Аутентификация LDAP

Аннотация

Этот документ является руководством по настройке сервера LDAP (в основном сервера OpenLDAP) для аутентификации в FreeBSD. Это полезно в ситуациях, когда множество серверов требуют одинаковых учётных записей пользователей, например, в качестве замены NIS.

Содержание

1. Предисловие 1
2. Настройка LDAP 1
3. Конфигурация клиента 6
4. Безопасность 11
Приложение A: Полезные инструменты 14
Приложение B: Сертификаты OpenSSL для LDAP 14

1. Предисловие

Этот документ предназначен для того, чтобы дать читателю достаточно знаний о LDAP для настройки сервера LDAP. В документе будет предпринята попытка объяснить использование net/nss_ldap и security/pam_ldap для клиентских машин, работающих с сервером LDAP.

По завершении читатель должен уметь настроить и развернуть сервер FreeBSD, способный размещать каталог LDAP, а также настроить и развернуть сервер FreeBSD, который может проходить аутентификацию с использованием каталога LDAP.

Эта статья не претендует на исчерпывающее описание вопросов безопасности, надёжности или лучших практик настройки LDAP и других обсуждаемых здесь сервисов. Хотя автор старается делать всё правильно, вопросы безопасности рассматриваются лишь в общих чертах. Данная статья должна рассматриваться только как теоретическая основа, а любая реальная реализация должна сопровождаться тщательным анализом требований.

2. Настройка LDAP

LDAP означает "Lightweight Directory Access Protocol" (облегчённый протокол доступа к каталогам) и является подмножеством X.500 Directory Access Protocol. Последние спецификации можно найти в RFC4510 и связанных документах. По сути, это база данных, которая предполагает более частые операции чтения, чем записи.

В примерах в этом документе будет использоваться LDAP-сервер OpenLDAP; хотя принципы, изложенные здесь, должны быть применимы к различным серверам, большая часть конкретных административных действий специфична для OpenLDAP. В портах доступно несколько версий сервера, например net/openldap26-server. Клиентские серверы потребуют соответствующие библиотеки net/openldap26-client.

Существуют (в основном) две области службы LDAP, которые требуют настройки. Первая — настройка сервера для правильного приёма соединений, а вторая — добавление записей в каталог сервера, чтобы инструменты FreeBSD знали, как с ним взаимодействовать.

2.1. Настройка сервера для подключений

Этот раздел относится конкретно к OpenLDAP. Если вы используете другой сервер, вам необходимо обратиться к документации этого сервера.

2.1.1. Установка OpenLDAP

Сначала установите OpenLDAP:

Пример 1. Установка OpenLDAP

```
# cd /usr/ports/net/openldap26-server
# make install clean
```

Это устанавливает бинарные файлы `slapd` и `slurpd`, а также необходимые библиотеки OpenLDAP.

2.1.2. Настройка OpenLDAP

Далее мы должны настроить OpenLDAP.

Вам следует требовать шифрование соединений с сервером LDAP; в противном случае пароли ваших пользователей будут передаваться в открытом виде, что считается небезопасным. Инструменты, которые мы будем использовать, поддерживают два очень похожих типа шифрования: SSL и TLS.

TLS означает "Transportation Layer Security". Службы, использующие TLS, обычно подключаются к тем же портам, что и аналогичные службы без TLS; таким образом, SMTP-сервер с поддержкой TLS будет ожидать подключений на порту 25, а LDAP-сервер — на порту 389.

SSL означает «Secure Sockets Layer», и службы, реализующие SSL, *не* работают на тех же портах, что и их аналоги без SSL. Таким образом, SMTPS использует порт 465 (а не 25), HTTPS — порт 443, а LDAPS — порт 636.

Причина, по которой SSL использует другой порт, чем TLS, заключается в том, что соединение TLS начинается как обычный текст и переключается на зашифрованный

трафик после директивы `STARTTLS`. Соединения SSL зашифрованы с самого начала. Кроме этого, существенных различий между ними нет.

Мы настроим OpenLDAP для использования TLS, так как SSL считается устаревшим.

