc' 'domainComponent' )
    DESC 'RFC1274/2247: domain component'
    EQUALITY caseIgnoreIA5Match
    SUBSTR caseIgnoreIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

# RFC2247
objectclass ( 1.3.6.1.4.1.1466.344 NAME 'dcObject'
    DESC 'RFC2247: domain component object'
    SUP top AUXILIARY MUST dc )

# From RFC2377
objectclass ( 1.3.6.1.1.3.1 NAME 'uidObject'
    DESC 'RFC2377: uid object'
    SUP top AUXILIARY MUST uid )

#
# From draft-zeilenga-ldap-nameref-xx.txt
#     used to represent referrals in the directory
#
attributetype ( 2.16.840.1.113730.3.1.34 NAME 'ref'
    DESC 'Named referral'
    EQUALITY caseExactMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
    USAGE distributedOperation )

objectclass ( 2.16.840.1.113730.3.2.6 NAME 'referral'
    DESC 'Named referral object'
    SUP top STRUCTURAL MUST ref )

#
# LDAPsubEntry
#     deprecated!
objectclass ( 2.16.840.1.113719.2.142.6.1.1
    NAME 'LDAPsubEntry'
    DESC 'LDAP Subentry'
    SUP top STRUCTURAL MAY cn )

#
# OpenLDAProotDSE
#     likely to change!
objectclass ( 1.3.6.1.4.1.4203.1.4.1
    NAME ( 'OpenLDAProotDSE' 'LDAProotDSE' )
    DESC 'OpenLDAP Root DSE object'
    SUP top STRUCTURAL MAY cn )

#
# From Cosine Pilot
#
attributetype ( 0.9.2342.19200300.100.1.37 NAME 'associatedDomain'
    EQUALITY caseIgnoreIA5Match
    SUBSTR caseIgnoreIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

#
# From U-Mich
#
attributetype ( 1.3.6.1.4.1.250.1.32
    NAME ( 'krbName' 'kerberosName' )
    DESC 'Kerberos Name'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
    SINGLE-VALUE )

#
# draft-zeilenga-ldap-features-xx.txt (supportedFeatures)
#
attributetype ( 1.3.6.1.4.1.4203.1.3.5
    NAME 'supportedFeatures'
    DESC 'features supported by the server'
    EQUALITY objectIdentifierMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.38

```

```

    USAGE dSAOperation )

#
# OpenLDAP specific schema items
#
attributetype ( 1.3.6.1.4.1.4203.1.3.1
    NAME 'entry'
    DESC 'OpenLDAP ACL entry pseudo-attribute'
    SYNTAX 1.3.6.1.4.1.4203.1.1.1
    SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.4203.1.3.2
    NAME 'children'
    DESC 'OpenLDAP ACL children pseudo-attribute'
    SYNTAX 1.3.6.1.4.1.4203.1.1.1
    SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

# Experimental ( subject to change )
#   this really shouldn't be published!
attributetype ( 1.3.6.1.4.1.4203.666.1.5
    NAME 'OpenLDAPaci'
    DESC 'OpenLDAP access control information'
    EQUALITY OpenLDAPaciMatch
    SYNTAX 1.3.6.1.4.1.4203.666.2.1
    USAGE directoryOperation )

```

Приложение E: OpenLDAP cosine.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом OpenLDAP.

```

# $OpenLDAP: pkg/ldap/servers/slapd/schema/cosine.schema,v 1.6.2.7 2002/01/09 16:49:09 kurt
Exp $
#
# RFC1274: Cosine and Internet X.500 schema
#
# This file contains LDAPv3 schema derived from X.500 COSINE "pilot"
# schema. As this schema was defined for X.500(89), some
# oddities were introduced in the mapping to LDAPv3. The
# mappings were based upon: draft-ietf-asid-ldapv3-attributes-03.txt
# (a work in progress)
#
# Note: It seems that the pilot schema evolved beyond what was
# described in RFC1274. However, this document attempts to describes
# RFC1274 as published.
#
# Depends on core.schema

attributetype ( 0.9.2342.19200300.100.1.2 NAME 'textEncodedORAddress'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

attributetype ( 0.9.2342.19200300.100.1.4 NAME 'info'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{2048} )

attributetype ( 0.9.2342.19200300.100.1.5
    NAME ( 'drink' 'favouriteDrink' )
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

attributetype ( 0.9.2342.19200300.100.1.6 NAME 'roomNumber'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

attributetype ( 0.9.2342.19200300.100.1.7 NAME 'photo'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.23{25000} )

# 9.3.8. User Class
#
# The User Class attribute type specifies a category of computer user.
# The semantics placed on this attribute are for local interpretation.
# Examples of current usage of this attribute in academia are
# undergraduate student, researcher, lecturer, etc. Note that the
# organizationalStatus attribute may now often be preferred as it makes
# no distinction between computer users and others.
#
# userClass ATTRIBUTE
# WITH ATTRIBUTE-SYNTAX

```

```

#         caseIgnoreStringSyntax
#         (SIZE (1 .. ub-user-class))
#         ::= {pilotAttributeType 8}
#
attributetype ( 0.9.2342.19200300.100.1.8 NAME 'userClass'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.9. Host
#
# The Host attribute type specifies a host computer.
#
# host ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#         caseIgnoreStringSyntax
#         (SIZE (1 .. ub-host))
#     ::= {pilotAttributeType 9}
#
attributetype ( 0.9.2342.19200300.100.1.9 NAME 'host'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.10. Manager
#
# The Manager attribute type specifies the manager of an object
# represented by an entry.
#
# manager ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#         distinguishedNameSyntax
#     ::= {pilotAttributeType 10}
#
attributetype ( 0.9.2342.19200300.100.1.10 NAME 'manager'
    EQUALITY distinguishedNameMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

# 9.3.11. Document Identifier
#
# The Document Identifier attribute type specifies a unique identifier
# for a document.
#
# documentIdentifier ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#         caseIgnoreStringSyntax
#         (SIZE (1 .. ub-document-identifier))
#     ::= {pilotAttributeType 11}
#
attributetype ( 0.9.2342.19200300.100.1.11 NAME 'documentIdentifier'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.12. Document Title
#
# The Document Title attribute type specifies the title of a document.
#
# documentTitle ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#         caseIgnoreStringSyntax
#         (SIZE (1 .. ub-document-title))
#     ::= {pilotAttributeType 12}
#
attributetype ( 0.9.2342.19200300.100.1.12 NAME 'documentTitle'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.13. Document Version
#
# The Document Version attribute type specifies the version number of a
# document.
#
# documentVersion ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#         caseIgnoreStringSyntax
#         (SIZE (1 .. ub-document-version))
#     ::= {pilotAttributeType 13}
#
attributetype ( 0.9.2342.19200300.100.1.13 NAME 'documentVersion'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.14. Document Author
#

```

```

# The Document Author attribute type specifies the distinguished name
# of the author of a document.
#
#     documentAuthor ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             distinguishedNameSyntax
#     ::= {pilotAttributeType 14}
#
attributetype ( 0.9.2342.19200300.100.1.14 NAME 'documentAuthor'
    EQUALITY distinguishedNameMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

# 9.3.15. Document Location
#
# The Document Location attribute type specifies the location of the
# document original.
#
#     documentLocation ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             caseIgnoreStringSyntax
#             (SIZE (1 .. ub-document-location))
#     ::= {pilotAttributeType 15}
#
attributetype ( 0.9.2342.19200300.100.1.15 NAME 'documentLocation'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.16. Home Telephone Number
#
# The Home Telephone Number attribute type specifies a home telephone
# number associated with a person. Attribute values should follow the
# agreed format for international telephone numbers: i.e., "+44 71 123
# 4567".
#
#     homeTelephoneNumber ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             telephoneNumberSyntax
#     ::= {pilotAttributeType 20}
#
attributetype ( 0.9.2342.19200300.100.1.20
    NAME ( 'homePhone' 'homeTelephoneNumber' )
    EQUALITY telephoneNumberMatch
    SUBSTR telephoneNumberSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.50 )

# 9.3.17. Secretary
#
# The Secretary attribute type specifies the secretary of a person.
# The attribute value for Secretary is a distinguished name.
#
#     secretary ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             distinguishedNameSyntax
#     ::= {pilotAttributeType 21}
#
attributetype ( 0.9.2342.19200300.100.1.21 NAME 'secretary'
    EQUALITY distinguishedNameMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

# 9.3.18. Other Mailbox
#
# The Other Mailbox attribute type specifies values for electronic
# mailbox types other than X.400 and rfc822.
#
#     otherMailbox ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             SEQUENCE {
#                 mailboxType PrintableString, -- e.g. Telemail
#                 mailbox IA5String -- e.g. X378:Joe
#             }
#     ::= {pilotAttributeType 22}
#
attributetype ( 0.9.2342.19200300.100.1.22 NAME 'otherMailbox'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.39 )

# 9.3.22. DNS ARecord
#
# The A Record attribute type specifies a type A (Address) DNS resource
# record [6] [7].
#
#     aRecord ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             DNSRecordSyntax
#     ::= {pilotAttributeType 26}
#
## incorrect syntax?

```

```

attributetype ( 0.9.2342.19200300.100.1.26 NAME 'aRecord'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

## missing from RFC1274
## incorrect syntax?
attributetype ( 0.9.2342.19200300.100.1.27 NAME 'mDRecord'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

# 9.3.23.  MX Record
#
# The MX Record attribute type specifies a type MX (Mail Exchange) DNS
# resource record [6] [7].
#
# mXRecord ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#     DNSRecordSyntax
# ::= {pilotAttributeType 28}
#
## incorrect syntax!!
attributetype ( 0.9.2342.19200300.100.1.28 NAME 'mXRecord'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

# 9.3.24.  NS Record
#
# The NS Record attribute type specifies an NS (Name Server) DNS
# resource record [6] [7].
#
# nSRecord ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#     DNSRecordSyntax
# ::= {pilotAttributeType 29}
#
## incorrect syntax!!
attributetype ( 0.9.2342.19200300.100.1.29 NAME 'nSRecord'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

# 9.3.25.  SOA Record
#
# The SOA Record attribute type specifies a type SOA (Start of
# Authority) DNS resource record [6] [7].
#
# sOARRecord ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#     DNSRecordSyntax
# ::= {pilotAttributeType 30}
#
## incorrect syntax!!
attributetype ( 0.9.2342.19200300.100.1.30 NAME 'sOARRecord'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

# 9.3.26.  CNAME Record
#
# The CNAME Record attribute type specifies a type CNAME (Canonical
# Name) DNS resource record [6] [7].
#
# cNAMERRecord ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#     iA5StringSyntax
# ::= {pilotAttributeType 31}
#
## incorrect syntax!!
attributetype ( 0.9.2342.19200300.100.1.31 NAME 'cNAMERRecord'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

# 9.3.28.  Associated Name
#
# The Associated Name attribute type specifies an entry in the
# organisational DIT associated with a DNS/NRS domain. See [3] for
# more details of usage of this attribute.
#
# associatedName ATTRIBUTE
#     WITH ATTRIBUTE-SYNTAX
#     distinguishedNameSyntax
# ::= {pilotAttributeType 38}
#
attributetype ( 0.9.2342.19200300.100.1.38 NAME 'associatedName'
    EQUALITY distinguishedNameMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

# 9.3.29.  Home postal address
#

```

```

# The Home postal address attribute type specifies a home postal
# address for an object. This should be limited to up to 6 lines of 30
# characters each.
#
#     homePostalAddress ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             postalAddress
#             MATCHES FOR EQUALITY
#     ::= {pilotAttributeType 39}
#
attributetype ( 0.9.2342.19200300.100.1.39 NAME 'homePostalAddress'
    EQUALITY caseIgnoreListMatch
    SUBSTR caseIgnoreListSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.41 )

# 9.3.30. Personal Title
#
# The Personal Title attribute type specifies a personal title for a
# person. Examples of personal titles are "Ms", "Dr", "Prof" and "Rev".
#
#     personalTitle ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             caseIgnoreStringSyntax
#             (SIZE (1 .. ub-personal-title))
#     ::= {pilotAttributeType 40}
#
attributetype ( 0.9.2342.19200300.100.1.40 NAME 'personalTitle'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.31. Mobile Telephone Number
#
# The Mobile Telephone Number attribute type specifies a mobile
# telephone number associated with a person. Attribute values should
# follow the agreed format for international telephone numbers: i.e.,
# "+44 71 123 4567".
#
#     mobileTelephoneNumber ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             telephoneNumberSyntax
#     ::= {pilotAttributeType 41}
#
attributetype ( 0.9.2342.19200300.100.1.41
    NAME ( 'mobile' 'mobileTelephoneNumber' )
    EQUALITY telephoneNumberMatch
    SUBSTR telephoneNumberSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.50 )

# 9.3.32. Pager Telephone Number
#
# The Pager Telephone Number attribute type specifies a pager telephone
# number for an object. Attribute values should follow the agreed
# format for international telephone numbers: i.e., "+44 71 123 4567".
#
#     pagerTelephoneNumber ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             telephoneNumberSyntax
#     ::= {pilotAttributeType 42}
#
attributetype ( 0.9.2342.19200300.100.1.42
    NAME ( 'pager' 'pagerTelephoneNumber' )
    EQUALITY telephoneNumberMatch
    SUBSTR telephoneNumberSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.50 )

# 9.3.33. Friendly Country Name
#
# The Friendly Country Name attribute type specifies names of countries
# in human readable format. The standard attribute country name must
# be one of the two-letter codes defined in ISO 3166.
#
#     friendlyCountryName ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             caseIgnoreStringSyntax
#     ::= {pilotAttributeType 43}
#
attributetype ( 0.9.2342.19200300.100.1.43
    NAME ( 'co' 'friendlyCountryName' )
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# 9.3.34. Unique Identifier
#
# The Unique Identifier attribute type specifies a "unique identifier"
# for an object represented in the Directory. The domain within which

```



```

# the identifier is unique, and the exact semantics of the identifier,
# are for local definition. For a person, this might be an
# institution-wide payroll number. For an organisational unit, it
# might be a department code.
#
#     uniqueIdentifier ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             caseIgnoreStringSyntax
#             (SIZE (1 .. ub-unique-identifier))
#     ::= {pilotAttributeType 44}
#
attributetype ( 0.9.2342.19200300.100.1.44 NAME 'uniqueIdentifier'
    EQUALITY caseIgnoreMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.35. Organisational Status
#
# The Organisational Status attribute type specifies a category by
# which a person is often referred to in an organisation. Examples of
# usage in academia might include undergraduate student, researcher,
# lecturer, etc.
#
# A Directory administrator should probably consider carefully the
# distinctions between this and the title and userClass attributes.
#
#     organizationalStatus ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             caseIgnoreStringSyntax
#             (SIZE (1 .. ub-organizational-status))
#     ::= {pilotAttributeType 45}
#
attributetype ( 0.9.2342.19200300.100.1.45 NAME 'organizationalStatus'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.36. Janet Mailbox
#
# The Janet Mailbox attribute type specifies an electronic mailbox
# attribute following the syntax specified in the Grey Book of the
# Coloured Book series. This attribute is intended for the convenience
# of U.K users unfamiliar with rfc822 and little-endian mail addresses.
# Entries using this attribute MUST also include an rfc822Mailbox
# attribute.
#
#     janetMailbox ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             caseIgnoreIA5StringSyntax
#             (SIZE (1 .. ub-janet-mailbox))
#     ::= {pilotAttributeType 46}
#
attributetype ( 0.9.2342.19200300.100.1.46 NAME 'janetMailbox'
    EQUALITY caseIgnoreIA5Match
    SUBSTR caseIgnoreIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )

# 9.3.37. Mail Preference Option
#
# An attribute to allow users to indicate a preference for inclusion of
# their names on mailing lists (electronic or physical). The absence
# of such an attribute should be interpreted as if the attribute was
# present with value "no-list-inclusion". This attribute should be
# interpreted by anyone using the directory to derive mailing lists,
# and its value respected.
#
#     mailPreferenceOption ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX ENUMERATED {
#             no-list-inclusion(0),
#             any-list-inclusion(1), -- may be added to any lists
#             professional-list-inclusion(2)
#             -- may be added to lists
#             -- which the list provider
#             -- views as related to the
#             -- users professional inter-
#             -- ests, perhaps evaluated
#             -- from the business of the
#             -- organisation or keywords
#             -- in the entry.
#         }
#     ::= {pilotAttributeType 47}
#
attributetype ( 0.9.2342.19200300.100.1.47
    NAME 'mailPreferenceOption'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

# 9.3.38. Building Name
#

```

```

# The Building Name attribute type specifies the name of the building
# where an organisation or organisational unit is based.
#
#     buildingName ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#             caseIgnoreStringSyntax
#             (SIZE (1 .. ub-building-name))
#     ::= {pilotAttributeType 48}
#
attributetype ( 0.9.2342.19200300.100.1.48 NAME 'buildingName'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

# 9.3.39. DSA Quality
#
# The DSA Quality attribute type specifies the purported quality of a
# DSA. It allows a DSA manager to indicate the expected level of
# availability of the DSA. See [8] for details of the syntax.
#
#     dSAQuality ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX DSAQualitySyntax
#         SINGLE VALUE
#     ::= {pilotAttributeType 49}
#
attributetype ( 0.9.2342.19200300.100.1.49 NAME 'dSAQuality'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.19 SINGLE-VALUE )

# 9.3.40. Single Level Quality
#
# The Single Level Quality attribute type specifies the purported data
# quality at the level immediately below in the DIT. See [8] for
# details of the syntax.
#
#     singleLevelQuality ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX DataQualitySyntax
#         SINGLE VALUE
#     ::= {pilotAttributeType 50}
#
attributetype ( 0.9.2342.19200300.100.1.50 NAME 'singleLevelQuality'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.13 SINGLE-VALUE )

# 9.3.41. Subtree Minimum Quality
#
# The Subtree Minimum Quality attribute type specifies the purported
# minimum data quality for a DIT subtree. See [8] for more discussion
# and details of the syntax.
#
#     subtreeMinimumQuality ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX DataQualitySyntax
#         SINGLE VALUE
#         -- Defaults to singleLevelQuality
#     ::= {pilotAttributeType 51}
#
attributetype ( 0.9.2342.19200300.100.1.51 NAME 'subtreeMinimumQuality'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.13 SINGLE-VALUE )

# 9.3.42. Subtree Maximum Quality
#
# The Subtree Maximum Quality attribute type specifies the purported
# maximum data quality for a DIT subtree. See [8] for more discussion
# and details of the syntax.
#
#     subtreeMaximumQuality ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX DataQualitySyntax
#         SINGLE VALUE
#         -- Defaults to singleLevelQuality
#     ::= {pilotAttributeType 52}
#
attributetype ( 0.9.2342.19200300.100.1.52 NAME 'subtreeMaximumQuality'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.13 SINGLE-VALUE )

# 9.3.43. Personal Signature
#
# The Personal Signature attribute type allows for a representation of
# a person's signature. This should be encoded in G3 fax as explained
# in recommendation T.4, with an ASN.1 wrapper to make it compatible
# with an X.400 BodyPart as defined in X.420.
#
#     IMPORT G3FacsimileBodyPart FROM { mhs-motis ipms modules
#     information-objects }
#
#     personalSignature ATTRIBUTE
#         WITH ATTRIBUTE-SYNTAX
#         CHOICE {
#             g3-facsimile [3] G3FacsimileBodyPart
#         }

```

```

# (SIZE (1 .. ub-personal-signature))
# ::= {pilotAttributeType 53}
#
attributetype ( 0.9.2342.19200300.100.1.53 NAME 'personalSignature'
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.23 )

# 9.3.44. DIT Redirect
#
# The DIT Redirect attribute type is used to indicate that the object
# described by one entry now has a newer entry in the DIT. The entry
# containing the redirection attribute should be expired after a
# suitable grace period. This attribute may be used when an individual
# changes his/her place of work, and thus acquires a new organisational
# DN.
#
# dITRedirect ATTRIBUTE
# WITH ATTRIBUTE-SYNTAX
# distinguishedNameSyntax
# ::= {pilotAttributeType 54}
#
attributetype ( 0.9.2342.19200300.100.1.54 NAME 'dITRedirect'
                EQUALITY distinguishedNameMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

# 9.3.45. Audio
#
# The Audio attribute type allows the storing of sounds in the
# Directory. The attribute uses a u-law encoded sound file as used by
# the "play" utility on a Sun 4. This is an interim format.
#
# audio ATTRIBUTE
# WITH ATTRIBUTE-SYNTAX
# Audio
# (SIZE (1 .. ub-audio))
# ::= {pilotAttributeType 55}
#
attributetype ( 0.9.2342.19200300.100.1.55 NAME 'audio'
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.4{25000} )

# 9.3.46. Publisher of Document
#
# The Publisher of Document attribute is the person and/or organization
# that published a document.
#
# documentPublisher ATTRIBUTE
# WITH ATTRIBUTE SYNTAX caseIgnoreStringSyntax
# ::= {pilotAttributeType 56}
#
attributetype ( 0.9.2342.19200300.100.1.56 NAME 'documentPublisher'
                EQUALITY caseIgnoreMatch
                SUBSTR caseIgnoreSubstringsMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# 8.3.2. Pilot Person
#
# The PilotPerson object class is used as a sub-class of person, to
# allow the use of a number of additional attributes to be assigned to
# entries of object class person.
#
# pilotPerson OBJECT-CLASS
# SUBCLASS OF person
# MAY CONTAIN {
#     userid,
#     textEncodedORAddress,
#     rfc822Mailbox,
#     favouriteDrink,
#     roomNumber,
#     userClass,
#     homeTelephoneNumber,
#     homePostalAddress,
#     secretary,
#     personalTitle,
#     preferredDeliveryMethod,
#     businessCategory,
#     janetMailbox,
#     otherMailbox,
#     mobileTelephoneNumber,
#     pagerTelephoneNumber,
#     organizationalStatus,
#     mailPreferenceOption,
#     personalSignature}
# ::= {pilotObjectClass 4}
#
objectclass ( 0.9.2342.19200300.100.4.4
              NAME ( 'pilotPerson' 'newPilotPerson' )
              SUP person STRUCTURAL

```

```

        MAY ( userid $ textEncodedORAddress $ rfc822Mailbox $
              favouriteDrink $ roomNumber $ userClass $
              homeTelephoneNumber $ homePostalAddress $ secretary $
              personalTitle $ preferredDeliveryMethod $ businessCategory $
              janetMailbox $ otherMailbox $ mobileTelephoneNumber $
              pagerTelephoneNumber $ organizationalStatus $
              mailPreferenceOption $ personalSignature )
    )

# 8.3.3. Account
#
# The Account object class is used to define entries representing
# computer accounts. The userid attribute should be used for naming
# entries of this object class.
#
# account OBJECT-CLASS
#     SUBCLASS OF top
#     MUST CONTAIN {
#         userid
#     }
#     MAY CONTAIN {
#         description,
#         seeAlso,
#         localityName,
#         organizationName,
#         organizationalUnitName,
#         host
#     }
#     ::= {pilotObjectClass 5}
#
objectclass ( 0.9.2342.19200300.100.4.5 NAME 'account'
    SUP top STRUCTURAL
    MUST userid
    MAY ( description $ seeAlso $ localityName $
          organizationName $ organizationalUnitName $ host )
    )

# 8.3.4. Document
#
# The Document object class is used to define entries which represent
# documents.
#
# document OBJECT-CLASS
#     SUBCLASS OF top
#     MUST CONTAIN {
#         documentIdentifier
#     }
#     MAY CONTAIN {
#         commonName,
#         description,
#         seeAlso,
#         localityName,
#         organizationName,
#         organizationalUnitName,
#         documentTitle,
#         documentVersion,
#         documentAuthor,
#         documentLocation,
#         documentPublisher
#     }
#     ::= {pilotObjectClass 6}
#
objectclass ( 0.9.2342.19200300.100.4.6 NAME 'document'
    SUP top STRUCTURAL
    MUST documentIdentifier
    MAY ( commonName $ description $ seeAlso $ localityName $
          organizationName $ organizationalUnitName $
          documentTitle $ documentVersion $ documentAuthor $
          documentLocation $ documentPublisher )
    )

# 8.3.5. Room
#
# The Room object class is used to define entries representing rooms.
# The commonName attribute should be used for naming pentries of this
# object class.
#
# room OBJECT-CLASS
#     SUBCLASS OF top
#     MUST CONTAIN {
#         commonName
#     }
#     MAY CONTAIN {
#         roomNumber,
#         description,
#         seeAlso,
#         telephoneNumber
#     }
#     ::= {pilotObjectClass 7}
#
objectclass ( 0.9.2342.19200300.100.4.7 NAME 'room'
    SUP top STRUCTURAL
    MUST commonName

```

```

        MAY ( roomNumber $ description $ seeAlso $ telephoneNumber )
    )

# 8.3.6. Document Series
#
# The Document Series object class is used to define an entry which
# represents a series of documents (e.g., The Request For Comments
# papers).
#
# documentSeries OBJECT-CLASS
#     SUBCLASS OF top
#     MUST CONTAIN {
#         commonName
#     }
#     MAY CONTAIN {
#         description,
#         seeAlso,
#         telephoneNumber,
#         localityName,
#         organizationName,
#         organizationalUnitName
#     }
#     ::= {pilotObjectClass 9}
#
objectclass ( 0.9.2342.19200300.100.4.9 NAME 'documentSeries'
    SUP top STRUCTURAL
    MUST commonName
    MAY ( description $ seeAlso $ telephonenumber $
        localityName $ organizationName $ organizationalUnitName )
)

# 8.3.7. Domain
#
# The Domain object class is used to define entries which represent DNS
# or NRS domains. The domainComponent attribute should be used for
# naming entries of this object class. The usage of this object class
# is described in more detail in [3].
#
# domain OBJECT-CLASS
#     SUBCLASS OF top
#     MUST CONTAIN {
#         domainComponent
#     }
#     MAY CONTAIN {
#         associatedName,
#         organizationName,
#         organizationalAttributeSet
#     }
#     ::= {pilotObjectClass 13}
#
objectclass ( 0.9.2342.19200300.100.4.13 NAME 'domain'
    SUP top STRUCTURAL
    MUST domainComponent
    MAY ( associatedName $ organizationName $ description $
        businessCategory $ seeAlso $ searchGuide $ userPassword $
        localityName $ stateOrProvinceName $ streetAddress $
        physicalDeliveryOfficeName $ postalAddress $ postalCode $
        postOfficeBox $ streetAddress $
        facsimileTelephoneNumber $ internationalISDNNumber $
        telephoneNumber $ teletexTerminalIdentifier $ telexNumber $
        preferredDeliveryMethod $ destinationIndicator $
        registeredAddress $ x121Address )
)

# 8.3.8. RFC822 Local Part
#
# The RFC822 Local Part object class is used to define entries which
# represent the local part of RFC822 mail addresses. This treats this
# part of an RFC822 address as a domain. The usage of this object
# class is described in more detail in [3].
#
# rFC822localPart OBJECT-CLASS
#     SUBCLASS OF domain
#     MAY CONTAIN {
#         commonName,
#         surname,
#         description,
#         seeAlso,
#         telephoneNumber,
#         postalAttributeSet,
#         telecommunicationAttributeSet
#     }
#     ::= {pilotObjectClass 14}
#
objectclass ( 0.9.2342.19200300.100.4.14 NAME 'RFC822localPart'
    SUP domain STRUCTURAL
    MAY ( commonName $ surname $ description $ seeAlso $ telephonenumber $
        physicalDeliveryOfficeName $ postalAddress $ postalCode $
        postOfficeBox $ streetAddress $
        facsimileTelephoneNumber $ internationalISDNNumber $
        telephoneNumber $ teletexTerminalIdentifier $
        telexNumber $ preferredDeliveryMethod $ destinationIndicator $

```

```

        registeredAddress $ x121Address )
    )
# 8.3.9.  DNS Domain
#
# The DNS Domain (Domain NameServer) object class is used to define
# entries for DNS domains. The usage of this object class is described
# in more detail in [3].
#
# dNSDomain OBJECT-CLASS
#     SUBCLASS OF domain
#     MAY CONTAIN {
#         ARecord,
#         MDRecord,
#         MXRecord,
#         NSRecord,
#         SOARecord,
#         CNAMERecord}
#     ::= {pilotObjectClass 15}
#
objectclass ( 0.9.2342.19200300.100.4.15 NAME 'dNSDomain'
    SUP 'domain' STRUCTURAL
    MAY ( ARecord $ MDRecord $ MXRecord $ NSRecord $
        SOARecord $ CNAMERecord )
)

# 8.3.10.  Domain Related Object
#
# The Domain Related Object object class is used to define entries
# which represent DNS/NRS domains which are "equivalent" to an X.500
# domain: e.g., an organisation or organisational unit. The usage of
# this object class is described in more detail in [3].
#
# domainRelatedObject OBJECT-CLASS
#     SUBCLASS OF top
#     MUST CONTAIN {
#         associatedDomain}
#     ::= {pilotObjectClass 17}
#
objectclass ( 0.9.2342.19200300.100.4.17 NAME 'domainRelatedObject'
    SUP top AUXILIARY
    MUST associatedDomain )

# 8.3.11.  Friendly Country
#
# The Friendly Country object class is used to define country entries
# in the DIT. The object class is used to allow friendlier naming of
# countries than that allowed by the object class country. The naming
# attribute of object class country, countryName, has to be a 2 letter
# string defined in ISO 3166.
#
# friendlyCountry OBJECT-CLASS
#     SUBCLASS OF country
#     MUST CONTAIN {
#         friendlyCountryName}
#     ::= {pilotObjectClass 18}
#
objectclass ( 0.9.2342.19200300.100.4.18 NAME 'friendlyCountry'
    SUP country STRUCTURAL
    MUST friendlyCountryName )

# 8.3.13.  Pilot Organization
#
# The PilotOrganization object class is used as a sub-class of
# organization and organizationalUnit to allow a number of additional
# attributes to be assigned to entries of object classes organization
# and organizationalUnit.
#
# pilotOrganization OBJECT-CLASS
#     SUBCLASS OF organization, organizationalUnit
#     MAY CONTAIN {
#         buildingName}
#     ::= {pilotObjectClass 20}
#
objectclass ( 0.9.2342.19200300.100.4.20 NAME 'pilotOrganization'
    SUP ( organization $ organizationalUnit ) STRUCTURAL
    MAY buildingName )

# 8.3.14.  Pilot DSA
#
# The PilotDSA object class is used as a sub-class of the dsa object
# class to allow additional attributes to be assigned to entries for
# DSAs.
#
# pilotDSA OBJECT-CLASS
#     SUBCLASS OF dsa
#     MUST CONTAIN {

```



```

#         dSAQuality}
# ::= {pilotObjectClass 21}
#
objectclass ( 0.9.2342.19200300.100.4.21 NAME 'pilotDSA'
              SUP dsa STRUCTURAL
              MAY dSAQuality )

# 8.3.15. Quality Labelled Data
#
# The Quality Labelled Data object class is used to allow the
# assignment of the data quality attributes to subtrees in the DIT.
#
# See [8] for more details.
#
# qualityLabelledData OBJECT-CLASS
#     SUBCLASS OF top
#     MUST CONTAIN {
#         dSAQuality}
#     MAY CONTAIN {
#         subtreeMinimumQuality,
#         subtreeMaximumQuality}
# ::= {pilotObjectClass 22}
objectclass ( 0.9.2342.19200300.100.4.22 NAME 'qualityLabelledData'
              SUP top AUXILIARY
              MUST dsaQuality
              MAY ( subtreeMinimumQuality $ subtreeMaximumQuality )
              )

```

Приложение E: OpenLDAP authldap.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом [Courier-MTA](#) и [Courier-IMAP](#) (in src/courier-0.xx/authldap.schema).

```

#$Id: authldap.schema,v 1.5 2004/04/18 15:54:38 mrsam Exp $
#
# OID prefix: 1.3.6.1.4.1.10018
#
# Attributes: 1.3.6.1.4.1.10018.1.1
#
# Depends on: nis.schema, which depends on cosine.schema

attributetype ( 1.3.6.1.4.1.10018.1.1.1 NAME 'mailbox'
                DESC 'The absolute path to the mailbox for a mail account in a non-default
                location'
                EQUALITY caseExactIA5Match
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.10018.1.1.2 NAME 'quota'
                DESC 'A string that represents the quota on a mailbox'
                EQUALITY caseExactIA5Match
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.10018.1.1.3 NAME 'clearPassword'
                DESC 'A separate text that stores the mail account password in clear text'
                EQUALITY octetStringMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.40{128})

attributetype ( 1.3.6.1.4.1.10018.1.1.4 NAME 'maildrop'
                DESC 'RFC822 Mailbox - mail alias'
                EQUALITY caseIgnoreIA5Match
                SUBSTR caseIgnoreIA5SubstringsMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )

attributetype ( 1.3.6.1.4.1.10018.1.1.5 NAME 'mailsource'
                DESC 'Message source'
                EQUALITY caseIgnoreIA5Match
                SUBSTR caseIgnoreIA5SubstringsMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.10018.1.1.6 NAME 'virtualdomain'
                DESC 'A mail domain that is mapped to a single mail account'
                EQUALITY caseIgnoreIA5Match
                SUBSTR caseIgnoreIA5SubstringsMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.10018.1.1.7 NAME 'virtualdomainuser'
                DESC 'Mailbox that receives mail for a mail domain'
                EQUALITY caseIgnoreIA5Match
                SUBSTR caseIgnoreIA5SubstringsMatch
                SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.10018.1.1.8 NAME 'defaultdelivery'

```

```

DESC 'Default mail delivery instructions'
EQUALITY caseExactIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

#
# Objects: 1.3.6.1.4.1.10018.1.2
#
objectclass ( 1.3.6.1.4.1.10018.1.2.1 NAME 'CourierMailAccount' SUP top AUXILIARY
DESC 'Mail account object as used by the Courier mail server'
MUST ( mail $ homeDirectory $ uidNumber $ gidNumber )
MAY ( mailbox $ uid $ cn $ gecos $ description $ loginShell $ quota $
      userPassword $ clearPassword $ defaultdelivery ) )

objectclass ( 1.3.6.1.4.1.10018.1.2.2 NAME 'CourierMailAlias' SUP top AUXILIARY
DESC 'Mail aliasing/forwarding entry'
MUST ( mail $ maldrop )
MAY ( mailsource $ description ) )

objectclass ( 1.3.6.1.4.1.10018.1.2.3 NAME 'CourierDomainAlias' SUP top AUXILIARY
DESC 'Domain mail aliasing/forwarding entry'
MUST ( virtualdomain $ virtualdomainuser )
MAY ( mailsource $ description ) )

```

Приложение E: OpenLDAP dyngroup.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом OpenLDAP.

```

# dyngroup.schema -- Dynamic Group schema
# $OpenLDAP: pkg/ldap/servers/slapd/schema/dyngroup.schema,v 1.6.2.4 2008/02/12 05:17:43
# quanah Exp $
## This work is part of OpenLDAP Software .
##
## Copyright 1998-2008 The OpenLDAP Foundation.
## All rights reserved.
##
## Redistribution and use in source and binary forms, with or without
## modification, are permitted only as authorized by the OpenLDAP
## Public License.
##
## A copy of this license is available in the file LICENSE in the
## top-level directory of the distribution or, alternatively, at
## .
#
# Dynamic Group schema (experimental), as defined by Netscape. See
# http://www.redhat.com/docs/manuals/ent-server/pdf/esadmin611.pdf
# page 70 for details on how these groups were used.
#
# A description of the objectclass definition is available here:
# http://www.redhat.com/docs/manuals/dir-server/schema/7.1/oc_dir.html#1303745
#
# depends upon:
#     core.schema
#
# These definitions are considered experimental due to the lack of
# a formal specification (e.g., RFC).
#
# NOT RECOMMENDED FOR PRODUCTION USE!  USE WITH CAUTION!
#
# The Netscape documentation describes this as an auxiliary objectclass
# but their implementations have always defined it as a structural class.
# The sloppiness here is because Netscape-derived servers don't actually
# implement the X.500 data model, and they don't honor the distinction
# between structural and auxiliary classes. This fact is noted here:
# http://forum.java.sun.com/thread.jspa?threadID=5016864&messageID=9034636
#
# In accordance with other existing implementations, we define it as a
# structural class.
#
# Our definition of memberURL also does not match theirs but again
# their published definition and what works in practice do not agree.
# In other words, the Netscape definitions are broken and interoperability
# is not guaranteed.
#
# Also see the new DynGroup proposed spec at
# http://tools.ietf.org/html/draft-haripriya-dynamicgroup-02

objectIdentifier NetscapeRoot 2.16.840.1.113730

objectIdentifier NetscapeLDAP NetscapeRoot:3
objectIdentifier NetscapeLDAPattributeType NetscapeLDAP:1

```

```

objectIdentifier NetscapeLDAPObjectClass NetscapeLDAP:2

objectIdentifier OpenLDAPExp11 1.3.6.1.4.1.4203.666.11
objectIdentifier DynGroupBase OpenLDAPExp11:8
objectIdentifier DynGroupAttr DynGroupBase:1
objectIdentifier DynGroupOC DynGroupBase:2

attributetype ( NetscapeLDAPAttributeType:198
    NAME 'memberURL'
    DESC 'Identifies an URL associated with each member of a group. Any type of labeled
URL can be used.'
    SUP labeledURI )

attributetype ( DynGroupAttr:1
    NAME 'dgIdentity'
    DESC 'Identity to use when processing the memberURL'
    SUP distinguishedName SINGLE-VALUE )

attributeType ( DynGroupAttr:2
    NAME 'dgAuthz'
    DESC 'Optional authorization rules that determine who is allowed to assume the
dgIdentity'
    EQUALITY authzMatch
    SYNTAX 1.3.6.1.4.1.4203.666.2.7
    X-ORDERED 'VALUES' )

objectClass ( NetscapeLDAPObjectClass:33
    NAME 'groupOfURLs'
    SUP top STRUCTURAL
    MUST cn
    MAY ( memberURL $ businessCategory $ description $ o $ ou $
        owner $ seeAlso ) )

# The Haripriya dyngroup schema still needs a lot of work.
# We're just adding support for the dgIdentity attribute for now...
objectClass ( DynGroupOC:1
    NAME 'dgIdentityAux'
    SUP top AUXILIARY
    MAY ( dgIdentity $ dgAuthz ) )

```

Приложение E: OpenLDAP inetOrgPerson.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом OpenLDAP.

```

# $OpenLDAP: pkg/ldap/servers/slapd/schema/inetorgperson.schema,
# v 1.4.2.6 2001/10/09 17:15:08 kurt Exp $
#
# InetOrgPerson (RFC2798)
#
# Depends upon
#   Definition of an X.500 Attribute Type and an Object Class to Hold
#   Uniform Resource Identifiers (URIs) [RFC2079]
#   (core.schema)
#
#   A Summary of the X.500(96) User Schema for use with LDAPv3 [RFC2256]
#   (core.schema)
#
#   The COSINE and Internet X.500 Schema [RFC1274] (cosine.schema)
#
# carLicense
# This multivalued field is used to record the values of the license or
# registration plate associated with an individual.
attributetype ( 2.16.840.1.113730.3.1.1
    NAME 'carLicense'
    DESC 'RFC2798: vehicle license or registration plate'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# departmentNumber
# Code for department to which a person belongs. This can also be
# strictly numeric (e.g., 1234) or alphanumeric (e.g., ABC/123).
attributetype ( 2.16.840.1.113730.3.1.2
    NAME 'departmentNumber'
    DESC 'RFC2798: identifies a department within an organization'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# displayName
# When displaying an entry, especially within a one-line summary list, it
# is useful to be able to identify a name to be used. Since other attri-

```

```

# bute types such as 'cn' are multivalued, an additional attribute type is
# needed. Display name is defined for this purpose.
attributetype ( 2.16.840.1.113730.3.1.241
    NAME 'displayName'
    DESC 'RFC2798: preferred name to be used when displaying entries'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
    SINGLE-VALUE )

# employeeNumber
# Numeric or alphanumeric identifier assigned to a person, typically based
# on order of hire or association with an organization. Single valued.
attributetype ( 2.16.840.1.113730.3.1.3
    NAME 'employeeNumber'
    DESC 'RFC2798: numerically identifies an employee within an organization'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
    SINGLE-VALUE )

# employeeType
# Used to identify the employer to employee relationship. Typical values
# used will be "Contractor", "Employee", "Intern", "Temp", "External", and
# "Unknown" but any value may be used.
attributetype ( 2.16.840.1.113730.3.1.4
    NAME 'employeeType'
    DESC 'RFC2798: type of employment for a person'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# jpegPhoto
# Used to store one or more images of a person using the JPEG File
# Interchange Format [JFIF].
# Note that the jpegPhoto attribute type was defined for use in the
# Internet X.500 pilots but no referencable definition for it could be
# located.
attributetype ( 0.9.2342.19200300.100.1.60
    NAME 'jpegPhoto'
    DESC 'a JPEG image'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.28 )

# preferredLanguage
# Used to indicate an individual's preferred written or spoken
# language. This is useful for international correspondence or human-
# computer interaction. Values for this attribute type MUST conform to
# the definition of the Accept-Language header field defined in
# [RFC2068] with one exception: the sequence "Accept-Language" ":"
# should be omitted. This is a single valued attribute type.
attributetype ( 2.16.840.1.113730.3.1.39
    NAME 'preferredLanguage'
    DESC 'RFC2798: preferred written or spoken language for a person'
    EQUALITY caseIgnoreMatch
    SUBSTR caseIgnoreSubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
    SINGLE-VALUE )

# userSMIMECertificate
# A PKCS#7 [RFC2315] SignedData, where the content that is signed is
# ignored by consumers of userSMIMECertificate values. It is
# recommended that values have a 'contentType' of data with an absent
# 'content' field. Values of this attribute contain a person's entire
# certificate chain and an smimeCapabilities field [RFC2633] that at a
# minimum describes their SMIME algorithm capabilities. Values for
# this attribute are to be stored and requested in binary form, as
# 'userSMIMECertificate;binary'. If available, this attribute is
# preferred over the userCertificate attribute for S/MIME applications.
## OpenLDAP note: ";binary" transfer should NOT be used as syntax is binary
attributetype ( 2.16.840.1.113730.3.1.40
    NAME 'userSMIMECertificate'
    DESC 'RFC2798: PKCS#7 SignedData used to support S/MIME'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.5 )

# userPKCS12
# PKCS #12 [PKCS12] provides a format for exchange of personal identity
# information. When such information is stored in a directory service,
# the userPKCS12 attribute should be used. This attribute is to be stored
# and requested in binary form, as 'userPKCS12;binary'. The attribute
# values are PFX PDUs stored as binary data.
## OpenLDAP note: ";binary" transfer should NOT be used as syntax is binary
attributetype ( 2.16.840.1.113730.3.1.216
    NAME 'userPKCS12'
    DESC 'RFC2798: PKCS #12 PFX PDU for exchange of personal identity information'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.5 )

```

```

# inetOrgPerson
# The inetOrgPerson represents people who are associated with an
# organization in some way. It is a structural class and is derived
# from the organizationalPerson which is defined in X.521 [X521].
objectclass ( 2.16.840.1.113730.3.2.2
  NAME 'inetOrgPerson'
  DESC 'RFC2798: Internet Organizational Person'
  SUP organizationalPerson
  STRUCTURAL
  MAY ( audio $ businessCategory $ carLicense $ departmentNumber $
    displayName $ employeeNumber $ employeeType $ givenName $
    homePhone $ homePostalAddress $ initials $ jpegPhoto $
    labeledURI $ mail $ manager $ mobile $ o $ pager $ photo $
    roomNumber $ secretary $ uid $ userCertificate $
    x500uniqueIdentifier $ preferredLanguage $
    userSMIMECertificate $ userPKCS12 )
)

```

Приложение E: OpenLDAP java.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом OpenLDAP.

