Создание самоподписанного сертификата для замены SSL-сертификата веб-интерфейса

Окружение

- **Версия РЕД ВИРТ:** 7.3

Современные браузеры уже несколько лет игнорируют поле CN в Subject при проверке соответствия имени сайта. Они смотрят исключительно на расширение Subject Alternative Name (SAN). Поэтому при замене или выпуске нового сертификата необходимо убедиться, что в нём присутствуют все доменные имена и IP-адреса, для которых будет использоваться сертификат.

Описание стенда

Создание сертификата с параметром SAN

Результат

Описание стенда

Для выпуска сертификата будет использоваться:

- Имя HostedEngine (имя хоста для конфигурации Standalone) - https://engine.mydomain.local.
- Имя домена - mydomain.local.
- IP-адрес HostedEngine (IP-адрес хоста для конфигурации Standalone) - 123.123.123.123.

Создание сертификата с параметром SAN

Для создания сертификата с параметром SAN необходимо выполнить следующие действия:

1. Сгенерируйте закрытый ключ /tmp/apache.key без пароля:

```
openssl genrsa -out /tmp/apache.key 2048
```

2. Создайте файл /tmp/san.cnf:

```
vi /tmp/san.cnf
```

3. Внесите в него параметры, которые будут использоваться при создании запроса на подпись сертификата (CSR).

   Пример содержимого:

```
[ req ]
default_bits       = 2048
distinguished_name  = req_distinguished_name
req_extensions     = san_req
prompt             = no

[ req_distinguished_name ]
C = RU                  # ваша страна
ST = Moscow             # регион
L = Moscow              # город
O = My Company          # организация
OU = IT                 # подразделение
CN = mydomain.local     # основное доменное имя

[ san_req ]
subjectAltName = DNS:engine.mydomain.local, DNS:mydomain.local,
IP:123.123.123.123 # все имена и адреса
```

   где:

   - [req] - общие настройки запроса;
   - default_bits - длина генерируемого закрытого ключа RSA в битах;
   - distinguished_name - указывает, в каком разделе конфигурационного файла находятся поля о владельце сертификата;
   - req_extensions - указывает, в каком разделе перечислены расширения запроса;
   - prompt - отключает интерактивный режим запроса данных;
   - [req_distinguished_name] - поля отличительного имени (владелец сертификата);
   - C - Country, необязательное поле, двухбуквенный код страны;
   - ST - State, необязательное поле, область или регион;
   - L - Locality, необязательное поле, город или населённый пункт;
   - O - Organization, необязательное поле, название организации;
   - OU - Organizational Unit, необязательное поле, подразделение внутри организации;
   - CN - Common Name, обязательное поле, основное доменное имя, для которого выпускается сертификат;

- [san_req] - альтернативные имена субъекта (SAN);
- subjectAltName - определяет список Subject Alternative Names, дополнительных доменных имён и IP-адресов, для которых сертификат также будет считаться действительным.

> **Примечание:**
>
> В параметрах о владельце сертификата в разделе [req_distinguished_name] необходимо написать значения в кавычках для случаев, когда используются спецсимволы. Для пробелов заключать в кавычки необязательно, но рекомендуется.
>
> Пример:
>
> ```
> O = My Company # можно, но не рекомендуется
> O = "My Company" # рекомендуется, но не обязательно
> O = "My # Company" # при спецсимволах кавычки обязательны
> ```

4. Создайте запрос на сертификат (CSR) с флагом:

   ```
   openssl req -new -key /tmp/apache.key -out /tmp/apache.csr -config /tmp/san.cnf
   ```

   где:

   - req – команда создания сертификатов или запросов на сертификаты;
   - -new - флаг, указывающий, что нужно создать новый CSR;
   - -key /tmp/apache.key - путь к файлу закрытого ключа, который будет использован для подписи CSR;
   - -out /tmp/apache.csr - путь к файлу, в который будет сохранён сгенерированный CSR;
   - -config /tmp/san.cnf - путь к конфигурационному файлу, из которого OpenSSL берёт настройки.