После установки OpenLDAP через порты следующие параметры конфигурации в /usr/local/etc/openldap/slapd.conf включат TLS:

```
security ssf=128

TLSCertificateFile /path/to/your/cert.crt
TLSCertificateKeyFile /path/to/your/cert.key
TLSCACertificateFile /path/to/your/cacert.crt
```

В этом примере `ssf=128` указывает OpenLDAP требовать 128-битное шифрование для всех соединений, как для поиска, так и для обновления. Этот параметр может быть настроен в соответствии с требованиями безопасности вашего сайта, но ослаблять его требуется редко, так как большинство библиотек LDAP-клиентов поддерживают стойкое шифрование.

Файлы cert.crt, cert.key и cacert.crt необходимы для аутентификации клиентами *вас* как действительного LDAP-сервера. Если вам просто нужен работающий сервер, вы можете создать самоподписанный сертификат с помощью OpenSSL:

Пример 2. Генерация RSA-ключа

```
% openssl genrsa -out cert.key 1024
Generating RSA private key, 1024 bit long modulus
.....................++++++
...++++++
e is 65537 (0x10001)

% openssl req -new -key cert.key -out cert.csr
```

На этом этапе вам будет предложено ввести некоторые значения. Вы можете ввести любые значения, какие пожелаете; однако важно, чтобы значение "Common Name" (Общее имя) было полным доменным именем сервера OpenLDAP. В нашем случае и в приведённых примерах сервер называется *server.example.org*. Неправильное указание этого значения может привести к сбоям при подключении клиентов. Это может стать причиной серьёзных проблем, поэтому внимательно следуйте этим шагам.

Наконец, запрос на подпись сертификата должен быть подписан:

Пример 3. Самостоятельная подпись сертификата

```
% openssl x509 -req -in cert.csr -days 365 -signkey cert.key -out cert.crt
```

```
Signature ok
subject=/C=AU/ST=Some-State/O=Internet Widgits Pty Ltd
Getting Private key
```

Это создаст самоподписанный сертификат, который можно использовать для директив в slapd.conf, где cert.crt и cacert.crt представляют собой один и тот же файл. Если вы планируете использовать множество серверов OpenLDAP (для репликации через `slurpd`), рекомендуется ознакомиться с разделом Сертификаты OpenSSL для LDAP, чтобы сгенерировать ключ центра сертификации (CA) и использовать его для подписи индивидуальных сертификатов серверов.

После этого добавьте следующее в /etc/rc.conf:

```
slapd_enable="YES"
```

Затем выполните `/usr/local/etc/rc.d/slapd start`. Это должно запустить OpenLDAP. Убедитесь, что он слушает порт 389 с помощью

```
% sockstat -4 -p 389
ldap     slapd      3261  7  tcp4   *:389                 *:*
```

2.1.3. Настройка клиента

Установите порт net/openldap26-client для библиотек OpenLDAP. Клиентские машины всегда будут иметь библиотеки OpenLDAP, так как они содержатся в пакетах security/pam_ldap и net/nss_ldap, по крайней мере, на данный момент.

Файл конфигурации для библиотек OpenLDAP — это /usr/local/etc/openldap/ldap.conf. Отредактируйте этот файл, указав следующие значения:

```
base dc=example,dc=org
uri ldap://server.example.org/
ssl start_tls
tls_cacert /path/to/your/cacert.crt
```

Важно, чтобы у ваших клиентов был доступ к cacert.crt, иначе они не смогут подключиться.

Существует два файла с именем ldap.conf. Первый — это данный файл, предназначенный для библиотек OpenLDAP и определяющий, как взаимодействовать с сервером. Второй — /usr/local/etc/ldap.conf, используется pam_ldap.

На этом этапе вы должны быть в состоянии выполнить `ldapsearch -Z` на клиентской

машине; `-Z` означает "использовать TLS". Если вы столкнётесь с ошибкой, значит что-то настроено неправильно; скорее всего, проблема в ваших сертификатах. Используйте `s_client` и `s_server` из openssl(1), чтобы убедиться, что они правильно настроены и подписаны.

