OpenSSL Certificate Creation and Implementation

Process Overview:

1) Install the OpenSSL toolkit

2) Create the Private Key and Certificate Signing Request (CSR) files.

3) Submit CSR to your certificate authority (CA) of choice (e.g. Thawte, GoDaddy, Verisign)

4) Retrieve your certificate from the certificate authority.

5) Download the “root certificate” from the same certificate authority

There are three files needed by OpenSSL for a successful SSL implementation: private key, certificate, root certificate. These are all different files and cannot be interchanged. The CSR (fourth file) is needed only to request the certificate and not needed by OpenSSL to run the secure website.

OpenSSL requires the “PEM” format. The PEM format is the most common format that Certificate Authorities issue certificates in. PEM certificates usually have extensions such as .pem, .crt, .cer, and .key. They are Base64 encoded ASCII files and contain "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" statements. Server certificates, intermediate certificates, and private keys can all be put into the PEM format.

Apache and other similar servers use PEM format certificates. Several PEM certificates, and even the private key, can be included in one file, one below the other, but most platforms, such as Apache, expect the certificates and private key to be in separate files.

Process Steps:

1) Install the OpenSSL toolkit

Shining Light Productions (http://slproweb.com/products/Win32OpenSSL.html) has the pre-compiled OpenSSL binaries for Windows. They also have links to the Microsoft Visual C++ Redistributables. Be careful to install the proper C++ Redistributables for your OS version.

Install the Visual C++ Redistributables first and then the OpenSSL binaries.

2) Create the Private Key and Certificate Signing Request (CSR) files.

From the command line on the server to be secured, enter the following syntax:

```
openssl req –newkey 2048 –nodes -keyout <myserver.key> –out
<server.csr>
```

Example:

```
openssl req –newkey 2048 –nodes -keyout  rentalregistry.key –out
rentalregistry.csr
```

After entering the above command, you will be prompted to respond to the following questions. Leave the "challenge password" and "optional company name" blank.

```
Country Name (2 letter code) [AU]: GB
State or Province Name (full name) [Some-State]: Yorks
Locality Name (eg, city) []: York
Organization Name (eg, company) [Internet Widgits Pty Ltd]: MyCompany Ltd
Organizational Unit Name (eg, section) []: IT
Common Name (eg, YOUR name) []: mysubdomain.mydomain.com
Email Address []:

Please enter the following 'extra' attributes to be sent with your certificate request

A challenge password []:
An optional company name []:
```

Once this process is complete, you'll have two files: The private key (rentalgegistry.key in the example); and the CSR (rentalregistry.csr in the example). Keep the Private Key file (don't misplace it!) and use the CSR for purchasing the certificate.

3) Submit CSR to your certificate authority (CA) of choice

When submitting the CSR to the CA, they'll ask what platform or web server the certificate will be installed on. Thawte does this and I assume others do as well. Select the Apache web server platform since it uses the OpenSSl toolkit for secure websites.

4) Retrieve your certificate from the certificate authority

When retrieving your certificate, select the X.509 format. Note that there may be several certificates to retrieve which all provide the certificate "chain" back to the CA.

If there are multiple certificates, the one OpenSSL requires is the “End Entity Certificate”. This is the one used for the certificate file.

5) Download the “root certificate” from the same certificate authority

The root certificate provides the certificate chain back to the CA. There may be several certificates that make up the root certificate. They can be combined into a single file. You must download the root certificate{s} from the same CA that generated your certificate.

Debugging

To check an MD5 hash of the public key to ensure that it matches with what is in a CSR or private key enter the following commands. The output strings from all of these commands should match exactly. If they don’t, then there’s an issue with one of the files.

openssl x509 -noout -modulus -in <certificate.crt> | openssl md5

openssl rsa -noout -modulus -in <privateKey.key> | openssl md5

openssl req -noout -modulus -in <CSR.csr> | openssl md5