5. Сгенерируйте закрытый ключ для ЦС:

   ```
   openssl genrsa -out /tmp/ca.key 2048
   ```

6. Создайте самоподписанный корневой сертификат:

   > **Примечание:**

Указана одна команда, не выполнять как отдельные строчки

```
openssl req -x509 -new -nodes -key /tmp/ca.key -sha256 -days 3650 -subj "/C=RU/ST=Moscow/L=Moscow/O=My Company CA/CN=My Root CA" -out /tmp/ca-cert-new.pem
```

где:

- req – команда создания сертификатов или запросов на сертификаты;
- -x509 – создание именно сертификата, а не запроса;
- -new – новый сертификат (потребуется ввести значения некоторых полей в сертификате);
- -nodes — не шифрует закрытый ключ (не требует парольной фразы);
- -key /tmp/ca.key – используемый закрытый ключ, из которого следует создать открытый;
- -sha256 – алгоритм хеширования для создания подписи ключа;
- -days 3650 – выпуск сертификата на 10 лет;
- -subj "..." - задаёт поля владельца (страна, регион, город, организация, общее имя) без интерактивного ввода;
- -out /tmp/ca-cert-new.pem – получаемый сертификат.

Параметры владельца сертификата ("/C=RU/ST=Moscow/L=Moscow/O=My Company CA/CN=My Root CA") указываются относительно вашей компании или могут не указываться. В случае если был выбран вариант не указывать, тогда стоит использовать вместо -subj "/C=RU/ST=Moscow/L=Moscow/O=My Company CA/CN=My Root CA" вариант с явным указанием пропусков:

```
-subj "/C=/ST=/L=/O=/CN=My Root CA"
```

или только с полем Common Name:

```
-subj "/CN=My Root CA"
```

ВАЖНО!

Основное имя **/CN** рекомендуется оставить, так как отсутствие поля не дает многим системам правильно воспринимать сертификат.

7. Подпишите запрос на выпуск сертификата (CSR) своим корневым сертификатом центра сертификации (ЦС):

Примечание:

Указана одна команда, не выполнять как отдельные строчки

```
openssl x509 -req -in /tmp/apache.csr -CA /tmp/ca-cert-new.pem -CAkey /tmp/ca.key -CAcreateserial -out /tmp/apache.cer -days 3650 -sha256 -extfile /tmp/san.cnf -extensions san_req
```

где:

- openssl x509 - утилита OpenSSL для работы с сертификатами в формате X.509;
- -req - указывает, что входной файл является запросом на подпись (CSR), а не готовым сертификатом. Без этого флага команда ожидала бы на входе сертификат;
- -in /tmp/apache.csr - путь к файлу CSR, который нужно подписать;
- -CA /tmp/ca-cert-new.pem - файл сертификата удостоверяющего центра, он будет использован как образец для подписи;
- -CAkey /tmp/ca.key - закрытый ключ CA, соответствующий сертификату, указанному в -CA. Именно этим ключом подписывается конечный сертификат;
- -CAcreateserial - автоматически создаёт файл серийного номера (обычно ca-cert-new.srl), если он не существует. В этом файле хранится следующий доступный серийный номер для подписываемых сертификатов. Без этого параметра нужно было бы вручную задать серийный номер через -CAserial;
- -out /tmp/apache.cer - путь к файлу, куда будет сохранён подписанный сертификат;
- -days 3650 - срок действия сертификата в днях (здесь 10 лет). Отсчитывается от текущей даты;
- -sha256 - алгоритм хэширования, используемый для подписи;
- -extfile /tmp/san.cnf - файл конфигурации, содержащий расширения для сертификата. Если не указать, расширения не будут добавлены;
- -extensions san_req - название раздела внутри файла san.cnf, где описаны нужные расширения. Без этого параметра OpenSSL ищет раздел по умолчанию (v3_req или req_ext).

Результат

Был создан следующий список файлов, которые потребуются для замены SSL-сертификата для веб-портала РЕД ВИРТ:

- /tmp/ca-cert-new.pem - корневой сертификат удостоверяющего центра. Устанавливается в корневые центры операционных систем или в хранилище сертификатов браузера .
- /tmp/apache.key - закрытый ключ сервера Apache. Хранится строго на сервере и никогда не передаётся, используется для расшифровки входящего TLS-трафика и для подписи CSR при создании запроса.
- /tmp/apache.cer - подписанный сертификат для веб-сервера. Этот сертификат устанавливается на веб-сервер и предъявляется клиентам при подключении по HTTPS, содержит открытый ключ, соответствующий apache.key, а также информацию о владельце и все альтернативные имена, указанные в san.cnf.

Источник: https://redvirt.red-soft.ru/base/poleznye-stati/red-virt-7-3/sozdanie-samopodpisannogo-sertifikata-dlya-zameny-ssl-sertifikata-veb-interfeysa-7/