2.2. Записи в базе данных

Аутентификация в каталоге LDAP обычно выполняется путем попытки привязаться к каталогу как подключающийся пользователь. Это делается путем установки "простой" привязки к каталогу с предоставленным именем пользователя. Если существует запись с `uid`, равным имени пользователя, и атрибут `userPassword` этой записи совпадает с предоставленным паролем, то привязка проходит успешно.

Первое, что нам нужно сделать, — это определить, где в каталоге будут находиться наши пользователи.

Базовой записью для нашей базы данных является `dc=example,dc=org`. Стандартное расположение пользователей, которое ожидает большинство клиентов, выглядит примерно как `ou=people,base`, поэтому здесь будет использоваться именно это. Однако следует помнить, что это настраивается.

Таким образом, запись ldif для организационного подразделения `people` будет выглядеть следующим образом:

```
dn: ou=people,dc=example,dc=org
objectClass: top
objectClass: organizationalUnit
ou: people
```

Все пользователи будут созданы как подразделы этой организационной единицы.

Некоторые размышления стоит посвятить классу объектов, к которому будут принадлежать ваши пользователи. Большинство инструментов по умолчанию используют `people`, что подходит, если вам просто нужно предоставить записи для аутентификации. Однако, если вы также собираетесь хранить информацию о пользователях в базе данных LDAP, вам, вероятно, стоит использовать `inetOrgPerson`, который имеет множество полезных атрибутов. В любом случае, соответствующие схемы должны быть загружены в slapd.conf.

Для этого примера мы будем использовать класс объекта `person`. Если вы используете `inetOrgPerson`, шаги практически идентичны, за исключением того, что атрибут `sn` является обязательным.

Чтобы добавить тестового пользователя с именем `tuser`, ldif-файл должен выглядеть так:

```
dn: uid=tuser,ou=people,dc=example,dc=org
objectClass: person
objectClass: posixAccount
objectClass: shadowAccount
```

```
objectClass: top
uidNumber: 10000
gidNumber: 10000
homeDirectory: /home/tuser
loginShell: /bin/csh
uid: tuser
cn: tuser
```

Я начинаю UID пользователей LDAP с 10000, чтобы избежать конфликтов с системными учётными записями; вы можете настроить любое число здесь, при условии что оно меньше 65536.

Также необходимы записи групп. Они настраиваются так же, как записи пользователей, но мы будем использовать стандартные значения, приведённые ниже:

```
dn: ou=groups,dc=example,dc=org
objectClass: top
objectClass: organizationalUnit
ou: groups

dn: cn=tuser,ou=groups,dc=example,dc=org
objectClass: posixGroup
objectClass: top
gidNumber: 10000
cn: tuser
```

Чтобы добавить их в вашу базу данных, вы можете использовать `slapadd` или `ldapadd` для файла, содержащего эти записи. Или вы можете использовать sysutils/ldapvi.

Утилита `ldapsearch` на клиентской машине теперь должна возвращать эти записи. Если это так, ваша база данных правильно настроена для использования в качестве сервера аутентификации LDAP.

3. Конфигурация клиента

Клиент уже должен иметь библиотеки OpenLDAP из Настройка клиента, но если вы устанавливаете несколько клиентских машин, вам потребуется установить net/openldap26-client на каждой из них.

Для аутентификации на сервере LDAP в FreeBSD необходимо установить два порта: security/pam_ldap и net/nss_ldap.

3.1. Аутентификация

security/pam_ldap настраивается через /usr/local/etc/ldap.conf.

Это *другой файл*, отличный от файла конфигурации библиотечных функций

> OpenLDAP, /usr/local/etc/openldap/ldap.conf; однако он принимает многие из тех же параметров; фактически он является надмножеством этого файла. В оставшейся части этого раздела ссылки на ldap.conf будут означать /usr/local/etc/ldap.conf.

Таким образом, мы захотим скопировать все исходные параметры конфигурации из openldap/ldap.conf в новый ldap.conf. После этого нам нужно указать security/pam_ldap, что искать на сервере каталогов.

