OpenSSL tips and common commands

OpenSSL is the de-facto tool for SSL on linux and other server systems. It providers both the library for creating SSL sockets, and a set of powerful tools for administrating an SSL enabled website. Following are a few common tasks you might need to perform with OpenSSL.

Generate a certificate request

Obtaining a signed SSL certificate envolves a number of buisness verification procedures and a sumbition of what is called a CSR ("Certificate signing request"). To generate the CSR, execute the following command.

openssl req -new -newkey rsa:1024 -nodes -keyout key.pem -out req.pem

Lets review the command:

* **req** activates the part of openssl that deals with certificate requests signing
* **-new** generate a new request
* **-newkey** generate a new private key
* **rsa:1024** 1024 is the bit length of the private key. Alternative you can use 2048 and 512, for larger or smaller keys but, please note that the strength of the key should match the type of service your certificate authority is providing to you.
* **-nodes** no des, stores the private key without protecting it with a passphrase. While this is not considered to be best practice, many people do not set a passphrase or later remove it, since services with pass phrase protected keys can not be auto-restarted without typing in the passphrase
* **-keyout key.pem** store the private key in a file called key.pem
* **-out req.pem** store the certificate request in a file called req.pem

This command will run interactivly and ask you a number of questions, please note that your answers will be double and cross checked by your certificate authority and that your answers must match any other legal documents regarding the registration of your company. Following are tips for proper answers:

Fill in your companies two letter country code, [consult wikipedia](http://en.wikipedia.org/wiki/ISO_3166-1_alpha-2#Officially_assigned_code_elements) if you are unsure which code to use.

Country Name (2 letter code) [AU]:

State for US, large administrative district for other countries:

State or Province Name (full name) [Some-State]:

City

Locality Name (eg, city) []:

Full company name, please copy this letter to letter from your companies registration forms. A difference such as using the sign & instead of the word "and" might cause your request to be rejected.

Organization Name (eg, company) [Internet Widgits Pty Ltd]:

Company sub-division or a product name

Organizational Unit Name (eg, section) []:

Your domain name, or in case of wildcard certificates, use an astrisk, like this: \*.mycompany.com

Common Name (eg, YOUR name) []:

Email to be displayed with the certificate

Email Address []:

Double check the information by using this command on your newly generated request:

openssl req -in req.pem -noout -text

Save your private key file, named key.pem, in a secure location. It will later be used to configure your web server. The request file, req.pem, should be sent to your certificate authority for signing.

Generate a self-signed key

You can generate a self-signed key for a development servers by following those steps:

Create an empty directory and step in to it. Execute the following command, please note that the backslash ("\") sign allow a single command to span over a number of lines. In our case it is used to fit the command in this document:

$ openssl req -x509 -days 365 -nodes -newkey rsa:1024 \

-keyout key.pem -out cert.pem

You can hit enter as an answer to all the questions to set the default except this one:

Common Name (eg, YOUR name) []:

type in the dns record used for your development server as an answer to this one.

Thats it, two new [PEM files](http://how2ssl.com/articles/working_with_pem_files/) will be created, "cert.pem" containing your certificate and "key.pem" containing the self signed key.

Testing SSL servers

You can use the OpenSSL built in client to connect to a web server and display the certificate chain. Replace your server address and port with your own:

$ openssl s\_client -connect www.facebook.com:443 -showcerts

Here is a typical output, with the certificate chain displayed:

CONNECTED(00000003)

depth=1 O = CA, OU = "CA", OU = CA, OU = CA

verify error:num=20:unable to get local issuer certificate

verify return:0

---

Certificate chain

0 s:/C=US/ST=California/L=Palo Alto/O=mysite/CN=mysite.com

i:/O=CA/OU=CA/OU=CA/OU=CA

-----BEGIN CERTIFICATE-----

MIIDnzCCAwigAwIBAgIQCSGX4cDpzQPaNSQ2VhCGgTANBgkqhkiG9w0BAQUFADCB

ujEfMB0GA1UEChMWVmVyaVNpZ24gVHJ1c3QgTmV0d29yazEXMBUGA1UECxMOVmVy

aVNpZ24sIEluYy4xMzAxBgNVBAsTKlZlcmlTaWduIEludGVybmF0aW9uYWwgU2Vy

A .... MANY LINES LIKE THAT .... .... MANY LINES LIKE THAT ....

gjRaROuWGxfY25KebCQpoBW2PJp3S1JmqHHyxjk4mzr+tzWK0Qn+tlBUy9igtkIh

VybjO+AxBZve1qyJIsVraz8wrw==

-----END CERTIFICATE-----

1 s:/O=CA/OU=CA/OU=CA/OU=CA

i:/C=US/O=CA/OU=CA

-----BEGIN CERTIFICATE-----

MIIDgzCCAuygAwIBAgIQRvzrurTQLw+SYJgjP5MHjzANBgkqhkiG9w0BAQUFADBf

MQswCQYDVQQGEwJVUzEXMBUGA1UEChMOVmVyaVNpZ24sIEluYy4xNzA1BgNVBAsT

LkNsYXNzIDMgUHVibGljIFByaW1hcnkgQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkw

A .... MANY LINES LIKE THAT .... .... MANY LINES LIKE THAT ....

OfamggNlEcS8vy2m9dk7CrWY+rN4uR7yK0xi1f2yeh3fM/1z+aXYLYwq6tH8sCi2

6UlIE0uDihtIeyT3ON5vQVS4q1drBt/HotSp9vE2YoCI8ot11oBx

-----END CERTIFICATE-----

---

Server certificate

subject=/C=US/ST=California/L=Palo Alto/O=mysite/CN=mysite.com

issuer=/O=CA/OU=CA/OU=CA/OU=CA

---

No client certificate CA names sent

---

SSL handshake has read 2007 bytes and written 343 bytes

---

New, TLSv1/SSLv3, Cipher is RC4-MD5

Server public key is 1024 bit

Secure Renegotiation IS NOT supported

Compression: NONE

Expansion: NONE

SSL-Session:

Protocol : SSLv3

Cipher : RC4-MD5

Session-ID: 244BE55....48F793

Session-ID-ctx:

Master-Key: 18674D2....B3465946941C0C77DF2DE

Key-Arg : None

PSK identity: None

PSK identity hint: None

Start Time: 1325335498

Timeout : 7200 (sec)

Verify return code: 20 (unable to get local issuer certificate)

---

you can copy parts of the output to a [PEM file](http://how2ssl.com/articles/working_with_pem_files/) and further inspect them with the verify openssl command.