```

# Java Object Schema
# $OpenLDAP: pkg/ldap/servers/slapd/schema/java.schema,
# v 1.1.2.2 2000/09/13 18:16:13 kurt Exp $
# depends upon core.schema

# Network Working Group                                V. Ryan
# Request for Comments: 2713                            S. Seligman
# Category: Informational                               R. Lee
#                                                       Sun Microsystems, Inc.
#                                                       October 1999
#
#
#      Schema for Representing Java(tm) Objects in an LDAP Directory
#
# Status of this Memo
#
#      This memo provides information for the Internet community. It does
#      not specify an Internet standard of any kind. Distribution of this
#      memo is unlimited.
#
# Copyright Notice
#
#      Copyright (C) The Internet Society (1999). All Rights Reserved.
#
# Abstract
#
#      This document defines the schema for representing Java(tm) objects in
#      an LDAP directory [LDAPv3]. It defines schema elements to represent
#      a Java serialized object [Serial], a Java marshalled object [RMI], a
#      Java remote object [RMI], and a JNDI reference [JNDI].
#
# [trimmed]
#
# 3 Attribute Type Definitions
#
#      The following attribute types are defined in this document:
#
#      javaClassName
#      javaClassNames
#      javaCodebase
#      javaSerializedData
#      javaFactory
#      javaReferenceAddress
#      javaDoc
#
# 3.1 javaClassName
#
#      This attribute stores the fully qualified name of the Java object's
#      "distinguished" class or interface (for example, "java.lang.String").
#      It is a single-valued attribute. This attribute's syntax is '
#      Directory String' and its case is significant.
#
#      ( 1.3.6.1.4.1.42.2.27.4.1.6
#        NAME 'javaClassName'
#        DESC 'Fully qualified name of distinguished Java class or
#          interface'
#        EQUALITY caseExactMatch
#        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
#        SINGLE-VALUE

```

```

#         )
#
# attributetype ( 1.3.6.1.4.1.42.2.27.4.1.6
#         NAME 'javaClassName'
#         DESC 'Fully qualified name of distinguished Java class or interface'
#         EQUALITY caseExactMatch
#         SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
#         SINGLE-VALUE )
#
# 3.2 javaCodebase
#
# This attribute stores the Java class definition's locations. It
# specifies the locations from which to load the class definition for
# the class specified by the javaClassName attribute. Each value of
# the attribute contains an ordered list of URLs, separated by spaces.
# For example, a value of "url1 url2 url3" means that the three
# (possibly interdependent) URLs (url1, url2, and url3) form the
# codebase for loading in the Java class definition.
#
# If the javaCodebase attribute contains more than one value, each
# value is an independent codebase. That is, there is no relationship
# between the URLs in one value and those in another; each value can be
# viewed as an alternate source for loading the Java class definition.
# See [Java] for information regarding class loading.
#
# This attribute's syntax is 'IA5 String' and its case is significant.
#
# ( 1.3.6.1.4.1.42.2.27.4.1.7
#     NAME 'javaCodebase'
#     DESC 'URL(s) specifying the location of class definition'
#     EQUALITY caseExactIA5Match
#     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
# )
#
# attributetype ( 1.3.6.1.4.1.42.2.27.4.1.7
#     NAME 'javaCodebase'
#     DESC 'URL(s) specifying the location of class definition'
#     EQUALITY caseExactIA5Match
#     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
#
# 3.3 javaClassNames
#
# This attribute stores the Java object's fully qualified class or
# interface names (for example, "java.lang.String"). It is a
# multivalued attribute. When more than one value is present, each is
# the name of a class or interface, or ancestor class or interface, of
# this object.
#
# This attribute's syntax is 'Directory String' and its case is
# significant.
#
# ( 1.3.6.1.4.1.42.2.27.4.1.13
#     NAME 'javaClassNames'
#     DESC 'Fully qualified Java class or interface name'
#     EQUALITY caseExactMatch
#     SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
# )
#
# attributetype ( 1.3.6.1.4.1.42.2.27.4.1.13
#     NAME 'javaClassNames'
#     DESC 'Fully qualified Java class or interface name'
#     EQUALITY caseExactMatch
#     SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
#
# 3.4 javaSerializedData
#
# This attribute stores the serialized form of a Java object. The
# serialized form is described in [Serial].
#
# This attribute's syntax is 'Octet String'.
#
# ( 1.3.6.1.4.1.42.2.27.4.1.8
#     NAME 'javaSerializedData'
#     DESC 'Serialized form of a Java object'
#     SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
#     SINGLE-VALUE
# )
#
# attributetype ( 1.3.6.1.4.1.42.2.27.4.1.8
#     NAME 'javaSerializedData'
#     DESC 'Serialized form of a Java object'
#     SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
#     SINGLE-VALUE )
#
# 3.5 javaFactory
#

```



```

# This attribute stores the fully qualified class name of the object
# factory (for example, "com.wiz.jndi.WizObjectFactory") that can be
# used to create an instance of the object identified by the
# javaClassName attribute.
#
# This attribute's syntax is 'Directory String' and its case is
# significant.
#
# ( 1.3.6.1.4.1.42.2.27.4.1.10
#   NAME 'javaFactory'
#   DESC 'Fully qualified Java class name of a JNDI object factory'
#   EQUALITY caseExactMatch
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
#   SINGLE-VALUE
# )
#
attributetype ( 1.3.6.1.4.1.42.2.27.4.1.10
  NAME 'javaFactory'
  DESC 'Fully qualified Java class name of a JNDI object factory'
  EQUALITY caseExactMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

# 3.6 javaReferenceAddress
#
# This attribute represents the sequence of addresses of a JNDI
# reference. Each of its values represents one address, a Java object
# of type javax.naming.RefAddr. Its value is a concatenation of the
# address type and address contents, preceded by a sequence number (the
# order of addresses in a JNDI reference is significant). For example:
#
#   #0#TypeA#ValA
#   #1#TypeB#ValB
#   #2#TypeC##r00ABXNyABpq...
#
# In more detail, the value is encoded as follows:
#
# The delimiter is the first character of the value. For readability
# the character '#' is recommended when it is not otherwise used
# anywhere in the value, but any character may be used subject to
# restrictions given below.
#
# The first delimiter is followed by the sequence number. The sequence
# number of an address is its position in the JNDI reference, with the
# first address being numbered 0. It is represented by its shortest
# string form, in decimal notation.
#
# The sequence number is followed by a delimiter, then by the address
# type, and then by another delimiter. If the address is of Java class
# javax.naming.StringRefAddr, then this delimiter is followed by the
# value of the address contents (which is a string). Otherwise, this
# delimiter is followed immediately by another delimiter, and then by
# the Base64 encoding of the serialized form of the entire address.
#
# The delimiter may be any character other than a digit or a character
# contained in the address type. In addition, if the address contents
# is a string, the delimiter may not be the first character of that
# string.
#
# This attribute's syntax is 'Directory String' and its case is
# significant. It can contain multiple values.
#
# ( 1.3.6.1.4.1.42.2.27.4.1.11
#   NAME 'javaReferenceAddress'
#   DESC 'Addresses associated with a JNDI Reference'
#   EQUALITY caseExactMatch
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
# )
#
attributetype ( 1.3.6.1.4.1.42.2.27.4.1.11
  NAME 'javaReferenceAddress'
  DESC 'Addresses associated with a JNDI Reference'
  EQUALITY caseExactMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

# 3.7 javaDoc
#
# This attribute stores a pointer to the Java documentation for the
# class. Its value is a URL. For example, the following URL points to
# the specification of the java.lang.String class:
# http://java.sun.com/products/jdk/1.2/docs/api/java/lang/String.html
#
# This attribute's syntax is 'IA5 String' and its case is significant.
#
# ( 1.3.6.1.4.1.42.2.27.4.1.12
#   NAME 'javaDoc'
#   DESC 'The Java documentation for the class'

```

```

#           EQUALITY caseExactIA5Match
#           SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
#       )
#
attributetype ( 1.3.6.1.4.1.42.2.27.4.1.12
    NAME 'javaDoc'
    DESC 'The Java documentation for the class'
    EQUALITY caseExactIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

# 4 Object Class Definitions
#
#   The following object classes are defined in this document:
#
#       javaContainer
#       javaObject
#       javaSerializedObject
#       javaMarshaledObject
#       javaNamingReference
#
# 4.1 javaContainer
#
#   This structural object class represents a container for a Java
#   object.
#
#       ( 1.3.6.1.4.1.42.2.27.4.2.1
#         NAME 'javaContainer'
#         DESC 'Container for a Java object'
#         SUP top
#         STRUCTURAL
#         MUST ( cn )
#       )
#
objectclass ( 1.3.6.1.4.1.42.2.27.4.2.1
    NAME 'javaContainer'
    DESC 'Container for a Java object'
    SUP top
    STRUCTURAL
    MUST cn )

# 4.2 javaObject
#
#   This abstract object class represents a Java object. A javaObject
#   cannot exist in the directory; only auxiliary or structural
#   subclasses of it can exist in the directory.
#
#       ( 1.3.6.1.4.1.42.2.27.4.2.4
#         NAME 'javaObject'
#         DESC 'Java object representation'
#         SUP top
#         ABSTRACT
#         MUST ( javaClassName )
#         MAY ( javaClassNames $
#             javaCodebase $
#             javaDoc $
#             description )
#       )
#
objectclass ( 1.3.6.1.4.1.42.2.27.4.2.4
    NAME 'javaObject'
    DESC 'Java object representation'
    SUP top
    ABSTRACT
    MUST javaClassName
    MAY ( javaClassNames $ javaCodebase $
        javaDoc $ description ) )

# 4.3 javaSerializedObject
#
#   This auxiliary object class represents a Java serialized object. It
#   must be mixed in with a structural object class.
#
#       ( 1.3.6.1.4.1.42.2.27.4.2.5
#         NAME 'javaSerializedObject'
#         DESC 'Java serialized object'
#         SUP javaObject
#         AUXILIARY
#         MUST ( javaSerializedData )
#       )
#
objectclass ( 1.3.6.1.4.1.42.2.27.4.2.5
    NAME 'javaSerializedObject'
    DESC 'Java serialized object'
    SUP javaObject
    AUXILIARY
    MUST javaSerializedData )

```

```

# 4.4 javaMarshaledObject
#
#   This auxiliary object class represents a Java marshalled object.  It
#   must be mixed in with a structural object class.
#
#   ( 1.3.6.1.4.1.42.2.27.4.2.8
#     NAME 'javaMarshaledObject'
#     DESC 'Java marshalled object'
#     SUP javaObject
#     AUXILIARY
#     MUST ( javaSerializedData )
#   )
#
objectclass ( 1.3.6.1.4.1.42.2.27.4.2.8
  NAME 'javaMarshaledObject'
  DESC 'Java marshalled object'
  SUP javaObject
  AUXILIARY
  MUST javaSerializedData )

# 4.5 javaNamingReference
#
#   This auxiliary object class represents a JNDI reference.  It must be
#   mixed in with a structural object class.
#
#   ( 1.3.6.1.4.1.42.2.27.4.2.7
#     NAME 'javaNamingReference'
#     DESC 'JNDI reference'
#     SUP javaObject
#     AUXILIARY
#     MAY ( javaReferenceAddress $
#           javaFactory )
#   )
#
objectclass ( 1.3.6.1.4.1.42.2.27.4.2.7
  NAME 'javaNamingReference'
  DESC 'JNDI reference'
  SUP javaObject
  AUXILIARY
  MAY ( javaReferenceAddress $ javaFactory ) )

# Full Copyright Statement
#
#   Copyright (C) The Internet Society (1999).  All Rights Reserved.
#
#   This document and translations of it may be copied and furnished to
#   others, and derivative works that comment on or otherwise explain it
#   or assist in its implementation may be prepared, copied, published
#   and distributed, in whole or in part, without restriction of any
#   kind, provided that the above copyright notice and this paragraph are
#   included on all such copies and derivative works.  However, this
#   document itself may not be modified in any way, such as by removing
#   the copyright notice or references to the Internet Society or other
#   Internet organizations, except as needed for the purpose of
#   developing Internet standards in which case the procedures for
#   copyrights defined in the Internet Standards process must be
#   followed, or as required to translate it into languages other than
#   English.
#
#   The limited permissions granted above are perpetual and will not be
#   revoked by the Internet Society or its successors or assigns.
#
#   This document and the information contained herein is provided on an
#   "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING
#   TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING
#   BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION
#   HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF
#   MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

```

Приложение E: OpenLDAP misc.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом OpenLDAP.

```

# misc.schema -- assorted schema definitions
# $OpenLDAP: pkg/ldap/servers/slapd/schema/misc.schema,v 1.27.2.4 2008/02/11 23:24:26 kurt
Exp $
## This work is part of OpenLDAP Software .
##
## Copyright 1998-2008 The OpenLDAP Foundation.
## All rights reserved.

```

```

##
## Redistribution and use in source and binary forms, with or without
## modification, are permitted only as authorized by the OpenLDAP
## Public License.
##
## A copy of this license is available in the file LICENSE in the
## top-level directory of the distribution or, alternatively, at
## <http://www.OpenLDAP.org/license.html>.
#
# Assorted definitions from several sources, including
# 'works in progress'. Contents of this file are
# subject to change (including deletion) without notice.
#
# Not recommended for production use!
# Use with extreme caution!

#-----
# draft-lachman-laser-ldap-mail-routing-02.txt !!!EXPIRED!!!
# (a work in progress)
#
attributetype ( 2.16.840.1.113730.3.1.13
    NAME 'mailLocalAddress'
    DESC 'RFC822 email address of this recipient'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )

attributetype ( 2.16.840.1.113730.3.1.18
    NAME 'mailHost'
    DESC 'FQDN of the SMTP/MTA of this recipient'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256}
    SINGLE-VALUE )

attributetype ( 2.16.840.1.113730.3.1.47
    NAME 'mailRoutingAddress'
    DESC 'RFC822 routing address of this recipient'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256}
    SINGLE-VALUE )

# I-D leaves this OID TBD.
# iPlanet uses 2.16.840.1.113.730.3.2.147 but that is an
# improperly delegated OID. A typo is likely.
objectclass ( 2.16.840.1.113730.3.2.147
    NAME 'inetLocalMailRecipient'
    DESC 'Internet local mail recipient'
    SUP top AUXILIARY
    MAY ( mailLocalAddress $ mailHost $ mailRoutingAddress ) )

#-----
# draft-srivastava-ldap-mail-00.txt !!!EXPIRED!!!
# (a work in progress)
#
attributetype ( 1.3.6.1.4.1.42.2.27.2.1.15
    NAME 'rfc822MailMember'
    DESC 'rfc822 mail address of group member(s)'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

#-----
# !!!no I-D!!!
# (a work in progress)
#
objectclass ( 1.3.6.1.4.1.42.2.27.1.2.5
    NAME 'nisMailAlias'
    DESC 'NIS mail alias'
    SUP top STRUCTURAL
    MUST cn
    MAY rfc822MailMember )

```

Приложение E: OpenLDAP nis.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом OpenLDAP.

```

# $OpenLDAP: pkg/ldap/servers/slapd/schema/nis.schema,
# v 1.1.8.5 2000/09/28 17:35:12 kurt Exp $
# Definitions from RFC2307 (Experimental)
# An Approach for Using LDAP as a Network Information Service

# Depends upon core.schema and cosine.schema

# Note: The definitions in RFC2307 are given in syntaxes closely related
# to those in RFC2252, however, some liberties are taken that are not

```

```

# supported by RFC2252. This file has been written following RFC2252
# strictly.

# OID Base is iso(1) org(3) dod(6) internet(1) directory(1) nisSchema(1).
# i.e. nisSchema in RFC2307 is 1.3.6.1.1.1
#
# Syntaxes are under 1.3.6.1.1.1.0 (two new syntaxes are defined)
#   validaters for these syntaxes are incomplete, they only
#   implement printable string validation (which is good as the
#   common use of these syntaxes violates the specification).
# Attribute types are under 1.3.6.1.1.1.1
# Object classes are under 1.3.6.1.1.1.2

# Attribute Type Definitions
attributetype ( 1.3.6.1.1.1.1.0 NAME 'uidNumber'
    DESC 'An integer uniquely identifying a user in an administrative domain'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.1 NAME 'gidNumber'
    DESC 'An integer uniquely identifying a group in an administrative domain'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.2 NAME 'gecos'
    DESC 'The GECOS field; the common name'
    EQUALITY caseIgnoreIA5Match
    SUBSTR caseIgnoreIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.3 NAME 'homeDirectory'
    DESC 'The absolute path to the home directory'
    EQUALITY caseExactIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.4 NAME 'loginShell'
    DESC 'The path to the login shell'
    EQUALITY caseExactIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.5 NAME 'shadowLastChange'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.6 NAME 'shadowMin'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.7 NAME 'shadowMax'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.8 NAME 'shadowWarning'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.9 NAME 'shadowInactive'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.10 NAME 'shadowExpire'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.11 NAME 'shadowFlag'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.12 NAME 'memberUid'
    EQUALITY caseExactIA5Match
    SUBSTR caseExactIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.1.1.1.13 NAME 'memberNisNetgroup'
    EQUALITY caseExactIA5Match
    SUBSTR caseExactIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.1.1.1.14 NAME 'nisNetgroupTriple'
    DESC 'Netgroup triple'
    SYNTAX 1.3.6.1.1.1.0.0 )

attributetype ( 1.3.6.1.1.1.1.15 NAME 'ipServicePort'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.16 NAME 'ipServiceProtocol'

```

```

SUP name )

attributetype ( 1.3.6.1.1.1.1.17 NAME 'ipProtocolNumber'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.18 NAME 'oncRpcNumber'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.19 NAME 'ipHostNumber'
DESC 'IP address as a dotted decimal, eg. 192.168.1.1, omitting leading zeros'
EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{128} )

attributetype ( 1.3.6.1.1.1.1.20 NAME 'ipNetworkNumber'
DESC 'IP network as a dotted decimal, eg. 192.168, omitting leading zeros'
EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{128} SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.21 NAME 'ipNetmaskNumber'
DESC 'IP netmask as a dotted decimal, eg. 255.255.255.0, omitting leading zeros'
EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{128} SINGLE-VALUE )

attributetype ( 1.3.6.1.1.1.1.22 NAME 'macAddress'
DESC 'MAC address in maximal, colon separated hex notation, eg. 00:00:92:90:ee:e2'
EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{128} )

attributetype ( 1.3.6.1.1.1.1.23 NAME 'bootParameter'
DESC 'rpc.bootparamd parameter'
SYNTAX 1.3.6.1.1.1.0.1 )

attributetype ( 1.3.6.1.1.1.1.24 NAME 'bootFile'
DESC 'Boot image name'
EQUALITY caseExactIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.1.1.1.26 NAME 'nisMapName'
SUP name )

attributetype ( 1.3.6.1.1.1.1.27 NAME 'nisMapEntry'
EQUALITY caseExactIA5Match
SUBSTR caseExactIA5SubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{1024} SINGLE-VALUE )

# Object Class Definitions

objectclass ( 1.3.6.1.1.1.2.0 NAME 'posixAccount' SUP top AUXILIARY
DESC 'Abstraction of an account with POSIX attributes'
MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
MAY ( userPassword $ loginShell $ gecos $ description ) )

objectclass ( 1.3.6.1.1.1.2.1 NAME 'shadowAccount' SUP top AUXILIARY
DESC 'Additional attributes for shadow passwords'
MUST uid
MAY ( userPassword $ shadowLastChange $ shadowMin $
shadowMax $ shadowWarning $ shadowInactive $
shadowExpire $ shadowFlag $ description ) )

objectclass ( 1.3.6.1.1.1.2.2 NAME 'posixGroup' SUP top STRUCTURAL
DESC 'Abstraction of a group of accounts'
MUST ( cn $ gidNumber )
MAY ( userPassword $ memberUid $ description ) )

objectclass ( 1.3.6.1.1.1.2.3 NAME 'ipService' SUP top STRUCTURAL
DESC 'Abstraction an Internet Protocol service.
Maps an IP port and protocol (such as tcp or udp)
to one or more names; the distinguished value of
the cn attribute denotes the service's canonical
name'
MUST ( cn $ ipServicePort $ ipServiceProtocol )
MAY ( description ) )

objectclass ( 1.3.6.1.1.1.2.4 NAME 'ipProtocol' SUP top STRUCTURAL
DESC 'Abstraction of an IP protocol. Maps a protocol number
to one or more names. The distinguished value of the cn
attribute denotes the protocol's canonical name'
MUST ( cn $ ipProtocolNumber $ description )
MAY description )

objectclass ( 1.3.6.1.1.1.2.5 NAME 'oncRpc' SUP top STRUCTURAL
DESC 'Abstraction of an Open Network Computing (ONC)
[RFC1057] Remote Procedure Call (RPC) binding.
This class maps an ONC RPC number to a name.
The distinguished value of the cn attribute denotes

```



```

        the RPC service"s canonical name"
        MUST ( cn $ oncrpcNumber $ description )
        MAY description )

objectclass ( 1.3.6.1.1.2.6 NAME 'ipHost' SUP top AUXILIARY
    DESC 'Abstraction of a host, an IP device. The distinguished
        value of the cn attribute denotes the host"s canonical
        name. Device SHOULD be used as a structural class'
    MUST ( cn $ ipHostNumber )
    MAY ( l $ description $ manager ) )

objectclass ( 1.3.6.1.1.2.7 NAME 'ipNetwork' SUP top STRUCTURAL
    DESC 'Abstraction of a network. The distinguished value of
        the cn attribute denotes the network"s canonical name'
    MUST ( cn $ ipNetworkNumber )
    MAY ( ipNetmaskNumber $ l $ description $ manager ) )

objectclass ( 1.3.6.1.1.2.8 NAME 'nisNetgroup' SUP top STRUCTURAL
    DESC 'Abstraction of a netgroup. May refer to other netgroups'
    MUST cn
    MAY ( nisNetgroupTriple $ memberNisNetgroup $ description ) )

objectclass ( 1.3.6.1.1.2.9 NAME 'nisMap' SUP top STRUCTURAL
    DESC 'A generic abstraction of a NIS map'
    MUST nisMapName
    MAY description )

objectclass ( 1.3.6.1.1.2.10 NAME 'nisObject' SUP top STRUCTURAL
    DESC 'An entry in a NIS map'
    MUST ( cn $ nisMapEntry $ nisMapName )
    MAY description )

objectclass ( 1.3.6.1.1.2.11 NAME 'ieee802Device' SUP top AUXILIARY
    DESC 'A device with a MAC address; device SHOULD be
        used as a structural class'
    MAY macAddress )

objectclass ( 1.3.6.1.1.2.12 NAME 'bootableDevice' SUP top AUXILIARY
    DESC 'A device with boot parameters; device SHOULD be
        used as a structural class'
    MAY ( bootFile $ bootParameter ) )

```

Приложение E: OpenLDAP openldap.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом OpenLDAP.

```

# $OpenLDAP: pkg/ldap/servers/slapd/schema/openldap.schema,v 1.24.2.4
# 2008/02/11 23:26:49 kurt Exp $
## This work is part of OpenLDAP Software .
##
## Copyright 1998-2008 The OpenLDAP Foundation.
## All rights reserved.
##
## Redistribution and use in source and binary forms, with or without
## modification, are permitted only as authorized by the OpenLDAP
## Public License.
##
## A copy of this license is available in the file LICENSE in the
## top-level directory of the distribution or, alternatively, at
## .

#
# OpenLDAP Project's directory schema items
#
# depends upon:
#     core.schema
#     cosine.schema
#     inetorgperson.schema
#
# These are provided for informational purposes only.

objectIdentifier OpenLDAProot 1.3.6.1.4.1.4203

objectIdentifier OpenLDAP OpenLDAProot:1
objectIdentifier OpenLDAPattributeType OpenLDAP:3
objectIdentifier OpenLDAPobjectClass OpenLDAP:4

objectClass ( OpenLDAPobjectClass:3
    NAME 'OpenLDAPorg'
    DESC 'OpenLDAP Organizational Object'
    SUP organization
    MAY ( buildingName $ displayName $ labeledURI ) )

```

```

objectClass ( OpenLDAPobjectClass:4
    NAME 'OpenLDAPou'
    DESC 'OpenLDAP Organizational Unit Object'
    SUP organizationalUnit
    MAY ( buildingName $ displayName $ labeledURI $ o ) )

objectClass ( OpenLDAPobjectClass:5
    NAME 'OpenLDAPperson'
    DESC 'OpenLDAP Person'
    SUP ( pilotPerson $ inetOrgPerson )
    MUST ( uid $ cn )
    MAY ( givenName $ labeledURI $ o ) )

objectClass ( OpenLDAPobjectClass:6
    NAME 'OpenLDAPdisplayableObject'
    DESC 'OpenLDAP Displayable Object'
    AUXILIARY
    MAY displayName )

```

Приложение E: OpenLDAP ppolicy.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом OpenLDAP.

```

# $OpenLDAP: pkg/ldap/servers/slapd/schema/ppolicy.schema,v 1.7.2.3 2008/02/11 23:26:49 kurt
Exp $
## This work is part of OpenLDAP Software .
##
## Copyright 2004-2008 The OpenLDAP Foundation.
## All rights reserved.
##
## Redistribution and use in source and binary forms, with or without
## modification, are permitted only as authorized by the OpenLDAP
## Public License.
##
## A copy of this license is available in the file LICENSE in the
## top-level directory of the distribution or, alternatively, at
## .
##
## Portions Copyright (C) The Internet Society (2004).
## Please see full copyright statement below.

# Definitions from Draft behera-ldap-password-policy-07 (a work in progress)
# Password Policy for LDAP Directories
# With extensions from Hewlett-Packard:
# pwdCheckModule etc.

# Contents of this file are subject to change (including deletion)
# without notice.
#
# Not recommended for production use!
# Use with extreme caution!

#Network Working Group                                J. Sermersheim
#Internet-Draft                                       Novell, Inc
#Expires: April 24, 2005                             L. Poitou
#                                                    Sun Microsystems
#                                                    October 24, 2004
#
#
# Password Policy for LDAP Directories
# draft-behera-ldap-password-policy-08.txt
#
#Status of this Memo
#
# This document is an Internet-Draft and is subject to all provisions
# of section 3 of RFC 3667. By submitting this Internet-Draft, each
# author represents that any applicable patent or other IPR claims of
# which he or she is aware have been or will be disclosed, and any of
# which he or she become aware will be disclosed, in accordance with
# RFC 3668.
#
# Internet-Drafts are working documents of the Internet Engineering
# Task Force (IETF), its areas, and its working groups. Note that
# other groups may also distribute working documents as
# Internet-Drafts.
#
# Internet-Drafts are draft documents valid for a maximum of six months
# and may be updated, replaced, or obsoleted by other documents at any
# time. It is inappropriate to use Internet-Drafts as reference
# material or to cite them other than as "work in progress."
#

```

```

# The list of current Internet-Drafts can be accessed at
# http://www.ietf.org/ietf/1id-abstracts.txt.
#
# The list of Internet-Draft Shadow Directories can be accessed at
# http://www.ietf.org/shadow.html.
#
# This Internet-Draft will expire on April 24, 2005.
#
#Copyright Notice
#
# Copyright (C) The Internet Society (2004).
#
#Abstract
#
# Password policy as described in this document is a set of rules that
# controls how passwords are used and administered in Lightweight
# Directory Access Protocol (LDAP) based directories. In order to
# improve the security of LDAP directories and make it difficult for
# password cracking programs to break into directories, it is desirable
# to enforce a set of rules on password usage. These rules are made to
#
# [trimmed]
#
#5. Schema used for Password Policy
#
# The schema elements defined here fall into two general categories. A
# password policy object class is defined which contains a set of
# administrative password policy attributes, and a set of operational
# attributes are defined that hold general password policy state
# information for each user.
#
#5.2 Attribute Types used in the pwdPolicy ObjectClass
#
# Following are the attribute types used by the pwdPolicy object class.
#
#5.2.1 pwdAttribute
#
# This holds the name of the attribute to which the password policy is
# applied. For example, the password policy may be applied to the
# userPassword attribute.
#
attributetype ( 1.3.6.1.4.1.42.2.27.8.1.1
    NAME 'pwdAttribute'
    EQUALITY objectIdentifierMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
#
#5.2.2 pwdMinAge
#
# This attribute holds the number of seconds that must elapse between
# modifications to the password. If this attribute is not present, 0
# seconds is assumed.
#
attributetype ( 1.3.6.1.4.1.42.2.27.8.1.2
    NAME 'pwdMinAge'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )
#
#5.2.3 pwdMaxAge
#
# This attribute holds the number of seconds after which a modified
# password will expire.
#
# If this attribute is not present, or if the value is 0 the password
# does not expire. If not 0, the value must be greater than or equal
# to the value of the pwdMinAge.
#
attributetype ( 1.3.6.1.4.1.42.2.27.8.1.3
    NAME 'pwdMaxAge'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )
#
#5.2.4 pwdInHistory
#
# This attribute specifies the maximum number of used passwords stored
# in the pwdHistory attribute.
#
# If this attribute is not present, or if the value is 0, used
# passwords are not stored in the pwdHistory attribute and thus may be
# reused.
#
attributetype ( 1.3.6.1.4.1.42.2.27.8.1.4
    NAME 'pwdInHistory'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )

```

#5.2.5 pwdCheckQuality

```
#
# {TODO: Consider changing the syntax to OID. Each OID will list a
# quality rule (like min len, # of special characters, etc). These
# rules can be specified outside this document.}
#
# {TODO: Note that even though this is meant to be a check that happens
# during password modification, it may also be allowed to happen during
# authN. This is useful for situations where the password is encrypted
# when modified, but decrypted when used to authN.}
#
# This attribute indicates how the password quality will be verified
# while being modified or added. If this attribute is not present, or
# if the value is '0', quality checking will not be enforced. A value
# of '1' indicates that the server will check the quality, and if the
# server is unable to check it (due to a hashed password or other
# reasons) it will be accepted. A value of '2' indicates that the
# server will check the quality, and if the server is unable to verify
# it, it will return an error refusing the password.
```

```
attributetype ( 1.3.6.1.4.1.42.2.27.8.1.5
    NAME 'pwdCheckQuality'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )
```

#5.2.6 pwdMinLength

```
#
# When quality checking is enabled, this attribute holds the minimum
# number of characters that must be used in a password. If this
# attribute is not present, no minimum password length will be
# enforced. If the server is unable to check the length (due to a
# hashed password or otherwise), the server will, depending on the
# value of the pwdCheckQuality attribute, either accept the password
# without checking it ('0' or '1') or refuse it ('2').
```

```
attributetype ( 1.3.6.1.4.1.42.2.27.8.1.6
    NAME 'pwdMinLength'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )
```

#5.2.7 pwdExpireWarning

```
#
# This attribute specifies the maximum number of seconds before a
# password is due to expire that expiration warning messages will be
# returned to an authenticating user.
#
# If this attribute is not present, or if the value is 0 no warnings
# will be returned. If not 0, the value must be smaller than the value
# of the pwdMaxAge attribute.
```

```
attributetype ( 1.3.6.1.4.1.42.2.27.8.1.7
    NAME 'pwdExpireWarning'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )
```

#5.2.8 pwdGraceAuthNLimit

```
#
# This attribute specifies the number of times an expired password can
# be used to authenticate. If this attribute is not present or if the
# value is 0, authentication will fail.
```

```
attributetype ( 1.3.6.1.4.1.42.2.27.8.1.8
    NAME 'pwdGraceAuthNLimit'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )
```

#5.2.9 pwdLockout

```
#
# This attribute indicates, when its value is "TRUE", that the password
# may not be used to authenticate after a specified number of
# consecutive failed bind attempts. The maximum number of consecutive
# failed bind attempts is specified in pwdMaxFailure.
```

```
#
# If this attribute is not present, or if the value is "FALSE", the
# password may be used to authenticate when the number of failed bind
# attempts has been reached.
```

```
attributetype ( 1.3.6.1.4.1.42.2.27.8.1.9
    NAME 'pwdLockout'
    EQUALITY booleanMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
    SINGLE-VALUE )
```

```

#5.2.10  pwdLockoutDuration
#
# This attribute holds the number of seconds that the password cannot
# be used to authenticate due to too many failed bind attempts. If
# this attribute is not present, or if the value is 0 the password
# cannot be used to authenticate until reset by a password
# administrator.

attributetype ( 1.3.6.1.4.1.42.2.27.8.1.10
    NAME 'pwdLockoutDuration'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )

#5.2.11  pwdMaxFailure
#
# This attribute specifies the number of consecutive failed bind
# attempts after which the password may not be used to authenticate.
# If this attribute is not present, or if the value is 0, this policy
# is not checked, and the value of pwdLockout will be ignored.

attributetype ( 1.3.6.1.4.1.42.2.27.8.1.11
    NAME 'pwdMaxFailure'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )

#5.2.12  pwdFailureCountInterval
#
# This attribute holds the number of seconds after which the password
# failures are purged from the failure counter, even though no
# successful authentication occurred.
#
# If this attribute is not present, or if its value is 0, the failure
# counter is only reset by a successful authentication.

attributetype ( 1.3.6.1.4.1.42.2.27.8.1.12
    NAME 'pwdFailureCountInterval'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )

#5.2.13  pwdMustChange
#
# This attribute specifies with a value of "TRUE" that users must
# change their passwords when they first bind to the directory after a
# password is set or reset by a password administrator. If this
# attribute is not present, or if the value is "FALSE", users are not
# required to change their password upon binding after the password
# administrator sets or resets the password. This attribute is not set
# due to any actions specified by this document, it is typically set by
# a password administrator after resetting a user's password.

attributetype ( 1.3.6.1.4.1.42.2.27.8.1.13
    NAME 'pwdMustChange'
    EQUALITY booleanMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
    SINGLE-VALUE )

#5.2.14  pwdAllowUserChange
#
# This attribute indicates whether users can change their own
# passwords, although the change operation is still subject to access
# control. If this attribute is not present, a value of "TRUE" is
# assumed. This attribute is intended to be used in the absense of an
# access control mechanism.

attributetype ( 1.3.6.1.4.1.42.2.27.8.1.14
    NAME 'pwdAllowUserChange'
    EQUALITY booleanMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
    SINGLE-VALUE )

#5.2.15  pwdSafeModify
#
# This attribute specifies whether or not the existing password must be
# sent along with the new password when being changed. If this
# attribute is not present, a "FALSE" value is assumed.

attributetype ( 1.3.6.1.4.1.42.2.27.8.1.15
    NAME 'pwdSafeModify'
    EQUALITY booleanMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
    SINGLE-VALUE )

# <MODIFIED> Mozilla extension

```

```

# 5.2.16 pwdMaxTotalAttempts

# This attribute may take one of three values:
# a. Zero (0) This indicates that repeat password checking is not invoked.
# The value 0 is defaulted if the attribute is not present.
# b. A positive number in the range 1 to 65535 defines the maximum number of
# failed bind attempts using a repeated password after which any password
# may not be used to authenticate.
# c. -1 (minus 1). This indicates that repeat password detection features are
# required but an unlimited number of repeat password attempts are allowed.
# When repeat password detection is invoked (pwdMaxTotalAttempts is either -1
# or in the range 1 to 65535) each failed password attempt is evaluated against
# the list maintained in pwdUniqueAttempts and if not present will be added to
# pwdUniqueAttempts and also to pwdFailureTime. Thus the first attempt to use
# any password will count toward the value defined in pwdMaxFailure. If the
# failed password is present in pwdUniqueAttempts it will only increment the
# appropriate counter in pwdUniqueAttempts if pwdMaxTotalAttempts is in the
# range 1 to 65535. When the sum of all non-expired counts in pwdUniqueAttempts
# equals pwdMaxTotalAttempts then the action defined by pwdLockout is invoked.
# If this attribute is not present (defaults to 0), or if the value is 0, no
# repeat password detection is invoked and any failed password attempt (whether
# repeat or unique) will count toward the value defined by pwdMaxFailure. If
# the value is set to -1 then an unlimited number of repeat password attempts
# are allowed.

attributetype ( 1.3.6.1.4.1.13769.1.3.1
    NAME 'pwdMaxTotalAttempts'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )
<:/MODIFIED>

# HP extensions
#
# pwdCheckModule
#
# This attribute names a user-defined loadable module that provides
# a check_password() function. If pwdCheckQuality is set to '1' or '2'
# this function will be called after all of the internal password
# quality checks have been passed. The function has this prototype:
#
# int check_password( char *password, char **errormessage, void *arg )
#
# The function should return LDAP_SUCCESS for a valid password.

attributetype ( 1.3.6.1.4.1.4754.1.99.1
    NAME 'pwdCheckModule'
    EQUALITY caseExactIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
    DESC 'Loadable module that instantiates "check_password() function'
    SINGLE-VALUE )

objectclass ( 1.3.6.1.4.1.4754.2.99.1
    NAME 'pwdPolicyChecker'
    SUP top
    AUXILIARY
    MAY ( pwdCheckModule ) )

#5.1 The pwdPolicy Object Class
#
# This object class contains the attributes defining a password policy
# in effect for a set of users. Section 10 describes the
# administration of this object, and the relationship between it and
# particular objects.
#
objectclass ( 1.3.6.1.4.1.42.2.27.8.2.1
    NAME 'pwdPolicy'
    SUP top
    AUXILIARY
    MUST ( pwdAttribute )
    MAY ( pwdMinAge $ pwdMaxAge $ pwdInHistory $ pwdCheckQuality $
        pwdMinLength $ pwdExpireWarning $ pwdGraceAuthNLimit $ pwdLockout
        $ pwdLockoutDuration $ pwdMaxFailure $ pwdFailureCountInterval $
        pwdMustChange $ pwdAllowUserChange $ pwdMaxTotalAttempts $ pwdSafeModify ) )

#5.3 Attribute Types for Password Policy State Information
#
# Password policy state information must be maintained for each user.
# The information is located in each user entry as a set of operational
# attributes. These operational attributes are: pwdChangedTime,
# pwdAccountLockedTime, pwdFailureTime, pwdHistory, pwdGraceUseTime,
# pwdReset, pwdPolicySubEntry.
#
#5.3.1 Password Policy State Attribute Option
#
# Since the password policy could apply to several attributes used to

```