Мы идентифицируем наших пользователей с помощью атрибута `uid`. Чтобы настроить это (хотя это значение по умолчанию), укажите директиву `pam_login_attribute` в ldap.conf:

Пример 4. Установка `pam_login_attribute`

```
pam_login_attribute uid
```

С таким набором, security/pam_ldap будет искать во всей LDAP-директории под `base` значение `uid=имя_пользователя`. Если оно найдёт одну и только одну запись, будет предпринята попытка привязаться к этой записи с использованием предоставленного пароля. Если привязка произойдёт успешно, доступ будет разрешён. В противном случае доступ будет запрещён.

Пользователи, чья оболочка не указана в /etc/shells, не смогут войти в систему. Это особенно важно, когда Bash установлен как оболочка пользователя на сервере LDAP. Bash не входит в стандартную установку FreeBSD. При установке из пакета или порта, он располагается в /usr/local/bin/bash. Убедитесь, что путь к оболочке на сервере указан правильно:

```
% getent passwd username
```

Если в последнем столбце вывода указан `/bin/bash`, есть два варианта действий. Первый — изменить запись пользователя на сервере LDAP на /usr/local/bin/bash. Второй вариант — создать символическую ссылку на клиентском компьютере LDAP, чтобы Bash находился в правильном месте:

```
# ln -s /usr/local/bin/bash /bin/bash
```

Убедитесь, что файл /etc/shells содержит записи для `/usr/local/bin/bash` и `/bin/bash`. После этого пользователь сможет входить в систему с Bash в качестве оболочки.

3.1.1. PAM

PAM, что означает "Pluggable Authentication Modules" (подключаемые модули аутентификации), является методом, с помощью которого FreeBSD аутентифицирует большинство своих сеансов. Чтобы указать FreeBSD, что мы хотим использовать LDAP-сервер, нам необходимо добавить строку в соответствующий файл PAM.

В большинстве случаев подходящий файл PAM — это /etc/pam.d/sshd, если вы хотите использовать SSH (не забудьте установить соответствующие параметры в /etc/ssh/sshd_config, иначе SSH не будет использовать PAM).

Чтобы использовать PAM для аутентификации, добавьте строку

```
auth  sufficient  /usr/local/lib/pam_ldap.so  no_warn
```

Строгое место, где эта строка появляется в файле, и какие опции указаны в четвертом столбце, определяют точное поведение механизма аутентификации; см. pam(d)

С такой конфигурацией вы сможете аутентифицировать пользователя в LDAP-каталоге. PAM выполнит привязку с вашими учётными данными, и в случае успеха сообщит SSH разрешить доступ.

Однако разрешать *каждому* пользователю в каталоге доступ к *каждой* клиентской машине — не лучшая идея. При текущей конфигурации пользователю для входа на машину достаточно наличия записи в LDAP. К счастью, есть несколько способов ограничить доступ пользователей.

ldap.conf поддерживает директиву `pam_groupdn`; каждая учётная запись, подключающаяся к этой машине, должна быть членом группы, указанной здесь. Например, если у вас есть

```
pam_groupdn cn=servername,ou=accessgroups,dc=example,dc=org
```

в ldap.conf, тогда только члены этой группы смогут войти в систему. Однако есть несколько моментов, которые следует учитывать.

Участники этой группы указываются в одном или нескольких атрибутах `memberUid`, и каждый атрибут должен содержать полное отличающееся имя участника. Таким образом, `memberUid: someuser` не сработает; необходимо указать:

```
memberUid: uid=someuser,ou=people,dc=example,dc=org
```

Кроме того, эта директива не проверяется в PAM во время аутентификации, она проверяется во время управления учётными записями, поэтому вам понадобится вторая строка в ваших PAM-файлах в разделе `account`. Это, в свою очередь, потребует, чтобы *каждый* пользователь был указан в группе, что не обязательно соответствует нашим желаниям. Чтобы избежать блокировки пользователей, не находящихся в LDAP, следует включить атрибут `ignore_unknown_user`. Наконец, следует установить опцию `ignore_authinfo_unavail`, чтобы не оказаться заблокированным на каждом компьютере при недоступности LDAP-сервера.

