Apache - Enabling HTTPS w/Certificate

1. Getting the required software

For an SSL encrypted web server you will need a few things. Depending on your install you may or may not have OpenSSL and mod_ssl, Apache's interface to OpenSSL. Use yum to get them if you need them.

```
yum install mod_ssl openssl
```

Yum will either tell you they are installed or will install them for you.

2. Generate a self-signed certificate

Using OpenSSL we will generate a self-signed certificate. If you are using this on a production server you are probably likely to want a key from a Trusted Certificate Authority, but if you are just using this on a personal site or for testing purposes a self-signed certificate is fine. To create the key you will need to be root so you can either su to root or use sudo in front of the commands

```
# Generate private key
openssl genrsa -out ca.key 2048

# Generate CSR
openssl req -new -key ca.key -out ca.csr

# Generate Self Signed Key
openssl x509 -req -days 365 -in ca.csr -signkey ca.key -out ca.crt
```

```
# Copy the files to the correct locations
cp ca.crt /etc/pki/tls/certs
cp ca.key /etc/pki/tls/private/ca.key
cp ca.csr /etc/pki/tls/private/ca.csr
```

	WARNING: Make sure that you **copy** the files and do not **move** them if you use SELinux. Apache will complain about missing certificate files otherwise, as it cannot read them because the certificate files do not have the right SELinux context.

If you have moved the files and not copied them, you can use the following command to correct the SELinux contexts on those files, as the correct context definitions for /etc/pki/* come with the bundled SELinux policy.

```
restorecon -RvF /etc/pki
```

Then we need to update the Apache SSL configuration file

```
vi +/SSLCertificateFile /etc/httpd/conf.d/ssl.conf
```

Change the paths to match where the Key file is stored. If you've used the method above it will be

```
SSLCertificateFile /etc/pki/tls/certs/ca.crt
```

Then set the correct path for the Certificate Key File a few lines below. If you've followed the instructions above it is:

```
SSLCertificateKeyFile /etc/pki/tls/private/ca.key
```

Quit and save the file and then restart Apache

```
/etc/init.d/httpd restart
```

All being well you should now be able to connect over https to your server and see a default Centos page. As the certificate is self signed browsers will generally ask you whether you want to accept the certificate.

3. Setting up the virtual hosts

Just as you set VirtualHosts for http on port 80 so you do for https on port 443. A typical VirtualHost for a site on port 80 looks like this

```
<VirtualHost *:80>
        <Directory /var/www/vhosts/yoursite.com/httpdocs>
        AllowOverride All
        </Directory>
        DocumentRoot /var/www/vhosts/yoursite.com/httpdocs
        ServerName yoursite.com
</VirtualHost>
```

To add a sister site on port 443 you need to add the following at the top of your file

```
NameVirtualHost *:443
```

and then a VirtualHost record something like this:

```
<VirtualHost *:443>
        SSLEngine on
        SSLCertificateFile /etc/pki/tls/certs/ca.crt
        SSLCertificateKeyFile /etc/pki/tls/private/ca.key
        <Directory /var/www/vhosts/yoursite.com/httpsdocs>
        AllowOverride All
        </Directory>
        DocumentRoot /var/www/vhosts/yoursite.com/httpsdocs
        ServerName yoursite.com
</VirtualHost>
```

Restart Apache again using

```
/etc/init.d/httpd restart
```

4. Configuring the firewall

You should now have a site working over https using a self-signed certificate. If you can't connect you may need to open the port on your firewall. To do this amend your iptables rules:

```
iptables -A INPUT -p tcp --dport 443 -j ACCEPT
/sbin/service iptables save
iptables -L -v
```

Revision #1
Created 31 August 2024 12:32:11 by Steve Ling
Updated 31 August 2024 12:33:20 by Steve Ling