```

# store passwords, each of the above operational attributes must have
# an option to specify which pwdAttribute it applies to. The password
# policy option is defined as the following:
#
# pwd-<passwordAttribute>
#
# where passwordAttribute a string following the OID syntax
# (1.3.6.1.4.1.1466.115.121.1.38). The attribute type descriptor
# (short name) MUST be used.
#
# For example, if the pwdPolicy object has for pwdAttribute
# "userPassword" then the pwdChangedTime operational attribute, in a
# user entry, will be:
#
# pwdChangedTime;pwd-userPassword: 20000103121520Z
#
# This attribute option follows sub-typing semantics. If a client
# requests a password policy state attribute to be returned in a search
# operation, and does not specify an option, all subtypes of that
# policy state attribute are returned.
#
#5.3.2 pwdChangedTime
#
# This attribute specifies the last time the entry's password was
# changed. This is used by the password expiration policy. If this
# attribute does not exist, the password will never expire.
#
# ( 1.3.6.1.4.1.42.2.27.8.1.16
# NAME 'pwdChangedTime'
# DESC 'The time the password was last changed'
# EQUALITY generalizedTimeMatch
# ORDERING generalizedTimeOrderingMatch
# SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
# SINGLE-VALUE
# USAGE directoryOperation )
#
#5.3.3 pwdAccountLockedTime
#
# This attribute holds the time that the user's account was locked. A
# locked account means that the password may no longer be used to
# authenticate. A 000001010000Z value means that the account has been
# locked permanently, and that only a password administrator can unlock
# the account.
#
# ( 1.3.6.1.4.1.42.2.27.8.1.17
# NAME 'pwdAccountLockedTime'
# DESC 'The time an user account was locked'
# EQUALITY generalizedTimeMatch
# ORDERING generalizedTimeOrderingMatch
# SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
# SINGLE-VALUE
# USAGE directoryOperation )
#
#5.3.4 pwdFailureTime
#
# This attribute holds the timestamps of the consecutive authentication
# failures.
#
# ( 1.3.6.1.4.1.42.2.27.8.1.19
# NAME 'pwdFailureTime'
# DESC 'The timestamps of the last consecutive authentication
# failures'
# EQUALITY generalizedTimeMatch
# ORDERING generalizedTimeOrderingMatch
# SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
# USAGE directoryOperation )
#
#5.3.5 pwdHistory
#
# This attribute holds a history of previously used passwords. Values
# of this attribute are transmitted in string format as given by the
# following ABNF:
#
# pwdHistory = time "#" syntaxOID "#" length "#" data
#
# time      = <generalizedTimeString as specified in 6.14
#             of [RFC2252]>
#
# syntaxOID = numericoid ; the string representation of the
#                         ; dotted-decimal OID that defines the
#                         ; syntax used to store the password.
#                         ; numericoid is described in 4.1
#                         ; of [RFC2252].
#
# length    = numericstring ; the number of octets in data.
#                         ; numericstring is described in 4.1
#                         ; of [RFC2252].

```

```

#
# data      = <octets representing the password in the format
#            specified by syntaxOID>.
#
# This format allows the server to store, and transmit a history of
# passwords that have been used. In order for equality matching to
# function properly, the time field needs to adhere to a consistent
# format. For this purpose, the time field MUST be in GMT format.
#
# ( 1.3.6.1.4.1.42.2.27.8.1.20
#   NAME 'pwdHistory'
#   DESC 'The history of user s passwords'
#   EQUALITY octetStringMatch
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
#   USAGE directoryOperation )
#
#5.3.6 pwdGraceUseTime
#
# This attribute holds the timestamps of grace authentications after a
# password has expired.
#
# ( 1.3.6.1.4.1.42.2.27.8.1.21
#   NAME 'pwdGraceUseTime'
#   DESC 'The timestamps of the grace authentication after the
#   password has expired'
#   EQUALITY generalizedTimeMatch
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
#
#5.3.7 pwdReset
#
# This attribute holds a flag to indicate (when TRUE) that the password
# has been updated by the password administrator and must be changed by
# the user on first authentication.
#
# ( 1.3.6.1.4.1.42.2.27.8.1.22
#   NAME 'pwdReset'
#   DESC 'The indication that the password has been reset'
#   EQUALITY booleanMatch
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
#   SINGLE-VALUE
#   USAGE directoryOperation )
#
#5.3.8 pwdPolicySubentry
#
# This attribute points to the pwdPolicy subentry in effect for this
# object.
#
# ( 1.3.6.1.4.1.42.2.27.8.1.23
#   NAME 'pwdPolicySubentry'
#   DESC 'The pwdPolicy subentry in effect for this object'
#   EQUALITY distinguishedNameMatch
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
#   SINGLE-VALUE
#   USAGE directoryOperation )
#
# <MODIFIED> Mozilla extension
# 5.3.10 pwdUniqueAttempts
# This attribute holds a history of previously failed passwords
# attempts up to the limit defined by pwdMaxFailure. Values of this
# attribute are transmitted in string format as given by the following
# ABNF:
#
# pwdUniqueAttempts = time "#" count "#" length "#" data
#
# time      = <generalizedTimeString as specified in 6.14
#            of [RFC2252]>.
#
# count     = Integer ; the count of uses of this password
#               ; Integer is described in 6.16 of [RFC2252]
#
# length    = numericstring ; the number of octets in data.
#               ; numericstring is described in 4.1
#               ; of [RFC2252].
#
# data      = <octets representing the password in the default hash format
#            for the server
#
# In order for equality matching to function properly, the time field
# needs to adhere to a consistent format. For this purpose, the time
# field MUST be in GMT (UCT) format. The time field is set only when
# the attribute is initially added.
#
# (1.3.6.1.4.1.13769.1.3.2
#   NAME 'pwdUniqueAttempts'
#   DESC 'History of unique passwords attempts'
#   EQUALITY octetStringMatch
#   SYNTAX 1.3.6.1.4.1.1466.115.121.1.40

```

```

# NO-USER-MODIFICATION
# USAGE directoryOperation )
# When a failed password attempt occurs and the value of pwdMaxTotalAttempts
# is non 0 then pwdUniqueAttempts is searched for a match. If a match occurs
# then the count field of this entry is incremented only if the value of
# pwdMaxTotalAttempts is positive. If no match occurs then an entry is made
# in pwdFailureTime and in pwdUniqueAttempts (with a count field value of 1).
# If pwdMaxTotalAttempts is positive then when the sum of the count field
# values of all items in pwdUniqueAttempts equals pwdMaxTotalAttempts the
# action taken is defined by pwdLockout.
</MODIFIED>

#Disclaimer of Validity
#
# This document and the information contained herein are provided on an
# "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS
# OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET
# ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED,
# INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE
# INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED
# WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
#
#Copyright Statement
#
# Copyright (C) The Internet Society (2004). This document is subject
# to the rights, licenses and restrictions contained in BCP 78, and
# except as set forth therein, the authors retain all their rights.

```

Приложение E: OpenLDAP qmail.schema

Этот набор схемы данных распространяется со стандартным дистрибутивом qmail.

```

#
# qmail-ldap v3 directory schema
#
# The official qmail-ldap OID assigned by IANA is 7914
#
# Created by: David E. Storey
# Modified and included into qmail-ldap by Andre Oppermann
#
# I've gone through this schema and I think it is now correct but I'm
# not 100% certain. The next release will clear it up.
#
# This schema depends on:
#   - core.schema
#   - cosine.schema
#
# Attribute Type Definitions

attributetype ( 1.3.6.1.4.1.7914.1.2.1.1 NAME 'qmailUID'
  DESC 'UID of the user on the mailsystem'
  EQUALITY integerMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.36
  SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.2 NAME 'qmailGID'
  DESC 'GID of the user on the mailsystem'
  EQUALITY integerMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.36
  SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.3 NAME 'mailMessageStore'
  DESC 'Path to the maildir/mbox on the mail system'
  EQUALITY caseExactIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.4 NAME 'mailAlternateAddress'
  SUBSTR caseIgnoreSubstringsMatch
  DESC 'Secondary (alias) mailaddresses for the same user'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.5 NAME 'mailQuota'
  DESC 'The amount of space the user can use until all further messages get bounced.'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
  SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.6 NAME 'mailHost'
  DESC 'On which qmail server the messagestore of this user is located.'

```

```

EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.7 NAME 'mailForwardingAddress'
  DESC 'Address(es) to forward all incoming messages to.'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.8 NAME 'deliveryProgramPath'
  DESC 'Program to execute for all incoming mails.'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.9 NAME 'qmailDotMode'
  DESC 'Interpretation of .qmail files: both, dotonly, ldaponly, ldapwithprog, none'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
  SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.10 NAME 'deliveryMode'
  DESC 'multi field entries of: normal, forwardonly, nombox, localdelivery, reply, echo'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.44 )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.11 NAME 'mailReplyText'
  DESC 'A reply text for every incoming message'
  SUBSTR caseIgnoreSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{4096}
  SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.12 NAME 'accountStatus'
  DESC 'The status of a user account: active, nopop, disabled'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
  SINGLE-VALUE )

# Object Class Definitions

objectclass ( 1.3.6.1.4.1.7914.1.2.2.1 NAME 'qmailUser'
  DESC 'QMail-LDAP User' SUP top AUXILIARY
  MUST ( mail $ uid )
  MAY ( mailMessageStore $ homeDirectory $ userPassword $
    mailAlternateAddress $ qmailUID $ qmailGID $ mailQuota $
    mailHost $ mailForwardingAddress $ deliveryProgramPath $
    qmailDotMode $ deliveryMode $ mailReplyText $
    accountStatus ) )

```

Приложение E: Samba samba3.schema

Эта схема распространяется со стандартным дистрибутивом [Samba 3.x](#) (во FreeBSD /usr/local/share/examples/samba/Ldap/samba.schema). Данный набор схемы идёт со стандартным портом FreeBSD. После чтения документации создаётся впечатление, что разработчики некоторых дистрибутивов linux забывают, что они делают общее дело, и переименовывают данный набор схемы в samba3.schema. Примечание: во избежание путаницы мы удалили закомментированную старую (историческую) секцию Samaba 2.x.

```

##
## schema file for Samba 3.x
## Schema for storing Samba user accounts and group maps in LDAP
## OIDs are owned by the Samba Team
##
## Prerequisite schemas - uid          (cosine.schema)
##                      - displayName (inetorgperson.schema)
##                      - gidNumber   (nis.schema)
##
## 1.3.6.1.4.1.7165.2.1.x - attributetypes
## 1.3.6.1.4.1.7165.2.2.x - objectclasses
##
#####
##          Attributes used by Samba 3.0 schema          ##
#####

##
## Password hashes
##
attributetype ( 1.3.6.1.4.1.7165.2.1.24 NAME 'sambaLMPassword'
  DESC 'LanManager Password'

```

```

EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{32} SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.25 NAME 'sambaNTPassword'
DESC 'MD4 hash of the unicode password'
EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{32} SINGLE-VALUE )

##
## Account flags in string format ([UWDX    ])
##
attributetype ( 1.3.6.1.4.1.7165.2.1.26 NAME 'sambaAcctFlags'
DESC 'Account Flags'
EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{16} SINGLE-VALUE )

##
## Password timestamps & policies
##
attributetype ( 1.3.6.1.4.1.7165.2.1.27 NAME 'sambaPwdLastSet'
DESC 'Timestamp of the last password update'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.28 NAME 'sambaPwdCanChange'
DESC 'Timestamp of when the user is allowed to update the password'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.29 NAME 'sambaPwdMustChange'
DESC 'Timestamp of when the password will expire'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.30 NAME 'sambaLogonTime'
DESC 'Timestamp of last logon'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.31 NAME 'sambaLogoffTime'
DESC 'Timestamp of last logoff'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.32 NAME 'sambaKickoffTime'
DESC 'Timestamp of when the user will be logged off automatically'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.48 NAME 'sambaBadPasswordCount'
DESC 'Bad password attempt count'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.49 NAME 'sambaBadPasswordTime'
DESC 'Time of the last bad password attempt'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

##
## string settings
##
attributetype ( 1.3.6.1.4.1.7165.2.1.33 NAME 'sambaHomeDrive'
DESC 'Driver letter of home directory mapping'
EQUALITY caseIgnoreIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{4} SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.34 NAME 'sambaLogonScript'
DESC 'Logon script path'
EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{255} SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.35 NAME 'sambaProfilePath'
DESC 'Roaming profile path'
EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{255} SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.36 NAME 'sambaUserWorkstations'
DESC 'List of user workstations the user is allowed to logon to'
EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{255} SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.37 NAME 'sambaHomePath'
DESC 'Home directory UNC path'
EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} )

```

```

attributetype ( 1.3.6.1.4.1.7165.2.1.38 NAME 'sambaDomainName'
    DESC 'Windows NT domain to which the user belongs'
    EQUALITY caseIgnoreMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} )

attributetype ( 1.3.6.1.4.1.7165.2.1.47 NAME 'sambaMungedDial'
    DESC '
    EQUALITY caseExactMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{1050} )

##
## SID, of any type
##

attributetype ( 1.3.6.1.4.1.7165.2.1.20 NAME 'sambaSID'
    DESC 'Security ID'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{64} SINGLE-VALUE )

##
## Primary group SID, compatible with ntSid
##

attributetype ( 1.3.6.1.4.1.7165.2.1.23 NAME 'sambaPrimaryGroupSID'
    DESC 'Primary Group Security ID'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{64} SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.51 NAME 'sambaSIDList'
    DESC 'Security ID List'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{64} )

##
## group mapping attributes
##
attributetype ( 1.3.6.1.4.1.7165.2.1.19 NAME 'sambaGroupType'
    DESC 'NT Group Type'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

##
## Store info on the domain
##

attributetype ( 1.3.6.1.4.1.7165.2.1.21 NAME 'sambaNextUserRid'
    DESC 'Next NT rid to give out for users'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.22 NAME 'sambaNextGroupRid'
    DESC 'Next NT rid to give out for groups'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.39 NAME 'sambaNextRid'
    DESC 'Next NT rid to give out for anything'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7165.2.1.40 NAME 'sambaAlgorithmicRidBase'
    DESC 'Base at which the samba RID generation algorithm should operate'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

#####
##          objectClasses used by Samba 3.0 schema          ##
#####

## The X.500 data model (and therefore LDAPv3) says that each entry can
## only have one structural objectclass.  OpenLDAP 2.0 does not enforce
## this currently but will in v2.1

##
## added new objectclass (and OID) for 3.0 to help us deal with backwards
## compatibility with 2.2 installations (e.g. ldapsam_compat)  --jerry
##
objectclass ( 1.3.6.1.4.1.7165.2.2.6 NAME 'sambaSamAccount' SUP top AUXILIARY
    DESC 'Samba 3.0 Auxilary SAM Account'
    MUST ( uid $ sambaSID )
    MAY ( cn $ sambaLMPassword $ sambaNTPassword $ sambaPwdLastSet $
        sambaLogonTime $ sambaLogoffTime $ sambaKickoffTime $
        sambaPwdCanChange $ sambaPwdMustChange $ sambaAcctFlags $
        displayName $ sambaHomePath $ sambaHomeDrive $ sambaLogonScript $

```



```

sambaProfilePath $ description $ sambaUserWorkstations $
sambaPrimaryGroupSID $ sambaDomainName $ sambaMungedDial $
sambaBadPasswordCount $ sambaBadPasswordTime))

##
## Group mapping info
##
objectclass ( 1.3.6.1.4.1.7165.2.2.4 NAME 'sambaGroupMapping' SUP top AUXILIARY
DESC 'Samba Group Mapping'
MUST ( gidNumber $ sambaSID $ sambaGroupType )
MAY ( displayName $ description $ sambaSIDList ))

##
## Whole-of-domain info
##
objectclass ( 1.3.6.1.4.1.7165.2.2.5 NAME 'sambaDomain' SUP top STRUCTURAL
DESC 'Samba Domain Information'
MUST ( sambaDomainName $ sambaSID )
MAY ( sambaNextRid $ sambaNextGroupRid $ sambaNextUserRid $
sambaAlgorithmicRidBase ) )

## used for idmap_ldap module
objectclass ( 1.3.6.1.4.1.7165.1.2.2.7 NAME 'sambaUnixIdPool' SUP top AUXILIARY
DESC 'Pool for allocating UNIX uids/gids'
MUST ( uidNumber $ gidNumber ) )

objectclass ( 1.3.6.1.4.1.7165.1.2.2.8 NAME 'sambaIdmapEntry' SUP top AUXILIARY
DESC 'Mapping from a SID to an ID'
MUST ( sambaSID )
MAY ( uidNumber $ gidNumber ) )