Ваш файл pam.d/sshd может в итоге выглядеть так:

Пример 5. Пример pam.d/sshd

```
auth            required        pam_nologin.so          no_warn
auth            sufficient      pam_opie.so             no_warn no_fake_prompts
auth            requisite       pam_opieaccess.so       no_warn allow_local
auth            sufficient      /usr/local/lib/pam_ldap.so      no_warn
auth            required        pam_unix.so             no_warn try_first_pass

account         required        pam_login_access.so
account         required        /usr/local/lib/pam_ldap.so      no_warn
ignore_authinfo_unavail ignore_unknown_user
```

Поскольку мы добавляем эти строки конкретно в pam.d/sshd, это повлияет только на SSH-сеансы. Пользователи LDAP не смогут войти через консоль. Чтобы изменить это поведение, изучите остальные файлы в /etc/pam.d и измените их соответствующим образом.

3.2. Name Service Switch

NSS — это служба, которая сопоставляет атрибуты с именами. Например, если файл принадлежит пользователю `1001`, приложение запросит у NSS имя для `1001` и может получить `bob`, `ted` или любое другое имя пользователя.

Теперь, когда информация о пользователях хранится в LDAP, необходимо указать NSS обращаться туда при запросах.

Порт net/nss_ldap выполняет это. Он использует тот же конфигурационный файл, что и security/pam_ldap, и после установки не должен требовать дополнительных параметров. Вместо этого остаётся просто отредактировать /etc/nsswitch.conf, чтобы воспользоваться преимуществами каталога. Просто замените следующие строки:

```
group: compat
passwd: compat
```

строкой

```
group: files ldap
passwd: files ldap
```

Это позволит сопоставлять имена пользователей с UID и UID с именами пользователей.

Поздравляем! Теперь у вас должна работать аутентификация через LDAP.

3.3. Предостережения

К сожалению, на момент написания этой документации FreeBSD не поддерживал изменение паролей пользователей с помощью passwd(1). В результате большинству администраторов приходится самостоятельно реализовывать решение. Здесь я привожу несколько примеров. Обратите внимание, что если вы пишете собственный скрипт для смены пароля, вам следует учитывать некоторые проблемы безопасности; см. Хранение паролей

Пример 6. Скрипт оболочки для изменения паролей

```
#!/bin/sh

stty -echo
read -p "Old Password: " oldp; echo
read -p "New Password: " np1; echo
read -p "Retype New Password: " np2; echo
stty echo

if [ "$np1" != "$np2" ]; then
  echo "Passwords do not match."
  exit 1
fi

ldappasswd -D uid="$USER",ou=people,dc=example,dc=org \
  -w "$oldp" \
  -a "$oldp" \
  -s "$np1"
```

Этот скрипт почти не проверяет ошибки, что более важно, он очень беспечен в том, как хранит ваши пароли. Если вы делаете что-то подобное, по крайней мере измените значение sysctl `security.bsd.see_other_uids`:

```
# sysctl security.bsd.see_other_uids=0
```

Более гибкий (и, вероятно, более безопасный) подход может быть реализован путем написания собственной программы или даже веб-интерфейса. Ниже приведена часть библиотеки на Ruby, которая позволяет изменять пароли в LDAP. Она используется как в командной строке, так и в вебе.

Пример 7. Ruby-скрипт для изменения паролей

```
require 'ldap'
require 'base64'
require 'digest'
```

```
require 'password' # ruby-password

ldap_server = "ldap.example.org"
luser = "uid=#{ENV['USER']},ou=people,dc=example,dc=org"

# get the new password, check it, and create a salted hash from it
def get_password
  pwd1 = Password.get("New Password: ")
  pwd2 = Password.get("Retype New Password: ")

  raise if pwd1 != pwd2
  pwd1.check # check password strength

  salt = rand.to_s.gsub(/0\./, '')
  pass = pwd1.to_s
  hash =
"{SSHA}"+Base64.encode64(Digest::SHA1.digest("#{pass}#{salt}")+salt).chomp!
  return hash
end

oldp = Password.get("Old Password: ")
newp = get_password

# We'll just replace it.  That we can bind proves that we either know
# the old password or are an admin.

replace = LDAP::Mod.new(LDAP::LDAP_MOD_REPLACE | LDAP::LDAP_MOD_BVALUES,
                        "userPassword",
                        [newp])

conn = LDAP::SSLConn.new(ldap_server, 389, true)
conn.set_option(LDAP::LDAP_OPT_PROTOCOL_VERSION, 3)
conn.bind(luser, oldp)
conn.modify(luser, [replace])
```

Хотя это не гарантирует отсутствия уязвимостей (например, пароль хранится в памяти), данный подход чище и гибче, чем простой скрипт на sh.

