Linux System Administration

Apache SSL Certificate Generation and Use

1. Create a local SSL Certificate repository

Log in to your machine either as the root user, or once logged in become the root user, then do:

```
# mkdir /etc/ssl/localcerts
```

2. Generate a locally signed Digital Certficate for Apache

We'll use openssl to generate a local server key, local server certificate, a CSR (Certificate Signing Request) and a server key that is unencrypted (no passphrase) to allow Apache to start without prompting for a passphrase.

Create our own self signed certificate:

```
# cd /etc/ssl/localcerts
# mkdir apache
# cd apache
```

Ubuntu uses a special wrapper program to create a self-signed certificate. You can create your own, manually generated certificates, but this method works fine for what we are doing. To generate your local certificate for apache do:

```
# make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/ssl/localcerts/apache/server.pem
```

When you are prompted to enter the host name to use in the SSL certificate enter:

```
pcX.ws.nsrc.org
```

and tab to "<OK>" and press ENTER to continue.

Now you have a local certificate named server.pem in the directory /etc/ssl/localcerts/apache.

3. Enable Apache SSL configuration for your default domain

We need to update the /etc/apache2/sites-available/default-ssl configuration file and enable the site for our server. First we edit the file:

```
# cd /etc/apache2/sites-available
# vi default-ssl
```

Find the line that says:

```
DocumentRoot /var/www
```

and change this to:

```
DocumentRoot /var/www/share/pcX.ws.nsrc.org
```

Now find the line that says:

```
#SSLCertificateFile    /etc/ssl/certs/ssl-cert-snakeoil.pem
```

And create a line just below that says:

```
SSLCertificateFile    /etc/ssl/localcerts/apache/server.pem
```

Now save the file and exit, then we'll enable the Apache SSL configuration.

```
        # a2ensite default-ssl
        # service apache2 restart

To verify that Apache will provide an encrypted connection to pcX.ws.nsrc.org open a web
browser
and go to:

        https://pcX.ws.nsrc.org/

You should receive a warning that the certificate is not trusted. Click "Continue" (this is
different
in each web browser) to view your home page.

4. Manually verify SSL certificate use on your web server
----------------------------------------------------------

You can use the built-in OpenSSL command line tool to connect to your web server and see
information about
the SSL certificate in use. To do this do (as root or a regular user):

        # openssl s_client -connect pcX.ws.nsrc.org:443

And you will see information about the SSL Digital Certificate for the site pcX.ws.nsrc.org.
You should see
something like:

subject=/CN=pcX.ws.nsrc.org
issuer=/CN=pcX.ws.nsrc.org
---
No client certificate CA names sent
---
SSL handshake has read 1004 bytes and written 319 bytes
---
New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
Server public key is 1024 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1
    Cipher    : DHE-RSA-AES256-SHA
    Session-ID: 18541F63DDD15E050A3C72ED9415CC9A00B7DCD0DC472919AE4E4B67E4D88837
    Session-ID-ctx:
    Master-Key:
20BC655CCF5BC3D3BECD1D04333F928CB1A756871E5ACBD94455DD324E7E62BE29D11664AFDD61257DB71CBE1B4A7FE
E
    Key-Arg   : None
    Start Time: 1334826634
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
---
closed


Press CTRL-C to exit from the program.

As you can see reasonable ciphers are available to ensure encrypted communication between the
server and
a client connecting via https (SSL).
```