Self-Signed_Certificate

Translation(s): none

Creating a Self-Signed Certificate is not very complicated. This guide will show you a step by step procedure how to do it on Debian.

Prerequisites

The first step is to make sure that openssl and a webserver package are on your system, serving web pages. For this page, we discuss use of the Apache server, but you can use nginx or another. There are links at the bottom of the page for information on using nginx. You can use dpkg -s PackageName to check (e.g. dpkg -s openssl). If not you can always download them from the package website. All you need to do is open a terminal and type

```
apt-get install apache2 openssl
```

or

```
aptitude install apache2 openssl
```

⚠ Be sure that you are root (su/sudo) ⚠ . If the packages are not installed on the server it will automatically download them from the package site and install.

⚠ Before configuring Apache2 to serve over HTTPS, you should confirm that it is working OK for normal HTTP traffic. You can check this by launching your browser and entering http://127.0.0.1/ in the address bar, if you are working directly on the server, or http://IP.ADDRESS.OF.SERVER, when working on a remote server. if you see "It Works!", its likely working.

STEP 2

The second step is to start creating the Certificate File. For that you can use the "openssl" command that will assist you in the process. The command is as follows

```
mkdir -p /etc/ssl/localcerts
openssl req -new -x509 -days 365 -nodes -out /etc/ssl/localcerts/apache.pem -
keyout /etc/ssl/localcerts/apache.key
chmod 600 /etc/ssl/localcerts/apache*
```

Where "/etc/ssl/" is the directory for certificates. "/etc/ssl/localcerts" is a good place to put your certificates, but you can add other directories there if you have/need certificates for different sites, mail server, etc. if you prefer. You must then, of course, make sure the vhost (file generally in /etc/apache/sites-available referencing a specific site) mentions the proper certificate with its path.

Before typing this command, it is advisable to look at the openssl man page

```
man openssl
```

to understand all of the openssl options. You can specify the encryption method, the valid duration of the certificate, and other parameters.

After typing in the command, you will be prompted to answer some questions. Go ahead and answer them 🙂

One of the questions that may confuse some, is that of the FQDN (fully qualified domain name).

This is the domain for which the certificate will be used (www.yoursite.tld). In the above displayed command, we named our certificate and key "apache.*", but when you have multiple certificates, they will require different names, or, as mentioned, should reside in different sub-directories of /etc/ssl. You could, in /etc/ssl/localcerts, have several certificates and name them according to domain (i.e. somesite.com.pem and somesites.com.key, othersite.net.pem and othersite.net.key, etc.).

To override the default number of days for which the certificate is valid, you can specify -days X, where X is some other number.

STEP 3

After the question period you have to make sure that the ssl mode is enabled. The command is

```
a2enmod ssl
```

STEP 4

Next step involves creating a default page. You can copy and modify the default site that is available in /etc/apache2/sites-available directory. (e.g. cp default ssl). Use your favorite text editor, for example nano to edit the new site configuration file. Modify the default site so the server will listen on port 443 for incoming secure connections. Example:

```
NameVirtualHost *:443
<VirtualHost *:443>
SSLEngine On
SSLCertificateFile /etc/ssl/localcerts/apache.pem
SSLCertificateKeyFile /etc/ssl/localcerts/apache.key
```

You may also need to specify the ServerName value (indicated here with "*") with the same domain name you gave to the apache2-ssl-certificate questions, or the IP address, if you have sites served on distinct IP addresses.

In the existing site configuration file, default, you will likely need to specify port 80 with *:80.

If you have multiple sites on your server that will use distinct certificates, you will want to indicate so in their vhosts for apache SNI to function. If you can, it's even better to assign each site its own IP address to avoid use of SNI, since it does not work with all clients, but that's another matter, entirely.

STEP 5

After creating your SSL site, its time to enable it. To enable your newly created site you need to run this command.

```
a2ensite sitename
```

Where "sitename" should be replace by the name of the site you have created in STEP 4, corresponding to the name of its vhost, and, consequently, its domain name (yoursite.tld).

STEP 6

It is very important to tell the server to listen on port 443. That's why you need to add a line to ports.conf (/etc/apache2/ports.conf) file. After the modification your file should look something like this:

```
Listen 443
Listen 80
```

STEP 7

Now restart the apache server to apply the changes.

```
/etc/init.d/apache2 restart
```

or

```
service apache2 restart
```

STEP 8

Open your browser and type:

```
https://IP.ADDRESS.OF.SERVER
```

where "IP.ADDRESS.OF.SERVER" is, you guessed it, the IP address of the server. If you have physical access and are working directly on the server, this could be

```
https://127.0.0.1
```

The loop-back address should take you to the default apache page and display the Certificate prompt.

If you have any questions or comments, please drop me a line at konrad@atwaterlibrary.ca

This site is dedicated to Daniel Enright, a man that opened my eyes on to the World of Linux.

See also

A good place to learn more about SSL certificates and apache is http://httpd.apache.org/docs/2.2/ssl/ssl_faq.html#aboutcerts

If you are using nginx, or some other web server, of course, it's a whole different ball game.

For nginx, see http://nginx.org/en/docs/http/configuring_https_servers.html

Generating the certificate with openssl will be the same, of course, but configuration of the server to access/use the certificate will be different.