4. Безопасность

Теперь, когда ваши машины (и, возможно, другие службы) проходят аутентификацию через ваш LDAP-сервер, этот сервер необходимо защитить как минимум так же, как /etc/master.passwd на обычном сервере, а возможно, и лучше, поскольку сломанный или взломанный LDAP-сервер нарушит работу всех клиентских служб.

Помните, что этот раздел не является исчерпывающим. Вам следует постоянно пересматривать свою конфигурацию и процедуры для улучшений.

4.1. Установка атрибутов только для чтения

Некоторые атрибуты в LDAP должны быть доступны только для чтения. Если оставить их доступными для записи пользователем, например, пользователь может изменить свой атрибут `uidNumber` на `0` и получить доступ `root`!

Для начала атрибут `userPassword` не должен быть доступен для чтения всем. По умолчанию любой, кто может подключиться к LDAP-серверу, может читать этот атрибут. Чтобы отключить это, добавьте следующее в slapd.conf:

Пример 8. Скрыть пароли

```
access to dn.subtree="ou=people,dc=example,dc=org"
  attrs=userPassword
  by self write
  by anonymous auth
  by * none

access to *
  by self write
  by * read
```

Это запретит чтение атрибута `userPassword`, но позволит пользователям изменять свои собственные пароли.

Кроме того, следует запретить пользователям изменять некоторые из своих атрибутов. По умолчанию пользователи могут изменять любые атрибуты (за исключением тех, которые сами схемы LDAP запрещают изменять), такие как `uidNumber`. Чтобы устранить эту уязвимость, измените приведённое выше на

Пример 9. Атрибуты только для чтения

```
access to dn.subtree="ou=people,dc=example,dc=org"
  attrs=userPassword
  by self write
  by anonymous auth
  by * none

access to attrs=homeDirectory,uidNumber,gidNumber
  by * read

access to *
  by self write
  by * read
```

Это предотвратит возможность пользователей маскироваться под других пользователей.

4.2. Определение учётной записи `root`

Часто учётная запись `root` или администратора для службы LDAP будет определена в файле конфигурации. Например, OpenLDAP поддерживает это, и это работает, но может привести к проблемам, если файл slapd.conf будет скомпрометирован. Возможно, лучше использовать это только для первоначальной настройки доступа к LDAP, а затем определить учётную запись `root` там.

Еще лучше определить учётные записи с ограниченными правами и полностью исключить учётную запись `root`. Например, пользователи, которые могут добавлять или удалять учётные записи, добавляются в одну группу, но сами не могут изменять состав этой группы. Такая политика безопасности поможет снизить последствия утечки пароля.

4.2.1. Создание группы управления

Предположим, вы хотите, чтобы ваш IT-отдел мог изменять домашние каталоги пользователей, но не хотите, чтобы все они могли добавлять или удалять пользователей. Способ сделать это — добавить группу для этих администраторов:

Пример 10. Создание группы управления

```
dn: cn=homemanagement,dc=example,dc=org
objectClass: top
objectClass: posixGroup
cn: homemanagement
gidNumber: 121 # required for posixGroup
memberUid: uid=tuser,ou=people,dc=example,dc=org
memberUid: uid=user2,ou=people,dc=example,dc=org
```

И затем измените атрибуты прав в slapd.conf:

Пример 11. ACL для группы управления домашним каталогом

```
access to dn.subtree="ou=people,dc=example,dc=org"
  attr=homeDirectory
  by dn="cn=homemanagement,dc=example,dc=org"
  dnattr=memberUid write
```

Теперь `tuser` и `user2` могут изменять домашние каталоги других пользователей.