objectclass ( 1.3.6.1.4.1.7165.1.2.2.9 NAME 'sambaSidEntry' SUP top STRUCTURAL
DESC 'Structural Class for a SID'
MUST ( sambaSID ) )

```

Open Source LDAP-проекты

Не так давно единственной доступной Open Source LDAP-системой был OpenLDAP (изначально основанный на коде Мичиганского университета), но сейчас положение дел изменилось. Вам могут понравиться [389 Directory Server](#) (бывший Fedora Directory Server), также основанный на разработках Мичиганского университета, [OpenDS](#) - оригинальная Java-реализация от Sun (ныне принадлежащая Oracle) или [ApacheDS \(Apache Directory\)](#) также написанный на Java. Все эти замечательные службы каталогов вместе с OpenLDAP предоставляют богатейший выбор возможностей и функций в сфере Open Source LDAP.

На наш взгляд, наличие нескольких проектов ничего, кроме пользы в деле поступательного развития служб каталогов, принести не может. В своём руководстве, в дополнение к информации об OpenLDAP, мы решили добавить документацию по ApacheDS как представителю второго поколения служб каталогов с рядом интересных возможностей, которые, как нам кажется, указывают то направление, в котором LDAP будет развиваться.

Нам бы очень хотелось, чтобы участники разных проектов меньше конфликтовали и не забывали про вежливость друг к другу, поскольку наличие разных реализаций идет на пользу не только пользователям, но и сообществам разработчиков. Далее приведена наша оценка этих проектов. Однако, следует учитывать, что каждый из них имеет уникальные особенности и множество верных последователей. Эта оценка лишь отражает нашу точку зрения в данный момент времени, и, конечно, она основана на том, что мы знаем.

Примечание: просто пропустите этот раздел, если Вам не нравится то, что здесь написано. Если же Вам кажется, что мы что-то упустили или неправильно поняли, и Ваш священный долг нам об этом сообщить, то на каждой странице Вы найдете две ссылки, по которым с нами можно связаться. Мы обязательно ответим Вам в том же духе, в котором Вы напишете.

OpenLDAP - широко известная и заслуженно считающаяся эталонной служба каталогов, основанная на оригинальном исходном коде Мичиганского университета. Первоначальный код с годами был существенно изменен, реструктурирован и доработан. Поскольку он написан на языке программирования C, мы считаем что он всегда будет работать быстрее, чем код, написанный на Java. Возможно, когда-нибудь это положение изменится (с введением Java-компиляторов и т. п.), но что-то не верится. Дискуссии на этот счёт ещё продолжаются. Одни утверждают, что с ростом производительности CPU разница между программами на этих языках теряет былое значение, другие же считают, что производительность ввода/вывода является существенным сдерживающим фактором в любых условиях. До версии 2.2 в OpenLDAP больше уделяли внимания реализации стандартов, нежели функциональным особенностям, но начиная с версии 2.4 система

приобрела такие значимые функции, как конфигурация времени исполнения, продвинутая репликация, и даже репликация с участием нескольких главных серверов (multi-master). Поскольку OpenLDAP реализует почти всё, что указано в стандартах, он может быть довольно сложен в настройке, особенно если Вы пытаетесь интегрировать его в гетерогенную среду AD. Кроме того, проект имеет особенность менять что-то в каждой новой версии. Так, конфигурационный файл, исправно работавший в версии x, ругается в версии y, а исправленный вариант ругается вновь в версии z. Это резко контрастирует с такими программами как Apache, где кажется, что конфигурационный файл имеет безграничную власть. И всё же мы считаем, что для больших проектов с достаточным количеством рабочих ресурсов (людей/знаний) и для проектов с неизвестным количеством ресурсов, OpenLDAP - правильный выбор, и он будет летать даже на относительно слабом железе.

Проект 389 Directory Services (бывший Fedora Directory Server) также основан на коде Мичиганского университета. В течение своей жизни он не раз преобразовывался и был известен как Netscape Directory Server, Redhat Directory Server и, наконец, стал именоваться 389. Написан на C и должен обладать сходной с OpenLDAP производительностью. Обладает продвинутыми функциями, такими как конфигурация в реальном времени, репликация с участием нескольких главных серверов (multi-master). Мы уверены, что новшества продолжают появляться. Отметим, что за первые 18 месяцев (или около того) своего существования под новым именем большинство изменений проекта было направлено на исправление ошибок и увеличение производительности, а не на наращивание функционала. Достоинно и похвально одновременно.

Мы не большие поклонники Java - складывается впечатление, что этот язык вобрал в себя худшие черты C++, да ещё и не имеет указателей (это была шутка, но если с чувством юмора у Вас туговато, просто вспомните, что мы программируем на C и Ruby, и посочувствуйте нам). Java - очень продуктивный и распространённый язык, позволяющий в короткие сроки написать значительное количество функций и обеспечить работу практически на любой платформе "из коробки", если только Вы готовы смириться с более скромной (по сравнению с C) производительностью и огромным объемом требуемой памяти. Оба новичка в классе LDAP с открытым исходным кодом (OpenDS и ApacheDS) решили использовать Java.

Всё более очевидными становятся реальные шаги Open Source LDAP (каталогов) в сторону упрощения установки и конфигурации безопасных каталогов, способных тесно интегрироваться с AD для обеспечения управления идентификацией и других возможностей в гетерогенных средах. Идеальным решением, на наш взгляд, будет установка в один клик всего сразу (LDAP, базы данных, kerberos, ssl), приемлемая рабочая конфигурация "из коробки" и простота расширения (функциональности и рабочих характеристик) с помощью инструментов высокого уровня. Мы считаем это основными требованиями к каталогам, содержащим от 100 до 200 000 записей. Если же записей больше, на первый план выходят увеличение ресурсов для их поддержки и соображения производительности. Тот факт, что LDAP основан на стандартах, позволяет относительно безболезненно мигрировать от одной реализации службы каталогов к другой и даёт пользователям возможность выбирать реализацию с оптимальными характеристиками (функциональности, либо производительности) в зависимости от изменения потребностей.

Мы решили сосредоточиться на ApacheDS - представителе нового поколения LDAP-серверов, приближающему к описанному нами выше уровню функциональности. В частности, мы отмечаем Directory Studio (основанный на Eclipse) и идеи проекта по оптимизации доступа к данным за счет использования возможностей классических транзакционных баз данных, таких как хранимые процедуры и триггеры. Решение сосредоточиться на ApacheDS совсем не говорит о нашем негативном отношении к OpenDS, за которым мы продолжаем следить с большим интересом.

LDAP для учёных ракетчиков - журнал изменений

Изменения, внесённые с момента предыдущей ревизии этой документации. Мы постоянно обновляем доступный on-line текст, а публикуем перечень изменений в данный журнал только с выходом новой версии. Периодически мы выпускаем новую версию этой документации, но и до момента выпуска помещаемые в данный журнал изменения уже доступны некоторое время on-line.

Версия 0.1.15 - 26 июля 2013 г.

1. Основное направление - Последовательное добавление материалов по ApacheDS и приведение материалов по OpenLDAP в соответствие с версией 2.4+.
2. Глава 2 - Изменение формулировок текста для большей (надеемся) ясности.
3. Глава 3 - Незначительные уточнения описания атрибутов x-ordered и исправление незначительных опечаток.
4. Глава 4 - Обновлён раздел Win2k по дистрибутивам OpenLDAP для Windows - примечания по

использованию cygwin или ApacheDS в Windows.

5. Глава 5 - Исправление незначительных опечаток.
6. Глава 5 - Исправление ошибки формата jpegphoto:< file:// (некорректное наличие пробела) в пункте 5.1.7.
7. Глава 5 - Шаг 3, исправление незначительных опечаток.
8. Глава 5 - Шаг 4, исправление незначительных опечаток, исправление uid в пункте 5.4.6.
9. Глава 6 - Дополнительные ссылки, незначительная ревизия текста.
10. Глава 6 - Индексирование - исправлено неверное определение subinitial и subfinal.
11. Глава 6 - rpolisy - в разделе разблокировки учётных записей имя атрибута исправлено на pwdLockout.
12. Глава 7 - Дополнительные примечания по синхронизации с использованием устаревшего демона slurpd.
13. Глава 7 - Уточнение процедуры синхронизации при репликации применительно как к slurpd, так и к syncrpl.
14. Глава 7 - Уточнение текста в разделе "отсылки". Добавлены процедуры и примечания по удалению записей отсылок.
15. Глава 7 - Новый раздел "псевдонимы" (aliases) - определение и описание с примерами.
16. Глава 8 - Исправление ошибок формата HTML.
17. Глава 14 - Исправление опечаток и дублированных аргументов в ldapadd, ldapmodify.
18. Глава 14 - В описание применения добавлен аргумент -M, необходимый для удаления отсылок.
19. Приложение А - Типы данных LDAP - изменение форматирования и незначительные уточнения типа Integer.
20. Приложение А - Индексирование - исправлено неверное определение subinitial и subfinal.
21. Приложение А - LDAP OID - незначительные изменения форматирования.
22. Приложение А - Фильтры соответствия компонентов LDAP - исправление ссылок на RFC.
23. Приложение С - Обновлён список RFC.
24. Приложение D - Незначительные уточняющие изменения в глоссарии.
25. Приложение Е - Объекты - изменение форматирования.

Версия 0.1.14 - 16 мая 2012 г.

1. Основное направление - Последовательное добавление материалов по ApacheDS и приведение материалов по OpenLDAP в соответствие с версией 2.4+.
2. Основное направление - Обновление ссылок с Fedora Directory Server на 389 Directory Server.
3. Глава 2 - Ревизия текста и незначительные расширения для повышения читабельности (хочется надеяться).
4. Глава 3 - Описание нестандартной возможности атрибутов X-ORDERED, используемой в OLC (cn=config).
5. Глава 3 - Незначительная ревизия и расширение текста для повышения читабельности.
6. Глава 4 - Обновлено руководство по установке ApacheDS в Windows. Подтверждается, что процесс установки не изменился и для Windows 7.
7. Глава 6 - cn=config, ошибочное указание oldRootDn в примечании исправлено на olcRootDn.
8. Глава 6 - Добавлена ссылка на наложение dynlist (динамические группы).
9. Глава 6 - Добавлено описание наложения dynlist.
10. Глава 6 - Изменена терминология описания cn=config с real-time configuration (rtc) на on-line configuration (OLC) для отражения префикса 'olc' в названиях директив.
11. Глава 6 - Добавление имён атрибутов olc (cn=config) ко всем документированным директивам конфигурации.
12. Глава 6 - Добавление объектных классов, используемых в OLC (cn=config).
13. Глава 6 - Добавление замечаний по использованию OLC (cn=config), охватывающих задачи, решаемые чаще всего, такие как добавление наборов схемы данных, ACL, модулей, баз данных и наложений.
14. Глава 6 - Наложение syncprov - добавлены имена атрибутов OLC (cn=config).
15. Глава 9 - Добавлен пример URL для поиска на localhost.
16. Глава 11 - Добавлена новая статья по динамическим группам.
17. Глава 12 - Добавлены дополнительные замечания по типам ошибок.
18. Глава 14 - Документация LDAPBrowser/Editor. Исправлены неработающие локальные ссылки.
19. Глава 14 - Добавлено описание флага -n для slaptest. Добавлены и изменены пояснения. Добавлены примеры.
20. Приложение А - Поисковые фильтры соответствия компонентов - откорректирован синтаксис и исправлены ошибки в примерах.
21. Приложение D - Добавлено пропущенное определение диапазона имён.
22. Приложение Е - Добавлены наборы схемы dyngroup.schema (для динамических групп) и misc.schema.
23. Приложение Е - config.ldif (cn=config) - добавлены ссылки для улучшения просмотра.
24. Приложение Е - Исправление иерархии объектных классов для inetOrgPerson.

Версия 0.1.13 - 10 ноября 2011 г.

1. Основное направление - Последовательное добавление материалов по ApacheDS и приведение материалов по OpenLDAP в соответствие с версией 2.4+.
2. Глава 2 - Добавление примечаний по DN, используемых для аутентификации. Исправление опечаток.
3. Глава 2 - В разделе 2.4.1 изменено 'одна или более дочерних записей' на 'ноль или более дочерних записей'.
4. Глава 3 - Исправление опечаток.
5. Глава 5 - В разделе 5.1.4 добавлены примечания по DN, используемых для аутентификации.
6. Глава 11 - Ошибка в HOWTO по настройке групп - некорректно определённое `dc=groups` исправлено на `ou=groups`.
7. Глава 14 - В начале раздела по `ldapdelete` добавлен недостающий 'dn:' в примере с `ldapmodify`.
8. Приложение А - Новая статья по DN, используемым для аутентификации (DN подключения или DN принципала).
9. Приложение А - Добавлены перекрёстные ссылки на статью о DN подключения.
10. Приложение D - Добавлено определение DN принципала.

Версия 0.1.12 - 4 августа 2010 г.

1. Основное направление - Последовательное добавление материалов по ApacheDS и приведение материалов по OpenLDAP в соответствие с версией 2.4+.
2. Страница по реализациям - Незначительные изменения формулировок.
3. Страница содержания - Исправлены несколько нерабочих ссылок в главе 11.
4. Глава 2 - Исправлены две ссылки на Relatively Distinguished Name.
5. Глава 3 - Объяснение параметра `USAGE` в определении атрибута.
6. Глава 4 - Обновление материалов по процедурам установки ApacheDS.
7. Глава 5 - В разделе 5.3.4 обновлены ACL по результатам повторного тестирования в OpenLDAP 2.4.16+.
8. Глава 6 - исправлена диаграмма, используемая в примере по директиве `access to` и озаглавленная "Общая и персональные адресные книги". В описание ACL в этом разделе добавлено примечание по использованию модификатора `expand` в версии 2.4. Текст последующих примечаний был обновлён для отражения замечаний по ACL.
9. Глава 6 - Удалён дублировавшийся текст.
10. Глава 6 - Ревизия текста по параметрам `group` и `peername` директивы `access to`, отражающая произошедшие изменения.
11. Глава 9 - Ethereal заменён на Wireshark.
12. Глава 12 - Обновлено сообщения об ошибках.
13. Глава 14 - Обновлено ссылка на LDAP Browser/Editor.
14. Глава 14 - Незначительное обновление текста примера `slappasswd`.
15. Приложение А - Добавлен дополнительный текст в описании DN/RDN.
16. Приложение А - Незначительные изменения в статье по ASN.1.
17. Приложение А - Добавлен дополнительный текст по определению именования корневой записи.
18. Приложение А - Новые статьи по X.500, RFC 2247 и простому именванию корневой записи.
19. Приложение В - Обновлено ссылка на LDAP Browser/Editor.
20. Приложение С - Обновлён список RFC.
21. Приложение D - Исправлена незначительная ошибка в описании `entryCSN`, добавлены дополнительные детали в описании `contextCSN`, `CSN` и `entryCSN`. Исправление ссылок. Исправление правописания.
22. Приложение D - Добавлены дополнительные детали по контексту именования.

Версия 0.1.11 - 12 июля 2009 г.

1. Основное направление - Последовательное добавление материалов по ApacheDS и приведение материалов по OpenLDAP в соответствие с версией 2.4+.
2. Глава 1 - Незначительные изменения текста.
3. Глава 2 - Незначительные изменения текста.
4. Глава 3 - В раздел 3.1 добавлены определения родительской, дочерней и одноуровневой записи. Добавлены незначительные разъяснения и ссылки по всему тексту.
5. Глава 5 - Незначительные изменения текста.
6. Глава 6 - Добавлены примечания по URL-форме параметра директивы `ServerID` и незначительные разъяснения.
7. Глава 6 - Добавлено описание директивы `password-hash`.
8. Глава 6 - Добавлено описание наложения политики паролей `ppolicy`.
9. Глава 6 - Добавлено описание расширенной версии `ppolicy`.

10. Глава 14 - Обновлён статус LDAP Browser/Editor.
11. Приложение А - Типы данных - добавлены дополнительные примечания по типу данных Integer.
12. Приложение А - Исправлена ссылка на поисковый фильтр соответствия компонентов.
13. Приложение А - OID - Обновлена ссылка со старого сайта oid.elibel.tm.fr на новый сайт www.oid-info.com.
14. Приложение А - Добавлена статья по наследованию объектных классов.
15. Приложение В - Обновлена ссылка со старого сайта oid.elibel.tm.fr на новый сайт www.oid-info.com.
16. Приложение В - Обновлена ссылка на LDAP Browser/Editor. В список LDAP-браузеров добавлен Apache LDAP Studio.

Версия 0.1.10 - 26 октября 2008 г.

1. Основное направление - Последовательное добавление материалов по ApacheDS и приведение материалов по OpenLDAP в соответствие с версией 2.4+.
2. Индексная страница/Содержание - Обновлено для отражения новых разделов.
3. Глава 1 - Исправлены опечатки.
4. Глава 2 - Исправлен файл LDIF в разделе 2.4.4, где было некорректно показано, что в ou=people используется inetOrgPerson.
5. Глава 2 - Ревизия текста раздела LDAP vs RDBMS. Небольшие исправления и обновления ссылок.
6. Глава 3 - Обновлён текст (в обзоре добавлено определение записи) и обновлены/добавлены диаграммы. Небольшие исправления и обновления ссылок.
7. Глава 3 - Описание rootDSE обновлено для соответствия версии 2.4.x.
8. Глава 5 - В разделе 5.3.1 исправлена диаграмма.
9. Глава 6 - Директива access - тип по умолчанию для dn.dnstyle изменён на base (exact), до версии 2.2+ был regex.
10. Глава 6 - Директива access - ревизия документации по элементу управления break.
11. Глава 6 - BDB - обновлены ссылки на сайт oracle вместо старого сайта sleepycat.com.
12. Глава 6 - Исправления в описании перехода от slapd.conf к cn=config.
13. Глава 6 - Включение описания директивы index (до этого была только ссылка на приложение А) и директивы mirrormode.
14. Глава 6 - Обновлена документация по наложениям syncprov и accesslog.
15. Глава 6 - Обновлена документация по директиве rootpw.
16. Глава 6 - Обновлена документация по директиве sync repl.
17. Глава 6 - Включение директив TLS в описание slapd.conf.
18. Глава 6 - Добавление некоторых директив (TLS_CACERT) в описание ldap.conf.
19. Глава 6 - Исправлено правописание!
20. Глава 7 - Исправлено правописание!
21. Глава 7 - Исправления и обновления материалов по конфигурации sync repl, delta-синхронизации и разнонаправленной репликации с несколькими главными серверами по результатам тестирования в версии 2.4.11.
22. Глава 8 - Исправления в файлах LDIF и тексте по использованию ldapadd/ldapmodify, отражающие тот факт, что эти утилиты больше не делают предположений, как им поступить.
23. Глава 11 - Добавлены дополнительные HOWTO (TLS и обеспечение безопасности).
24. Глава 11 - Примечания перенесены в приложение А.
25. Глава 12 - Добавлены дополнительные причины появления некоторых сообщений об ошибках.
26. Глава 12 - Добавлены дополнительные сообщения об ошибках.
27. Глава 14 - Переупорядочены аргументы ldapsearch.
28. Глава 14 - Переупорядочены аргументы ldapadd/ldapmodify, материал обновлён до версии 2.4+.
29. Глава 14 - Переупорядочены аргументы ldapdelete, добавлено описание формата файла (-f).
30. Глава 14 - Добавлена дополнительная информация по применению и настройке LDAPBrowser/Editor.
31. Глава 14 - Добавлены опции командной строки slapd.
32. Глава 15 - Обзор безопасности. Зарезервировано место под описание конфигурации SASL и TLS.
33. Глава 15 - Добавлена конфигурация TLS.
34. Приложение А - Новая статья по выживанию в ASN.1.
35. Приложение А - Примечания из главы 11 перенесены в приложение А.
36. Приложение А - Типы данных - незначительные исправления в определении DirectoryString.
37. Приложение А - Новая статья "Поисковые фильтры соответствия компонентов" - неполная, поскольку такие фильтры в настоящее время не работают в OpenLDAP (версия 2.4.8).
38. Приложение В - Обновлены сведения по ресурсам.
39. Приложение С - Добавлены RFC (RFC 3641 и 4792).

Версия 0.1.9 - 1 февраля 2008 г.

1. Основное направление - Последовательное добавление материалов по ApacheDS и приведение материалов по OpenLDAP в соответствие с версией 2.4+.
2. Индексная страница/Содержание - Обновлено.
3. Глава 2 - Добавление раздела по обобщениям возможностей LDAP. Незначительные исправления.
4. Глава 2 - Расширение и доработка раздела по отсылкам и репликации.
5. Глава 3 - Атрибуты - добавление текста.
6. Глава 6 - Глава переименована в "Конфигурационные файлы" (охватывает файлы slapd.conf и ldap.conf OpenLDAP, а также server.xml ApacheDS).
7. Глава 6 - Директивы OpenLDAP: loglevel - дополнительные детали, directory - незначительные исправления.
8. Глава 6 - Директива OpenLDAP access to <what> - изменение параметра на attrs (было attr) - похоже, в OpenLDAP собираются избавляться от attr, изменено описание использования модификатора expand с regex.
9. Глава 6 - Добавлены директивы OpenLDAP: overlay, replica, replogfile, syncrepl, readonly, updatedn, updateref, replicationinterval, referral, moduleload и modulepath.
10. Глава 6 - OpenLDAP - добавлена документация по наложениям accesslog, chain, syncprov (репликация). Кроме того, зарезервировано место для rcache (прокси), rpolicy (политика паролей), rwm (перезапись DN).
11. Глава 6 - OpenLDAP - переход к использованию cn=config.
12. Глава 7 - Глава переименована в "Репликация и отсылки".
13. Глава 7 - Добавлен раздел 7.2 - репликация (slurpd и syncrepl).
14. Глава 7 - Добавлен раздел 7.2 - разнонаправленная репликация syncrepl с несколькими главными серверами (N-Way Multi-Master).
15. Глава 7 - Добавлен раздел 7.2 - репликация syncrepl с использованием delta-синхронизации.
16. Глава 7 - Добавлен раздел 7.3 - отсылки и сцепление.
17. Глава 8 - Добавлены дополнительные детали по удалению атрибутов и записей с помощью LDIF.
18. Глава 8 - Зарезервировано место под описание DSML.
19. Глава 9 - LDAP URL - незначительные исправления и разъяснения. Обновлено в соответствии с требованиями RFC 4516. Домены в примерах изменены на example.com.
20. Глава 11 - Добавлены ссылки на репликацию slurpd, отсылки, сцепление отсылок, репликацию syncrepl, delta-синхронизацию.
21. Глава 14 - Инструменты LDAP - информация по ldapsearch обновлена до версии 2.4+.
22. Глава 14 - Инструменты LDAP - информация по slapadd обновлена до версии 2.4+.
23. Глава 14 - Инструменты LDAP - информация по slapcat обновлена до версии 2.4+.
24. Глава 14 - Инструменты LDAP - информация по slapindex обновлена до версии 2.4+.
25. Глава 14 - Инструменты LDAP - добавлена документация по ldappasswd и slaptest.
26. Приложение А - OID - дополнительные детали.
27. Приложение А - Поисковые фильтры - незначительные исправления. Обновлено в соответствии с требованиями RFC 4515 (включены возможности расширения правил соответствий).
28. Приложение А - DN-RDN - разъяснение, что DN/RDN формируются содержимым входящих в них атрибутов.
29. Приложение В - Добавлены материалы по Open Source LDAP-серверам.
30. Приложение С - Обновлён список RFC.
31. Приложение D - ASN.1 - улучшены ссылки. Дополнительные материалы.
32. Приложение Е - Добавлен файл cn=schema.ldif для cn=config.

Версия 0.1.8 - 21 декабря 2007 г.

1. Основное направление - Последовательное добавление материалов по Open Source ApacheDS.
2. Глава 4 - Добавление материалов по установке OpenLDAP и ApacheDS.
3. Глава 5 - Удаление пометки version: 1 и соответствующих ей комментариев в файлах LDIF для устранения проблемы с OpenLDAP 2.2+.
4. Глава 8 - Исправлена ошибка в директиве newrdn - должна включаться информация об объектном классе.
5. Приложение В - Добавлена информация по Apache Directory Server и другим инструментам, обновлена ссылка на LDAPBrowser/Editor.
6. Приложение С - Обновлён список RFC.

Версия 0.1.7 - 22 марта 2007 г.

1. Основное направление - Последовательное изменение домена с mydomain.com на example.com.
2. Глава 3 - Исправлена опечатка ABSTRACT (было ABTRACT)

3. Глава 5 - Незначительные исправления опечаток.
4. Глава 6 - База данных bdb - исправлен неверный прототип searchstack.
5. Приложение А - Головная боль с именованием корневой записи - разъяснения формулировок.
6. Приложение А - OID - OID OpenLDAP исправлен на 1.3.6.1.4.1.4203 (был неверно указан 4303), а также незначительное исправление опечаток.
7. Приложение D - Добавлено описание Organizational Unit.

Версия 0.1.6 - 19 января 2006 г.

1. Основное направление - Последовательное изменение домена с mydomain.com на example.com.
2. Глава 3 - Добавление пояснения, что objectClass также является атрибутом, по которому можно осуществлять поиск.
3. Глава 5 - Шаги 1, 2, 3, 4 - В slapd.conf добавлено примечание по директиве index objectClass.

Версия 0.1.5 - январь 2006 г.

1. Содержание.
2. Основное направление - Последовательное изменение домена с mydomain.com на example.com.
3. Глава 5 - Шаги 1, 2, 3 и 4 - Удаление директив dbnosync, dirtyread, searchstack из файлов slapd.conf.
4. Глава 5 - Шаги 1, 2, 3 и 4 - Изменение директивы index sn eq,sub (удалены излишние индексы subinitial, subany и subfinal) в файлах slapd.conf.
5. Глава 5 - Шаг 4 - Добавление комментариев к объектным классам и атрибутам.
6. Глава 5 - Шаг 4 - Изменено расположение скрипта запуска/остановки slapd.sh для [bsd].
7. Глава 5 - Шаг 5 - single-sign-on SSO - статья не закончена, просто зарезервировано место.
8. Глава 11 - Незначительные исправления в описании конфигурации с несколькими DIT.
9. Приложение А - Разъяснения по RDN-DN: когда DN должен быть уникальным, а когда - нет.

Версия 0.1.4 - 24 октября 2004 г.

1. Содержание.
2. Глава 1 and 2 - Многочисленные грамматические исправления и разъяснения - спасибо Seemant Kulleen.
3. Глава 6 - Незначительное исправление параметра rootDSE.
4. Приложение Е - В список часто используемых атрибутов добавлен userPassword.

Версия 0.1.3 - 5 сентября 2004 г.

1. Содержание.
2. Содержание - Исправлена нумерация главы 5.
3. Глава 5 - Исправлена нумерация разделов.
4. Приложение Е - Объектные классы и наборы схемы данных LDAP - добавлены перекрёстные ссылки в наборах схемы samba, courier и qmail.

Версия 0.1.2 - 10 августа 2004 г.

1. Содержание.
2. Глава 5 - Исправлена серьёзная ошибка в LDIF для шага 1, отклоняемого в OpenLDAP 2.2
3. Приложение В - Ресурсы LDAP - добавлены ссылки на некоторые web-сайты.
4. Приложение С - LDAP RFC - добавлены RFC 3829, 3866 и 3045.
5. Приложение Е - Объектные классы и наборы схемы данных LDAP - добавлены наборы схемы samba3 и courier (email).

Версия 0.1.1 - 24 июля 2004 г.

1. Содержание.
2. Глава 5 - Примеры - Создание и добавление объектных классов, атрибутов и наборов схемы данных.
3. Глава 11 - HOWTO - Конфигурирование групп пользователей.
4. Глава 11 - HOWTO - Конфигурирование нескольких DIT.
5. Приложение В - Ресурсы LDAP - исправлен URL для LDAP Browser/editor, добавлены некоторые комментарии по инструментам.

Версия 0.1.0 - 21 июня 2004 г.

1. Содержание.
2. Глава 1 - Соглашения и терминология

3. Глава 2 - Обзор LDAP
4. Глава 3 - Схема данных, объектные классы и атрибуты LDAP.
5. Глава 5 - Примеры - Простой каталог.
6. Глава 5 - Примеры - Повышение уровня безопасности.
7. Глава 5 - Примеры - Расширение иерархии.
8. Глава 6 - slapd.conf - access.
9. Глава 6 - slapd.conf - argsfile.
10. Глава 6 - slapd.conf - attributetype.
11. Глава 6 - slapd.conf - concurrency.
12. Глава 6 - slapd.conf - conn_max_pending.
13. Глава 6 - slapd.conf - conn_max_auth.
14. Глава 6 - slapd.conf - defaultsearchbase.
15. Глава 6 - slapd.conf - gentlehup.
16. Глава 6 - slapd.conf - idletimeout.
17. Глава 6 - slapd.conf - include.
18. Глава 6 - slapd.conf - loglevel.
19. Глава 6 - slapd.conf - objectclass.
20. Глава 6 - slapd.conf - pidfile.
21. Глава 6 - slapd.conf - referral.
22. Глава 6 - slapd.conf - schemadn.
23. Глава 6 - slapd.conf - sizelimit.
24. Глава 6 - slapd.conf - sockbuf_max_incoming.
25. Глава 6 - slapd.conf - sockbuf_max_incoming_auth.
26. Глава 6 - slapd.conf - threads.
27. Глава 6 - slapd.conf - timelimit.
28. Глава 6 - slapd.conf - database - bdb.
29. Глава 6 - slapd.conf - rootpw.
30. Глава 6 - slapd.conf - suffix.
31. Глава 6 - slapd.conf - rootdn.
32. Глава 8 - LDIF - формат.
33. Глава 9 - LDAP URL.
34. Приложение А - Заметки и пояснения - Головная боль с именованием корневой записи.
35. Приложение А - Заметки и пояснения - DN и RDN.
36. Приложение А - Заметки и пояснения - Несколько DIT.
37. Приложение А - Заметки и пояснения - Поисковые фильтры.
38. Приложение А - Заметки и пояснения - OID
39. Приложение А - Заметки и пояснения - Определение иерархии объектных классов в LDIF.
40. Приложение А - Заметки и пояснения - Типы данных LDAP.
41. Приложение В - Ресурсы LDAP.
42. Приложение С - RFC и стандарты X.500.
43. Приложение D - Глоссарий.
44. Приложение Е - Часто используемые объектные классы и атрибуты.

LDAP для учёных ракетчиков — список доработок

Пугающе длинный список того, что ещё не сделано.

Если Вы хотите внести свою лепту — выберите тему, которая Вам интересна, и отправьте нам письмо со своим текстом. Наградой Вам будет наше хорошее отношение и тёплое чувство того, что Вы помогли человечеству в его неуёмном стремлении к знаниям. Вот и всё. Приведённый ниже список не означает, что мы будем вносить изменения в указанном порядке.

1. Содержание — заголовки глав.
2. Глава 3 — Атрибуты — COLLECTIVE.
3. Глава 3 — Атрибуты — NO-USER-MODIFICATION.
4. Глава 3 — Атрибуты — USAGE.
5. Глава 5 — Примеры — технология единого входа.
6. Глава 5 — Примеры — отсылки и репликация.
7. Глава 6 — Завершение описания OLC (cn=config).
8. Глава 6 — Конфигурация — access — ldapfilter.
9. Глава 6 — Конфигурация — attributeoptions.

10. Глава 6 — Конфигурация — limits.
11. Глава 6 — Конфигурация — objectidentifier.
12. Глава 6 — Конфигурация — password-hash.
13. Глава 6 — Конфигурация — password-crypt-salt-format.
14. Глава 6 — Конфигурация — rootDSE examples.
15. Глава 6 — Конфигурация — sasl-authz-policy.
16. Глава 6 — Конфигурация — sasl-host.
17. Глава 6 — Конфигурация — sasl-realm.
18. Глава 6 — Конфигурация — sasl-regexp.
19. Глава 6 — Конфигурация — sasl-secprops.
20. Глава 6 — Конфигурация — srvtab.
21. Глава 6 — Конфигурация — ucddata-path.
22. Глава 6 — Конфигурация — database — ldbm.
23. Глава 6 — Конфигурация — database — sql.
24. Глава 6 — Конфигурация — database — ldap.
25. Глава 6 — Конфигурация — database — meta.
26. Глава 6 — Конфигурация — database — dnssrv.
27. Глава 6 — Конфигурация — database — null.
28. Глава 6 — Конфигурация — database — monitor.
29. Глава 6 — Конфигурация — database — passwd.
30. Глава 6 — Конфигурация — database — perl.
31. Глава 6 — Конфигурация — database — shell.
32. Глава 6 — Конфигурация — lastmod.
33. Глава 6 — Конфигурация — maxderefddepth.
34. Глава 6 — Конфигурация — subordinate.
35. Глава 6 — Конфигурация — allow — proxy_authz_anon.
36. Глава 6 — Конфигурация — disallow — tls_2_anon — tls_authc.
37. Глава 6 — ldap.conf — добавить большинство директив.
38. Глава 7 — Запуск репликации при использовании cn=config.
39. Глава 8 — LDIF — директива control.
40. Глава 8 — LDIF — base 64.
41. Глава 9 — Функциональная модель.
42. Глава 10 — LDAP API.
43. Глава 12 — Устранение неполадок — Стандартные ошибки — дополнительные пояснения.
44. Глава 12 — Устранение неполадок — записи журнала OpenLDAP.
45. Глава 13 — Производительность.
46. Глава 14 — Примеры для ldapsearch, ldapdelete, ldapsearch, ldapmodrdn, slapadd, slapcat, slapindex.
47. Глава 14 — Добавить описание slapdn, slapdauth, ldapwhoami.
48. Глава 15 — Безопасность — SASL.
49. Приложение A — Типы данных.
50. Приложение A — Индексирование.
51. Преобразование руководства в форматы DocBook/PDF.

Технологические материалы ZyTrax

В этом разделе собран перевод некоторых технологических материалов корпорации ZyTrax.

Криптография	Обзоры и учебные материалы по криптографии, аутентификации, авторизации, хэшам, отпечаткам, MAC, цифровым подписям и другим сногшибательным вещам.
SSL/TLS/X.509	Всё, от чего вы так долго прятали голову в песок: SSL/TLS и сертификаты X.509, в том числе генерация самоподписанных сертификатов средствами OpenSSL. Фильм ужасов. После прочтения этого руководства требуется как минимум два дня отдыха.
Kerberos	Kerberos V. Единственный на сегодняшний день протокол сетевой аутентификации (и авторизации). Его использование в Active Directory (AD). И в GSSAPI. Голова начинает болеть уже сейчас.



Руководство по выживанию — шифрование, аутентификация

Это руководство по выживанию охватывает такие ошеломительные темы как криптография, шифрование, авторизация и аутентификация. Для тех, кто сомневается в своей математической подготовке (и даже для тех, кто не сомневается) хочется отметить, что используемая в криптографии математика может ужаснуть кого угодно, и в этом руководстве она либо вообще не рассматривается, либо рассматривается поверхностно. Данное руководство сконцентрировано на функциональности системы и состоит исключительно из описаний и разъяснений, а не из конкретных команд или реализаций. Большинство из рассматриваемых здесь материалов служит основой для [SSL/TLS и сертификатов X.509](#).

Содержание:

- [Терминология](#)
- [Общий обзор криптографии](#)
- [Симметричное шифрование](#)
- [Асимметричное шифрование](#)
- [Отпечатки и хэши](#)
- [Аутентификационный код сообщения \(MAC\)](#)
- [Цифровые подписи](#)

Терминология

Неполный перечень терминов, используемых в защите информации. Многие из этих терминов разъясняются далее в тексте, другие остались сами по себе и приведены здесь отчасти для того, чтобы напугать читателя, но, главным образом потому, что мы понятия не имеем, что они означают.

Аутентификация (Authentication)	Процесс или процедуры, используемые для проверки того, что данные или информация, заявленные как поступившие из какого-то источника, могли поступить только из этого источника. Методы, используемые для аутентификации, включают цифровые подписи , MAC и даже добрые старые пароли. Как правило, после аутентификации Вам ещё нужно быть авторизованным для доступа к ресурсам или информации.
Авторизация (Authorization)	После того, как пользователи прошли аутентификацию, они, как правило, авторизованы (на основании их учётных данных для входа или свойств учетной записи) на получение или запрет доступа к определённым системным ресурсам, таким как файлы, приложения или сетевые ресурсы. Термин привилегия иногда используется как синоним авторизации. Так, пользователь может иметь достаточно привилегий для доступа к ресурсу X, но не к ресурсу Y, или, иными словами, они авторизованы на доступ к X, но не авторизованы на доступ к Y.
Шифр, шифрование (Cipher)	Шифр иначе называется алгоритмом шифрования. В английском языке под этим термином также иногда понимается процесс шифрования.
Открытый текст (Clear Text или Plain Text)	Блок данных, к которому не применялся процесс шифрования, либо блок данных, полученный в результате операции расшифрования или дешифрования.
Удостоверяющие данные (Credentials)	Причудливое название того, что большинство из нас назвало бы паролем (хотя они могут принимать и иные формы, такие как аппаратный токен или биометрические данные). Ваши удостоверяющие данные являются одним из способов доказать, что Вы именно тот, за кого себя выдаёте. Так как Вы должны быть единственным человеком (или, в некоторых случаях, группой лиц) кто знает или имеет доступ к Вашим удостоверяющим данным, то когда Вы предоставляете их в системе или в сети это доказывает, что Вы именно тот, за кого себя выдаёте. После выполнения неких форм обмена данными, которые будут включать в себя предоставление Ваших удостоверяющих данных (например, набор пароля) Вы становитесь аутентифицированы . Как правило, после аутентификации Вам ещё нужно быть авторизованным для доступа к ресурсам или информации.
Расшифрование (Decipher)	Применение алгоритма дешифрования к зашифрованному блоку или тексту с предоставлением соответствующих ключей, в результате чего он превращается в открытый текст. Многие (но не все) шифры используют один и тот же алгоритм для шифрования и дешифрования.
Шифрование (Encryption)	Процесс преобразования данных с использованием шифра (или алгоритма шифрования). Используемые методы могут быть симметричными или асимметричными .

Хэш (Hash) Он же — отпечаток (digest), или односторонний хэш (one-way hash). Алгоритм для уменьшения бесконечно большого блока данных до уникальной строки октетов меньшего и фиксированного размера, например, 64 или 256 байт. Хэши используются для обеспечения целостности сообщений, а также в [MAC](#) или [цифровых подписях](#).



Общий обзор криптографии

Согласно словарю Уэбстера криптография — это "шифрование и дешифрование сообщений с помощью секретного кода или шифра; также: компьютеризированное кодирование и декодирование информации".

Криптографией называют процесс преобразования (шифрования) данных, представленных "открытым текстом", с помощью некоторого метода (шифра или алгоритма шифрования) в абракадабру, которую можно трансформировать (расшифровать) обратно в открытый текст, только если получатель обладает неким секретным знанием, таким как ключ или набор ключей.

Исторически секретное знание представляли из себя сами шифры или алгоритмы шифрования. Например, такой шифр, как замена каждой буквы на предшествующую ей в алфавите, который порой используют дети или мы сами, когда посылаем секретные послания своим друзьям. Слабое место такого подхода в том, что если метод или алгоритм шифрования раскрыт, все коммуникации, зашифрованные с помощью этого алгоритма или шифра, могут быть преобразованы в обычный текст (расшифрованы). Нужно будет разработать новый алгоритм и распространить по всем заинтересованным сторонам.

Современная криптография предполагает, что злоумышленники рано или поздно раскроют криптографический алгоритм. По причинам, которые мы увидим позднее, в действительности сами алгоритмы находятся в свободном доступе. Секретное знание же теперь состоит в уникальном ключе или ключах, используемых этими алгоритмами для преобразования данных. Если этот ключ скомпрометирован (получен или украден злоумышленником), то путём простой замены ключа, оставляя алгоритм без изменений, можно возобновить конфиденциальные коммуникации. Злоумышленнику ничего не остаётся, как приниматься за раскрытие ключа заново, причём знаний у него будет не больше, чем перед началом предыдущего взлома, а мы, в свою очередь, постараемся тщательнее следить за своей ключевой информацией.

Криптографические алгоритмы не являются абсолютно закрытыми, наоборот, информация о них обычно находится в свободном доступе и они вновь и вновь подвергаются атакам определённого круга исследователей и специалистов (тестеров-взломщиков), которые жить без этого не могут. Только те алгоритмы, которые длительное время выдерживали повторяющиеся атаки, используются затем в промышленной эксплуатации. Поскольку исследования в сфере криптографических алгоритмов продолжаются, иногда это приводит к необходимости замены казалось бы зрелого и надёжного алгоритма, когда в нём обнаруживается уязвимость. В качестве недавнего примера здесь можно привести теоретические уязвимости, обнаруженные в алгоритме создания отпечатков MD5. Хотя нельзя исключать того, что для нахождения ключа злоумышленник может использовать атаки методом грубой силы (то есть простым перебором), криптографические системы используют концепцию, известную как *вычислительная невозможность*, которая, попросту, означает, что атака методом грубой силы будет потреблять неприемлемо большое количество ресурсов и времени. Вычислительная невозможность основывается на достигнутом в определённый момент уровне технологий и, следовательно, не является абсолютной величиной. Так, например, в некоторых алгоритмах время от времени, по мере увеличения вычислительных мощностей, увеличивается размер ключа.

Если злоумышленники получили секретный ключ или ключи (путём воровства, подбора, удачи или иных мелкопакостных средств), они конечно же не станут трубить об этом на весь мир, чтобы пользователь сразу же поменял ключи. Вместо этого они продолжат тихо и спокойно прослушивать якобы безопасные коммуникации. Это, конечно же, является серьёзной проблемой и обычно решается некоей комбинацией содержания ключей в среде "доказанного вмешательства (tamper-proof)" (которая уничтожит ключи при попытке компрометации) или "оповещения о вмешательстве (tamper-aware)" (она же HSM, которая предупредит пользователя, если ключ скомпрометирован), либо путём регулярной смены ключей, — так называемым обслуживанием ключей, — которая просто минимизирует потенциальные риски.

Криптография может использоваться для решения трёх задач:

1. **Конфиденциальность:** Только стороны, участвующие в коммуникации, могут понять сообщения или данные, посылаемые между этими сторонами.

2. **Аутентификация:** Данные могут поступать только из распознанного источника.

3. **Целостность:** Данные, полученные одной стороной, являются именно теми данными, которые были отправлены другой стороной; во время передачи они не подвергались манипуляциям или компрометации.

Решение одной или нескольких из перечисленных задач может обеспечиваться применением как единственного алгоритма, так и комбинацией алгоритмов или методов.

Сначала обратимся к основным методам. Современные криптографические методы могут быть симметричными или асимметричными.

Примечание: Один из лучших источников информации по криптографии — публикации Национального Института стандартов и технологий США (NIST). В частности, в SP800-57 (на текущий момент ревизия 3, 2012 год) обсуждается управление ключами и приводится превосходный и тщательный анализ как методов криптографии, так и угроз. Кроме того, там даны (в таблицах 2 и 4) практические рекомендации по размерам ключей для различных алгоритмов. Заинтересованным читателям мы настоятельно рекомендуем хорошенько изучить этот достойный, хотя и большой, документ, как основательное и весьма практичное посвящение в тайные знания криптографии. На этой странице мы ссылаемся также на ряд других документов NIST SP.



Симметричная криптография

Симметричные алгоритмы шифрования, называемые также системами с одним ключом (single-key), с общим секретом (shared-secret), или даже системами с закрытым ключом (private-key), используют единственный ключ (или набор ключей) для шифрования и дешифрования данных. Единственный ключ (общий секрет) должен быть безопасным образом распространён между сторонами, которые будут его использовать, перед началом самих безопасных коммуникаций. У систем с общим секретом существует как минимум два узких места. Во-первых, ключ должен быть распространён безопасным способом с помощью процесса, называемого управлением ключами, который сам по себе нетривиален. Во-вторых, ответственность за безопасность уже распространённого ключа лежит на всех сторонах коммуникации: "Самому себе я доверяю, но могу ли я доверять всем остальным сторонам, что они не допустили огласки ключа?". Если ключ, являющийся общим секретом, скомпрометирован на любой из сторон, то он скомпрометирован для всех использующих его сторон. Симметричные алгоритмы используют значительно меньше вычислительных ресурсов, нежели их асимметричные коллеги. Как правило, они являются единственным приемлемым способом шифрования объемных потоков данных.

Примеры распространённых симметричных алгоритмов шифрования: DES (Data Encryption Standard, он же Data Encryption Algorithm (DEA)), Triple DES (TDES, он же TDEA (Triple DEA)), AES (Advanced encryption Standard), IDEA (International Data encryption Algorithm), а также RC4 (Rivest Cipher 4 — по состоянию на 2013 год рассматривается как потенциально подверженный взлому, хотя факты успешных атак ещё не доказаны и не опубликованы), типичные размеры ключей 64, 128, 192 или 256 бит. На рисунке 1 демонстрируется рабочее применение общего секрета в классических конфиденциальных коммуникациях.

Примечание: Термин "общий секрет (shared secret)", описывающий единственный ключ (или набор ключей), используемый или разделяемый обеими сторонами коммуникации, не следует путать с "разделением секрета (secret sharing)", описывающим процесс, при котором общий или единственный секретный ключ разбивается на части и распределяется между несколькими лицами, чтобы сделать его более защищённым.

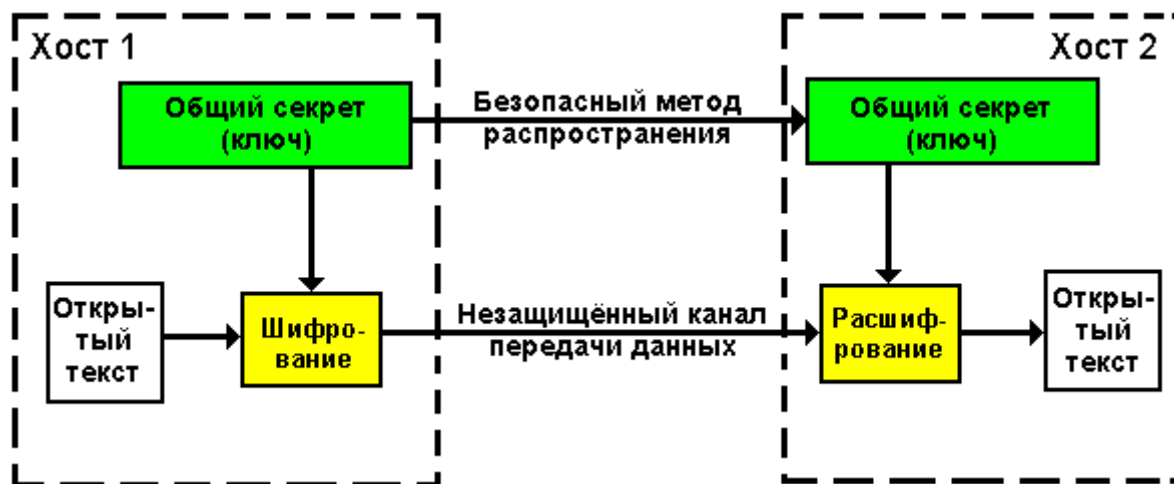


Рисунок 1 — Симметричная криптография

Асимметричная криптография

Асимметричные алгоритмы шифрования используют пару ключей, — открытый и закрытый, — и обычно называются криптографическими системами с открытым ключом или, иногда, несекретным шифрованием (своего рода оксюморон). В таких системах данные (на жаргоне — открытый текст), зашифрованные одним ключом, могут быть расшифрованы только парным ему ключом. Имея один ключ, вычислительно невозможно вывести парный ему ключ. Принцип работы асимметричной криптографии заключается в предоставлении одного ключа, называемого открытым, в широкий доступ, при сохранении другого ключа, называемого (сюрприз!) закрытым, в тайне. У этой технологии есть интересный побочный эффект. Если сообщение было зашифровано закрытым ключом и может быть расшифровано парным ему открытым ключом, то только владелец закрытого ключа мог произвести шифрование. Это свойство используется в [цифровых подписях](#) и описано ниже. По сравнению со своими симметричными коллегами, асимметричные алгоритмы потребляют значительное количество вычислительных ресурсов и потому обычно не применяются для шифрования объёмных потоков данных.

Наиболее широкое распространение среди криптосистем с открытым ключом получили RSA (по первым буквам фамилий разработчиков: Rivest, Shamir и Adelman) и Криптография на основе эллиптических кривых (Elliptic Curve Cryptography, ECC). Типичные размеры ключа в системах с открытым ключом RSA — 512, 1024, а в последнее время 2048 бит (текущие рекомендации US NIST на период с 2010 по 2030 год), а то и больше, если кому-то доставляет удовольствие чрезмерно перегружать процессор. Размеры ключа в ECC — 112, 224, 256, 384 или 512 бит. Открытый ключ из пары закрытого и открытого ключей может без нарушений безопасности храниться в общедоступных сервисах, таких как DNS, а закрытый ключ должен храниться в надёжном защищённом месте (возможно даже в аппаратных модулях безопасности (Hardware Security Modules, HSMs)). На рисунке 2 демонстрируется использование криптографии с открытым ключом для классических конфиденциальных коммуникаций.

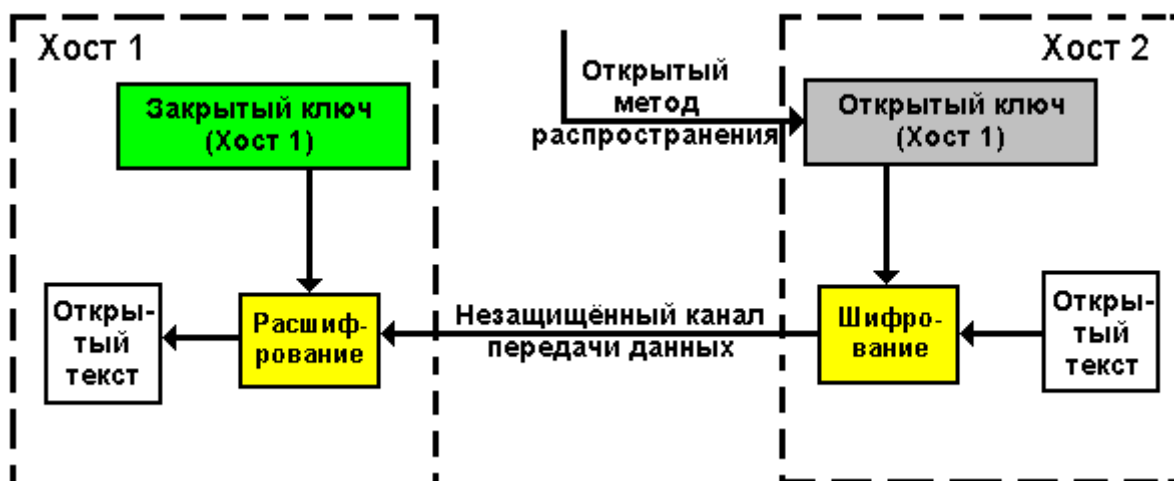


Рисунок 2 — Асимметричная криптография

Для достижения конфиденциальности сообщение, посылаемое от хоста 2 на хост 1 зашифровывается с использованием открытого ключа хоста 1. Это сообщение можно расшифровать только с помощью закрытого ключа хоста 1. Если хосту 1 требуется отправить конфиденциальное сообщение на хост 2, то он должен получить открытый ключ хоста 2 (не показано на рисунке 2, чтобы не загромождать).

У систем с открытым ключом есть одно существенное ограничение. Они полагаются на знание или доверие тому, что открытый ключ, который будет использоваться в коммуникациях с человеком и организацией, действительно является открытым ключом человека и организации и не был подменён злонамеренной третьей стороной. Обычно это достигается двумя основными методами: инфраструктурой открытых ключей (Public Key Infrastructure, PKI), или, в более общем случае, использованием доверенной третьей стороны. Третья сторона безопасно управляет открытыми ключами и подтверждает их подлинность. Если некто запрашивает у третьей стороны (в контексте сертификатов X.509 называемой удостоверяющим центром (Certificate Authority, CA)) открытый ключ объекта X, он доверяет, что ему был предоставлен корректный открытый ключ. Третья сторона, на основании проведенных проверок подлинности (аттестаций, нотариальных заверений и других), достоверно заявляет, что X — это единственный глобально-уникальный объект X. Наиболее распространённый способ информирования о том, что открытые ключи были удостоверены третьей стороной — встраивание их в [сертификаты X.509](#) (или [SSL](#)), подписанные цифровой подписью издателя сертификата (обычно удостоверяющего центра).



Отпечатки сообщений (хэши)

Для обеспечения целостности данных сообщение можно просто зашифровать. Чтобы изменить содержащиеся в сообщении данные, злоумышленник должен обладать единственным ключом (в симметричных системах) или закрытым ключом (в асимметричных системах). Однако криптосистемы используют сложные математические функции и потому потребляют много ресурсов процессора. Шифрование всех сообщений подряд приводит к неприемлемо высокому потреблению ресурсов, и это особенно неприятно, когда конфиденциальность данных не требуется. К счастью, для уменьшения нагрузки можно использовать другие методы. Самый распространённый — нересурсоёмкая процедура, называемая односторонним хэшем, просто хэшем, или чаще **отпечатком сообщения (message digest)** (в обиходе просто **digest**). Алгоритм хэшей или отпечатков создаёт уникальный и относительно небольшого фиксированного размера (независимо от исходной длины сообщения) блок данных, вернуть который в исходное состояние невозможно. Полученные хэшированные или отпечатанные блоки иногда называются **отпечатками пальцев**, поскольку они уникально описывают исходный открытый текст. Отправляемое сообщение обычно включает сразу и открытый (незашифрованный) текст и отпечаток этого сообщения. Алгоритм хэширования применяется к полученному открытому тексту, и, если результат совпадает с отпечатком сообщения, это означает, что полученные данные не были изменены. В некотором смысле отпечатки сообщений аналогичны по концепции с контрольными суммами, но значительно отличаются от них математическими свойствами.

Самые распространённые формы отпечатков сообщений — MD5 (Message Digest 5) и SHA-1, а в последнее время SHA-224, SHA-256, SHA-384 и SHA-512 (последние четыре — представители семейства алгоритмов SHA-2 (Secure Hash Algorithm 2)). Рисунок 3 демонстрирует использование отпечатков сообщений.

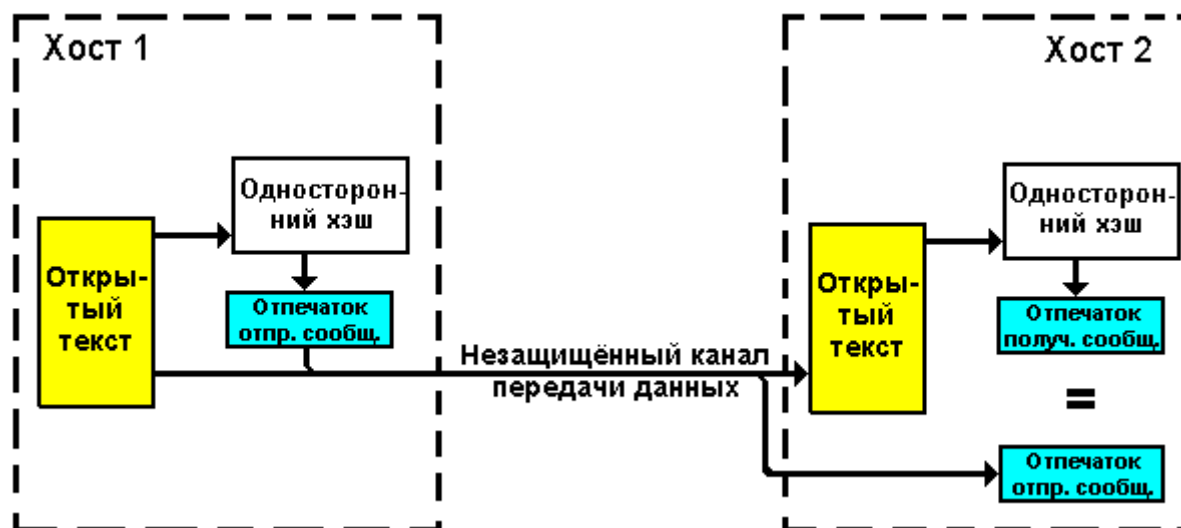


Рисунок 3 — Отпечатки сообщений

У простого добавления отпечатка сообщения к блоку открытого текста есть очевидное слабое место: путём замены сразу и открытого текста, и его отпечатка злоумышленник может подменить исходное сообщение или даже послать ложное сообщение от имени легального отправителя.

Примечание: Таблица 1 документа NIST SP800-107 содержит оценку относительной "силы" каждого алгоритма создания хэшей или отпечатков сообщений.



Аутентификационный код сообщения (Message Authentication Code, MAC)

Для аутентификации и поддержания целостности данных может использоваться так называемый аутентификационный код сообщения (MAC). MAC сочетает в себе отпечаток сообщения и общий (секретный) ключ. Часть, являющаяся ключом, аутентифицирует отправителя, а часть, представляющая собой хэш или отпечаток, обеспечивает целостность данных. Существуют две формы MAC. MAC, основанный на симметричном блочном шифре (таком как TDEA или AES), называется CMAC. MAC, основанный на алгоритме хеширования (создания отпечатка сообщения), называется Hashed MAC (HMAC) и, вероятно, используется наиболее широко.

Самые распространённые формы HMAC — HMAC-MD5, HMAC-SHA-1, а в последнее время HMAC-SHA-224, HMAC-SHA-256 и HMAC-SHA-384. На рисунке 4 показано, как используется MAC. Применяемый в HMAC **закрытый ключ** может быть сгенерирован различными методами (определёнными в SP 800-107, ревизия 1). На упрощённом уровне этот ключ HMAC может быть рассмотрен как простая "**соль**", назначение которой — аутентифицировать источник. **Примечание:** Алгоритм хеширования MD5 и, косвенно, любые использующие его алгоритмы, такие как HMAC-MD5, получили в большинстве документов IETF статус "нерекомендуемые" в связи с некоторыми теоретическими уязвимостями, опубликованными в начале 2005 года. Тем не менее, наличие этих недостатков не говорит о том, что использовать данные алгоритмы нельзя.

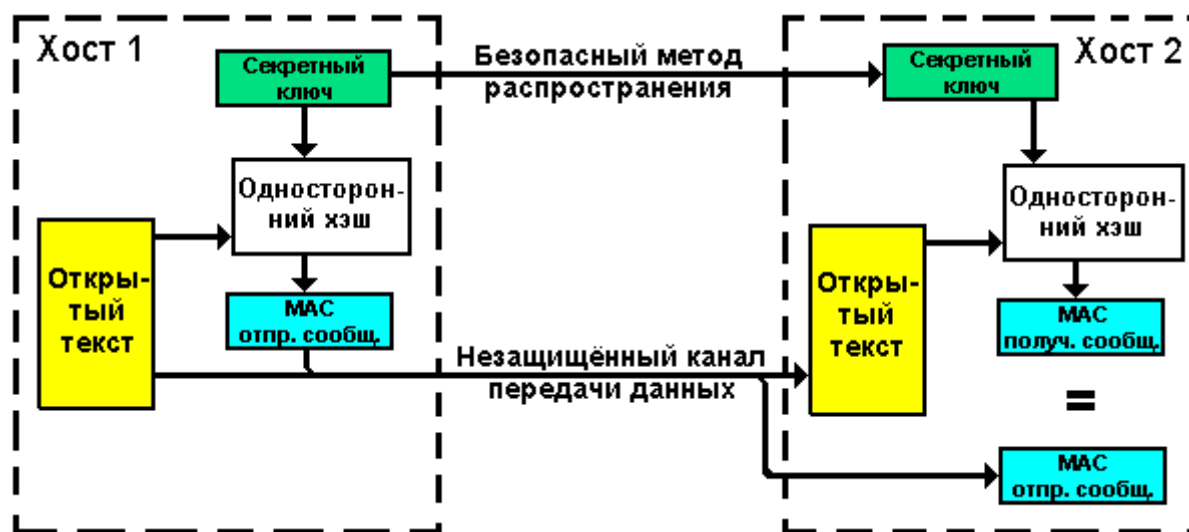


Рисунок 4 — Аутентификационный код сообщения (MAC)

На рисунке 4 показано шифрование только отпечатка или хэша. В результате мы получаем гарантию целостности сообщения. Если требуется также обеспечение конфиденциальности сообщения, то можно зашифровать его целиком (сразу и открытый текст, и хэш, или отпечаток). Протокол [TLS \(SSL\)](#) обеспечивает эту (последнюю) возможность в фазе записи данных.

<криптографические заморочки> Показанный на рисунке 4 ключ назван **секретным ключом**, но это также и **закрытый ключ**. Он назван **секретным ключом**, поскольку для того, чтобы оставаться в тайне, он должен постоянно быть закрытым. Аналогично, и даже более корректно, он может быть назван **закрытым ключом** (хотя термин **закрытый ключ** имеет подтекст асимметричной криптографии), поскольку он должен постоянно быть закрытым для того, чтобы оставаться в тайне. Иногда в жизни так трудно разобраться... **< / криптографические заморочки >**



Цифровые подписи

В мире асимметричной криптографии, или криптографии с открытым ключом, для аутентификации и поддержания целостности данных используются так называемые цифровые подписи. Отправляемое сообщение, опять же, хэшируется с помощью, скажем, MD5, SHA-1, SHA-256 или SHA-384, чтобы можно было убедиться в целостности данных. Полученный в результате отпечаток сообщения затем зашифровывается с помощью закрытого ключа отправителя. Сообщение в виде открытого текста и зашифрованный отпечаток отправляются вместе другой стороне. Получатель расшифровывает отпечаток сообщения с помощью открытого ключа отправителя, применяет алгоритм хэширования к полученному открытому тексту, и, если результаты совпали, то подтверждается сразу и подлинность отправителя, и целостность данных.

Наиболее распространённые алгоритмы цифровых подписей — RSA-MD5, RSA-SHA-1, RSA-SHA-256, RSA-SHA-384, DSA (Digital Signature Architecture, стандарт правительства США, определён в FIPS-186, ревизия 4) и ECDSA (Elliptic Curve Digital Signature Algorithm, определён в FIPS-186, ревизия 4). Типичные размеры ключа для систем цифровых подписей RSA и DSA — 768, 1024, а в последнее время 2048 бит (текущие рекомендации US NIST на период с 2010 по 2030 год), а то и больше. Размеры ключа цифровых подписей ECDSA для аналогичного уровня обеспечения безопасности значительно меньше и обычно составляют 112, 128, 192 или 256 бит. На рисунке 5 показано, как используется цифровая подпись. **Примечание:** Алгоритм хеширования MD5 и, косвенно, любые использующие его алгоритмы, такие как RSA-MD5, получили в большинстве документов IETF статус "нерекомендуемые" в связи с некоторыми теоретическими уязвимостями, опубликованными в начале 2005 года. Тем не менее, наличие этих недостатков не говорит о том, что использовать данные алгоритмы нельзя.

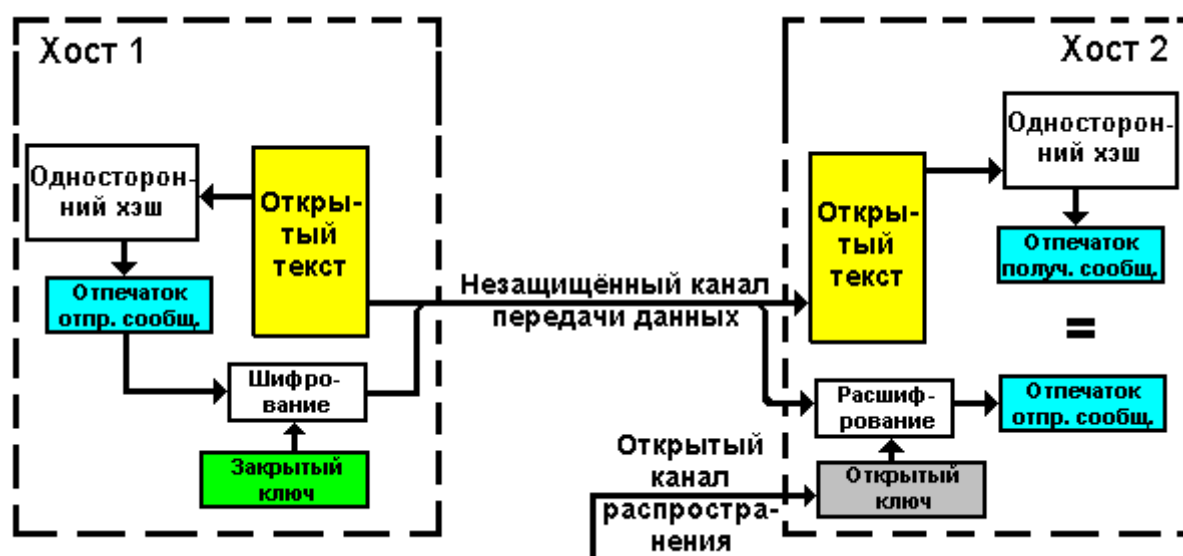


Рисунок 5 — Цифровая подпись

Примечание: Как показано на рисунке, с помощью находящегося в свободном доступе открытого ключа любой может расшифровать сообщение и сделать с него контрольный отпечаток, а зашифровать его может только обладатель закрытого ключа — таким образом обеспечивается гарантия подлинности источника. Лежащий в основе цифровой подписи отпечаток сообщения обеспечивает его целостность. Такое применение отличается от обеспечения конфиденциальности сообщения, при котором отправитель шифрует данные с помощью открытого ключа получателя, а расшифровать сообщение можно только с помощью закрытого ключа получателя.



Руководство по выживанию — TLS/SSL и сертификаты SSL (X.509)

Это руководство по выживанию рассматривает такую вводящую в ступор тему, как сертификаты TLS/SSL и X.509 (SSL), в том числе самоподписанные сертификаты. В нём присутствуют элементы того, что часто называют инфраструктурой открытых ключей (Public Key Infrastructure, PKI).

То, что мы в просторечии называем сертификатами SSL, в действительности должно называться сертификатами X.509. Термин "сертификат SSL" получил распространение в связи с адаптацией Netscape формата сертификата X.509 (одного из [стандартов служб каталогов X.500 ITU](#)) при разработке этой

компанией оригинальных версий протокола SSL в те доисторические времена, когда мир был ещё молод, а Интернет был дружелюбным местом. Термин "сертификат SSL" сохраняется только ради предоставления некоего права выбора: можно сказать "сертификат SSL" или "сертификат X.509", что Вам больше по вкусу.

В настоящий момент руководство включает SSL, TLS, некоторые детали о X.509 и их применении, а также кое-какие разъяснения о типах сертификатов, в том числе о сертификатах EV, и о процессе доверия.

Создание [самоподписанных сертификатов](#) представлено в виде рабочего примера использования пакета [OpenSSL](#).

Вы можете либо купить сертификат SSL (X.509), либо сгенерировать свой собственный (самоподписанный) для тестирования или, в зависимости от приложения, даже применения в производственной среде. **Хорошие новости:** При использовании самоподписанных сертификатов Вы сэкономите несметное количество денег. **Плохие новости:** При использовании самоподписанных сертификатов никто кроме Вас и (возможно) Ваших ближайших родственников не сможет им доверять. Но, прежде чем выкладывать все свои сбережения за новенький блестящий сертификат X.509 (SSL) или даже за более дорогой сертификат EV SSL (X.509), Вам, вероятно, хотелось бы знать, что они делают и как они это делают. И если Вы впадаете в ступор как только люди начинают говорить об SSL, защите информации и сертификатах — самое время туда впасть. Это руководство не будет лёгким чтивом.

<укоренившиеся привычки> В приведённом ниже списке RFC гиперссылки указывают на текстовые версии, которые мы скопировали на наш сайт, когда эти RFC были выпущены. Мы начинали делать это очень давно, когда мир был ещё молод и RFC содержались в каких-то странных местах, иногда меняющих своё местоположение, а производительность и доступность этих репозиторий оставляла желать лучшего (мягко сказано). Сегодня всё это далеко не так. У IETF и IANA есть солидные веб-сайты с прекрасной производительностью и постоянно совершенствующимися возможностями. Тем не менее, мы упорно продолжаем следовать нашей укоренившейся привычке без всякой видимой причины (трюки новые, а пёс-то старый...).

Если Вы хотите/предпочитаете/нуждаетесь в более продвинутом представлении RFC, то следующая информация специально для Вас. Основной репозиторий RFC поддерживается [IETF](#), текстовые версии можно найти по адресу www.ietf.org/rfc/rfcXXXX.txt (где XXXX — состоящий из 4 цифр номер RFC, при необходимости дополненный слева нулями). RFC в текстовом формате также можно посмотреть по адресу <http://datatracker.ietf.org/doc/rfcXXXX/> (где XXXX — состоящий из 4 цифр номер RFC, при необходимости дополненный слева нулями) здесь же содержится различная статусная информация по RFC, а также ссылки на просмотр в альтернативных форматах, таких как PDF и HTML. Наконец, существует [страница поиска по RFC](#).

Если же Вам всего лишь нужно почитать эти несчастные RFC, просто смело жмите на наши ссылки. **< /укоренившиеся привычки>**

Примечание переводчика: в русском варианте страницы ссылки указывают на HTML-версии RFC на сайте <http://tools.ietf.org/html/>.

Время от времени мы обновляем эту страницу и ведём [журнал изменений](#).

Содержание:

[Обзор протокола TLS/SSL](#)

[Шифрование — обзор](#)

[Обзор сертификата X.509](#)

[Типы и терминология сертификатов X.509](#)

[Цепочки сертификатов X.509](#)

[Протоколы, связанные с сертификатами \(CMP, CMC/CMS, SCVP, OCSP, HTTP\)](#)

[Протокол онлайн-получения статуса сертификата \(OCSP\)](#)

[Формат сертификата X.509](#)

[Списки отзыва сертификатов X.509 \(CRL\)](#)

[Процесс доверия — удостоверяющие центры и сертификаты X.509](#)

[Сертификаты X.509 расширенной валидации \(EV\)](#)

[Пример — создание самоподписанных сертификатов \(OpenSSL, несколько методов\)](#)

[Пример — создание подчинённых УЦ, промежуточных сертификатов и кросс-сертификатов \(OpenSSL\)](#)

[Пример — создание сертификатов с несколькими именами хоста \(OpenSSL\)](#)

[Замечания по форматам сертификатов](#)

[RFC по теме](#)

Протокол TLS/SSL

Основное предназначение сертификатов SSL (X.509) — использование совместно с протоколом TLS/SSL. Secure Sockets Layer (SSL, уровень защищённых сокетов) — протокол, изначально разработанный Netscape в 1992 году для безопасного обмена информацией между web-сервером и браузером по незащищённым сетям. Претерпел несколько изменений, нынешняя [версия 3](#) датируется 1995 годом и используется в различных клиент-серверных приложениях. В связи с распадом Netscape спецификации SSL в дальнейшем обновляться не будут. IETF стандартизировала Transport Layer Security (TLS, безопасность транспортного уровня) версии 1, незначительно отличающийся от SSL, в [RFC 2246](#), версии 1.1 в [RFC 4346](#), наконец версии 1.2 в [RFC 5246](#). Кроме того, в [RFC 3546](#) определено несколько расширений для случаев, когда TLS используются в системах с ограниченной пропускной способностью, таких как беспроводные сети, в [RFC 6066](#) определён ряд расширений TLS, внесённых в формат расширенного приветствия клиента (представленного в TLS1.2), в [RFC 6961](#) определён метод уменьшения трафика, когда клиент запрашивает у сервера информацию о состоянии сертификата.

При первоначальной установке защищённого соединения, в зависимости от реализации, будут проводиться переговоры о поддержке конкретного протокола из набора SSLv3, TLSv1, TLSv1.1 или TLSv1.2. Все так привыкли к названию SSL, что в большинстве случаев то, что называют SSL, скорее всего является использованием TLS — например, OpenSSL поддерживает и SSL (версии 3) и TLS (версий 1, 1.1 и 1.2). Поскольку SSL и TLS отличаются друг от друга лишь в деталях, дальнейшее описание относится к обоим протоколам. **Примечание:** SSLv2 был запрещён [RFC 6176](#), в котором приводится ужасающий список всех его недостатков.

TLS/SSL работают поверх TCP, но ниже по уровню тех конечных пользовательских протоколов, которые они защищают, таких как HTTP или IMAP (как показано на рисунке 1):

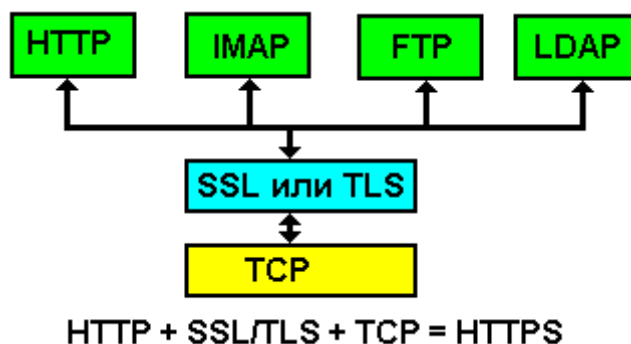


Рисунок 1 — Уровень TLS/SSL

За самими протоколами TLS/SSL не закреплено какого-либо общеизвестного номера порта — вместо этого при использовании с протоколом более высокого уровня, например HTTP, принято обозначать, что этот протокол будет использовать защищённый вариант (HTTPS в случае HTTP), у которого как раз есть общеизвестный номер порта (или порт по умолчанию). Обозначение HTTPS просто указывает на то, что нормальный протокол HTTP будет работать поверх соединения TLS/SSL, которые в свою очередь работают поверх TCP. В случае HTTPS общеизвестный номер порта — 443, в случае IMAPS — 993, POP3S — 995 и так далее.

Дальнейшее описание требует некоторого знакомства с терминами MAC (Message Authentication Code, аутентификационный код сообщения), односторонние хэши, симметричные и асимметричные криптографические алгоритмы. Для тех, кто с ними не знаком, рекомендуется почитать [это руководство по выживанию, касающееся криптографии](#). Осторожно, прочтение этого материала может вызывать переутомление и фобии.

Примечания:

1. Родственный протокол, Datagram Transport Layer Security (DTLS, безопасность транспортного уровня для дейтаграмм), определяет сервис безопасности при использовании UDP ([RFC 6347](#)). Поскольку его принципы аналогичны TLS, в дальнейшем в данном руководстве DTLS не обсуждается.
2. Термин TLS 1.2 Suite B (определённый в [RFC 6460](#)) определяет набор шифров (смотрите ниже), совместимых с NSA Suite B Cryptography, и имеет значение только при использовании TLS для приложений в интересах национальной безопасности США.

Обзор — установление защищенного соединения

При установлении защищенного соединения с помощью TLS/SSL, например при использовании HTTPS (порт по умолчанию — 443), между клиентом, который всегда является инициатором соединения, и сервером происходит обмен сообщениями. Первый набор сообщений называется протоколом рукопожатия (Handshake Protocol), после обмена этими сообщениями и клиент и сервер входят в протокол записи (или данных) (Record (Data) Protocol). При обмене сообщениями во время протокола рукопожатия достигаются следующие цели:

1. Устанавливается, какой вариант протокола из поддерживаемого (в зависимости от реализации) набора, — SSLv3, TLSv1, TLSv1.1, TLSv1.2, — будет использоваться. Всегда выбирается самый последний из возможных вариантов, то есть у TLSv1 всегда будет приоритет перед SSLv3 в случае, если и клиент и сервер поддерживают оба варианта. Клиент предлагает список вариантов, а сервер выбирает из предложенного списка.
2. Отправляются аутентификационные данные. Обычно сервер посылает аутентификационную информацию в форме сертификата (встроенную в сертификат) X.509 (SSL), но протокол поддерживает и другие методы.
3. Устанавливается идентификатор (ID) сессии, таким образом, при необходимости сессия может быть перезапущена.
4. Происходят переговоры о наборе шифров, состоящем из алгоритма обмена ключами вместе с типом алгоритма шифрования объемных данных и типом **MAC**, которые будут использоваться в последующей сессии обмена данными (протокол записи). Обычно в качестве алгоритма обмена ключами используется **асимметричный алгоритм (с закрытым и открытым ключом)**, такой как RSA, DSA или ECC (Elliptic Curve Cipher, шифр на основе эллиптических кривых, смотрите [RFC 5289](#)). Асимметричные алгоритмы потребляют очень много ресурсов процессора и потому для последующего шифрования объемных данных (протокол записи) используются **симметричные шифры**. Алгоритмы обмена ключами применяются для передачи информации, на основании которой могут быть независимо вычислены сеансовые ключи для симметричного шифра. MAC используется для защиты целостности передаваемых/получаемых данных во время протокола записи.

Конечно, это упрощенная схема, и во время установления соединения может совершаться обмен и другими данными, например, в процессе так называемой взаимной аутентификации у клиента для его аутентификации может быть запрошен сертификат X.509 (SSL), но описанный нами процесс — это *наиболее распространенный* случай, его мы и проиллюстрировали на рисунке 2:

Последовательность обмена сообщениями протоколов TLS/SSL

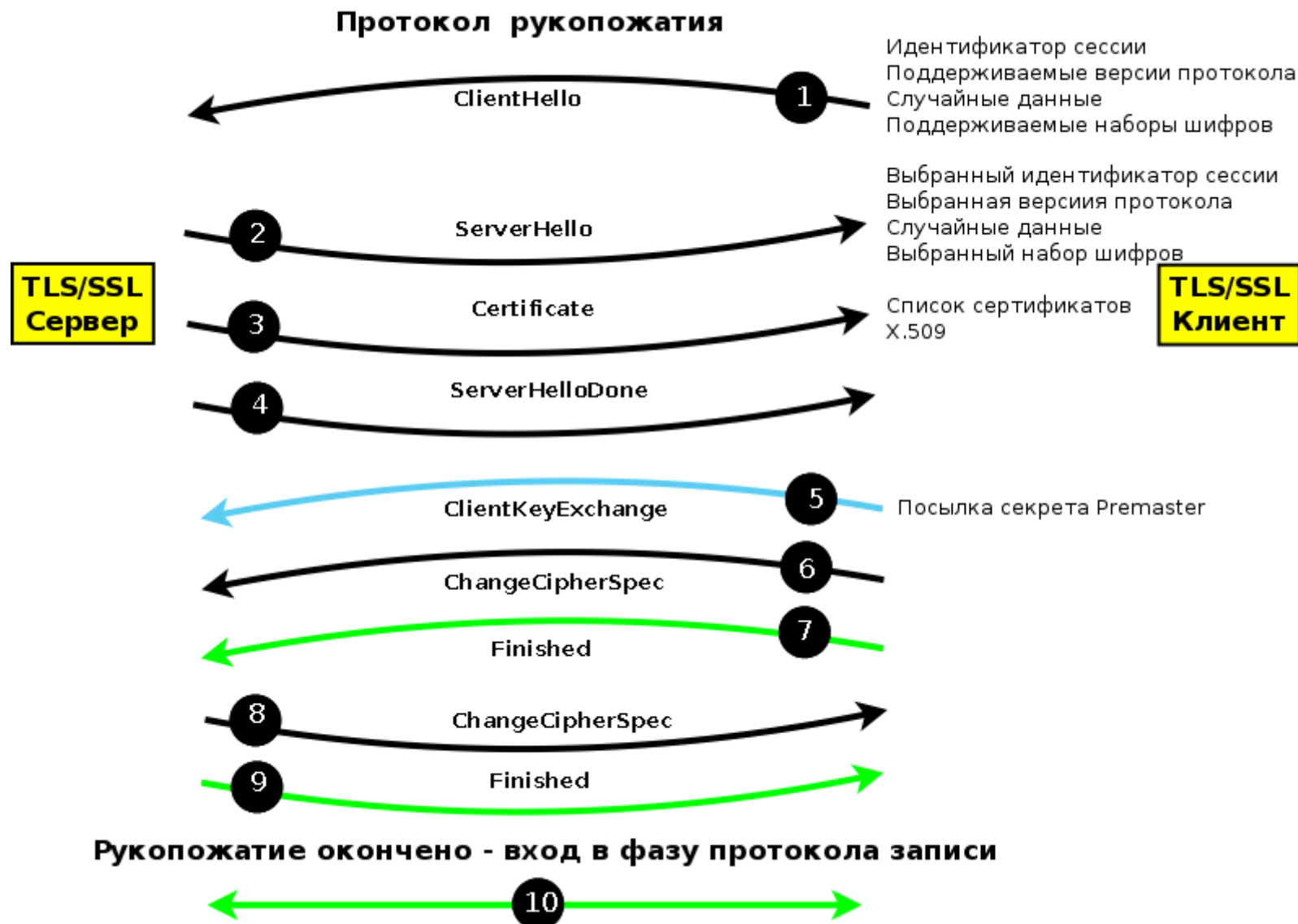


Рисунок 2 - Последовательность обмена сообщениями протоколов TLS/SSL

Примечания:

1. В фазе протокола рукопожатия проводятся переговоры и устанавливается соединение, а в фазе протокола записи происходит передача (инкапсуляция) зашифрованного потока данных, таких как HTTP, SMTP или IMAP.
2. На рисунке 2 чёрными стрелками показаны сообщения, отправляемые открытым текстом, синими — сообщения, отправляемые с использованием открытого ключа, предоставленного сервером (с помощью шифра обмена ключами), зелёными — сообщения, отправляемые с использованием того алгоритма шифрования объёмных данных и того MAC, о которых стороны договорились в процессе переговоров.
3. TLS/SSL позволяют договориться об использовании какого-либо алгоритма сжатия данных (как составной части набора шифров). С учётом скорости современных сетей сжатие данных используется всё реже или вообще не используется, и обычно значение выбранного алгоритма сжатия устанавливается в NULL (не используется).

TLS/SSL — детальное описание

В данном разделе приводится более подробное описание обмена сообщениями протоколов TLS/SSL (смотрите рисунок 2 выше) для тех, кто любит покопаться во внутренностях. Если Вам удобнее мириться с тем, что "в мире происходит много необычного", лучше [пропустите этот раздел](#), чтобы не рисковать рассудком.

1. **ClientHello:** Сообщение ClientHello предлагает список поддерживаемых версий/вариантов протоколов, поддерживаемые наборы шифров в порядке предпочтения и список алгоритмов сжатия (обычно NULL).

Клиент также посылает случайное значение размером в 32 байта (отметка текущего времени), которое позднее будет использоваться для вычисления симметричного ключа, и идентификатор сессии, который будет равен нулю, если не было предыдущих сессий, либо ненулевому значению, если клиент считает, что предыдущая сессия существует.

2. Каждый набор шифров состоит из алгоритма обмена ключами, алгоритма шифрования объёмных данных и алгоритма MAC (хэширования).
3. Набор шифров, используемый и клиентом и сервером, при первоначальном соединении таков:

