

Managing self signed SSL certificates

Alessandro Dotti Contra
alessandro@hyboria.org

May 1, 2010

Abstract

If you ever need to set up some SSL based secure services, you'll have to deal with certificates and certificate authorities. This article is a really dry explanation of how you can set up a self certificate authority and then manage your SSL certificates.

Create a root CA certificate

The first thing to do is to create a root CA certificate and a private CA key.

Prepare the working environment

Before we can start working with certificates, some files and directories need to be created.

```
$ mkdir CA
$ cd CA
$ mkdir newcerts private requests
$ echo '01' > serial
$ touch index.txt
```

The resulting files and directories will be used as follows:

`CA` is the root directory for the CA working area;

`newcerts` will store all the issued certificates;

`private` will store all the private key related to the issued certificates;

`requests` will store all the certificate requests;

`serial` contains the number OpenSSL will use for the next certificate issued;

`index.txt` will contain some resume informations for the certificates issued.

Create the certificate and the private key

Now we can create the root CA certificate and the private CA key.

```
$ openssl req -new -x509 -extensions v3_ca \  
    -keyout private/cakey.pem -out cacert.pem \  
    -days 365
```

This command will generate two files: `cacert.pem` is the root CA certificate and `private/cakey.pem` is the root CA private key. The `cacert.pem` file could be distributed to your clients, while the `cakey.pem` will be used later to sign certificate requests.

The content of the certificate can be viewed with the following command:

```
$ openssl x509 -text -in cacert.pem
```

The `cacert.pem` must also be copied to the `newcerts` directory. The name must begin with the serial number of the certificate (which is 00 for the CA certificate).

```
$ cp cacert.pem newcerts/00.pem
```

Create and sign certificate requests

Once the root CA certificate and private key are ready, you're able to create the certificates you need for your services. A certificate request is needed first, then it will be signed with the CA private key.

```
$ openssl req -new -nodes -out req.pem
```

This command will create the certificate request `req.pem` that needs to be signed and the private key `key.pem` for your host or service.

The request and the key just generated must be stored:

```
$ cp key.pem private/[hostname|service]-key.pem  
$ cp req.pem requests/[hostname|service]-req.pem
```

Now the request must be signed by the CA:

```
$ openssl ca -out cert.pem -in req.pem
```

The file `cert.pem` is the signed certificate. You will also find a copy of the certificate in the `newcerts` directory; the name begins with the appropriate serial number.

If your application requires it, you can merge the key and the certificate in a single file:

```
$ cat key.pem cert.pem > key-cert.pem
```

The files `cert.pem` and `key.pem` (or `key-cert.pem` if needed) must be installed on the target host.

Revoke certificates

To revoke a certificate:

```
$ openssl ca -revoke newcerts/[serial].pem
```

You can find the appropriate serial number browsing the `index.txt` file.
Now you have to update your certificate revocation list:

```
$ openssl ca -gencrl -out crl.pem
```

The content of the certificate revocation list can be view as follows:

```
$ openssl crl -text -in crl.pem
```

The certificate revocation list should be made available trough your web site.

Renew certificates

To renew and expired certificate, you need to first revoke the expired certificate and then resign the original request.

```
$ openssl ca -revoke newcerts/[serial].pem
```

```
$ openssl ca -out cert.pem -in req.pem
```

References

- [1] Daniele Giacomini: *Appunti di informatica libera - cap.198: Introduzione a OpenSSL*
- [2] Marcus Redivo: *Creating and using SSL certificates*

Copyright ©2010 Alessandro Dotti Contra

This work is licensed under the terms of the Creative Commons NonCommercial-Attribution-ShareAlike license. A reference copy of this license can be found at the following address:

<http://creativecommons.org/licenses/by-nc-sa/2.5/>