В этом примере мы предоставили подмножество административных полномочий определённым пользователям, не давая им власти в других областях. Идея заключается в том, чтобы вскоре ни одна учётная запись пользователя не обладала полномочиями учётной записи `root`, но каждое полномочие, которое имел root, было бы у хотя бы одного пользователя. Учётная запись `root` тогда становится ненужной и может быть удалена.

4.3. Хранение паролей

По умолчанию OpenLDAP сохраняет значение атрибута `userPassword` так же, как и любые другие данные: в открытом виде. В большинстве случаев оно кодируется в base64, что обеспечивает достаточную защиту, чтобы честный администратор не узнал ваш пароль, но не более того.

Хорошей идеей является хранение паролей в более безопасном формате, таком как SSHA (солёный SHA). Это выполняется любой программой, которую вы используете для изменения паролей пользователей.

Приложение A: Полезные инструменты

Вот несколько других программ, которые могут быть полезны, особенно если у вас много пользователей и вы не хотите настраивать всё вручную.

security/pam_mkhomedir — это модуль PAM, который всегда завершается успешно; его назначение — создавать домашние каталоги для пользователей, у которых их нет. Если у вас есть десятки серверов-клиентов и сотни пользователей, гораздо удобнее использовать этот модуль и настроить скелетные каталоги, чем подготавливать каждый домашний каталог вручную.

sysutils/ldapvi — это отличная утилита для редактирования значений LDAP в синтаксисе, подобном LDIF. Каталог (или его подраздел) отображается в редакторе, выбранном переменной окружения `EDITOR`. Это позволяет легко вносить масштабные изменения в каталог без необходимости написания собственного инструмента.

security/openssh-portable обладает возможностью обращаться к серверу LDAP для проверки SSH-ключей. Это очень удобно, если у вас много серверов и вы не хотите копировать свои открытые ключи на все из них.

Приложение B: Сертификаты OpenSSL для LDAP

Если вы размещаете два или более LDAP-серверов, вам, вероятно, не захочется использовать самоподписанные сертификаты, так как каждый клиент придется настраивать для работы с каждым сертификатом. Хотя это возможно, это не так просто, как создать собственный центр сертификации и подписать сертификаты ваших серверов с его помощью.

Шаги здесь представлены как есть, без попыток объяснить происходящее — дополнительные пояснения можно найти в openssl(1) и связанных справочных страницах.

Для создания центра сертификации нам просто нужны самоподписанный сертификат и ключ. Шаги для этого следующие

Пример 12. Создание сертификата

```
% openssl genrsa -out root.key 1024
% openssl req -new -key root.key -out root.csr
% openssl x509 -req -days 1024 -in root.csr -signkey root.key -out root.crt
```

Это будут корневой ключ и сертификат ЦС. Возможно, вы захотите зашифровать ключ и хранить его в прохладном, сухом месте; любой, кто получит к нему доступ, сможет выдавать себя за один из ваших LDAP-серверов.

Затем, используя первые два шага, описанные выше, создайте ключ ldap-server-one.key и запрос на подпись сертификата ldap-server-one.csr. После подписания запроса с помощью root.key вы сможете использовать ldap-server-one.* на своих LDAP-серверах.

Не забудьте использовать полное доменное имя для атрибута "common name" при создании запроса на подпись сертификата; в противном случае клиенты будут отклонять соединение с вами, и диагностика этой проблемы может быть очень сложной.

Для подписи ключа используйте `-CA` и `-CAkey` вместо `-signkey`:

Пример 13. Подписывать как Центр Сертификации

```
% openssl x509 -req -days 1024 \
-in ldap-server-one.csr -CA root.crt -CAkey root.key \
-out ldap-server-one.crt
```

Результирующий файл будет представлять собой сертификат, который можно использовать на ваших LDAP-серверах.

Наконец, чтобы клиенты доверяли всем вашим серверам, распространите файл root.crt (*сертификат*, а не ключ!) на каждом клиенте и укажите его в директиве `TLSCACertificateFile` в файле ldap.conf.