Администрирование Linux. Часть5.

Пособие для дистанционных курсов.
http://linuxnavigator.ru.

21.07.2009

Оглавление

СОЗДАНИЕ ЦЕНТРА СЕРТИФИКАЦИИ.

Сертификаты нам потребуются для нормальной работы почты, OpenVPN и многих других сервисов. Мы поднимем свой собственный центр сертификации. Для наших задач совсем не надо подписывать сертификаты в коммерческих центрах, 100 зеленых енотов за один сертификат в год (или сколько они сейчас стоят?), это много.

Я не буду подробно описывать весь функционал программы ssl, которую мы будем использовать для создания сертификатов. Это отдельная тема, очень большая тема. Если вас заинтересует более подробное изучение этого вопроса — читайте man-ы.

Если посмотреть на программу ssl, при её вызове всегда указывается команда и параметры этой команды. Мы будем пользоваться только тремя командами:

req — запросы на создание сертификатов.
ca — управление центром сертификации.
dhparam — создание и управление параметрами Diffie-Hellman.

После указания команды идут её параметры. Параметров много, и изучать их мы будем постепенно.

Сначала мы создадим корневой сертификат нашего сервера сертификации. Затем для каждого сервера или службы мы будем выдавать свой собственный сертификат и подписывать его своим же ключиком.

ИЗМЕНЕНИЯ В КОНФИГУРАЦИОННОМ ФАЙЛЕ OPENSSL.CONF

В CentOS 5 конфигурационный файл openssl находится в директории */etc/pki/tls*. Перейдём в эту директорию и отредактируем файл.

```
# cd /etc/pki/tls
#
```

Откроем на редактирование файл *openssl.conf*. Что бы в дальнейшем не писать много дополнительных параметров в командной строке, в конце файла добавим две секции server и client.

```
[ server ]
basicConstraints=CA:FALSE
nsCertType = server
nsComment  = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always


[ client ]
basicConstraints=CA:FALSE
nsCertType  = client
nsComment   = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
```

Первая секция описывает генерацию ключей для серверов, вторая для клиентов. Тут не определяется, какой конкретно клиент или сервер описывается. При желании можно добавить отдельные секции для каждого типа сертификатов, которые вы собираетесь выписывать.

Найдите в файле секцию *[req_distinguished_name]* и замените значения по умолчанию. В моем случае эта секция выглядит так:

```
[ req_distinguished_name ]
countryName                 = Country Name (2 letter code)
countryName_default         = RU
```

```
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name
stateOrProvinceName_default     = Moscow

localityName                    = Locality Name (eg, city)
localityName_default            = Moscow

0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = Student server

organizationalUnitName          = Organizational Unit Name
organizationalUnitName_default  = Student server

commonName = Common Name (eg, your name or your server\'s hostname)
commonName_max                  = 64

emailAddress                    = Email Address
emailAddress_max                = 64
emailAddress_default            = student@st1.kryukov.biz
```

В этом месте вы подставите свои значении. Делается это для удобства, что бы потом каждый раз не вводить параметры.

Принцип написания параметров очень простой. Рассмотрим поле *emailAddress*.

```
emailAddress                    = Email Address
emailAddress_max                = 64
emailAddress_default            = student@st1.kryukov.biz
```

Когда вы запустите программу, она отобразит следующую информацию:

```
Email Address [student@st1.kryukov.biz]:
```

Первый параметр описывает название, которое будет выводиться. Второй — значение по умолчанию. Третий — максимальный размер.

Ну и ещё одно небольшое изменения в секции *[CA_default]*:

```
dir             = ../CA
certificate     = $dir/CA.crt
private_key     = $dir/private/CA.key
RANDFILE        = /var/log/messages
```

В этой секции мы определяем параметры корневого сертификата нашего центра сертификации.

Обратите внимание на параметр *RANDFILE*. Обычно его делают равным */dev/random* или */dev/urandom*. Но, по моему мнению, лучше пользоваться каким либо, часто изменяем файлом журнальной регистрации. Раз в сутки файл */var/log/message* подвергается ротации и содержимое этого файла не повторяется.

В параметре dir мы используем относительный путь. Т.е. все действия, которые мы будем производить, следует делать строго в директории */etc/pki/tls*.

ДОПОЛНИТЕЛЬНЫЕ ФАЙЛЫ.

