Сертификаты и TLS

HTTPS — это надстройка над протоколом HTTP, дополнительный слой шифрования под названием TLS/SSL или просто SSL.
Протокол HTTPS позволяет нам зашифровать наше соединение, сделав его безопасным, а также удостовериться, что сервер, с которым общается наш браузер, — это действительно тот, за кого себя выдаёт.

Протокол HTTP работает так: клиент отправлял запрос серверу и получал ответы. В HTTPS появляется дополнительный этап — рукопожатие (handshake). Он нужен для того, чтобы установить защищённое соединение.
Рукопожатие происходит в несколько этапов.

1. Приветствие. Клиент и сервер обмениваются информацией о том, какие версии SSL и алгоритмы шифрования они поддерживают.
2. Далее сервер отправляет клиенту открытый ключ и сертификат. **Сертификат** — это сущность, которая связывает открытый ключ с информацией, кому он принадлежит, при помощи подписи. Электронно-цифровая подпись на сертификате означает, что кто-то удостоверяет, что данный открытый ключ действительно принадлежит человеку или компании. Если сертификат, который предоставляет веб-сервер, не подписан удостоверяющим центром, которому мы доверяем, браузер покажет предупреждение.
3. Обмен ключами. TLS/SSL — гибридная система. Она использует и асимметричное шифрование, и симметричное. Поэтому мы используем асимметричную криптографию для того, чтобы безопасно передать ключ, сгенерированный на клиенте (используя публичный ключ, взятый из сертификата). Сервер расшифрует его с помощью своего приватного ключа, который надёжно хранится на сервере и никому больше не известен). Сложность безопасного хранения симметричного ключа частично облегчается тем, что ключ жив\т только в рамках одного сеанса.

Настройка HTTPS на веб-сервере

Пропишем в файле /etc/hosts какое-нибудь доменное имя для сервера, чтобы мы могли обращаться к веб-страничке не по IP-адресу, а по доменному имени.

```
sudo vim /etc/hosts
127.0.0.1 site.local
```

Сгенерируем сертификат при помощи программы OpenSSL.

Начнём с приватного ключа. Его мы создадим командой:

```
cd /tmp
openssl genrsa -out server.key 4096
```

Теперь, прежде чем мы сможем сгенерировать сертификат, нам потребуется запрос на подпись сертификата, CSR.
openssl req -new -key server.key –out server.csr

Эта команда означает:

- req — указывает, что мы запрашиваем CSR;
- -out — указывает имя файла, в котором будет сохранён ваш CSR;
- -key — имя приватного ключа.

Далее в интерактивном диалоге предоставим информацию, которая будет включена в сертификат.
Так как у нас будет самоподписанный сертификат, то есть издавать его будем мы сами себе, подпишем его сами. Для этого:

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

Параметр -days установлен на 365, что означает, что сертификат действителен в течение года. После этого сертификат потребуется поменять.
Теперь у нас есть самоподписанный сертификат и секретный ключ.

Чтобы просмотреть информацию в сертификате, используется следующая команда:

```
openssl x509 -noout -text -in server.crt
```

Чтобы удостовериться, соответствует ли данный ключ данному сертификату, нужно сравнить модули — фрагменты служебной информации, хранящиеся в сертификатах, ключах и CSR:

```
openssl x509 -noout -modulus -in ssl_certificate.crt | openssl md5
```

Команда для вывода модуля приватного ключа:

```
openssl rsa -noout -modulus -in private.key | openssl md5
```

Настроим nginx для работы по HTTPS.
Создадим в папке /etc/nginx папку ssl для того, чтобы хранить в ней сертификат и ключ.

```
sudo mkdir ssl
sudo mv /tmp/server.* .
```

Для настройки откроем конфигурационный файл:

```
sudo vim site-available/default
```

Добавим:

```
ssl_certificate /etc/nginx/ssl/server.crt;
ssl_certificate_key /etc/nginx/ssl/server.key;
```

Впишем в server_name то же доменное имя, что и добавляли в /etc/hosts.
Теперь давайте перезапустим nginx для того, чтобы наши изменения вступили в силу:

```
systemctl restart nginx
```

Мы видим, что наше соединение шифруется, однако браузер не доверяет сертификату сервера. Чтобы исправить это, мы можем издать свой корневой сертификат и подписывать им другие сертификаты. Затем добавить в доверенные удостоверяющие центры в операционной системе и браузере наш корневой сертификат.