HOWTO certificates

1. Introduction

How you handle certificates depends a great deal on what your role is.
Your role can be one or several of:

  - User of some client application
  - User of some server application
  - Certificate authority

This file is for users who wish to get a certificate of their own.
Certificate authorities should read https://www.openssl.org/docs/apps/ca.html.

In all the cases shown below, the standard configuration file, as
compiled into openssl, will be used. You may find it in /etc/,
/usr/local/ssl/ or somewhere else. By default the file is named
openssl.cnf and is described at https://www.openssl.org/docs/apps/config.html.
You can specify a different configuration file using the
'-config {file}' argument with the commands shown below.

2. Relationship with keys

Certificates are related to public key cryptography by containing a
public key. To be useful, there must be a corresponding private key
somewhere. With OpenSSL, public keys are easily derived from private
keys, so before you create a certificate or a certificate request, you
need to create a private key.

Private keys are generated with 'openssl genrsa -out privkey.pem' if
you want a RSA private key, or if you want a DSA private key:
'openssl dsaparam -out dsaparam.pem 2048; openssl gendsa -out privkey.pem dsaparam.pem'.

The private keys created by these commands are not passphrase protected;
it might or might not be the desirable thing. Further information on how to
create private keys can be found at https://www.openssl.org/docs/HOWTO/keys.txt.
The rest of this text assumes you have a private key in the file privkey.pem.

3. Creating a certificate request

To create a certificate, you need to start with a certificate request
(or, as some certificate authorities like to put it, "certificate
signing request", since that's exactly what they do, they sign it and
give you the result back, thus making it authentic according to their
policies). A certificate request is sent to a certificate authority
to get it signed into a certificate. You can also sign the certificate
yourself if you have your own certificate authority or create a
self-signed certificate (typically for testing purpose).

The certificate request is created like this:

```
  openssl req -new -key privkey.pem -out cert.csr
```

Now, cert.csr can be sent to the certificate authority, if they can
handle files in PEM format. If not, use the extra argument '-outform'
followed by the keyword for the format to use (see another HOWTO
<formats.txt?>). In some cases, -outform does not let you output the
certificate request in the right format and you will have to use one

of the various other commands that are exposed by openssl (or get creative and use a combination of tools).

The certificate authority performs various checks (according to their policies) and usually waits for payment from you. Once that is complete, they send you your new certificate.

Section 5 will tell you more on how to handle the certificate you received.

4. Creating a self-signed test certificate

You can create a self-signed certificate if you don't want to deal with a certificate authority, or if you just want to create a test certificate for yourself. This is similar to creating a certificate request, but creates a certificate instead of a certificate request. This is NOT the recommended way to create a CA certificate, see https://www.openssl.org/docs/apps/ca.html.

```
  openssl req -new -x509 -key privkey.pem -out cacert.pem -days 1095
```

5. What to do with the certificate

If you created everything yourself, or if the certificate authority was kind enough, your certificate is a raw DER thing in PEM format. Your key most definitely is if you have followed the examples above. However, some (most?) certificate authorities will encode them with things like PKCS7 or PKCS12, or something else. Depending on your applications, this may be perfectly OK, it all depends on what they know how to decode. If not, There are a number of OpenSSL tools to convert between some (most?) formats.

So, depending on your application, you may have to convert your certificate and your key to various formats, most often also putting them together into one file. The ways to do this is described in another HOWTO <formats.txt?>, I will just mention the simplest case. In the case of a raw DER thing in PEM format, and assuming that's all right for your applications, simply concatenating the certificate and the key into a new file and using that one should be enough. With some applications, you don't even have to do that.

By now, you have your certificate and your private key and can start using applications that depend on it.