Для работы нам потребуются несколько дополнительных файлов и директорий.

```
# touch ../CA/index.txt
# echo 01 > ../CA/serial
# mkdir ../CA/newcerts
```

СОЗДАНИЕ СЕРТИФИКАТА ЦЕНТРА СЕРТИФИКАЦИИ.

Сейчас мы создадим ключ и сертификат нашего центра сертификации. Делается это очень просто:

```
# openssl req -days 3650 -nodes -new -x509 –keyout \
../CA/private/CA.key -out ../CA/CA.crt
```

Программа задаст несколько вопросов. На все вопросы нажимайте *Enter*. Таким образом мы согласимся со значениями по умолчании. На все кроме *Common Name*, тут вы должны ввести имя своего сервера, например, *st1.kryukov.biz*.

Сейчас мы не будем шифровать полученый ключ (параметр –nodes), мы это делаем для упрощения дальнейших действий. Если параметр –nodes не указывать, при генерации ключа, вас попросят ввести пароль. Затем, при любом использовании ключа, например для подписи сертификатов, вас всегда будут спрашивать этот пароль.

Очень важно понимать, что если вы забудете пароль на ключ — вы не сможете пользоваться этим ключем. И вам придется переделать все сертифкаты вашей компании!

Параметр *–days* определяет, сколько дней будет действителен сертификат. В нашем случае мы выписываем сертификат на десять лет. Для корневого сертифката это нормально.

Пареметр *–new* говорит, что будет генерироваться новый запрос на сертификат.

Параметр *–x509* говорит программе, что вместо файла запроса на сертифкат сгенерировать самоподписанный сертификат. Что нам в итоге и надо.

Параметр *–keyout* определяет файл, в который будет помещен ключ.

Перметр *–out* определяет файл, в который будет помещен сертификат.

Обязательно закрываем доступ к секретному ключу.

```
# chmod 600 ../CA/private/CA.key
```

СОЗДАНИЕ СЕРТИФИКАТА СЕРВЕРА.

Сейчас мы создадим универсальный сертификат для различных программа, которые будут работать на нашем сервере. Конечно, можно для кажого сервиса выписывать отдельный сертификат: для WEB сервера, почтового сервера, VPN сервера и т.д. Но если вы владеете центром сертификации (CA), то выписывать отдельные сертификаты дл каждого сервиса не обязательно.

Создаём запрос на сертификацию:

```
# openssl req -days 3650 -nodes -new -keyout \
private/st1.kryukov.biz.key –out \
certs/st1.kryukov.biz.csr -extensions server
```

На все вопросы отвечаем по умолчанию — *Enter*. Кроме вопросов:

Common Name — тут вводим имя вашего сервера, например, *st1.kryukov.biz*.

A challenge password — вводим пароль.

Большинство параметров вам уже знакомы. Единственный новый параметр *–extensions*. В качестве аргумента ему указывают раздел в конфигурационном файле, который необходимо использовать программе.

```
[ server ]
basicConstraints=CA:FALSE
nsCertType = server
nsComment  = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
```

Теперь подпишем полученный запрос на сертификацию:

```
# openssl ca -days 3650 -out certs/st1.kryukov.biz.pem \
-in certs/st1.kryukov.biz.csr -extensions server
```

На все вопросы отвечаем *y*.

Закрываем доступ к файлу ключа:

```
# chmod -w private/st1.kryukov.biz.key
```

Удаляем файл запроса на сертификацию:

```
# rm certs/st1.kryukov.biz.csr
```

В результате мы получили:

/etc/pki/CA/private/CA.key — приватный ключ центра сертификации. Никому не даем, прячем ото всех!
/etc/pki/CA/CA.crt — публичный сертификат центра сертификации.
/etc/pki/tls/private/st1.kryukov.biz.key — приватный ключ нашего сервера. Прячем ото всех!
/etc/pki/tls/certs/st1.kryukov.biz.pem — сертификат нашего сервера.

Для наших задач этого достаточно. В дальнейшем, максимум, что нам потребуется — выписывать ключи для клиентов OpenVPN. Ключ для клиентов генерируется точно так же как и для сервера, единственное исключение — *-extensions client*.

На одном из следующих занятий мы посмотрим на программу оболочку, упрощающую работу с сертификатами.