```
TLS_NULL_WITH_NULL_NULL (0x00, 0x00)
# первый NULL — алгоритм обмена ключами
# следующий за ним WITH NULL определяет алгоритм шифрования объёмных данных
# последний NULL определяет MAC
```

Это значение говорит о том, что шифрование выполняться не будет и потому все сообщения до Client Key Exchange (ClientKeyMessage) будут отправляться открытым текстом.

4. Типичный набор шифров:

```
TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x00, 0x0A)
# RSA — алгоритм обмена ключами
# WITH 3DES EDE CBC определяет алгоритм шифрования объёмных данных
# (Triple DES с циклической цепочкой блоков)
# SHA — это MAC (хэш)
```

Примечания:

1. Корректные значения наборов шифров можно найти в приложениях C RFC по TLS ([RFC 2246](#) для TLS 1, [RFC 4346](#) для TLS 1.1 и [RFC 5246](#) для TLS 1.2). Значения для ECC (шифров на основе эллиптических кривых) приведены в [RFC 4492](#) и [RFC 7027](#).
2. Слово EXPORT, встречающееся в описаниях некоторых наборов шифров, говорит о том, что эти шифры являются экспортируемыми, то есть некоторые шифры разрешено использовать только в определённых странах (смотрите документацию [Бюро промышленности и безопасности \(BIS\) Министерства торговли США](#) и [Вассенарские соглашения](#)), и это необходимо учитывать при настройке системы, которая будет использоваться на международном уровне.
3. Расширения, определённые в [RFC 3546](#) и в основном используемые в беспроводных сетях, могут быть применены в сообщении ClientHello на манер обеспечения обратной совместимости. [RFC 6066](#) значительно увеличивает количество расширений TLS, многие из которых могут использоваться в обычных (не беспроводных) сетях. Сервер волен молча проигнорировать те расширения, которые он не понимает.
4. В [RFC 6066](#) представлен тип расширения TLS Certificate Status, в котором, по существу, сказано: "Я (клиент) абсолютно не доверяю твоему (сервера) сертификату, но я буду абсолютно доверять твоему (сервера) ответу на мой запрос (Certificate Status) о валидности сертификата(!)". Ответ на запрос Certificate Status (обычно получаемый с помощью [OCSP](#)) посылается в сообщении **CertificateStatus** сразу после сообщения **Certificate** (смотрите ниже). Видимо, запросы Certificate Status стали настолько популярны (среди беспроводных устройств?), что есть риск падения OCSP-серверов. В [RFC 6961](#) представлено расширение "certificate_request_v2", снижающее объёмы трафика от сервера TLS серверу OCSP путём разрешения первому из них хранить ответы OCSP, а также от сервера TLS клиенту TLS путём разрешения первому из них отправлять всю необходимую информацию, в том числе о промежуточных сертификатах, в одном сообщении **CertificateStatus**.
5. **ServerHello**: Сообщение ServerHello возвращает выбранные вариант/номер версии протокола, набор шифров и алгоритм сжатия. Сервер посылает случайное значение размером в 32 байта (отметка текущего времени), которое позднее будет использоваться для вычисления симметричных ключей. Если идентификатор сессии в сообщении ClientHello был равен нулю, сервер создаст и вернёт идентификатор сессии. Если в сообщении ClientHello был предложен идентификатор предыдущей сессии, известный данному серверу, то протокол рукопожатия будет проведён по упрощённой схеме. Если клиент предложил неизвестный серверу идентификатор сессии, сервер возвращает новый идентификатор сессии и протокол рукопожатия проводится по полной схеме.
6. **Certificate**: Сервер посылает свой [сертификат X.509](#), содержащий открытый ключ сервера, алгоритм

которого должен совпадать с алгоритмом обмена ключами в выбранном наборе шифров. Протокол предлагает и другие методы доставки открытого ключа, — он может, например, быть записью DNS KEY RR, — но сертификат X.509 является стандартным общепринятым методом. Цель данного сообщения — получение клиентом из доверенного источника открытого ключа сервера, который затем может использоваться этим клиентом для отправки зашифрованного сообщения.

Примечания:

1. Хотя в этом сообщении обычно посылается только один сертификат, можно послать и так называемую связку сертификатов (certificate bundle) — более одного сертификата в PEM-файле. Например, связки сертификатов могут быть определены с помощью директивы Apache **SSLCertificateChainFile**, тогда как единичный сертификат должен быть определён директивой **SSLCertificateFile**. Обычно связки используются при наличии [кросс-сертификатов](#), появляющихся в результате некоторой формы реструктуризации сертификата, например, при корпоративном поглощении одной компании другой или изменении ключа/размера ключа удостоверяющего центра (CA).
2. Протокол DNSSEC DANE ([RFC 6698](#)) позволяет клиентам получать копию сертификата X.509 сервера с помощью запроса к системе DNS. Однако, сертификаты, полученные по системе DNS с помощью DANE, являются дополнительными по отношению к тем, которые получены в процессе нормального обмена сертификатами TLS/SSL, и позволяют патологическим параноикам провести дополнительную верификацию.
7. **ServerDone**: Это сообщение указывает на окончание серверной части данной последовательности диалога и побуждает клиента продолжить последовательность протокола. **Примечание:** В этом месте сервер может запросить клиентский сертификат для выполнения взаимной аутентификации. Последовательность обмена клиентским сертификатом была опущена из описания последовательности протокола, поскольку обычно она не используется и её включение усложнило бы данное описание.
Примечание: Если во время переговоров об установлении TLS/SSL сервер запросил сертификат клиента, то клиент должен отправить свой сертификат (в том же формате, который определен для сервера, с тем исключением, что [RFC 6066](#) позволяет любому клиенту посылать URL сертификата вместо полного сертификата) непосредственно за сообщением **ServerDone** и до сообщения **ClientKeyExchange**.
8. **ClientKeyExchange**: Клиент вычисляет так называемый ключ pre-master key, используя случайные числа (или отметки текущего времени) сервера и клиента. Он шифрует этот ключ с помощью открытого ключа сервера, полученного из предоставленного сертификата X.509. Только сервер может расшифровать данное сообщение своим закрытым ключом. Обе стороны независимо друг от друга вычисляют общий секретный ключ master key из ключа pre-master, используя определённый в стандарте алгоритм. Любые сеансовые ключи, которые могут потребоваться, будут производными от этого ключа master key.
9. **ChangeCipherSpec — клиент**: Это сообщение указывает, что весь последующий трафик, исходящий от данного клиента, будет зашифрован с помощью выбранного (в результате переговоров) алгоритма шифрования объёмных данных и будет содержать MAC, сформированный по выбранному алгоритму. Номинально это сообщение всегда будет шифроваться текущим шифром, который в стадии установления соединения будет NULL, и, следовательно, данное сообщение отображено на диаграмме как отправляемое в открытом виде. Несмотря на то, что это сообщение показано на диаграмме протокола как отправляемое отдельно, часто оно объединяется с сообщением Client Key Exchange.
10. **Finished — клиент**: Это сообщение содержит все сообщения, отправленные и полученные во время протокола рукопожатия, за исключением сообщения Finished. Оно шифруется с помощью алгоритма шифрования объёмных данных и хэшируется с помощью алгоритма MAC, о которых договорились стороны. Если сервер может расшифровать и верифицировать это сообщение (содержащее все предыдущие сообщения), используя независимо вычисленный им сеансовый ключ, значит диалог был успешным. Если же нет, на этом месте сервер прерывает сессию и отправляет сообщение Alert с некоторой (возможно, неконкретной) информацией об ошибке.
11. **ChangeCipherSpec — сервер**: Это сообщение указывает, что весь последующий трафик, исходящий от данного сервера, будет зашифрован с помощью выбранного (в результате переговоров) алгоритма шифрования объёмных данных и будет содержать MAC, сформированный по выбранному алгоритму. Номинально это сообщение всегда будет шифроваться текущим шифром, который в стадии

установления соединения будет NULL, и, следовательно, данное сообщение отображено на диаграмме как отправляемое в открытом виде. Получение данного сообщения неявно говорит клиенту о том, что сервер получил и смог обработать сообщение Finished этого клиента.

12. **Finished — сервер:** Это сообщение содержит все сообщения, отправленные и полученные во время протокола рукопожатия, за исключением сообщения Finished. Оно шифруется с помощью алгоритма шифрования объемных данных и включает MAC, о которых договорились стороны. Если клиент может расшифровать это сообщение, используя независимо вычисленный им сеансовый ключ, значит диалог был успешным. Если же нет, клиент прерывает соединение и выдаёт сообщение Alert и подходящий (хотя и не всегда конкретный) код ошибки.
13. **Record Protocol (протокол записи):** Последующие сообщения между сервером и клиентом шифруются с помощью алгоритма шифрования объемных данных и включают MAC, о которых договорились стороны.

Примечания:

1. Случайные значения, посылаемые клиентом и сервером, и последующий секретный ключ pre-master включают в себя значение времени величиной два байта (для предотвращения атак повторения) и потому, как и во всех криптографических системах, и клиент и сервер должны использовать синхронизированный источник времени, такой как [NTP](#).
2. При прерывании соединения клиентом или сервером с выдачей сообщения Alert прилагаемый код ошибки может быть неточным (и, зачастую, бесполезным), чтобы не предоставлять другой стороне дополнительной информации, которую можно было бы использовать в последующих атаках.



Обзор сертификатов X.509 (SSL)

Оригинальный стандарт ITU-T X.509, в котором сертификаты получили своё несчастливое имя, — один из серии стандартов спецификации каталогов X.500. Использование сертификатов X.509 в Интернет стандартизировано IETF в [RFC 5280](#), определяющем формат сертификата X.509, а также в [RFC 4210](#), определяющем протокол управления сертификатами (Certificate Management Protocol, CMP), который используется для запросов на доступ и обработку сертификатов X.509 (есть также ряд дополнительных [протоколов](#), используемых для обработки и проверки сертификатов). Наконец, в [RFC 3739](#) определяется то, что называется **квалифицированным сертификатом (Qualified Certificate)**, относящимся к **Европейской директиве по электронным подписям (директива 1999/93/EC)**.

X.509 открывает совершенно новый и удивительный мир терминологии. В частности, для обозначения полей в сертификате в нём используется термин "отличительное имя" (Distinguished Name, DN). DN определены IETF в [серии RFC по LDAP](#), конкретнее, в RFC 4514. Также используются термины "абстрактная нотация синтаксиса 1" (Abstract Syntax Notation 1, ASN.1) и "идентификатор объекта" (Object Identifier, OID), описанные в серии стандартов X.680 ITU. Наконец, при кодировании используются "особые (или уникальные) правила кодирования" (Distinguished Encoding Rules, DER), описанные в X.690 ITU.

Также существует большое количество связанных с управлением сертификатами стандартов, помеченных как PKCS#X (где X — число), например, PKCS#10 определяет формат запроса на подпись сертификата (Certificate Signing Request, CSR). Они относятся к стандартам, определенным [RSA Laboratories](#). Некоторые из этих стандартов были воспроизведены практически без изменений в качестве RFC, например, упомянутый выше PKCS#10 был опубликован как [RFC 2986](#) (обновлён в [RFC 5967](#)). В дополнение к стандартам от IETF, RSA и ITU-T, X.509 был дополнительно стандартизирован в ряде стран, а также некоторыми промышленными организациями. Те, кто следит за этим вопросом, а также те, кто занимается реализацией, обращают внимание на то, что подобная множественность стандартов может привести к серьезным проблемам в реализации и интерпретации. Вот довольно подробный, большой и пугающий [набор заметок по реализации X.509 от Peter Gutmann](#).

Сертификат X.509 выполняет две различные функции:

1. Сертификат X.509 (в настоящее время X.509v3) служит контейнером для [открытого ключа](#), который может быть использован для проверки или валидации **конечного субъекта (end entity, EE)**, такого как веб-сайт или LDAP-сервер. Этот субъект определён в поле [subject](#) сертификата. Субъект в поле **subject** описывается в форме уникального или отличительного имени (Distinguished Name, DN), —

широко используется в LDAP, — состоящего из ряда относительных уникальных имён (Relative Distinguished Name, RDN), каждое из которых представляет собой содержащий данные элемент, называемый атрибутом (Attribute). В частности, атрибут CN (commonName), образующий RDN уникального имени, обычно содержит описание **конечного субъекта**, которому выдан сертификат. Примером CN может быть адрес веб-сайта, такой как CN=www.example.com. Полное DN в поле **subject** может содержать один или более из следующих RDN: CN= (commonName, общепринятое имя, наименование конечного субъекта, например, website или www.example.com), C= (страна), ST= (штат или область в составе страны), L= (местоположение, номинально определяет адрес, но используется в разных целях, за исключением сертификатов EV, где его предназначение строго определено), OU= (organizationalUnitName, название подразделения компании или какая-либо иная подструктура), O= (organizationName, обычно название компании).

2. Сертификат X.509 подписан цифровой подписью доверенной удостоверяющей организации (обычно называемой удостоверяющим центром (Certificate Authority) или просто CA), уникальное имя (DN) которой указано в поле **issuer** сертификата. Это делается, во-первых, для того, чтобы убедиться, что данный сертификат не был подделан, а во-вторых, для того, чтобы подтвердить (удостоверить), что данный открытый ключ для субъекта, указанного в поле **subject** на самом деле является открытым ключом для данного **субъекта**. Этот процесс **доверия** мы опишем **далее**. Подписывающая организация может быть удостоверяющим центром (Certification Authority, CA), регистрационным центром (Registration Authority, RA) или каким-то другим промежуточным центром (таким как подчинённый удостоверяющий центр (subordinate CA)), кроме того сертификат может быть **самоподписанным**.

Примечание: Закрытый ключ, ассоциированный с открытым ключём в сертификате X.509 пользователя, всегда хранится у пользователя и никогда не разглашается подписывающей организации.

Отдельная структура X.509, называемая **списком аннулированных (или отозванных) сертификатов (Certificate Revocation List, CRL, в настоящий момент CRLv2)** предоставляет информацию о сертификатах, которые по тем или иным причинам были аннулированы или признаны недействительными. По сути, CRL представляют собой старомодный "пакетный" способ работы с аннулированными сертификатами. В них содержатся большие (порой даже очень большие) списки всех сертификатов, которые были отозваны. Если проверяемого (по серийному номеру) сертификата нет в списке CRL, предполагается, что он ещё действителен. У списков CRL есть множество проблем: они могут обновляться только периодически и только УЦ (издателем сертификатов), из-за большого размера CRL браузеры скачивают их (если вообще это делают) неохотно и нерегулярно. Короче говоря, это не очень удобный и эффективный способ. Для проверки текущего статуса конкретного сертификата (опять же, определяемого по серийному номеру) всё чаще и чаще используется онлайн-способ (**OCSP**).

Большая часть информации в этом разделе сфокусирована на использовании X.509 для валидации коммуникаций сервера. X.509 может также использоваться и для других целей, таких как идентификация пользователей (включая цифровые подписи) и S/MIME, которые мы обязательно рассмотрим, когда (если) голова болеть перестанет.



Типы сертификатов X.509 и терминология

При обсуждении вопросов, связанных с сертификатами X.509 (SSL), используется несметное количество терминов. Иногда они применяются последовательно, но в основном — нет, что только добавляет неразберихи. Даже в RFC, посвящённым сертификатам, нет строго определения терминологии, пожалуй, лучше всего с этим обстоят дела в **RFC 4210**. Как правило, удостоверяющие центры (УЦ) предлагают несколько типов сертификатов. За исключением **сертификатов EV** и **квалифицированных сертификатов**, имеющих точное назначение и определение, все эти типы сертификатов, по большому счёту, — маркетинговая концепция, цель которой — предоставить выбор по цене/функциональности. Поэтому мы ограничимся лишь поверхностным обзором подобных типов сертификатов, в лучшем случае предоставляя немного технической терминологии. Наконец, не все УЦ одинаковы. Хотя большинство УЦ являются нарочито профессиональными организациями, которые регулярно проверяются или сертифицируются теми или иными национальными организациями, это не всегда так (на сайте УЦ ищите ссылки по аттестации, сертификации и аудиту и смотрите, что там). При покупке сертификата необходимо проявлять бдительность!

Далее предпринята попытка определить наиболее часто используемые термины по удостоверяющим центрам и типам сертификатов в свете того, что было сказано в предыдущем параграфе. Предлагаемые пояснения взяты из технической документации (в основном из RFC, особенно RFC 4210), а также с веб-сайтов

удостоверяющих центров.

Удостоверяющий центр (УЦ, он же корневой УЦ, англ. Certificate Authority (CA) a.k.a. root CA):

термин "удостоверяющий центр" определяется как субъект, подписывающий сертификаты, в которых выполнены следующие условия: поля **issuer** и **subject** совпадают, поле **KeyUsage** установлено в **keyCertSign** и/или в поле **basicConstraints** атрибут **ca** установлен в TRUE. Как правило, в цепочке сертификатов сертификат корневого УЦ является сертификатом самого верхнего уровня, однако RFC 4210 определяет, что корневым УЦ может быть любой издатель (**issuer**), сертификат которого получен конечным субъектом (например, браузером) в результате доверенного процесса получения. Поскольку окончательное решение о выдаче любого сертификата остаётся за этим УЦ, термины и условия любого промежуточного сертификата могут быть изменены данным субъектом.

Регистрационный центр (РЦ, он же регистрационный УЦ, англ. Registration Authority (RA) a.k.a.

Registration CA): Регистрационный центр (RA) может потребоваться в определенных условиях для обработки специфических характеристик сертификата, например, РЦ может быть делегирован Национальным удостоверяющим центром для специализации на персональных сертификатах, а другой РЦ может специализироваться на сертификатах серверов. РЦ, если таковые вообще имеются, являются, по сути, неким средством для административного удобства. РЦ могут подписывать сертификаты (как подчинённые УЦ), но могут также, выполнив соответствующие проверки конечного субъекта, передавать запрос на подписание корневому УЦ.

Подчинённый центр (подчинённый УЦ, англ. Subordinate Authority a.k.a. Subordinate CA): Общий термин. Любой субъект, подписывающий сертификаты, но не являющийся корневым УЦ. Некоторые подчинённые УЦ - особенно те, которые работают под полным контролем владельца корневого УЦ - могут быть помечены как УЦ (будет присутствовать расширение **BasicConstraints** и **ca** будет установлено в **True**). То есть, если РЦ подписывает сертификаты, то он делает это как подчинённый УЦ, и если он работает под контролем корневого УЦ, то он также может быть помечен, как УЦ.

Промежуточный центр (промежуточный УЦ, англ. Intermediate Authority a.k.a. Intermediate CA):

Неточный термин, иногда используется для определения субъекта, создающего промежуточные сертификаты, и, таким образом, охватывает РЦ и подчинённые УЦ.

Кросс-сертификаты (они же сертификаты сцепления или мостовые сертификаты, англ. Cross

certificates a.k.a. Chain or Bridge certificate): Кросс-сертификат представляет собой сертификат, в котором субъекты в полях **subject** и **issuer** не совпадают, но оба являются удостоверяющими центрами (присутствует расширение **BasicConstraints** и поле **ca** установлено в **True**). Как правило, такие сертификаты используются, когда УЦ изменил некоторые элементы своей политики издания сертификатов (новая дата истечения срока действия ключа или новый ключ), либо когда один УЦ был поглощён другим, и сертификаты, изданные поглощённым УЦ, привязываются к новому владельцу, чтобы можно было отказаться от использования ранее выданных корневых сертификатов. При использовании в данном контексте термин сертификат сцепления указывает на то, что была создана новая привязка, и иногда называется мостовым сертификатом (он привязывается к новому УЦ или выполняет функции моста к нему). Кросс-сертификаты могут быть установлены на сервере (как часть связки сертификатов - смотрите примечание в разделе [Протокол TLS, сообщение Certificate](#)), но при использовании для обеспечения обратной совместимости, например, при обработке EV-сертификата несовместимым с EV клиентом, кросс-сертификат устанавливается на клиенте. За исключением того, что поле **ca** установлено в **True** (что, откровенно говоря, мало что меняет) кросс-сертификат представляет собой обыкновенный промежуточный сертификат.

Промежуточные сертификаты (они же сертификаты сцепления, англ. Intermediate certificates a.k.a.

Chain certificates): Неточный термин, применяемый к любому сертификату, не подписанному корневым УЦ. Промежуточные сертификаты формируют [цепочку](#), в которой на пути от сертификата конечного субъекта до корневого сертификата может быть сколько угодно промежуточных сертификатов. Промежуточные сертификаты могут быть выпущены подчинёнными УЦ, РЦ или даже напрямую корневым УЦ (хотя технически их следует называть кросс-сертификатами) для различных целей: организации перехода, поглощения или даже просто для дифференциации брендов. В данном контексте термин **сцепление** бессмысленен (хотя звучит умно и помпезно), он просто указывает на то, что сертификат является частью какой-то цепочки.

Связка сертификатов (англ. Certificate Bundle): Общий термин, указывающий на то, что в один файл

(обычно в формате [PEM](#)) объединены несколько сертификатов X.509. Связки сертификатов могут передаваться во время протокола рукопожатия TLS/SSL. Обычно связки сертификатов используются для административных целей во время некоторых изменений состояния УЦ, например, поглощения, изменения политики, ключа, даты истечения срока действия ключа и т.п.

Квалифицированные сертификаты (англ. Qualified certificates): Определённый в [RFC 3739](#) термин "квалифицированные сертификаты" относится к персональным сертификатам (а не к сертификатам серверов) и ссылается на Директиву Европейского союза об электронной подписи (1999/93/EC), сфокусированную на единообразном определении индивидуума в целях электронной подписи, авторизации или аутентификации. В частности, данное RFC позволяет содержать в поле **subject** в порядке очередности атрибуты **commonName** (CN=), **givenName** (GN=) или **pseudonym**=, также может присутствовать поле **subjectDirectoryAttributes**, содержащее какие-либо из атрибутов **dateOfBirth**=, **placeOfBirth**=, **gender**=, **countryOfCitizenship**= и **countryOfResidence**= . Наконец, в этом RFC определены два новых расширения **biometricInfo** и **Qualified Certificate statements (qcStatements)**. Квалифицированный сертификат определяется по наличию расширения **qcStatements** со значением **qcStatement-2**. Большинство правительств конкретных стран добавили для включения в такие сертификаты ряд дополнительных полей. В некоторых случаях на то были реальные причины, в других — просто желание продемонстрировать свои амбиции и сделать невыносимой жизнь тех, кто занимается реализацией стандартов сертификатов.

Неквалифицированные сертификаты (англ. non-Qualified certificates): Вообще, так называют персональные сертификаты, не удовлетворяющие требованиям стандарта квалифицированных сертификатов. Издатели **квалифицированных сертификатов** также употребляют этот термин по отношению к другим сертификатам с намёком на то, что эти сертификаты более низкого качества.

Мультихостовые сертификаты (англ. Multi-host certificates): Сертификат сервера обычно в поле **subject** содержит атрибут **CN=hostname**, например, **CN=www.example.com**. Имя хоста разрешается системой DNS, которая может выдавать для этого хоста несколько IP-адресов (если в DNS имеется несколько [записей А или AAAA](#)). В этом случае серверный сертификат X.509 (SSL) с одним и тем же именем хоста может быть реплицирован на все подобные хосты (очевидно, соответствующий закрытый ключ также должен быть реплицирован на каждый хост, что может представлять собой проблему при использовании аппаратных криптографических устройств, в этом случае некоторые УЦ с удовольствием продадут Вам дополнительные сертификаты (которые будут называться мультихостовыми) чтобы обойти эту проблему). Если же присутствует несколько имён хостов, таких как **www.example.com**, **example.com** или **www1.example.com**, в таком случае требуется специальный подход и специальные типы сертификатов, описанные ниже как **мультидоменные сертификаты** и **сертификаты с возможностью подстановки**.

Мультидоменные сертификаты (англ. Multi-domain certificates): Некоторые УЦ продают мультидоменные сертификаты, охватывающие несколько доменных имён, например, таких как **www.example.com**, **example.com** или даже **www.example.net**. Это достигается путём использования нескольких записей в поле **subjectAltName** и описывается [далее](#). С технической точки зрения на этот процесс практически нет ограничений, например, в одном сертификате X.509 (SSL) могут поддерживаться **www.example.com** и **www.example.net**, но у большинства УЦ предусмотрены на это какие-либо коммерческие ограничения, которые, впрочем, обычно можно преодолеть путём дополнительных капиталовложений. Иногда для этих целей можно использовать описанные ниже **сертификаты с возможностью подстановки**, но имена хостов в таких сертификатах ограничены одним доменом.

Сертификаты с возможностью подстановки (англ. Wildcard certificates): Некоторые УЦ продают сертификаты с возможностью подстановки, в которых поле **subject** содержит **CN=*.example.com** (* — это шаблон подстановки). Такие сертификаты поддерживают любое имя хоста в составе домена, то есть ***.example.com** будет поддерживать **www.example.com** и **mail.example.com**, но не **example.com** и, очевидно, не **example.net** (для этих целей могут подойти описанные выше **мультидоменные сертификаты**).

Сертификаты EV (сертификаты расширенной валидации они же расширенные сертификаты, англ. EV (Extended Validation) Certificates a.k.a. Extended Certificates): Сертификаты EV отличаются наличием расширения **CertificatePolicies**, содержащего зарегистрированные OID в поле **policyIdentifier**. Подробнее сертификаты EV описаны [ниже](#).

Сертификаты DV (сертификаты валидации домена, англ. DV (Domain Validation) Certificates): Некоторые УЦ выпускают то, что называется сертификатами валидации домена (DV). Этот термин не используется повсеместно. Подразумевается, что удостоверяющий центр подтверждает только тот факт, что лицо или организация, запросившая данный сертификат, является владельцем доменного имени. То есть значение **CN=** в полях **subject** или **subjectAltName**, например, **www.example.com**, может рассматриваться как верное, а вот информация об организации (**C=**, **ST=**, **L=**, **OU=** или **O=**) не должна рассматриваться как верная, и эти значения должны быть либо пустыми, либо содержать соответствующий текст, например, "not valid" ("неверно").

Сертификаты OV (сертификаты валидации организации, англ. OV (Organizational Validation)

Certificates): Некоторые УЦ выпускают то, что называется сертификатами валидации организации (OV). Этот термин не используется повсеместно. Подразумевается, что удостоверяющий центр подтверждает тот факт, что лицо или организация, запросившая данный сертификат, является владельцем доменного имени, а также подтверждает предоставленные в сертификате сведения об организации. То есть значения CN=, C=, ST=, L=, OU= или O= в полях **subject** или **subjectAltName** могут рассматриваться как верные. Хотя на первый взгляд это выглядит весьма основательно, всё же такие сертификаты не дотягивают до уровня [сертификатов EV](#), требующих дополнительных условий квалификации.

Внутридоменные сертификаты (англ. Domain-Only Certificates): Общий термин, обычно несёт негативную окраску, применяется к сертификатам, в которых процесс верификации конечного пользователя, с точки зрения тех, кто применяет данный термин, варьируется от поверхностного до вообще никакого.



Цепочки сертификатов X.509

Сертификаты X.509 могут формировать **цепочки**, то есть они могут быть подписаны одним или более промежуточными удостоверяющими центрами в иерархической манере, либо сертификат может быть просто подписан корневым УЦ напрямую. Концепция регистрационных центров (РЦ), выступающих в роли промежуточных подписывающих удостоверяющих центров, представлена в упомянутых выше RFC. Понятие "регистрационный центр (РЦ)" (иногда в стандартах EV именуемый "подчинённым УЦ"), вводится для описания организации, у которой на самом деле был приобретён сертификат X.509, например, лицензированного агента, сертификат которого подписан (возможно, также в результате иерархической цепочки подписания) удостоверяющим центром, представляющим собой УЦ самого верхнего уровня или корневой УЦ. По структуре чем-то похоже на операторов реестра DNS и регистраторов, если организация DNS Вам знакома. В [RFC 4158](#) есть полезная, но крайне запутанная и неудобоваримая информация о том, как можно надёжно построить цепочку сертификатов с помощью пар полей **subject** и **issuer**, дополняя их, кроме всего прочего, полями **SubjectKeyIdentifier** и **AuthorityKeyIdentifier**.

Сертификат самого верхнего уровня в иерархии подписания называется корневым сертификатом, сертификатом УЦ или даже иногда корневым сертификатом УЦ. Корневые сертификаты получают каким-либо доверенным способом (в случае браузеров они распространяются с программным обеспечением браузера и периодически обновляются) и при валидации цепочки сертификатов обычно называются якорями доверия. При получении от сервера в процессе рукопожатия TLS/SSL сертификата (или связки сертификатов) конечного субъекта, программа, получившая сертификат, должна проверить его и весь путь до корневого сертификата (сертификата УЦ), включая, при необходимости, все промежуточные сертификаты (которые, опять же, обычно распространяются с программным обеспечением браузера). Корневой сертификат (сертификат УЦ) распознаётся по наличию одинаковых значений в полях **issuer** и **subject**, по тому, что поле **KeyUsage** установлено в **keyCertSign** и/или в поле **BasicConstraints** атрибут **CA** установлен в TRUE. Процесс построения цепочки сертификатов описан в [RFC 4158](#), валидация цепочки сертификатов описана в [RFC 5280](#). Различные варианты подписания сертификатов показаны на рисунке 3:

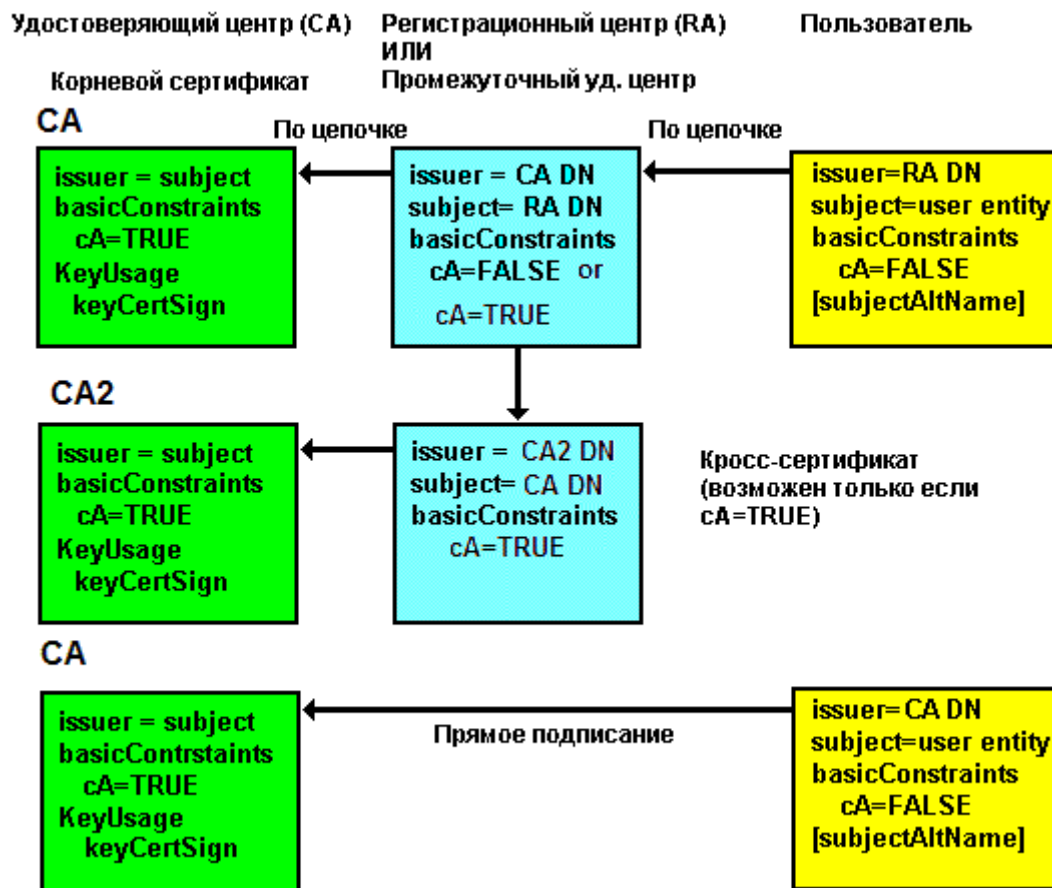


Рисунок 3 — Цепочки сертификатов X.509

Использование сертификатов X.509

Издатель сертификата (**issuer**) идентифицируется с использованием формата Distinguished Name (DN, уникального имени), который в оригинале разрабатывался для представления расположения объекта в DIT (информационном дереве каталога) DAP или LDAP. DN не следует путать с сетевым адресом или URL/URI. Обычно DN имеет формат **CN=Type of Certificate, OU=Certificate Division, O=Certificate Company name, C=Country** (**CN=, OU=, O=, C=**), но может иметь и более простой формат **OU=, O=, C=**, или даже **CN=, O=, C=**, наконец (чтобы немного запутать ситуацию) он может иметь формат **CN=, OU=, DC=, DC=**. В общем, формат может быть разным — вариантов много. DN состоит из нескольких разделённых запятыми RDN (Relative Distinguished Names, относительных уникальных имён), то есть CN= или C= представляют собой RDN в составе DN. Сертификат X.509 не содержит URI для получения по сети каких либо сертификатов для образования цепочки, но он может содержать (в других полях) URI для получения **CRL**. Используя сертификаты приложения, — к примеру, браузеры или почтовые клиенты, — должны заранее получить корневой сертификат, а если существует цепочка сертификатов, то и все промежуточные сертификаты, каким-либо способом, по сети или нет. Корневые сертификаты наиболее известных УЦ распространяются с браузерами (и доступны для соответствующих им почтовых клиентов). Также у браузеров и почтовых клиентов имеются пользовательские интерфейсы для добавления новых или замены существующих сертификатов. В Firefox: "Инструменты" (Tools)->"Настройки" (Options)->"Дополнительные" (Advanced)->Вкладка "Шифрование" (Encryption)->"Просмотр сертификатов" (View Certificates), в MSIE: "Сервис" (Tools)->"Свойства обозревателя" (Internet Options)->Вкладка "Содержание" (Content)->Кнопка "Сертификаты" (Certificates)->Вкладка "Доверенные корневые центры сертификации" (Trusted Root Certification Authorities).

Примечание: Распространяемые с программным обеспечением большинства браузеров (и почтовых клиентов) корневые сертификаты добавляются в соответствии с критериями, определяемыми поставщиком браузера, и варьирующимися от "кто больше заплатит" до тотальной проверки легитимности УЦ и соответствия их другим требованиям.

При обращении приложения к сервису с поддержкой TLS/SSL, в процессе **диалога рукопожатия TLS/SSL** оно получает сертификат (или связку сертификатов) сервера. После этого приложение (например, браузер)

выделит атрибут CN из DN в поле `subject` (и/или в поле `subjectAltName`, смотрите [RFC 6125](#)) для проверки субъекта (скажем, адреса веб-сервера, такого как `www.example.com`). Затем наше приложение будет использовать DN в поле `issuer` сертификата X.509 сервера для нахождения соответствующего корневого сертификата в своём локальном хранилище (и, если такового не обнаружено, генерации исключения — обычно в виде непонятного, пугающего пользователя диалога). Если найден валидный корневой сертификат, то тем самым подлинность предоставленного сервером сертификата считается установленной. В итоге (если всё прошло успешно) открытый ключ, содержащийся в сертификате X.509, считается безопасным (доверенным) и пригодным для проведения коммуникаций с указанным в поле `subject` субъектом.

Серверам обычно требуется только свой собственный сертификат (сертификаты) и не требуется корневых сертификатов — они просто посылают свои сертификаты клиенту, а валидации сертификатов не производят. Однако TLS/SSL позволяют проводить взаимную валидацию, в этом случае и сервер и клиент отправляют свои сертификаты. Если приложение требует, чтобы клиент предоставлял свой сертификат, то сервер должен будет произвести валидацию клиентского сертификата, для чего у него должны иметься все необходимые корневые и промежуточные сертификаты, полученные любым путём, например, по почте или с веб-сайтов удостоверяющих центров.

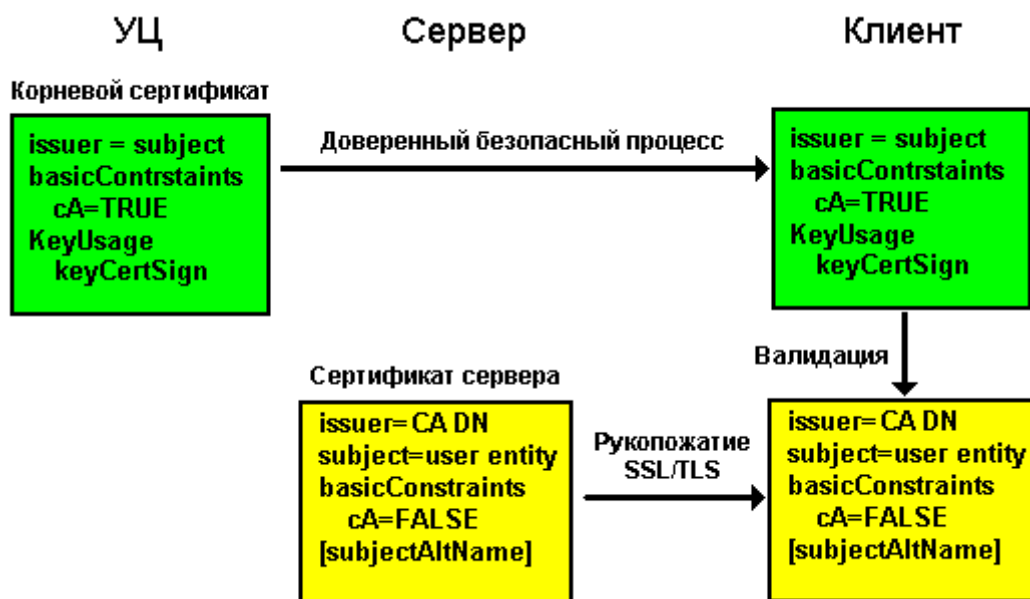


Рисунок 4 — Использование сертификатов X.509

Сертификат X.509 представляет собой структуру данных. Различные протоколы позволяют производить манипуляции с сертификатами через коммуникационную сеть. Такие **операции с сертификатами X.509**, как отправка запроса на подписание, получение подписанного сертификата и другие, определены, в частности, в протоколе управления сертификатами (Certificate Management Protocol, CMP, [RFC 4210](#)) и в PKCS #10 ([RFC 2986](#)), и описаны [далее в этом руководстве](#).



Протоколы, связанные с сертификатами (CMP, CMC/CMS, CRMF, SCVP, OCSP, HTTP)

Беглый обзор: Для манипуляций с сертификатами и списками отзыва сертификатов (Certificate Revocation Lists, CRL) определен ряд протоколов. Протокол управления сертификатами (Certificate Management Protocol, CMP, [RFC 4210](#), обновлён в [RFC 6712](#)) предоставляет методы для манипуляции сертификатами (формат сообщений сертификатов (Certificate Request Message Format, CRMF) определён в [RFC 4211](#)). В RFC 4210 определено несколько коммуникационных методов (таких как HTTPS), которые могут быть использованы для транспортировки запросов сертификата по сети, но не дано явного определения транспортного протокола для обработки сертификатов (смотрите CMC/CMS ниже).

Протокол CMC/CMS (Certificate Management over CMS, управление сертификатами поверх CMS), определён в [RFC 5272](#), [RFC 5273](#), [RFC 5274](#) и обновлён в [RFC 6402](#), позволяет осуществлять безопасную транспортировку запросов сертификата от запрашивающей стороны к УЦ (включая любые промежуточные РЦ) и обратно. Формат сообщения может быть либо PKCS #10 ([RFC 2896](#)), либо CMRF ([RFC 4211](#)). Этот протокол следует называть CMC, но иногда его называют CMS. Технически, CMS — это Cryptographic Message Syntax

(синтаксис криптографического сообщения), в котором описывается только формат конверта запроса и ответа (для PKCS #10 или CRMF). Данный протокол определяет ряд операций, которые могут быть выполнены над сертификатами, в том числе обновление якорей доверия (корневых сертификатов), поддерживаемых у запрашивающей сертификат стороны. Это большой и насыщенный (сложный) протокол.

Протокол онлайн-получения статуса сертификата (Online Certificate Status Protocol, OCSP, [RFC 6960](#)) — это протокол для онлайн-проверки подлинности сертификатов. [RFC 6066](#) расширяет стандарт TLS чтобы клиент мог запрашивать OCSP-статус сертификата во время фазы протокола рукопожатия, а [RFC 6961](#) определяет упрощённый формат 'certificate_request_v2', позволяющий снизить объёмы трафика сервера OCSP.

Протокол серверной валидации сертификатов (Server-Based Certificate Validation Protocol, SCVP, [RFC 5055](#)) позволяет делегировать недоверенному серверу некоторые клиентские функции, а именно построение и валидацию пути сертификации. Если же сервер SCVP является ещё и доверенным, то он может предоставлять дополнительную функциональность, например, проверку статуса сертификата (как OCSP) или получение промежуточных сертификатов (как CMP).

В [RFC 4387](#) определены некоторые ограниченные манипуляции с сертификатами X.509, ключами и списками CRL по протоколу HTTP с использованием метода GET (как раз этот метод мы использовали, когда последний раз покупали сертификат X.509). [RFC 4386](#) определяет использование DNS-записей [SRV](#) для обнаружения репозитория сертификатов и сервисов [OCSP](#).

Протокол управления сертификатами (Certificate Management Protocol, CMP)

Протокол управления сертификатами (CMP) определён в [RFC 4210](#) (обновлён в [RFC 6712](#)), формат сообщений запроса сертификатов (Certificate Request Message Format, CRMF) определён в [RFC 4211](#).

В дальнейшем описание будет расширено.

Протокол серверной валидации сертификатов (Server-Based Certificate Validation Protocol, SCVP)

В дальнейшем будет приведено описание. Варианты OCSP в значительной степени заменяют этот протокол.

Управление сертификатами поверх CMS (Certificate Management over CMS, CMC / CMS)

В дальнейшем будет приведено описание.

Протокол онлайн-получения статуса сертификата (Online Certificate Status Protocol, OCSP)

Протокол онлайн-получения статуса сертификата (OCSP) определён в [RFC 6960](#), а переработанный, обеспечивающий высокую производительность формат сообщений определён в [RFC 5019](#) (The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments (облегчённый профиль OCSP для крупномасштабных сред)). В этом формате существенно уменьшен размер OCSP-запросов для одного сертификата и исключено большинство необязательных (OPTIONAL) полей в запросах и ответах (смотрите примечание ниже). [RFC 6960](#) (отменяющий действие [RFC 2560](#) и [RFC 6277](#)) просто уточняет некоторые моменты в оригинальных RFC для достижения большей совместимости с [RFC 5019](#), а также добавляет свои собственные, совершенно новые туманные места в спецификацию. В [RFC 6961](#) определён новый формат 'certificate_request_v2', позволяющий серверам кэшировать (сохранять) ответы, а также позволяющий посылать в одном сообщении информацию о всех имеющих отношение к запросу сертификатах (включая промежуточные).

OCSP — это онлайн альтернатива [спискам отзыва сертификатов \(Certificate Revocation List, CRL\)](#). Если удостоверяющий центр поддерживает сервис OCSP, URI этого сервиса обычно указывается в поле [AuthorityInfoAccess](#) (AIA) сертификата. Клиент отправляет запрос, опционально подписанный электронной подписью, идентифицирующий сертификат, который требуется проверить, и получает от сервера OCSP подписанный ответ, указывающий статус сертификата как действующий (good), отозванный (revoked) или неизвестный (unknown). В RFC разрешено передавать запросы и ответы посредством ряда различных транспортных протоколов (в частности, в качестве примеров упомянуты LDAP, SMTP и HTTP) с указанием конкретной транспортной схемы в URI поля AIA сертификата, подвергающегося проверке. Также RFC

определяет формат сообщений HTTP (используются GET и POST) при транспортировке OCSP. Формат сообщений запроса и ответа:

```
# Формат запроса
OCSPRequest
  TBSRequest
    version          0 = v1
    requestorName    OPTIONAL
    requestList (один или несколько, в случае RFC 5019 - только один)
    reqCert
      hashAlgorithm   AlgorithmIdentifier
      issuerNameHash   Хэш DN издателя
      issuerKeyHash     Хэш открытого ключа издателя
      serialNumber     CertificateSerialNumber
      singleRequestExtensions OPTIONAL
    requestExtensions OPTIONAL
    optionalSignature OPTIONAL
```

Примечания:

1. Подпись запроса OCSP является необязательной, и, если она присутствует, в поле **requestorName** должен быть указан субъект, подписавший сообщение. Очевидно, для проверки подписи у получателя сообщения должен быть открытый ключ субъекта, подписавшего сообщение (или его делегированного агента).
2. Поставщики сертификатов EV (Extended Validation) обязаны предоставлять сервис OCSP, для всех остальных типов сертификатов этот сервис является опциональным.

```
# Формат ответа
OCSPResponse
  responseStatus
    successful        0 -- ответ содержит корректный статус
    malformedRequest  1 -- неверный формат запроса
    internalError      2 -- внутренняя ошибка сервера издателя
    tryLater           3 -- повторите попытку позже
                     -- (4) не используется
    sigRequired        5 -- запрос должен быть подписан
    unauthorized       6 -- запрос неавторизован
  responseBytes      OPTIONAL
  responseType      OID (1.3.6.1.5.5.7.48.1.1 =BasicOCSPResponse)
  BasicOCSPResponse
    response
      ResponseData
        version        0 - версия, по умолчанию v1
        responderID    ОДНО ИЗ ДВУХ:
          byName        1 - имя
          byKey         2 - хэш открытого ключа издателя
        producedAt     GeneralizedTime
      responses
        certID
          hashAlgorithm   AlgorithmIdentifier
          issuerNameHash   -- хэш DN издателя
          issuerKeyHash    -- хэш открытого ключа издателя
          serialNumber     CertificateSerialNumber
        certStatus      CertStatus
          good            0
          revoked         1
          unknown         2
        thisUpdate      GeneralizedTime
        nextUpdate      GeneralizedTime OPTIONAL
        singleExtensions OPTIONAL
        responseExtensions OPTIONAL
        signatureAlgorithm AlgorithmIdentifier,
        signature        Хэш данных ответа
        certs            OPTIONAL
```

Примечания:

1. Статус **good** в поле **certStatus** согласно RFC означает "не отозван", но в полях расширения (Extensions) может быть доступна дополнительная информация.
2. Статус **unauthorized** в поле **responseStatus** указывает на то, что сервер, выдающий ответ, не имеет достоверной информации по этому сертификату.
3. В [RFC 6960](#) определено несколько расширений для сообщений, кроме того, в них можно включать любые расширения [CRL \(RFC 2459\)](#).

4. Обычно ответ подписывается УЦ, издавшим сертификат, указанный в поле **serialNumber**, но протокол позволяет подписывать ответ делегированному удостоверяющему центру. В этом случае ответ должен включать сертификат (содержащий открытый ключ делегированного центра, подписавшего ответ) в поле **certs**, и этот сертификат должен быть подписан издателем сертификата, указанного в поле **serialNumber**.
5. [RFC 5019](#) определяет усовершенствованный формат сообщений (полностью совместимый с полным форматом OCSP) путём исключения большинства необязательных (OPTIONAL) полей с целью снижения нагрузки на сеть при передаче сообщений. Нововведения в протоколе: поддержка OCSP поверх HTTP только с использованием методов GET и POST. Нововведения в запросах: запрос ограничивается единственным сертификатом (значение поля **requestList** будет 1), обходится без необязательного поля **singleRequestExtensions**, необязательных структур **requestExtensions** и **optionalSignature** (если запрос подписан, отвечающий сервер вправе проигнорировать подпись). Нововведения в ответах: путём принудительного ограничения на один сертификат в запросе, RFC 5019 уже уменьшает сложность (и размер) ответа, кроме того, исключено необязательное поле **responseExtensions** (но поле **singleResponseExtensions** может быть включено). Опять же, если ответ подписан делегированным УЦ, он должен включать сертификат (содержащий открытый ключ делегированного центра, подписавшего ответ) в поле **certs**, и этот сертификат должен быть подписан издателем сертификата, указанного в поле **serialNumber**.

Узкие места OCSP

OCSP разрабатывался как система реального времени (в противоположность пакетной природе классических списков CRL), так что теоретически клиенты могут проверить статус любого сертификата до его принятия и использования. В частности, при обработке [сертификата EV](#) любая реализация клиента (например, браузер) в буквальном смысле обязана выполнить OCSP-проверку чтобы реализовать концепцию безопасности EV. Это означает, к примеру, что в результате каждого обращения к сервису HTTPS может (а в случае EV должна) происходить дополнительная проверка УЦ посредством обращения к сервису OCSP. Поскольку всё больше сайтов реализуют благонамеренную, но, по существу, ошибочную политику использования HTTPS для всего подряд (вместо того, чтобы настроить более значимый и надёжный сервис DNSSEC), производительность серверов OCSP резко снижается и они просто встают на колени от своего рода DDoS-атаки (хоть и из благих побуждений). [RFC 6066](#) расширяет стандарт TLS, чтобы клиент мог запрашивать статус сертификата у сервера TLS (**OCSPStatusRequest**), тем самым, возможно, ещё более усугубляя проблему загрузки OCSP, упростив реализацию опроса OCSP и поощряя каждого клиента его выполнять. В [RFC 6961](#) предпринята попытка решить проблему загрузки OCSP путём использования нового запроса TLS 'certificate_request_v2', который, предположительно, позволяет (хотя при опросе OCSP сервер не обязательно должен придерживаться такого поведения) кэшировать ответы OCSP на сервере (за счёт чего загрузка снижается) и посылать несколько сообщений о статусе сертификатов (включая все промежуточные сертификаты) в одном блоке.



Формат сертификата X.509

В этом разделе в упомянутых (но не всеобъемлющих) подробностях описывается формат и назначение основных полей (технически называемых атрибутами) сертификата X.509. Этот формат определён в [RFC 5280](#).

Сертификат X.509 представляет собой структуру [ASN.1](#), закодированную с помощью определённых в X.690 уникальных правил кодирования (Distinguished Encoding Rules, DER), и включающую в себя многочисленные ссылки на глобально уникальные [идентификаторы объектов \(Object Identifiers, OID\)](#).

Примечание: В X.509 (и LDAP) многие используют бестолковую псевдоВенгерскую нотацию (или lowerCamelCase, если Вы предпочитаете этот термин). В общем случае стоит помнить, что плохие реализации X.509 чувствительны к регистру символов, хорошие — нет. Более того, все правила соответствия LDAP, относящиеся к манипулированию DN, являются нечувствительными к регистру; это означает, что имена атрибутов нечувствительны к регистру.

Выводимый OpenSSL сертификат X.509 выглядит так (основные поля описаны ниже):

Certificate:
Data:

```
Version: 3 (0x2)
Serial Number:
bb:7c:54:9b:75:7b:28:9d
Signature Algorithm: sha1WithRSAEncryption
Issuer: C=MY, ST=STATE, O=CA COMPANY NAME, L=CITY, OU=X.509, CN=CA ROOT
Validity
Not Before: Apr 15 22:21:10 2008 GMT
Not After : Mar 10 22:21:10 2011 GMT
Subject: C=MY, ST=STATE, L=CITY, O=ONE INC, OU=IT, CN=www.example.com
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
RSA Public Key: (1024 bit)
Modulus (1024 bit):
00:ae:19:86:44:3c:dd:38:df:e2:41:5f:d8:86:19:
69:7e:85:d7:1d:ae:1e:eb:87:b0:5f:fc:f3:db:e3:
aa:82:76:d6:42:05:f1:0e:5c:5a:a2:8d:f6:d3:00:
37:04:96:13:06:16:e6:d1:67:14:69:cd:85:df:a7:
b3:ac:a2:6c:33:cd:d6:00:3d:24:99:fa:4b:81:07:
0c:b2:5a:fe:06:16:da:34:66:63:78:31:7d:11:5e:
63:de:9e:ee:76:8b:0c:12:af:fb:f2:28:0a:76:5b:
99:20:b8:f7:c0:9c:e8:89:c5:d0:1e:e5:07:c8:bd:
38:c8:52:97:cc:76:c9:c8:2b
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Basic Constraints:
CA:FALSE
Netscape Comment:
OpenSSL Generated Certificate
X509v3 Subject Key Identifier:
EE:D9:4A:74:03:AC:FB:2C:FD:43:C7:58:6E:2E:6A:88:BA:65:61:CC
X509v3 Authority Key Identifier:
keyid:54:0D:DE:E3:37:23:FF:2E:E8:03:0A:2C:52:FE:FC:C0:C8:13:72:80

Signature Algorithm: sha1WithRSAEncryption
52:3d:bc:bd:3f:50:92:67:a3:d3:6f:37:a9:3f:89:b5:16:5b:
9c:0d:32:25:32:91:c7:bf:f6:0d:f8:6d:1c:09:45:2f:3f:b9:
18:b7:1c:8d:7c:06:33:ef:ca:e0:92:a3:90:3f:7c:4e:16:87:
67:ae:7c:2c:1a:43:e5:3a:24:d9:c3:7d:cf:bf:eb:01:9d:c1:
f0:bb:0f:15:de:d5:9e:42:9d:f8:7f:0d:5b:af:59:80:d1:aa:
cc:db:31:1b:d4:7f:f3:f1:71:25:85:c9:8b:78:3e:13:ac:11:
51:35:49:8d:c3:9a:bb:9a:89:2c:ef:7f:90:f9:05:b3:65:98:
b8:74
```

Сертификат X.509 состоит из следующих полей (атрибутов):

Version	В общем случае это поле будет указывать на версию 3 (X.509v3). Поскольку нумерация начинается с нуля, значение этого поля будет 2. Если поле опущено, подразумевается версия 1 (значение 0).
Serial Number	Положительное число размером максимум до 20 октетов. Определяет уникальный серийный номер каждого сертификата, выпущенного конкретным удостоверяющим центром (то есть сам по себе, в глобальном смысле, не являющийся уникальным) и использующийся, кроме всего прочего, в списках отзыва сертификатов (CRL) .
Signature	Значением данного поля должен быть тот же OID , что указан в поле SignatureAlgorithm ниже.
Issuer	DN (Distinguished Name) удостоверяющего центра, подписавшего и, следовательно, издавшего данный сертификат. Это может быть корневой или некорневой удостоверяющий центр. Значение поля issuer может состоять из подмножества набора атрибутов domainComponent (DC=), countryName (C=), commonName (CN=), surname (SN=), givenName (GN=), pseudonym=, serialNumber=, title=, initials=, organizationName (O=), organizationalUnitName (OU=), stateOrProvinceName (ST=) и localityName (L=). Должны поддерживаться только атрибуты CN=, C=, ST=, O=, OU= и serialNumber=, остальные являются опциональными (serialNumber= встречается редко, но для сертификатов EV является обязательным, а L= встречается часто (хотя и трактуется неправильно), но в сертификатах EV используется как поясняющее значение). Теоретически, в RFC 5280 разрешены любые глобально уникальные методы построения DN (формат X.500 O=, C= или формат IETF DC=, DC=), но де-факто все придерживаются формата X.500. DN издателя может выглядеть примерно так:

```
# Разбиение на две строки показано исключительно для удобства
C=MY,ST=some state,L=some city,O=Expensive Certs inc,
OU=X.509 certs,CN=very expensive certs
# Существуют различные интерпретации полей RDN,
# далее представлены общепринятые значения.
# C = двухбуквенный код страны согласно ISO3166
# ST = штат или область (край)
# L = местоположение, обычно - город
# O = организация - название компании
```

```
# OU = подразделение организации, обычно - тип сертификата или бренд
# CN = общепринятое имя, обычно - наименование продукта или бренд
```

Validity

Представляет собой два подполя (атрибута) **notBefore** и **notAfter**, определяющих период времени, в который сертификат является действительным, или, если присутствует только подполе **NotAfter**, его значение определяет тот момент, когда пользователю снова придётся раскошелиться удостоверяющему центру (издателю). Даты до 2049 года включительно кодируются как UTCTime (формат YYMMDDHHMMSSZ, да-да, год обозначается двумя цифрами), а после 2050 года — как GeneralizedTime (формат YYYYMMDDHHMMSSZ, год обозначается четырьмя цифрами). Z в формате представления времени — символьная константа, указывающая на использование часового пояса Zulu Time (UTC, он же время по Гринвичу), при её отсутствии подразумевается локальное время.

Subject

DN, определяющий субъекта, ассоциированного с этим сертификатом. Если это корневой сертификат, в полях **issuer** и **subject** будут одинаковые DN (и в поле **BasicConstraints** CA будет установлено в TRUE). В сертификате конечного пользователя/сервера DN в поле **subject** тем или иным способом идентифицирует конечного субъекта. Как правило, атрибут CN (RDN) данного DN идентифицирует некоторое уникальное имя конечного субъекта, который будет сертифицирован или аутентифицирован. Значение поля **subject** может состоять из подмножества набора атрибутов domainComponent (DC=), countryName (C=), commonName (CN=), surname (SN=), givenName (GN=), pseudonym, serialNumber, title, organizationName (O=), organizationalUnitName (OU=), stateOrProvinceName (ST=) и localityName (L=). DN субъекта для аутентификации доступа к веб-сайту может выглядеть так:

```
# Разбиение на две строки показано исключительно для удобства
C=MY,ST=another state,L=another city,O=my company,OU=certs,
CN=www.example.com
# Существуют различные интерпретации полей RDN,
# далее представлены общепринятые значения.
# В случае персонального сертификата в этом поле могут встречаться
# GN=, SN= или pseudonym=
# C = двухбуквенный код страны согласно ISO3166
# ST = штат или область (край)
# L = местоположение, обычно - город
# O = организация - название компании
# OU = подразделение организации, обычно - подразделение или наименование группы
# объектов
# CN = общепринятое имя, обычно - наименование конечного субъекта, например
www.example.com
```

Однако некоторые сертификаты используют DN из поля **issuer** и заменяют атрибут CN (RDN) именем аутентифицируемого субъекта. Исходя из такой логики, DN субъекта, сформированный на основании DN поля **issuer** из примера выше, будет выглядеть так:

```
# Разбиение на две строки показано исключительно для удобства
C=MY,ST=some state,L=some city,O=Expensive Certs inc,
OU=X.509 certs,CN=www.example.com
```

В приведённом примере CN=www.example.com будет срабатывать, если при доступе к веб-сайту используется URL <http://www.example.com>. Если же для доступа к этому сайту также используется <http://example.com>, процесс проверки сертификата завершится неудачей. В этом случае можно использовать поле **subjectAltName** для расширения сферы действия сертификата, чтобы включить example.com (или любые другие доменные имена, на которые распространяется действие этого сертификата).

Хотя большинство сертификатов в настоящее время (2011 год) используют для описания субъекта RDN CN= поля **subject**, RFC 6125 рекомендует использовать поле **subjectAltName**, содержащее наименование конечного субъекта, и чтобы при валидации наименования субъекта значение поля **subjectAltName** имело бы приоритет перед значением атрибута CN= в поле **subject**. Возможно, через некоторое время все приложения и издатели сертификатов будут придерживаться этих принципов, и потому, для обработки всех возможных вариантов, наименование субъекта следует помещать и в значение атрибута CN= поля **subject**, и в поле **subjectAltName**.

Поле **subject** может быть пустым, в этом случае субъект, который будет аутентифицирован, определяется в поле **subjectAltName**.

Форма CN=www.example.com/emailAddress=me@example.com встречается часто и обычно создаётся по умолчанию инструментами OpenSSL при генерации [запроса на подписание сертификата \(Certificate Signing Request, CSR\)](#). В этой форме определяется второй атрибут **emailAddress**, который может присутствовать или не присутствовать. Большинство

удостоверяющих центров требуют, чтобы строка с указанием адреса электронной почты оставалась пустой при создании CSR с помощью OpenSSL, возможно затем, чтобы продать Вам ещё один сертификат для защиты адресов электронной почты. Или это кажется Вам слишком циничным?

SubjectPublicKeyInfo	Содержит два подполя (атрибута), algorithm (OID алгоритма открытого ключа из списка, определённого в RFC 3279) и subjectPublicKey (открытый ключ субъекта в виде строки бит). Пример атрибута algorithm при использовании OID алгоритма RSA (rsaEncryption): 1.2.840.113549.1.1.1
IssuerUniqueIdentifier	Необязательное поле. Определяет уникальное значение для издателя. В RFC рекомендуется, чтобы это поле НЕ присутствовало.
SubjectUniqueIdentifier	Необязательное поле. Определяет уникальное значение для субъекта, который будет аутентифицирован. В RFC рекомендуется, чтобы это поле НЕ присутствовало.
Extensions	Великое множество расширений определено в различных RFC, далее представлены только самые значимые (по нашему скромному мнению). Расширения могут быть помечены как критичные (CRITICAL). Примечание: Пометка CRITICAL имеет интересную и тонкую интерпретацию. Если обрабатывающее сертификат программное обеспечение видит значение с пометкой CRITICAL (которую оно всегда может интерпретировать), но не может распознать данное расширение, оно ДОЛЖНО отказаться от обработки и НЕ принимать этот сертификат.
AuthorityInfoAccess	Расширение широко известно по аббревиатуре AIA (не путайте с AIA в LDAP — это разные вещи). Используется для хранения информации о сервисах УЦ, в том числе Online Certificate Status Protocol (OCSP) любых форматов. Интересно, что хотя сертификаты EV требуют, чтобы служба определения статуса сертификата работала в режиме 24x7 (и чаще всего такая служба реализуется с помощью OCSP), в действующих стандартах EV использование этого поля не рассматривается. AuthorityInfoAccess = AuthorityInfoAccessSyntax # могут существовать несколько элементов AuthorityInfoAccessSyntax, # каждый состоит из: accessMethod = OID # может принимать значения: # 1.3.6.1.5.5.7.48.1 = ocsp # accessLocation = URI сервиса OCSP УЦ # для УЦ-издателя # 1.3.6.1.5.5.7.48.2 = caIssuers # используется только если сертификат подписан не корневым УЦ # accessLocation = URI описания корневого УЦ accessLocation = URI (исключительно адрес электронной почты или X.500 DirectoryString)
authorityKeyIdentifier	Необязательное расширение. Может содержать три поля: keyIdentifier (0) KeyIdentifier authorityCertIssuer (1) GeneralNames authorityCertSerialNumber (2) CertificateSerialNumber Стандарт рекомендует использовать значение keyIdentifier для всех сертификатов, кроме корневого. Обычно keyIdentifier — это 160-битный хэш SHA-1 поля subjectKeyInfo, но определены и другие методы. Наличие этого поля упрощает создание пути (цепочки) сертификатов, позволяет удостоверяющим центрам иметь несколько корневых сертификатов, у каждого из которых может быть разный ключ, на который ссылается это расширение.
subjectKeyIdentifier	Необязательное расширение, но стандарт рекомендует использовать это значение во всех сертификатах в качестве вспомогательного средства для построения пути (цепочки) сертификатов. Обычно SubjectKeyIdentifier — это 160-битный хэш SHA-1 поля subjectKeyInfo, но определены и другие методы.
KeyUsage	Определяет цели, для которых может использоваться открытый ключ, и может принимать следующие значения: digitalSignature (0) nonRepudiation (1) keyEncipherment (2) dataEncipherment (3) keyAgreement (4) keyCertSign (5) # указывает на то, что это сертификат УЦ cRLSign (6) encipherOnly (7) decipherOnly (8) Если задано значение keyCertSign, то в расширении BasicConstraints значение cA ДОЛЖНО БЫТЬ также установлено в TRUE, однако, если присутствует расширение BasicConstraints, в котором cA установлено в TRUE, то наличия KeyUsage

ExtendedKeyUsage	Если это расширение присутствует, в нём уточняются цели, для которых может быть использован открытый ключ. Значение этого поля должно соотноситься со значением поля KeyUsage. Оно может быть одним из следующих: <pre> serverAuth (1) TLS-аутентификация сервера WWW (действительно с digitalSignature, keyEncipherment или keyAgreement) clientAuth (2) TLS-аутентификация клиента WWW (действительно с digitalSignature или keyAgreement) codeSigning (3) Подписание доступного для скачивания исполняемого кода (действительно с digitalSignature) emailProtection (4) Защита электронной почты (действительно с digitalSignature, nonRepudiation, и/или (keyEncipherment или keyAgreement)) timeStamping (8) Привязка хэша объекта ко времени (действительно с digitalSignature и/или nonRepudiation) OCSPSigning (9) Подписание ответов OCSP (действительно с digitalSignature и/или nonRepudiation) </pre>
Basic Constraints	Логическое значение, определяющее, является ли данный сертификат корневым сертификатом (сертификатом УЦ) (TRUE) или нет (FALSE). В дополнении может также присутствовать необязательный атрибут pathLenConstraint, определяющий максимальную глубину цепочки сертификатов. <pre> ca TRUE FALSE pathLenConstraint INTEGER </pre>
CRL Distribution Points	Необязательное, но РЕКОМЕНДУЕМОЕ RFC расширение (попробуй разберись). Определяет одно или несколько URL (и другую опциональную информацию), по которым можно получить списки отзыва сертификатов (CRL) для удостоверяющего центра, выпустившего данный сертификат (issuer). Каждое расположение CRL, называемое DistributionPoint (пункт распространения), имеет следующий формат: <pre> # (0) = OPTIONAL distributionPoint DistributionPointName (0) # points to a structure defined below reasons ReasonFlags (0) cRLIssuer GeneralNames (0) # DN издателя CRL, если он не совпадает с # издателем сертификата. # Данное DN может использоваться в поисковых запросах LDAP # DistributionPointName может быть ЛИБО fullName GeneralNames # может содержать URI, # например, http://crl.example.com, # или название протокола, например, HTTP, LDAP, FTP и т.п. # используемого для получения CRL # ЛИБО nameRelativeToCRLIssuer # RDN, добавляемое к DN издателя для получения CRL # ReasonFlags может принимать одно из следующих значений: unused 0 keyCompromise 1 cACompromise 2 affiliationChanged 3 superseded 4 cessationOfOperation 5 certificateHold 6 privilegeWithdrawn 7 aCCompromise 8 </pre>
CertificatePolicies	Необязательное расширение. Может быть использовано для идентификации конкретных политик удостоверяющего центра issuer. Данное расширение позволяет указывать как OID (в CertPolicyId), так и другие опциональные атрибуты, которые, по существу, содержат (ссылаются на) читабельный текст, но RFC РЕКОМЕНДУЕТ использовать только OID. OID в данном поле также используется для идентификации сертификата EV. Значение может быть любым из приведённых ниже атрибутов: <pre> policyIdentifier CertPolicyId # OID policyQualifiers # разрешено несколько значений # обычно представляет собой URI на текстовое # изложение политики или просто текст </pre>
subjectAltName	Необязательное расширение, но RFC 6125 рекомендует, чтобы в сертификатах серверов это поле всегда присутствовало и в нём содержалось имя субъекта, для которого выпущен сертификат (обосновывается это тем, что поле dDSName этого расширения как раз и было определено для хранения имени сервера, а в атрибуте CN= поля subject может содержаться информация любого формата). Расширение может быть использовано для определения дополнительных субъектов, охватываемых действием данного сертификата (поле subject не пустое, либо как альтернатива полю subject (поле subject пустое, а расширение subjectAltName помечено как CRITICAL). Значение может быть любым из приведённых ниже атрибутов: <pre> otherName Пары type=, value= включая имена Kerberos и rfc822Name SRVName (RFC 4985) email me@example.com </pre>

dNSName	host1.example.com
x400Address	Почтовый адрес X.400, если Вы в курсе, что это
directoryName	Альтернативный DN
ediPartyName	Материалы EDI
uniformResourceIdentifier	URI ldap://ldap.example.com
iPAddress	192.168.0.1
registeredID	OID

Разрешается присутствие нескольких записей. Использование **subjectAltName** — путь решения проблемы нескольких имён DNS у одного сервера. Например, если доступ к серверу осуществляется с использованием `https://www.example.com` и `https://example.com`, то `www.example.com` может присутствовать в атрибуте CN= поля subject, а `example.com` может присутствовать в поле dNSName этого расширения, или, чаще, оба имени `www.example.com` и `example.com` будут присутствовать в записях dNSName. **Примечание:** в этих полях могут быть любые доменные имена, например, сразу и `www.example.com`, и `www.example.net`. Требование общего корня доменного имени не выдвигается. Не все удостоверяющие центры поддерживают использование **subjectAltName**. Генерация записей subjectAltName в самоподписанных сертификатах средствами OpenSSL — несколько грубоватый процесс, тем не менее, он описан [ниже](#).

SignatureAlgorithm Алгоритм (идентифицируемый по [OID](#)), используемый для подписания сертификата, и любые ассоциированные с ним параметры. Например, OID цифровой подписи RSA с SHA1 (`sha1WithRSAEncryption`):

1.2.840.113549.1.1.5

Это значение должно совпадать с указанным в поле [signature](#) сертификата. Список действительных для использования с сертификатами X.509 OID определён в [RFC 3279](#).

SignatureValue Битовая строка, содержащая цифровую подпись.

Все поля, за исключением **SignatureAlgorithm** и **SignatureValue** закодированы с помощью правил кодирования ASN.1 DER (Distinguished Encoding Rules), определённых в стандарте X.690 ITU-T. Цифровая подпись охватывает все закодированные DER поля, кроме, очевидно, самой себя.

Некоторые организации, занимающиеся стандартизацией (национальные, промышленные организации или правительственные агентства) определяют специфичные профили, в которых уточняется значение некоторых полей или атрибутов, либо определяются специфичные поля. Всё очень запутанно.



Списки отзыва сертификатов (Certificate Revocation Lists, CRL)

Списки отзыва сертификатов (CRL) — метод, с помощью которого сертификаты могут быть признаны недействительными до наступления даты, указанной в поле [NotAfter](#) сертификата. Обычно CRL выпускаются тем же УЦ, который выпустил и сам сертификат. CRL могут быть получены различными методами, например, посредством LDAP, HTTP или FTP. Формат CRL (в частности CRLv2, текущая версия) определён в [RFC 5280](#), а затем обновлён в [RFC 6818](#).

Теоретически, при получении сертификата от сервера, программное обеспечение, получившее сертификат, должно проверить, что этот сертификат отсутствует в CRL (не был отозван). Чтобы свести к минимуму возможность ошибочного приёма отозванного сертификата, каждый раз при получении сертификата от сервера должна быть получена последняя версия CRL. Поскольку CRL содержит список всех отозванных сертификатов какого-либо УЦ, он может быть внушительных размеров и создавать, таким образом, неприемлемую избыточность трафика при первоначальном соединении TLS/SSL. Кроме того, в зависимости от политики удостоверяющего центра, CRL могут обновляться каждый час, 12 часов, раз день и т.д. Суть в том, что полученные файлы CRL, даже если мы запрашиваем их снова и снова при каждом получении сертификата, всё равно могут содержать просроченные данные.

На практике получение CRL каждый раз при проверке сертификата выполняется довольно редко (если вообще выполняется), многие системы полагаются на кэши CRL, которые периодически обновляются. В RFC 5280 для описания частоты обновления CRL используется обтекаемый термин **suitably-recent (достаточно свежие)**. Онлайн версия списков отзыва, известная как [Online Certificate Status Protocol \(OCSP\)](#), определена в [RFC 2560](#), переработана в [RFC 5019](#) и обновлена в [RFC 6277](#). Формат CRL:

Version	Необязательное поле. В общем случае это поле будет указывать на версию 2 (CRLv2). Поскольку нумерация начинается с нуля, значение этого поля будет 1. Если поле опущено, подразумевается версия 2 (значение 1).

Signature	Значением данного поля должен быть тот же OID, что указан в поле SignatureAlgorithm ниже.						
Issuer	DN удостоверяющего центра, подписавшего и, следовательно, издавшего данный CRL. Это может быть корневой или некорневой удостоверяющий центр. Смотрите также описание поля issuer сертификата.						
ThisUpdate	Время и дата создания CRL. Формат даты может быть UTCTime (YYMMDDHHMMSSZ) или GeneralizedTime (YYYYMMDDHHMMSSZ). Z в формате представления времени — символьная константа, указывающая на использование часового пояса Zulu Time (UTC, он же время по Гринвичу), при её отсутствии подразумевается локальное время.						
NextUpdate	Время и дата создания следующего CRL. Формат даты может быть UTCTime (YYMMDDHHMMSSZ) или GeneralizedTime (YYYYMMDDHHMMSSZ). Z в формате представления времени — символьная константа, указывающая на использование часового пояса Zulu Time (UTC, он же время по Гринвичу), при её отсутствии подразумевается локальное время.						
RevokedCertificates	В CRL отозванные сертификаты идентифицируются по их серийным номерам. Серийные номера не глобально уникальны, но уникальность поддерживается в серийных номерах сертификатов, выпущенных одним УЦ, таким образом, для получения уникальности клиенты должны использовать комбинацию серийного номера и поля issuer. <table><tr><td>userCertificate</td><td>CertificateSerialNumber</td></tr><tr><td>revocationDate</td><td>Time</td></tr><tr><td>crlEntryExtensions</td><td>Extensions OPTIONAL</td></tr></table>	userCertificate	CertificateSerialNumber	revocationDate	Time	crlEntryExtensions	Extensions OPTIONAL
userCertificate	CertificateSerialNumber						
revocationDate	Time						
crlEntryExtensions	Extensions OPTIONAL						
Extensions	Несколько расширений CRL определены в RFC 3280 , ни одно из них не помечено как критичное (CRITICAL) и многие из них могут присутствовать лишь в случае непрямого CRL, то есть CRL, выпущенного сторонним УЦ (не тем УЦ, который выпустил отзываемый сертификат).						
SignatureAlgorithm	Алгоритм (идентифицируемый по OID), используемый для подписания сертификата, и любые ассоциированные с ним параметры. Например, OID цифровой подписи RSA с SHA1 (sha1WithRSAEncryption): <table><tr><td>1.2.840.113549.1.1.5</td></tr></table> Это значение должно совпадать с указанным в поле signature CRL. Список действительных для использования с сертификатами X.509 OID определён в RFC 3279.	1.2.840.113549.1.1.5					
1.2.840.113549.1.1.5							
SignatureValue	Битовая строка, содержащая цифровую подпись.						



Процесс доверия — удостоверяющие центры и сертификаты X.509

Как было сказано выше, сертификат X.509 является доверенным источником открытого ключа субъекта, указанного в сертификате (обычно указывается в атрибуте CN поля [subject](#) или в дополнении [subjectAltName](#)). Доверие формируется во время процесса создания сертификата. В зависимости от УЦ, процесс получения сертификата X.509 будет различаться в деталях, но, в общем случае, он состоит из следующих шагов:

1. Первое (самого высокого уровня) звено в цепи доверия — это удостоверяющий центр (УЦ). УЦ считаются доверенными организациями либо потому, что они сами так заявляют, либо потому, что они удовлетворяют каким-либо стандартам. Самой признанной организацией в сфере стандартизации УЦ в Северной Америке является [WebTrust](#), и большинство УЦ указывают в документации, что они подвергались ревизии аккредитованным аудитором WebTrust (хотя в последнее время всё чаще подобные проверки выполняют и другие крупные международные аудиторские конторы). Сам факт основания УЦ является первым шагом в создании линии доверия. В какой-то момент времени УЦ генерирует одну или несколько пар асимметричных ключей и с помощью закрытого ключа выпускает [самоподписанный сертификат](#) (поля [issuer](#) и [subject](#) данного сертификата X.509 совпадают и в расширении [BasicConstraints](#) CA установлен в TRUE). Этот сертификат является корневым сертификатом удостоверяющего центра, а закрытый ключ, парный к которому открытый ключ содержится в корневом сертификате, используется для подписания пользовательских сертификатов. Корневые сертификаты (или сертификаты УЦ) распространяются среди конечных пользователей посредством какого-либо доверенного процесса, чаще всего с инсталлятором браузера.

2. Пользователь, желающий получить сертификат, рассматривает продукты различных удостоверяющих центров (УЦ) и выбирает наиболее подходящий для него УЦ. Выбранному УЦ (или одному из его агентов — регистрационных центров (РЦ)) подаётся заявка на конкретный тип сертификата SSL (X.509). В зависимости от типа сертификата затребывается и (как правило) проверяется различная информация, такая как наименование бизнеса, регистрационный номер бизнеса или иная идентификационная информация. Опять же, в зависимости от типа сертификата, могут быть затребованы дополнительные подтверждения идентичности. В основном вся эта информация подвергается ручной обработке (хотя процесс может быть в большей или меньшей степени автоматизирован). В результате этих проверок устанавливается следующее звено в цепи доверия.

Удостоверяющий центр, сам по себе являющийся доверенным, уполномочен достоверно (доверенно) заявить о том (установить то), что пользователь является именно тем, за кого он себя выдаёт. В большей или меньшей степени.

3. После выбора продукта SSL, предоставления необходимой идентификационной информации и проверки её удостоверяющим центром, пользователю предлагается сгенерировать набор асимметричных ключей и с помощью закрытого ключа подписать запрос на подписание сертификата (Certificate Signing Request, CSR), в котором, кроме прочей информации, будет содержаться открытый ключ из сгенерированной пары ключей. Формат CSR определён в [RFC 2986](#) (перепубликация стандарта RSA, PKCS#10), обновлён в [RFC 5967](#) и содержит следующие данные:

Version	Version 0.
Subject	DN, определяющее субъекта, который будет ассоциирован с сертификатом. В общем случае, в зависимости от политик УЦ, это должно быть то значение, которое будет помещено в поле subject возвращаемого сертификата X.509.
SubjectPublicKeyInfo	Содержит два подполя (атрибута): algorithm (OID алгоритма шифрования с открытым ключом из списка, определённого в RFC 3279) и subjectPublicKey (открытый ключ субъекта в виде битовой строки). Пример OID алгоритма RSA (rsaEncryption): 1.2.840.113549.1.1.1
Attributes	Необязательное поле. Может содержать несколько подполей (атрибутов), пока определены следующие: challengePassword (пароль для дополнительной защиты CSR — большинство УЦ настаивают, чтобы этот атрибут НЕ присутствовал) и unstructuredName (любой подходящий текст — опять же, обычно УЦ не требуют его наличия или игнорируют, если таковой имеется).
SignatureAlgorithm	Алгоритм (идентифицируемый по OID), используемый для подписания CSR, и любые ассоциированные с ним параметры. Например, OID цифровой подписи RSA с SHA1 (sha1WithRSAEncryption): 1.2.840.113549.1.1.5 Примечание: CSR подписывается с помощью закрытого ключа из пары закрытого и открытого ключей, соответствующий открытый ключ помещается в поле SubjectPublicKeyInfo этого CSR.
SignatureValue	Битовая строка, содержащая цифровую подпись.

Процедура создания CSR описана [далее](#).

4. CSR загружается на сервер УЦ (обычно по FTP или HTTP). УЦ использует данные из CSR и, возможно, другую информацию, полученную из заявки на сертификат SSL, для создания сертификата X.509 пользователя, обычно со сроком действия от 1 до 3 лет. Наконец, УЦ подписывает сертификат пользователя (поля [SignatureAlgorithm](#) и [SignatureValue](#)) используя закрытый ключ, парный к которому открытый ключ содержится в корневом сертификате УЦ. Сертификат X.509 тем или иным способом (FTP/HTTP/EMAIL) отправляется пользователю.
5. Таким образом, цикл доверия завершается цифровой подписью удостоверяющего центра. Цифровая подпись пользовательского сертификата может быть верифицирована только с помощью открытого

ключа издателя (поле [issuer](#)), который содержится в корневом сертификате УЦ, получаемом путём [какого-либо \(доверенного\) процесса](#) (обычно с инсталлятором браузера).

При получении браузером сертификата от веб-сайта (во время протокола рукопожатия TLS/SSL) должен быть проверен весь путь вплоть до корневого сертификата (сертификата УЦ). На этом пути может быть один или несколько уровней сертификатов, в зависимости от поставщика сертификата. Корневые сертификаты (сертификаты УЦ), а также какие-либо необходимые промежуточные сертификаты, обычно распространяются в составе программного обеспечения основных браузеров. Распространяемые таким способом корневые сертификаты (сертификаты УЦ) часто называются якорями доверия — широко распространённый термин, используемый для описания любой базовой структуры или информации, полученной путём доверенного распространения, с помощью которой можно проверить/аутентифицировать полученную информацию. В [RFC 5914](#) (и немного в [RFC 5937](#)) определено, каким образом могут быть организованы, использованы и обработаны такие якоря доверия в конкретных случаях применения сертификатов X.509/SSL.



Сертификаты EV

Сертификаты расширенной валидации (Extended Validation, EV) определены [CA/Browser Forum](#) и включают в себя сочетание продвинутых методов канцелярских проверок и технических процессов. Цель сертификатов EV — обеспечение повышенного доверия к конечным пользователям, хотя в стандартах прямо заявляется, что эти сертификаты не гарантируют, что владелец сертификата занимается заявленной деятельностью (бизнесом) — только то, что владелец действительно существует.

В браузерах, поддерживающих сертификаты EV, при защищённом соединении пользователь видит обычный значок замка, но при применении сертификата EV строка состояния подсвечивается зелёным цветом. В настоящий момент сертификаты EV поддерживают следующие браузеры: MSIE (только 7) и последние версии Konqueror, Firefox (v3+) и Opera(9.5+). Обычно сертификаты EV значительно дороже, нежели другие типы сертификатов. Характеристики стандарта издания сертификатов EV:

1. Удостоверяющий центр должен быть аттестован на соответствие EV и подвергаться ежегодной переаттестации.
2. При расширенной проверке пользователя, кроме всего прочего, требуется, чтобы УЦ убедился, что пользователь, подавший заявку, действительно является владельцем того доменного имени, аутентификацию которого он стремится получить!
3. Удостоверяющие центры следуют практикам, изложенным в [RFC 3647](#), который имеет статус информационного (INFORMATIONAL), то есть все остальные могут его не придерживаться.
4. Стандартизируется использование и значение некоторых атрибутов DN в поле [subject](#), а также добавляются несколько новых. В частности, следующие атрибуты определены как требуемые (REQUIRED) — эквивалент ключевого слова MUST из RFC в стандартах EV:
 1. O - organizationalName (OID 2.5.4.10). Полное юридическое название организации пользователя.
 2. businessCategory (OID 2.5.4.15). Определяет, какие категории сертификатов могут подойти, могут быть разделы 5b, 5c или 5d.
 3. C - Country - (OID 2.5.4.6), ST - stateOrProvinceName - (OID 2.5.4.8) и L - localityName - (OID 2.5.4.7). Значения всех этих атрибутов должны соответствовать легальной юрисдикции бизнес-субъекта (а не, скажем, расположению веб-сервера).
 4. serialNumber (OID 2.5.4.5). Регистрационный номер бизнеса.
 5. CN - CommonName - (OID 2.5.4.3). Доменное имя хоста, например, [www.example.com](#).
5. Обязательное требование: предоставление EV-удостоверяющими центрами онлайн-овых (24x7) возможностей проверки статуса любого сертификата EV. Обычно это достигается путём использования [Online Certificate Status Protocol \(OCSP\)](#) ([RFC 2560](#)).
6. Обязательное требование: сертификаты EV могут выдаваться только для аутентификации серверов (по крайней мере пока), то есть в атрибуте CN= должно быть имя сервера, такое как [www.example.com](#) или [mail.example.com](#).
7. Точное указание на то, что это именно сертификат EV, производится путём использования зарегистрированного OID (уникального для каждого УЦ) в поле [CertificatePolicies](#).

Примеры — самоподписанные сертификаты

В этом разделе показано использование команд [OpenSSL](#) для выполнения задач, связанных с сертификатами X.509. Для иллюстрации всех функций работы с сертификатами он охватывает генерацию того, что в просторечии называется **самоподписанными сертификатами**. Термин "самоподписанные" требует небольшого разъяснения, поскольку у него два потенциальных значения.

Если подходить строго, то этот термин означает сертификат, в котором значения полей [issuer](#) и [subject](#) совпадают. В этом смысле все корневые сертификаты являются самоподписанными. Второе значение — то, когда пользователь становится сам себе удостоверяющим центром (УЦ), что является альтернативой покупке сертификата X.509 у официально признанных УЦ, таких как Verisign или Thawte (и других), которые уже являются доверенными (их корневые сертификаты (сертификаты УЦ) предустанавливаются на компьютер пользователя основными инструментами работы с сертификатами, например, браузерами). Полученный браузером с веб-сайта во время фазы протокола рукопожатия TLS/SSL сертификат пользователя (конечного субъекта), изданный одним из официально признанных УЦ, может быть отслежен (и, следовательно, аутентифицирован) браузером вплоть до удостоверяющего центра (УЦ) с использованием поля [issuer](#) полученного сертификата. Чтобы браузер не выводил сообщений об ошибках, должен быть предустановлен корневой сертификат УЦ (а также любые промежуточные сертификаты). Для определения установленных сертификатов используется термин **якорь доверия**. Корневые сертификаты большинства коммерческих УЦ и многих национальных УЦ распространяются и устанавливаются с программным обеспечением браузеров, таких как MSIE, Firefox, Opera и т.д.

Далее в этом разделе рассматриваются самоподписанные сертификаты по второму значению этого термина — пользователь становится сам себе удостоверяющим центром. Такая форма **самоподписанных сертификатов** может быть использована либо во время тестирования, либо в рабочей среде, например, когда пользователь хочет организовать контроль доступа в какой-либо частной сети, закрытой группе пользователей или каком-либо другом сообществе. В этом случае отношения доверия, обычно ассоциированные с внешним УЦ, могут рассматриваться как неявные из-за характера организации подписания таких сертификатов.

Когда программное обеспечение с поддержкой TLS/SSL (браузер) впервые встречает самоподписанный сертификат и подразумевается, что у браузера нет копии соответствующего корневого сертификата, будет сгенерировано сообщение, спрашивающее пользователя, желает ли он принять сертификат. С другой стороны, самоподписанный корневой сертификат может быть [предустановлен или импортирован на компьютер пользователя](#), в этом случае браузер не будет генерировать сообщения об ошибке.

Примечание по срокам действия сертификатов и размерам ключей: Большинство сертификатов, описанных в последующих подразделах, имеют срок действия от 1 до 3 лет. Однако, большинство коммерческих корневых сертификатов, поставляемых с браузерами, имеют сроки действия от 10 до 20 лет! Нет ничего страшного в использовании достаточно длительных сроков действия сертификатов, чтобы не беспокоиться об их постоянном перевыпуске. Текущая (2011 год) рекомендация по длине ключа RSA, — 2048 бит, — действительна до 2030 года (вследствие чего срок действия многих ныне действующих корневых сертификатов истекает около 2028 года, давая им запас времени в пару лет для увеличения размера ключа, прежде чем ключи размером 2048 бит не перестанут быть легитимными в 2030 году). [Рекомендации RSA по силе и требуемым срокам действия ключей](#). Аналогичные рекомендации по размеру (2048 бит) и срокам действия (до 2030 года) ключей также содержатся в специальной публикации 800-57, часть 1, ревизия 2, таблица 4 Национального института стандартов и технологий США (National Institute of Standards and Technology, US NIST).

Обзор процесса

В приведённых далее последовательностях действий используются команды OpenSSL (тестировалось с OpenSSL 0.9.8n и 1.0.0e) для генерации самоподписанных сертификатов X.509, которые могут быть установлены и использованы серверными системами TLS/SSL, такими как веб, FTP, LDAP или почтовыми системами (агентами SMTP). Команды OpenSSL используют множество параметров, в том числе ответы по умолчанию, сроки действия и поля сертификатов, из файла **openssl.cnf** (в FreeBSD — /etc/ssl/openssl.cnf), и если Вы собираетесь работать с сертификатами всерьёз, стоит просмотреть и, по мере необходимости, отредактировать этот файл, чтобы потом меньше было печатать параметров в командной строке.

Примечание: Есть серьёзные опасения, что ошибки, допущенные при редактировании openssl.cnf или при

экспериментах с CA.pl (полезный скриптовый файл), могут привести к падению небес на землю. Сделайте резервную копию обоих файлов, прежде чем начинать что-либо делать, тогда Вы в любой момент сможете вернуть Вашу систему в первоначальное состояние, если что-то пойдёт не так. Помните: всё что Вы делаете, Вы делаете на свой страх и риск.

Создаваемые при работе с сертификатами файлы будут содержать либо открытые ключи, которые обычно не требуют особенной защиты, либо закрытые ключи, которые необходимо хорошо защищать. Будут ли оба типа файлов храниться в одной структуре каталогов или нет — зависит от локальной политики безопасности. Многое также зависит от требований окружения. Как и с любым сложным программным обеспечением, существует огромное множество способов что-либо сделать, но из них лишь два-три могут оказаться действительно полезными методами (Really Useful Methods, RUM™).

Примечание по работе в FreeBSD: На свежееустановленной FreeBSD 8.1 в установке **openssl** по умолчанию (0.9.8n) отсутствует скрипт CA.pl. Если Вы запрашивали инсталляцию исходных кодов, то его можно найти в /usr/src/crypto/openssl/apps/CA.pl. Другой вариант — использовать систему портов (/usr/ports/security/openssl), а затем выполнить **make patch** (просто скачать и распаковать openssl, но не устанавливать его), после чего найти скрипт с помощью **find ./ -name "CA.pl"**. Поскольку PERL в FreeBSD больше не устанавливается по умолчанию, его нужно установить самостоятельно.

Метод 1: Быстрое создание корневого сертификата и сертификата сервера.

Метод 2: Быстрое создание единого сертификата, который будет корневым сертификатом и сертификатом сервера.

Метод 3: Корневой сертификат УЦ и несколько сертификатов.

Метод 3A: Создание подчинённых УЦ, промежуточных сертификатов и кросс-сертификатов.

Метод 1: Корневой сертификат, сертификат сервера

Создаётся сертификат УЦ, то есть корневой сертификат, который можно импортировать в браузер в целях тестирования, а также один сертификат сервера, который может быть использован сервером, например, Apache. Для упрощения процесса используется стандартный скрипт OpenSSL CA.pl (для упрощения примеров мы переместили его в /etc/ssl, Вы же указывайте своё расположение).

Примечание: Для генерации сертификатов, которые можно было бы использовать с несколькими именами сервера, например, example.com, www.example.com или www.example.net требуется процесс, описанный ниже.

```
# По умолчанию используется алгоритм RSA с ключами 1024 бит (2011 год).
# О том как изменить значения по умолчанию смотрите в пункте 1 метода 3.
# Создаём директорию и самоподписанный корневой сертификат УЦ:
cd /etc/ssl
./CA.pl -newca
# Запрашиваемая здесь последовательность атрибутов относится к DN УЦ.
# По умолчанию корневой сертификат создаётся в
# demoCA/cacert.pem
# а закрытый ключ УЦ в
# demoCA/private/cakey.pem
# Срок действия cacert.pem - 3 года

./CA.pl -newreq
# Создаём запрос на подписание сертификата (CSR)
# Запрашиваемая последовательность атрибутов относится к DN сервера
# Важно лишь значение атрибута CN (CommonName),
# где следует указать имя веб-сервера или другого сервера,
# например, www.example.com, ldap.example.com или mail.example.com

./CA.pl -sign
# Подписываем CSR и оставляем сертификат в
# /usr/local/openssl/newcert.pem
# Срок действия newcert.pem - 1 год.
# Закрытый ключ в
# /usr/local/openssl/newkey.pem

# Удаляем парольную фразу из newkey.pem
# чтобы сервер не запрашивал пароль при загрузке
openssl rsa -in newkey.pem -out keyout.pem

# ВНИМАНИЕ: этот файл содержит закрытый ключ, и права только на чтение
# должны быть только у владельца этого файла
```



```
# Переместите и переименуйте файлы согласно требованиям сервера.
# Необходимы оба файла newcert.pem и keyout.pem.
# Пример для apache:
SSLCertificateFile /path/to/newcert.pem
SSLCertificateKeyFile /path/to/keyout.pem
# Пример для OpenLDAP:
TLSCACertificateFile /path/to/cacert.pem
TLSCertificateFile /path/to/newcert.pem
TLSCertificateKeyFile /path/to/keyout.pem
```

Дополнительные пояснения и примечания смотрите [далее](#). Файл demoCA/cacert.pem, являющийся корневым сертификатом, может быть скопирован и [импортирован в браузер](#).

Метод 2: Только сертификат сервера

Это самый простой (в одну команду) путь создания сертификата X.509, который можно будет использовать и как сертификат сервера, и как корневой сертификат (сертификат УЦ). Поскольку этот сертификат самоподписанный (и присутствует поле CA:True), он может быть [импортирован в браузер или другое клиентское ПО](#) как корневой сертификат (сертификат УЦ). Запрашиваемая последовательность атрибутов относится к DN того сервера, для которого генерируется сертификат. В частности, атрибут CN должен определять имя хоста сервиса, такое как www.example.com, а не имя хоста компьютера. То есть, если веб-сервис имеет адрес https://www.example.com, а выполняется на хосте webserver.example.com, следует использовать CN=www.example.com. Далее показан стандартный диалог OpenSSL, значения #обрамлённые_знаками_решётки# должны быть заменены на требуемые, например, #имя_сервера# следует заменить на действительное имя сервера, который подлежит аутентификации, скажем, www.exmple.com или ldap.example.com.

Примечание: Для генерации сертификатов, которые можно было бы использовать с несколькими именами сервера, например, example.com, www.example.com или www.example.net требуется процесс, описанный [ниже](#).

```
cd /etc/ssl
openssl req -x509 -nodes -days 1059 -newkey rsa:2048 \
-keyout testkey.pem -out testcert.pem

Generating a 2048 bit RSA private key
.....+++++
.....+++++
writing new private key to 'testkey.pem'
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:#код_вашей_страны#
State or Province Name (full name) [Some-State]:#название_вашего_субъекта_страны#
Locality Name (eg, city) []:#название_вашего_города#
Organization Name (eg, company) [Internet Widgets Pty Ltd]:#название_вашей_организации#
Organizational Unit Name (eg, section) []:#необязательное_поле#
Common Name (eg, YOUR name) []:#имя_сервера#
Email Address []:..

# Создаём сертификат testcert.pem
# и незашифрованный закрытый ключ testkey.pem
# которому необходимо немедленно дать права
# только на чтение только владельцу:
# chown user:group testkey.pem
# chmod 0400 testkey.pem

# Переместите и переименуйте файлы согласно требованиям сервера.
# Необходимы оба файла testcert.pem и testkey.pem.
# Пример для apache:
SSLCertificateFile /path/to/testcert.pem
SSLCertificateKeyFile /path/to/testkey.pem
# Пример для TLS-сервера OpenLDAP (slapd.conf)
TLSCertificateFile /path/to/testcert.pem
TLSCertificateKeyFile /path/to/testkey.pem
# Пример для TLS-клиента OpenLDAP (ldap.conf)
TLS_CACERT /path/to/testcert.pem
```

Примечание: Параметр **-x509** используется для создания самоподписанного сертификата УЦ. **-nodes** подавляет диалог запроса парольной фразы. **-days 1059** устанавливает срок действия сертификата: 2 года 329 дней.

<ляп> Читатели могут задаться вопросом, зачем здравомыслящему человеку может потребоваться создавать сертификат сроком действия 2 года 329 дней? Тому есть две причины. Во-первых, если можно так сделать, то почему бы и нет? Во-вторых, внимательные читатели могли заметить, что 3 года составляет 1095 дней а не 1059, которые были использованы в тестах. Если честно, при первоначальном запуске тестов мы перепутали 5 и 9 (по молодости лет, плохому зрению или глупости — решайте сами) и вместо того, чтобы перезапустить тесты с 1095 днями, мы оставили 1059 и придумали эту глупую отмазку. Наконец, поскольку предполагается, что текущие рекомендации по размеру ключей в 2048 бита будут действительны до 2030 года, можно вполне разумно задать это значение в **-days 6205** (по состоянию на 2013 год): 17 лет с учётом високосных лет.**</ляп>**

Метод 3: Корневой сертификат УЦ и несколько сертификатов

Если Вы собираетесь генерировать несколько сертификатов, скажем, для использования во внутренней системе, этому стоит посвятить некоторое время и усилия. Мы пробовали несколько методов и этот, на наш взгляд, самый простой. В нём используется стандартный скрипт CA.pl для создания удостоверяющего центра, при этом инициализируются множество директорий и файлов, которые очень проблематично настроить другим способом, а затем используются команды openssl для генерации CSR и подписания сертификатов, поскольку таким способом достигается больший контроль над переменными и (относительно) меньший уровень проблем.

1. **Размещение и предварительная подготовка.** Необязательный этап, позволяющий печатать меньше параметров в командной строке. Решите, где Вы собираетесь построить свой репозиторий сертификатов. Для наглядности мы создадим его в /etc/ssl, и это размещение будет использоваться далее по тексту. Измените его в соответствии с Вашими потребностями. Переместите скрипт CA.pl (если Вы не можете его найти, используйте *locate CA.pl*) в /etc/ssl/CA.pl либо туда, где Вы собираетесь создавать репозиторий сертификатов. Мы также переименовали этот скрипт в ca.pl из-за патологического отвращения к клавише shift, но если Вы не разделяете наших взглядов, просто используйте CA.pl в примерах.

Файлы ca.pl (или CA.pl) и /etc/ssl/openssl.cnf оказывают существенное влияние на работу как самого скрипта, так и команды openssl. Произведённые нами изменения показаны ниже, для удобства в файле CA.pl (ca.pl) приведены только изменённые строки:

```
# ПРЕДУПРЕЖДЕНИЕ: Следует сделать и хранить копии первоначальных файлов
# CA.pl и openssl.cnf прежде чем вносить какие-либо изменения
# Срок действия корневого сертификата - 10 лет, можете установить другой,
# но большинство общедоступных УЦ предоставляют корневые сертификаты,
# действительные до 2028
$CADAYS="-days 3650";    # 10 лет
# по умолчанию $CADAYS="-days 1059";    # 3 года

# Изменяем директорию УЦ
$CATOP="./ca";
# по умолчанию $CATOP="./demoCA";
# при изменении этой переменной требуются соответствующие изменения в openssl.cnf
```

Чтобы упростить себе жизнь, мы сделали следующие изменения в /etc/ssl/openssl.cnf (опять же, приведены только изменённые строки):

```
# ПРЕДУПРЕЖДЕНИЕ: Следует сделать и хранить копии первоначальных файлов
# CA.pl и openssl.cnf прежде чем вносить какие-либо изменения
[ CA_default ]
# изменение директории как в ca.pl
dir = ./ca          # Тут будет всё
# dir = ./demoCA    # по умолчанию

# Параметр копирования расширений: используйте с осторожностью
copy_extensions = copy
# раскомментируйте эту директиву если требуется сертификат для нескольких имён DNS,
# иначе оставьте закомментированной (по умолчанию)

# Срок действия сертификата изменён на 3 года.
# Можно опустить опцию -days
default_days = 1059 # 3 года
# по умолчанию default_days = 365 # 1 год

[ policy_match ]
# Добавление этой строки в предыдущий раздел
# приведёт к добавлению атрибута L= в DN issuer и subject,
# в противном случае этот атрибут будет опущен - на Ваш выбор
localityName = optional
```

```
[req]
default bits = 2048 # текущая рекомендация на 2011-2030 годы
# default_bits = 1024 # значение в оригинальном файле

[ req_distinguished_name ]
# Чтобы меньше печатать установите корректные значения _default.
# Отредактируйте или добавьте в нужное место
countryName_default          = MY
stateOrProvinceName_default  = STATE
localityName_default         = CITY
organizationName_default     = ONE INC
organizationalUnitName_default = IT
```

Стоит проверить все остальные значения этого файла — вдруг Вы захотите что-то изменить.

2. **Создание удостоверяющего центра.** Первая команда создаёт корневой сертификат удостоверяющего центра (УЦ) и некоторые другие служебные файлы. Создаётся пара ключей — открытый и закрытый. Открытый ключ записывается в корневой сертификат /etc/ssl/ca/cacert.pem, закрытый ключ — в /etc/ssl/ca/private/cakey.pem. Формат файла по умолчанию — [PEM](#). Запрашиваемые детали DN будут использованы для формирования полей [issuer](#) и [subject](#) этого корневого сертификата, а также для поля [issuer](#) всех последующих подписанных сертификатов, укажите их в соответствии со своими требованиями. Парольная фраза обязательна, она устанавливается для защиты закрытого ключа и будет запрашиваться при всех последующих подписаниях сертификатов, так что её стоит запомнить:

```
cd /etc/ssl
./ca.pl -newca

CA certificate filename (or enter to create) #ENTER

Making CA certificate ...
Generating a 2048 bit RSA private key
.....+++++
.....+++++
writing new private key to './ca/private/cakey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [MY]:
State or Province Name (full name) [STATE]:
Locality Name (eg, city) [CITY]:
Organization Name (eg, company) [ONE INC]:CA COMPANY NAME
Organizational Unit Name (eg, section) [IT]:X.509
Common Name (eg, YOUR name) []:CA ROOT
Email Address []:.

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /etc/ssl/openssl.cnf
Enter pass phrase for ./ca/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
  Serial Number:
    bb:7c:54:9b:75:7b:28:9c
  Validity
    Not Before: Apr 15 21:07:36 2008 GMT
    Not After : Apr 13 21:07:36 2018 GMT
  Subject:
    countryName          = MY
    stateOrProvinceName  = STATE
    organizationName     = CA COMPANY NAME
    localityName         = CITY
    organizationalUnitName = X.509
    commonName           = CA ROOT
  X509v3 extensions:
    X509v3 Subject Key Identifier:
      54:0D:DE:E3:37:23:FF:2E:E8:03:0A:2C:52:FE:FC:C0:C8:13:72:80
    X509v3 Authority Key Identifier:
      keyid:54:0D:DE:E3:37:23:FF:2E:E8:03:0A:2C:52:FE:FC:C0:C8:13:72:80
    DirName:/C=MY/ST=STATE/O=CA COMPANY NAME/L=CITY/OU=X.509/CN=CA ROOT
    serial:BB:7C:54:9B:75:7B:28:9C
```

```
X509v3 Basic Constraints:
CA:TRUE
Certificate is to be certified until Apr 13 21:07:36 2018 GMT (3650 days)
Write out database with 1 new entries
Data Base Updated
```

Создаётся стандартная структура директорий:

```
ca                # cacert.pem (корневой сертификат)
                  # serial (отслеживание серийных номеров)
                  # crlnumber (серийные номера CRL)
                  # index.txt
ca/private/cakey.pem # закрытый ключ УЦ
ca/newcerts        # копии всех созданных сертификатов
ca/crl             # опциональное место размещения созданных CRL
ca/certs           # опциональное место размещения созданных сертификатов
```

Срок действия созданного корневого сертификата (ca/cacert.pem) — 10 лет (3650 дней) согласно отредактированному значению в файле ca.pl. Соответствующий закрытый ключ (ca/private/cakey.pem) защищён парольной фразой PEM, но всё равно его следует защищать установкой минимально возможных прав доступа (назначаемые по умолчанию права 0644 неоправданно высоки). Запрашиваемый Challenge Password — простой пароль, который может быть использован для защиты доступа к сертификатам, CSR и последующим сертификатам X.509. Большинство (если не все) публичные удостоверяющие центры не приветствуют установку этого пароля и сопутствующего опционального названия компании (Optional Company Name), и во всех примерах это поле оставляется незаполненным. Если Вы хотите отключить запрос этих значений, отредактируйте /etc/ssl/openssl.cnf:

```
[ req_attributes ]
# Закомментируйте указанные строки
#challengePassword      = A challenge password
#challengePassword_min  = 4
#challengePassword_max  = 20
#unstructuredName       = An optional company name
```

3. **Создание запроса на подписание сертификата (CSR).** Запрашиваемые в этом случае атрибуты DN будут относиться к полю [subject](#) итогового сертификата.

Эта команда может использоваться и для создания запроса CSR (Certificate Signing Request) в публичный УЦ, а поскольку большинство УЦ не приветствуют паролей, просто нажмите ENTER при запросе **A challenge password** (о том, как совсем отключить эти запросы смотрите выше). Опциональное значение emailAddress также остаётся пустым, главным образом потому, что для сертификата сервера оно не имеет никакого значения, а также потому, что в большинстве запросов в публичные УЦ оно не заполняется.

```
openssl req -nodes -new -newkey rsa:2048 \
-keyout ca/private/cert1key.pem -out ca/certs/cert1csr.pem

Generating a 2048 bit RSA private key
.....++++++
.....++++++
writing new private key to 'ca/private/cert1key.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [MY]:
State or Province Name (full name) [STATE]:
Locality Name (eg, city) [CITY]:
Organization Name (eg, company) [ONE INC]:
Organizational Unit Name (eg, section) [IT]:
Common Name (eg, YOUR name) []:www.example.com
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Эта команда создаёт новый запрос на подписание сертификата (в /etc/ssl/ca/certs/cert1csr.pem), который затем должен быть подписан с помощью закрытого ключа УЦ, и закрытый ключ (который, за счёт использования параметра **-nodes**, не будет защищён парольной фразой — ведь мы будем использовать его в серверном приложении) в /etc/ssl/ca/private/cert1key.pem. Значение CN (в примере — www.example.com) — это используемое приложением имя сервиса. Оно могло бы быть ldap.example.com, либо любым другим подходящим именем. Имейте также ввиду, что такой сертификат сработает при доступе на https://www.example.com, но если для доступа к тому же сервису используется https://example.com, то Вам перед созданием CSR нужно добавить поле subjectAltName с помощью [описанной ниже процедуры](#) (и использовать в секции VirtualHost настроек Apache директиву **ServerAlias**). Имя сервиса не следует путать именем хоста сервера. Так, если к веб-сервису обращаются как к https://www.example.com, а он выполняется на хосте webserver.example.com, в CN нужно использовать www.example.com.

Для просмотра запроса сертификата выполните:

```
openssl req -in ca/certs/cert1csr.pem -noout -text
Certificate Request:
  Data:
    Version: 0 (0x0)
    Subject: C=MY, ST=STATE, L=CITY, O=ONE INC, OU=IT, CN=www.example.com
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      RSA Public Key: (1024 bit)
        Modulus (1024 bit):
          00:ae:19:86:44:3c:dd:38:df:e2:41:5f:d8:86:19:
          69:7e:85:d7:1d:ae:1e:eb:87:b0:5f:fc:f3:db:e3:
          aa:82:76:d6:42:05:f1:0e:5c:5a:a2:8d:f6:d3:00:
          37:04:96:13:06:16:e6:d1:67:14:69:cd:85:df:a7:
          b3:ac:a2:6c:33:cd:d6:00:3d:24:99:fa:4b:81:07:
          0c:b2:5a:fe:06:16:da:34:66:63:78:31:7d:11:5e:
          63:de:9e:ee:76:8b:0c:12:af:fb:f2:28:0a:76:5b:
          99:20:b8:f7:c0:9c:e8:89:c5:d0:1e:e5:07:c8:bd:
          38:c8:52:97:cc:76:c9:c8:2b
        Exponent: 65537 (0x10001)
    Attributes:
      a0:00
  Signature Algorithm: sha1WithRSAEncryption
  79:f5:20:45:6c:ec:8e:ae:3a:75:70:67:6d:cb:a9:a6:df:80:
  f1:8f:43:1b:c5:ce:dd:41:19:e9:8e:72:06:d3:05:b8:23:e6:
  98:88:3d:a4:96:62:27:e6:6f:af:fd:7a:b9:20:ec:c6:8d:de:
  2b:26:5c:a3:b5:0f:20:09:d3:e8:90:71:0f:dd:81:82:83:94:
  bb:ac:fa:09:42:5c:7a:91:19:9a:4a:98:7c:92:7d:3d:42:d0:
  66:2a:87:ff:b5:58:bd:41:08:1f:4f:02:04:4d:ca:75:6c:5b:
  bd:61:cd:c5:89:7c:e0:0d:dd:50:1f:90:42:5f:d6:a6:31:71:
  40:7d
```

4. **Создание и подписание сертификата конечного пользователя.** Эта команда принимает на входе CSR (ca/certs/cert1csr.pem) и создаёт сертификат конечного пользователя (ca/certs/cert1.pem):

```
openssl ca -policy policy_anything \
-in ca/certs/cert1csr.pem -out ca/certs/cert1.pem

Using configuration from /usr/local/openssl/openssl.cnf
Enter pass phrase for ./ca/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
  Serial Number:
    bb:7c:54:9b:75:7b:28:9d
  Validity
    Not Before: Apr 15 22:21:10 2008 GMT
    Not After : Mar 10 22:21:10 2011 GMT
  Subject:
    countryName           = MY
    stateOrProvinceName   = STATE
    localityName          = CITY
    organizationName      = ONE INC
    organizationalUnitName = IT
    commonName            = www.example.com
  X509v3 extensions:
    X509v3 Basic Constraints:
      CA:FALSE
    Netscape Comment:
      OpenSSL Generated Certificate
    X509v3 Subject Key Identifier:
      EE:D9:4A:74:03:AC:FB:2C:FD:43:C7:58:6E:2E:6A:88:BA:65:61:CC
    X509v3 Authority Key Identifier:
      keyid:54:0D:DE:E3:37:23:FF:2E:E8:03:0A:2C:52:FE:FC:C0:C8:13:72:80
```



```
Certificate is to be certified until Mar 10 22:21:10 2011 GMT (1059 days)
Sign the certificate? [y/n]:y
```

```
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Поле **issuer** полученного в результате сертификата формируется из поля **subject** корневого сертификата (ca/cacert.pem), а поле **subject** нового сертификата — из CSR. Многие значения по умолчанию команда берёт из openssl.cnf: срок действия (default_days =), ключ подписания (private_key =) и DN издателя из корневого сертификата (certificate =). В зависимости от требований Вашей системы параметр - **policy policy_anything** может быть обязательным. Он указывает на определённую в файле openssl.cnf секцию **policy_anything**, которая позволяет задавать в DN поля subject сертификата значения атрибутов, отличные от соответствующих атрибутов в DN поля issuer. При отсутствии этого параметра по умолчанию используется -**policy policy_match**, накладывающее ограничения на значения RDN C=, ST= и O=.

Для просмотра полученного в результате сертификата выполните:

```
openssl x509 -in ca/certs/cert1.pem -noout -text

Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      bb:7c:54:9b:75:7b:28:9d
    Signature Algorithm: sha1WithRSAEncryption
    Issuer: C=MY, ST=STATE, O=CA COMPANY NAME, L=CITY, OU=X.509, CN=CA ROOT
    Validity
      Not Before: Apr 15 22:21:10 2008 GMT
      Not After : Mar 10 22:21:10 2011 GMT
    Subject: C=MY, ST=STATE, L=CITY, O=ONE INC, OU=IT, CN=www.example.com
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      RSA Public Key: (1024 bit)
      Modulus (1024 bit):
        00:ae:19:86:44:3c:dd:38:df:e2:41:5f:d8:86:19:
        69:7e:85:d7:1d:ae:1e:eb:87:b0:5f:fc:f3:db:e3:
        aa:82:76:d6:42:05:f1:0e:5c:5a:a2:8d:f6:d3:00:
        37:04:96:13:06:16:e6:d1:67:14:69:cd:85:df:a7:
        b3:ac:a2:6c:33:cd:d6:00:3d:24:99:fa:4b:81:07:
        0c:b2:5a:fe:06:16:da:34:66:63:78:31:7d:11:5e:
        63:de:9e:ee:76:8b:0c:12:af:fb:f2:28:0a:76:5b:
        99:20:b8:f7:c0:9c:e8:89:c5:d0:1e:e5:07:c8:bd:
        38:c8:52:97:cc:76:c9:c8:2b
      Exponent: 65537 (0x10001)
    X509v3 extensions:
      X509v3 Basic Constraints:
        CA:FALSE
      Netscape Comment:
        OpenSSL Generated Certificate
      X509v3 Subject Key Identifier:
        EE:D9:4A:74:03:AC:FB:2C:FD:43:C7:58:6E:2E:6A:88:BA:65:61:CC
      X509v3 Authority Key Identifier:
        keyid:54:0D:DE:E3:37:23:FF:2E:E8:03:0A:2C:52:FE:FC:C0:C8:13:72:80

    Signature Algorithm: sha1WithRSAEncryption
    52:3d:bc:bd:3f:50:92:67:a3:d3:6f:37:a9:3f:89:b5:16:5b:
    9c:0d:32:25:32:91:c7:bf:f6:0d:f8:6d:1c:09:45:2f:3f:b9:
    18:b7:1c:8d:7c:06:33:ef:ca:e0:92:a3:90:3f:7c:4e:16:87:
    67:ae:7c:2c:1a:43:e5:3a:24:d9:c3:7d:cf:bf:eb:01:9d:c1:
    f0:bb:0f:15:de:d5:9e:42:9d:f8:7f:0d:5b:af:59:80:d1:aa:
    cc:db:31:1b:d4:7f:f3:f1:71:25:85:c9:8b:78:3e:13:ac:11:
    51:35:49:8d:c3:9a:bb:9a:89:2c:ef:7f:90:f9:05:b3:65:98:
    b8:74
```

Полезные команды

Проверка и просмотр сертификатов. Может случиться неприятность и произойти повреждение файлов. Чтобы убедиться в целостности сертификатов, используйте такие команды:

```
openssl x509 -in certificate-name.pem -noout -text
# отображает сертификат целиком

openssl x509 -in certificate-name.pem -noout -dates
```



```
# выводит только даты начала и окончания срока действия
openssl x509 -in certificate-name.pem -noout -purpose
# выводит список всех возможных целей применения сертификата
openssl x509 -in certificate-name.pem -noout -purpose - dates
# выводит срок действия и цели применения сертификата
```

5. **Импорт самоподписанного корневого сертификата.** Созданный на этапе 2 файл в формате PEM (ca/cacert.pem) может быть импортирован непосредственно в MSIE и Firefox, чтобы они не выдавали запросы о недоверенных сертификатах.

MSIE: MSIE предпочитает работать с сертификатами, имеющими суффикс .cer, но, к счастью, принимает файлы в формате PEM, так что просто переименуйте ca/cacert.pem в ca/cacert.cer. Скопируйте сертификат на машину с MSIE с помощью какого-либо портативного носителя и используйте MSIE->Меню "Сервис" (Tools)->"Свойства обозревателя" (Internet Options)->Вкладка "Содержание" (Content)->Кнопка "Сертификаты" (Certificates)->Кнопка "Импорт" (Import), а затем выполните запросы мастера импорта.

В окружении AD сертификаты также можно распространить с помощью объектов групповых политик (Group Policy Object, GPO). Загрузите "Администрирование" (Administrative Tools)->"Управление групповой политикой" (Group Policy Manager)->Раскройте пункт "Домены" (Domains)->Щёлкните правой кнопкой мыши на "Default Domain Policy" и выберите "Изменить" (Edit)->Раскройте пункт "Конфигурация компьютера" (Computer configuration)->Раскройте пункт "Политики"->Раскройте пункт "Конфигурация Windows" (Windows Settings)->Раскройте пункт "Параметры безопасности" (Security Settings)->Раскройте пункт "Политики открытого ключа" (Public Key Settings)->Щёлкните правой кнопкой мыши на "Доверенные корневые центры сертификации" (Trusted Root Certificate Authorities)->Выберите "Импорт" (Import), а затем выполните запросы мастера импорта.

Firefox: Firefox принимает файлы с суффиксами и .pem и .cer. Выберите меню "Инструменты" (Tools)->"Настройки" (Options)->"Дополнительные" (Advanced)->Вкладка "Шифрование" (Encryption)->"Просмотр сертификатов" (View Certificates)->Вкладка "Центры сертификации" (Authorities)->Нажмите кнопку "Импортировать" (Import) и выберите файл.

Метод 3А: Создание подчинённых УЦ, промежуточных сертификатов и кросс-сертификатов

В методе 3 создаётся простая цепочка из двух сертификатов: сертификата конечного субъекта (сервера) и корневого сертификата УЦ. В методе 3А производится исследование того, что можно добавить к этой структуре (по тем или иным причинам). Мы создадим подчинённый УЦ, возможно, второй корневой УЦ, а также, основываясь на этой структуре, создадим всевозможные кросс-сертификаты и промежуточные сертификаты.

1. **Основные настройки:** Выполните этапы 1 и 2 [метода 3](#). Создаётся просто структура директорий и наш первый корневой УЦ. Если Вы уже прошли все этапы метода 3, то ни одно из имён файлов, создаваемых далее не будет конфликтовать с уже созданными, так что Вы ничего не потеряете и Ваша система не нарушится. По завершении этого процесса будут созданы следующие директории и файлы:

```
ca                # cacert.pem (корневой сертификат)
                  # serial (отслеживание серийных номеров)
                  # crlnumber (серийные номера CRL)
                  # index.txt
ca/private/cakey.pem # закрытый ключ УЦ
ca/newcerts        # копии всех созданных сертификатов
ca/crl             # опциональное место размещения созданных CRL
ca/certs           # опциональное место размещения созданных сертификатов
```

Замечание по стратегии: Команда openssl берёт многие из своих расширенных параметров из конфигурационного файла (openssl.cnf). До этого момента мы предполагали, что изменения нужно производить именно в этом стандартном файле, поскольку он обычно используется при выполнении всех таких команд. Но для текущей задачи такой подход не подойдёт. Вместо этого мы рекомендуем скопировать стандартный конфигурационный файл (openssl.cnf), дать этому новому файлу подходящее имя (которое легко запомнить) и использовать аргумент **-config filename** для выбора, в зависимости от задачи, соответствующего конфигурационного файла. Это позволяет работать быстрее (со временем), не путаться (со временем) и использовать конфигурацию повторно (изначально).

2. Создаём подчинённый УЦ:

Создаём новые директории, скажем, subca и subca/private (с теми же правами, что и ca/private). Этот шаг просто позволит поддерживать порядок (в своём роде):

```
cd /etc/ssl
mkdir ca/subca
mkdir ca/subca/private
# создаём необходимый пустой файл базы данных для издателя subca
touch ca/subca/index.txt
```

Редактируем openssl.cnf, создаём секцию [sub_ca], она может находиться в любом месте файла, но для определённости создадим её перед секцией [user_certs]. Различные параметры в этой секции будут использоваться для переопределения обычных параметров, используемых при подписании сертификата.

```
# новая секция
[ sub_ca ]
basicConstraints=CA:TRUE
# другой вариант:
# basicConstraints=CA:TRUE,pathlen:0
# pathlen:0 ограничивает subCA, чтобы он мог
# подписывать только сертификаты конечных субъектов

# необязательная строка тестового комментария
# nsComment="Most Trusted Certificate in the World"
nsComment="OpenSSL Generated Certificate"
# следующие два параметра - обычные для всех некорневых сертификатов
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer

# следующая строка лишь демонстрирует позицию секции sub_ca в файле
[ user_cert ]
```

3. Создаём сертификат подчинённого УЦ:

Создаём CSR для подчинённого УЦ и подписываем его с использованием корневого сертификата УЦ:

```
# ПРИМЕЧАНИЕ: На этих этапах используется корневой УЦ, созданный на этапах 1 и 2 метода 3.

# Создаём CSR для подчинённого УЦ
openssl req -new -keyout ca/subca/private/subca1key.pem \
-out ca/subca/subca1csr.pem
Generating a 2048 bit RSA private key
.....++++++
.....++++++
writing new private key to 'ca/subca/private/subca1key.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [MY]:
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) [Some City]:
Organization Name (eg, company) [My Company Name]:
Organizational Unit Name (eg, section) []:Certs
Common Name (eg, YOUR name) []:subCA1

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:

# Примечания:
# 1. Значение Organizational Unit не так важно, несёт больше описательную нагрузку.
# 2. Значение Common Name - имя (название), данное подчинённому УЦ.

# Теперь подпишем этот запрос с использованием ключа корневого УЦ.
# Для установки определённых нами расширений используем -extensions sub_ca.
openssl ca -policy policy_anything -in ca/subca/subca1csr.pem \
-out ca/subca/subca1cert.pem -extensions sub_ca

# Выводим полученный сертификат:
openssl x509 -in ca/subca/subca1cert.pem -noout -text
```

```

Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      c6:bd:b2:ce:22:bc:4d:57
    Signature Algorithm: sha1WithRSAEncryption
    Issuer: C=MY, ST=Some-State, O=My company name, OU=Certs, CN=Root CA1
    Validity
      Not Before: Dec  9 20:40:18 2011 GMT
      Not After : Dec  6 20:40:18 2021 GMT
    Subject: C=MY, ST=Some-State, L=Some City, O=My company name, OU=Certs, CN=subCA1
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      RSA Public Key: (2048 bit)
        Modulus (2048 bit):
          00:a9:f3:02:01:c9:e3:76:e6:dd:cc:ad:d0:2d:6d:
          01:b6:27:c8:a0:9c:f3:a0:85:b8:c7:30:56:18:cb:
          34:7d:db:b5:e1:9c:8a:72:6a:91:26:52:f9:71:b7:
          a1:47:ab:68:a0:ae:96:db:f5:e5:5a:fd:a8:4f:2b:
          94:57:ab:a3:ea:c7:b9:ab:fd:16:86:36:3e:43:27:
          ae:9a:c8:3f:92:ad:ec:a4:3b:9d:6d:3b:87:9d:62:
          b5:bf:75:40:46:5b:69:80:02:db:fe:0d:f7:49:c0:
          40:b9:cf:b4:3e:7e:0f:52:c5:12:58:9d:4b:77:41:
          a0:f8:9d:36:74:e4:f9:47:c1:04:b1:5e:aa:6b:f6:
          cd:32:ec:91:a3:e6:51:36:9e:b8:9b:fc:00:80:41:
          c7:2d:4d:8a:e4:b9:46:a9:1c:d4:5c:97:14:57:21:
          e2:ff:e8:ff:28:53:5e:bb:13:93:43:86:4d:2a:a6:
          a0:93:8d:2a:61:c0:b6:e8:4e:87:64:de:09:97:66:
          c8:f7:ae:66:00:e8:47:76:00:80:12:51:f5:da:f0:
          82:f0:6c:b3:b1:00:2d:1a:20:5c:32:ea:d3:72:bc:
          0c:a6:82:b8:44:f5:96:0b:68:65:a8:d4:31:81:5e:
          f0:37:71:5d:e3:c7:3d:59:ff:32:32:49:6b:b6:e5:
          55:87
        Exponent: 65537 (0x10001)
    X509v3 extensions:
      X509v3 Basic Constraints:
        CA:TRUE
      X509v3 Key Usage:
        Certificate Sign, CRL Sign
      Netscape Comment:
        OpenSSL Generated Certificate
      X509v3 Subject Key Identifier:
        58:47:30:77:3F:EF:AC:3D:5A:A3:BB:F6:7F:D2:CE:95:08:77:23:A0
      X509v3 Authority Key Identifier:
        keyid:FB:7B:FB:7B:23:87:8B:FC:3D:97:33:11:6B:0D:37:A0:21:8D:E6:CD

    Signature Algorithm: sha1WithRSAEncryption
      43:b5:e2:8d:4d:07:56:45:20:b3:0b:ac:fd:cc:36:c0:3b:df:
      86:19:eb:96:9b:4a:e0:b5:e7:41:73:54:84:8f:ea:15:48:35:
      69:15:2f:29:0f:90:79:e3:68:e6:08:4d:98:7b:1c:f3:34:14:
      1d:4d:21:dc:0a:2c:1b:90:7b:b8:23:1b:9c:48:1a:b5:ba:91:
      44:8e:18:05:1c:ee:48:d8:ef:ab:b7:e5:f6:86:98:bf:17:cd:
      4b:40:d3:51:d2:cf:c2:43:14:a3:38:9a:72:30:c7:94:64:e3:
      93:80:5c:42:31:79:ba:10:34:d0:96:45:7c:59:67:f5:f0:dd:
      a9:aa:3d:22:97:d2:d2:05:4d:7b:aa:06:27:8a:69:9d:3c:94:
      85:95:b6:7d:49:a5:bd:42:90:57:87:af:44:0f:b2:db:98:d0:
      d6:fd:9f:65:90:70:a2:af:9d:14:34:5d:2a:7b:28:12:37:e7:
      76:0c:b0:58:a1:b7:45:6f:d7:26:dc:cb:f2:39:76:7d:08:5c:
      22:f9:99:fc:84:89:b5:7c:e0:75:eb:db:2d:18:8d:d9:da:c8:
      4d:f5:cd:47:ad:f3:49:7e:fc:96:4c:2e:e3:92:bd:a3:d8:83:
      12:2c:a2:7c:eb:dc:45:e0:61:a0:e1:31:b1:03:4b:55:e8:cf:
      e0:f3:2b:72

```

Четыре расширения **X509v3 extensions**, — следствие использования аргумента **-extensions sub_ca**, — переопределяют обычные характеристики сертификата. Если от корневого УЦ требуется подписать обычный сертификат конечного субъекта, просто опустите этот аргумент. Технически, получившийся в результате сертификат (ca/subca/subca1cert.pem) — это **кросс-сертификат**, поскольку и полученный сертификат, и сертификат издателя имеют поле **basicConstraints**, в котором **ca** установлено в **True**.

4. Подписание сертификата конечного субъекта подчинённым УЦ:

Чтобы подписать сертификат конечного субъекта с использованием ключа подчинённого УЦ, нам сначала нужно скопировать файл openssl.cnf и переименовать его, скажем, в subca1.cnf. Теперь внесём изменения в subca1.cnf (эти изменения просто сообщат openssl где брать различные файлы, относящиеся к subCA, тогда как стандартный файл openssl.cnf продолжит ссылаться на информацию о корневом УЦ):

```

[ CA_default ]
# Показаны только те значения, которые были изменены.
database       = $dir/subca/index.txt # индексный файл базы данных.

```

```

certificate      = $dir/subca/subcalcert.pem      # сертификат subCA
# Следующий параметр говорит о том, что сертификаты от обоих центров
# (корневого и подчинённого) будут иметь сквозную (единую) нумерацию.
# Чтобы изменить это, скопируйте ca/serial в ca/subca/serial
# и модифицируйте параметр.
serial           = $dir/serial                   # текущий серийный номер
# Эти параметры задаются, если будут выпускаться отдельные CRL,
# в противном случае оставьте их без изменений
crl_dir          = $dir/subca/crl                # месторасположение подписанных crl
crlnumber        = $dir/subca/crlnumber          # текущий номер crl

# Закомментируйте, если собираетесь использовать CRL V1
crl              = $dir/subca/crl.pem            # текущий CRL

private key      = $dir/subca/private/subcalkey.pem # закрытый ключ subCA
RANDFILE        = $dir/subca/private/.rand       # закрытый файл со случайным числом

```

Теперь создадим CSR на сертификат конечного субъекта и подпишем его ключом subCA:

```

# Создаём CSR:
openssl req -new -nodes -keyout ca/private/user1key.pem \
-out ca/certs/user1csr.pem
# Подписываем его ключом subCA с использованием -config subca1.cnf:
openssl ca -policy policy_anything -in ca/certs/user1csr.pem \
-out ca/certs/user1cert.pem -config subca1.cnf
# Выводим получившийся сертификат:
openssl x509 -in ca/certs/user1cert.pem -noout -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            c6:bd:b2:ce:22:bc:4d:58
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=MY, ST=Some-State, L=Some City, O=My company name, OU=Certs, CN=subCA1
        Validity
            Not Before: Dec  9 21:06:43 2011 GMT
            Not After : Dec  6 21:06:43 2021 GMT
        Subject: C=MY, ST=Some-State, L=Some City, O=My company name, OU=Server,
CN=www.example.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (2048 bit)
            Modulus (2048 bit):
                00:c3:f4:dc:07:08:30:3a:3d:72:3e:71:e2:8f:c8:
                42:a4:00:6c:9a:e4:5d:fd:3e:28:e0:90:84:1f:10:
                3d:c1:f5:22:88:82:52:2b:c3:b5:68:a7:cc:6a:04:
                60:77:cc:ac:e1:03:36:53:25:1e:d2:5b:c1:74:1f:
                ec:87:6a:a3:d7:9b:a7:11:6b:c0:ac:0d:9d:5c:a0:
                4d:17:a1:98:c5:c6:1e:3f:b6:17:ab:a3:9c:10:df:
                bf:cb:82:31:ab:c0:30:5f:58:95:c3:43:72:3a:51:
                3d:e0:f4:a1:df:df:64:98:f4:ef:d7:a6:ac:21:20:
                0d:2b:21:6e:54:98:c5:2b:d3:46:94:48:03:85:61:
                a1:bf:27:2d:3c:be:fc:5f:21:d6:82:47:4d:59:48:
                3f:f2:7c:c8:6f:04:d5:4f:64:1a:f0:e7:e7:dc:77:
                10:7f:1d:82:68:3a:ba:c9:29:79:a9:44:5b:c7:02:
                c1:59:79:4a:25:c0:e8:f1:89:89:6a:fb:81:6c:2c:
                23:b1:2c:cf:ba:e4:ac:ec:dc:f7:92:bd:aa:b1:03:
                8a:20:26:06:87:5a:d8:91:7e:4e:88:11:a8:93:b0:
                d8:e0:81:9e:86:06:94:2e:f6:fd:14:b0:91:28:59:
                a8:45:fd:c5:d7:a4:04:82:b5:80:a9:28:2b:1e:f2:
                af:dd
            Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                B1:5A:23:4E:C8:2B:FD:98:AD:01:7A:CA:73:D3:26:84:75:13:6C:78
            X509v3 Authority Key Identifier:
                keyid:58:47:30:77:3F:EF:AC:3D:5A:A3:BB:F6:7F:D2:CE:95:08:77:23:A0

        Signature Algorithm: sha1WithRSAEncryption
        50:4b:8e:50:8f:fa:f4:98:ee:37:9e:90:f0:18:af:f4:a4:21:
        57:a7:ee:d3:5e:8d:f5:f6:25:d3:84:58:33:cd:b7:ce:88:e4:
        9d:5c:f3:6f:3e:b9:b5:eb:8b:4a:48:25:e6:56:5f:da:83:63:
        d8:a8:8e:9a:dc:6e:d1:ca:d2:69:dc:31:85:3b:93:8e:41:6a:
        38:f9:e3:87:47:81:2a:eb:24:c4:03:57:43:1c:5e:2b:07:e2:
        a4:cb:f1:1d:c0:de:52:8b:f8:a2:59:d7:59:6c:8b:6e:a6:f6:
        85:6a:e4:43:05:1c:ad:07:81:db:d7:c7:fe:02:c5:42:b5:89:
        76:47:b5:6c:6e:09:d1:82:20:5d:40:93:5b:85:fb:92:c5:92:
        b6:09:57:0e:b9:36:fa:86:3a:23:89:a6:af:16:b5:0a:15:a7:
        b1:73:27:38:10:49:f3:ae:83:7f:26:65:51:1f:9f:32:7a:49:
        26:cf:d4:ef:9b:18:6f:e1:35:04:30:02:b5:ef:a6:29:0f:1e:
        b9:d4:82:d1:25:d3:ce:c2:f5:cb:80:5e:bb:7c:95:fc:fc:88:

```

```
84:da:0e:36:3a:2b:e7:9e:65:41:f2:7e:b9:fa:a7:4e:04:4d:
3d:97:52:28:1f:a6:9d:2e:54:e5:b8:45:98:33:46:0f:2c:93:
ac:58:be:eb
```

В данном случае издатель сертификата — **subCA1**, а не **root CA1**, как было при подписании запроса на создание сертификата подчинённого УЦ. Чтобы разрешить проверку подлинности сертификата конечного субъекта по всей цепочке, в браузер должны быть импортированы ОБА сертификата: корневого УЦ (ca/cacert.pem) и подчинённого УЦ (ca/subca/subca1cert.pem).

5. Другие возможности:

Используя подобную технику, можно создать второй корневой УЦ (root CA2), при этом, понятное дело, потребуется ещё одна копия openssl.cnf, либо второй подчинённый УЦ (subCA2), опять же, с уникальным конфигурационным файлом. Короче, любой каприз...

Сертификаты с несколькими именами сервера

Если сертификат предназначен для использования на хосте, у которого больше одного имени DNS, скажем, https://www.example.com, https://example.com или даже www.example.net, то вам нужно указать использование поля subjectAltName в запросе на подписание сертификата (CSR). Чтобы это сделать, отредактируйте файл openssl.cnf как показано ниже (приведены только изменённые строки):

```
# Сначала найдите секцию [CA_Default]
[CA_Default]
....
# Раскомментируйте или добавьте
copy_extensions = copy
# Это заставит функцию са скопировать поля расширений из CSR.
# Эти манипуляции следует произвести на хосте, создающем
# сертификаты на основе CSR.

# Теперь найдите секцию [v3_req]
[v3_req]
....
# Добавьте следующую строку с нужными именами хостов:
subjectAltName = "DNS:www.example.com,DNS:example.com,DNS:example.net"
# Это следует сделать на хосте, который генерирует CSR.
```

При запуске команды на создание CSR добавьте к её аргументам **-reqexts v3_req** только в том случае, когда Вам нужно включить дополнительные поля **subjectAltName**. Хотя в приведённом примере показано добавление трёх имён, на практике можно добавить любое количество. Каждое включение имеет формат DNS:hostname.domain.name, несколько включений должны быть разделены запятыми, а все вместе они должны быть заключены в кавычки. Вот запрос на создание CSR из этапа 3 метода 3 с дополнительным аргументом:

```
openssl req -nodes -new -newkey rsa:2048 \
-keyout ca/private/cert1key.pem -out ca/certs/cert1csr.pem -reqexts "v3_req"
```

Если при создании CSR не указывался аргумент **-reqexts**, по этому CSR будет создан обычный сертификат с единственным именем сервера. Приведённый пример работает как дополнение рассмотренного выше метода 3 и представляет собой наиболее гибкий подход.

Примечание: Если Вы хотите выпустить несколько сертификатов с разным набором имён серверов в каждом, то лучшей стратегией будет простое копирование и переименование стандартного конфигурационного файла (openssl.cnf) во что-то более осмысленное, а затем использование аргумента **-config filename** для выбора каждого из файлов по мере необходимости. Количество подобных конфигурационных файлов может быть любым, просто помните их имена и предназначение.

Если Вы хотите создать сертификаты с несколькими именами сервера в методах 1 или 2, то в дополнение к приведённым выше правкам openssl.cnf добавьте это:

```
# Найдите секцию [req]
[req]
....
# Добавьте или раскомментируйте:
req_extensions = v3_req
```

При такой модификации поля subjectAltName будут безусловно добавляться в каждый CSR.

Замечания по форматам сертификатов

В качестве формата по умолчанию OpenSSL поддерживает формат почты повышенной конфиденциальности Privacy Enhanced Mail (PEM). Изначальный формат сертификатов X.509 — ASN.1 DER (бинарный формат). PEM кодирует бинарный блок DER в base64 (RFC 3548), при этом создаётся текстовая версия (набор символов ASCII/IA5) которая может, кроме всего прочего, пересылаться почтовыми системами. Такой формат также включает заголовочную и завершающую строки, каждая из которых начинается и заканчивается ровно пятью дефисами и представляет собой пригодное для чтения людьми описание того содержимого в формате base64, которое заключено между этими строками. Файлы PEM выглядят примерно так:

```
-----BEGIN CERTIFICATE-----
MIIDHDCCAowGAwIBAgIJAlt8VJt1eyicMA0GCSqGSIb3DQEBBQUAMGgxCzAJBgNV
BAYTAk1ZMQ4wDAYDVQQIEwVTVFEFURTEYMBYGA1UEChMPQ0EgQ09NUEFOWSBOQU1F
MQ0wCwYDVQQHEwRDSVRZMQ4wDAYDVQQLEwVYLjUwOTEQMA4GA1UEAxMHQ0EgUk9P
VDAeFw0wODA0MTUyMTA3MzZaFw0xODA0MTMyMTA3MzZaMGgxCzAJBgNVBAYTAk1Z
MQ4wDAYDVQQIEwVTVFEFURTEYMBYGA1UEChMPQ0EgQ09NUEFOWSBOQU1FMQ0wCwYD
VQQHEwRDSVRZMQ4wDAYDVQQLEwVYLjUwOTEQMA4GA1UEAxMHQ0EgUk9PVDCBnzAN
BgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA0YMzbKpyk0+02R90E0BAMwGSl8M5QAum
e8zdgmxkLf/AazWqyY9YZSMYV3CF6l3HLoSpvLkjLz4E+78Y15btg0P/54gdSNWz
T6YkDd4Te22bt5HZA4UH4GTBn1M7TCXQBMb+XNtj7PM1ehuazyiHWFYoEbN5ywJ
7M2SttAdOWUCAwEAAaOBzTCByjAdBgNVHQ4EFgQUVA3e4zcj/y7oAwosUv78wMgT
coAwgZoGA1UdIwSBkjCBj4AUVA3e4zcj/y7oAwosUv78wMgTcoChbKRqMGgxCzAJ
BgNVBAYTAk1ZMQ4wDAYDVQQIEwVTVFEFURTEYMBYGA1UEChMPQ0EgQ09NUEFOWSBO
QU1FMQ0wCwYDVQQHEwRDSVRZMQ4wDAYDVQQLEwVYLjUwOTEQMA4GA1UEAxMHQ0Eg
Uk9PVIIJAlt8VJt1eyicMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEFBQADgYEA
S41UTHVbVdu7Vv1Cg40oeU+bRghsSTJuZRG+M4T+sG8CEV0uDzxYYGkwDgC9a7Yb
Cfh/ea7F1El1Ym1Zj2v3wLhRl1CIRrJSdLFxPNKKoLG5FmVjY4xSGr63PFHq0rtZ
NH5lEmZybl+m2frlkjUv9KAvxclZnIFgovdu8YPMDds=
-----END CERTIFICATE-----

BLAH BLAH BLAH

-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4, ENCRYPTED
DEK-Info: DES-EDE3-CBC, 6EF6203EF1A9533A

r7LMq15wr100mMsD84KyNo+5yY2SfY5DFXhQU+lqVzjLBhHsP3Fa9xpsSgcfPxVz
El3/msvQ98BkaMihajEn5f2Ux0ZcjLCk5lnBHbmXTvOSQhwIBF0/8fUyb1CAyr7c
njh4XpwzGBjuIBKhj8uW98tV02kk/LrFf2rhL97UYNv0j9neFFVt30DTJw1Riu1N
nAZCrTmhlfkVEVZDqEEKCVGPnAiWTU8ACMUXj4eJog3eelotFuyZ1JagBpTlZFEF
f6uoSk8HBZLItWTxqRuBRVb8jqE2Bhdp9hvvjdja9XIrAPGQJ0u2QVw==
-----END RSA PRIVATE KEY-----
```

Текст BEGIN CERTIFICATE и END CERTIFICATE из примера может принимать различные значения, описывающие функциональность и формат материала, содержащегося между строками-ограничителями. В любом файле PEM может быть более одного элемента (каждый из которых ограничен последовательностью ---BEGIN -----END), а также этот файл может содержать другую информацию до, после и между такими элементами, но не внутри ограничителей. Формат PEM определён в [RFC 1421](#) для использования с S/MIME ([RFC 3850](#)).

```
# Ключи
key.pem # ключ RSA или DSA в формате PEM
key.der # ключ RSA или DSA в формате DER

# Сертификаты:
# могут содержать текст + закрытый ключ + сертификат
# в одном файле, либо ключи могут быть в отдельном файле;
# обычно, если ключ зашифрован, он должен быть в отдельном файле.
cert.pem # формат PEM
cert.der # формат DER
cert.p12 # формат PKCS12
```

Формат PKCS12 (стандарт лаборатории RSA) обычно используется браузерами:

```
# Конвертация PKCS12 в PEM:
openssl pkcs12 -clcerts -nokeys -in cert.p12 -out usercert.pem
# извлечение только сертификата

openssl pkcs12 -nocerts -in cert.p12 -out userkey.pem
# извлечение только ключа

# Конвертация PEM в PKCS12:
openssl pkcs12 -export -out cert.p12 -inkey ./userkey.pem -in ./usercert.pem
```




RFC по теме

Список RFC, относящихся к TLS и сертификатам X.509. Большинство из них упоминается в тексте этого документа. Список не полный, но мы обновляем его по мере обновления текста или при публикации подходящего RFC. Так что в конечном счёте он станет исчерпывающим.

- RFC 2986** PKCS #10: Certification Request Syntax Specification Version 1.7
PKCS #10: Спецификация синтаксиса запроса сертификации, версия 1.7
M. Nystrom, B. Kaliski. Ноябрь 2000. Формат: TXT=27794 байт, отменяет RFC2314, обновлён в RFC5967, статус: INFORMATIONAL.
- RFC 4210** Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)
Протокол управления сертификатами (CMP) инфраструктуры открытых ключей X.509 Интернет
C. Adams, S. Farrell, T. Kause, T. Mononen. Сентябрь 2005. Формат: TXT=212013 байт, отменяет RFC2510, обновлён в RFC6712, статус: PROPOSED STANDARD.
- RFC 4211** Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF).
Формат сообщения запроса сертификата (CRMF) инфраструктуры открытых ключей X.509 Интернет
J. Schaad. Сентябрь 2005. Формат: TXT=86136 байт, отменяет RFC2511, статус: PROPOSED STANDARD.
- RFC 5019** The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments
Облегчённый профиль протокола онлайн-получения статуса сертификата (OCSP) для высоко нагруженных сред
A. Deacon, R. Hurst. Сентябрь 2007. Формат: TXT=46371 байт, статус: PROPOSED STANDARD.
- RFC 5246** The Transport Layer Security (TLS) Protocol Version 1.2
Протокол TLS, версия 1.2
T. Dierks, E. Rescorla. Август 2008. Формат: TXT=222395 байт, отменяет RFC3268, RFC4346, RFC4366, обновляет RFC4492, обновлён в RFC5746, RFC5878, RFC6176, статус: PROPOSED STANDARD.
- RFC 5272** Certificate Management over CMS (CMC)
Управление сертификатами поверх CMS (CMC)
J. Schaad, M. Myers. Июнь 2008. Формат: TXT=167138 байт, отменяет RFC2797, обновлён в RFC6402, статус: PROPOSED STANDARD.
- RFC 5273** Certificate Management over CMS (CMC): Transport Protocols
Управление сертификатами поверх CMS (CMC): Транспортные протоколы
J. Schaad, M. Myers. Июнь 2008. Формат: TXT=14030 байт, обновлён в RFC6402, статус: PROPOSED STANDARD.
- RFC 5274** Certificate Management Messages over CMS (CMC): Compliance Requirements
Управление сертификатами поверх CMS (CMC): Технические требования
J. Schaad, M. Myers. Июнь 2008. Формат: TXT=27380 байт, обновлён в RFC6402, статус: PROPOSED STANDARD.
- RFC 5746** Transport Layer Security (TLS) Renegotiation Indication Extension
Расширение индикации повторных переговоров TLS
E. Rescorla, M. Ray, S. Dispensa, N. Oskov. Февраль 2010. Формат: TXT=33790 байт, обновляет RFC5246, RFC4366, RFC4347, RFC4346, RFC2246, статус: PROPOSED STANDARD.
- RFC 5878** Transport Layer Security (TLS) Authorization Extensions
Расширения авторизации TLS
M. Brown, R. Housley. Май 2010. Формат: TXT=44594 байт, обновляет RFC5246, статус: EXPERIMENTAL.
- RFC 5912** New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)
Новые модули ASN.1 для инфраструктуры открытых ключей с использованием X.509

(PKIX)

P. Hoffman, J. Schaad. Июнь 2010. Формат: TXT=216154 байт, обновлён в RFC6960, статус: INFORMATIONAL.

[RFC 5967](#)

The application/pkcs10 Media Type

Медиа-тип application/pkcs10

S. Turner. Август 2010. Формат: TXT=10928 байт, обновляет RFC2986, статус: INFORMATIONAL.

[RFC 6066](#)

Transport Layer Security (TLS) Extensions: Extension Definitions

Расширения TLS: определение расширения

D. Eastlake 3rd. Январь 2011. Формат: TXT=55079 байт, отменяет RFC4366, статус: PROPOSED STANDARD.

[RFC 6125](#)

Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)

Представление и верификация идентификационной сущности основанного на домене сервиса приложений в PKIX в контексте TLS

P. Saint-Andre, J. Hodges. Март 2011. Формат: TXT=136507 байт, статус: PROPOSED STANDARD.

[RFC 6176](#)

Prohibiting Secure Sockets Layer (SSL) Version 2.0

Запрет SSL версии 2.0

S. Turner, T. Polk. Март 2011. Формат: TXT=7642 байт, обновляет RFC2246, RFC4346, RFC5246, статус: PROPOSED STANDARD.

[RFC 6402](#)

Certificate Management over CMS (CMC) Updates

Обновления CMC

J. Schaad. Ноябрь 2011. Формат: TXT=66722 байт, обновляет RFC5272, RFC5273, RFC5274, статус: PROPOSED STANDARD.

[RFC 6712](#)

Internet X.509 Public Key Infrastructure — HTTP Transfer for the Certificate Management Protocol (CMP)

Инфраструктура открытых ключей X.509 Интернет — передача CMP по HTTP

T. Kaue, M. Peylo. Сентябрь 2012. Формат: TXT=21308 байт, обновляет RFC4210, статус: PROPOSED STANDARD.

[RFC 6960](#)

X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP

Протокол OCSP инфраструктуры открытых ключей X.509 Интернет

S. Santesson, M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams. Июнь 2013. Формат: TXT=82037 байт, отменяет RFC2560, RFC6277, обновляет RFC5912, статус: PROPOSED STANDARD.

[RFC 6961](#)

The Transport Layer Security (TLS) Multiple Certificate Status Request Extension

Расширение TLS запроса статуса нескольких сертификатов

Y. Pettersen. Июнь 2013. Формат: TXT=21473 байт, статус: PROPOSED STANDARD.

[RFC 7027](#)

Elliptic Curve Cryptography (ECC) Brainpool Curves for Transport Layer Security (TLS)

Криптография на основе эллиптических кривых (ECC): кривые Brainpool для TLS

J. Merkle, M. Lochter. Октябрь 2013. Формат: TXT=16366 байт, обновляет RFC4492, статус: INFORMATIONAL.

[RFC 7030](#)

Enrollment over Secure Transport

Регистрация поверх безопасного транспорта

M. Pritikin, Ed., P. Yee, Ed., D. Harkins, Ed.. Октябрь 2013. Формат: TXT=123989 байт, статус: PROPOSED STANDARD.

Изменения страницы

В нижней части страницы помещается корректная дата последнего изменения.

Ноябрь 2013: Страница переделана в HTML5

Ноябрь 2013: Порядок терминов на странице изменён с SSL/TLS на TLS/SSL для отображения

современных тенденций.

Ноябрь 2013: Обновлён рисунок 2.

Октябрь 2013: В список RFC добавлены RFC 7027 и 7030. Обновлён текст, касающийся ECC.



Copyright © 1994-2013 ZyTrax, Inc. Все права защищены.
Переведено участниками проекта Pro-LDAP.ru в 2011-2013 г.