OpenSSL Cheatsheet

trimstray/the-book-of-secret-knowledge

Testing connection to remote host

```
echo | openssl s_client -connect google.com:443 -showcerts
```

Testing connection to remote host (with SNI support)

```
echo | openssl s_client -showcerts -servername google.com -connect google.com:443
```

Testing connection to remote host with specific ssl version

```
openssl s_client -tls1_2 -connect google.com:443
```

Testing connection to remote host with specific ssl cipher

```
openssl s_client -cipher 'AES128-SHA' -connect google.com:443
```

Generate private key

```
# _len: 2048, 4096
( _fd="private.key" ; _len="4096" ; \
openssl genrsa -out ${_fd} ${_len} )
```

Generate private key with passphrase

```
# _ciph: des3, aes128, aes256
# _len: 2048, 4096
( _ciph="aes128" ; _fd="private.key" ; _len="4096" ; \
openssl genrsa -${_ciph} -out ${_fd} ${_len} )
```

Remove passphrase from private key

```
( _fd="private.key" ; _fd_unp="private_unp.key" ; \
openssl rsa -in ${_fd} -out ${_fd_unp} )
```

Encrypt existing private key with a passphrase

```
# _ciph: des3, aes128, aes256
( _ciph="aes128" ; _fd="private.key" ; _fd_pass="private_pass.key" ; \
openssl rsa -${_ciph} -in ${_fd} -out ${_fd_pass}
```

Check private key

```
( _fd="private.key" ; \
openssl rsa -check -in ${_fd} )
```

Get public key from private key

```
( _fd="private.key" ; _fd_pub="public.key" ; \
openssl rsa -pubout -in ${_fd} -out ${_fd_pub} )
```

Convert DER to PEM

```
( _fd_der="cert.crt" ; _fd_pem="cert.pem" ; \
openssl x509 -in ${_fd_der} -inform der -outform pem -out ${_fd_pem} )
```

Convert PEM to DER

```
( _fd_der="cert.crt" ; _fd_pem="cert.pem" ; \
openssl x509 -in ${_fd_pem} -outform der -out ${_fd_der} )
```

Checking whether the private key and the certificate match

```
(openssl rsa -noout -modulus -in private.key | openssl md5 ; \
openssl x509 -noout -modulus -in certificate.crt | openssl md5) | uniq
```

Generate private key + csr

```
( _fd="private.key" ; _fd_csr="request.csr" ; _len="4096" ; \
openssl req -out ${_fd_csr} -new -newkey rsa:${_len} -nodes -keyout ${_fd} )
```

Generate csr

```
( _fd="private.key" ; _fd_csr="request.csr" ; \
openssl req -out ${_fd_csr} -new -key ${_fd} )
```

Generate csr (metadata from exist certificate)

```
( _fd="private.key" ; _fd_csr="request.csr" ; _fd_crt="cert.crt" ; \
openssl x509 -x509toreq -in ${_fd_crt} -out ${_fd_csr} -signkey ${_fd} )
```

Generate csr with -config param

```
( _fd="private.key" ; _fd_csr="request.csr" ; \
openssl req -new -sha256 -key ${_fd} -out ${_fd_csr} \
-config <(
cat <<-EOF
[req]
default_bits = 2048
prompt = no
default_md = sha256
req_extensions = req_ext
distinguished_name = dn

[ dn ]
C=<two-letter ISO abbreviation for your country>
ST=<state or province where your organization is legally located>
L=<city where your organization is legally located>
O=<legal name of your organization>
OU=<section of the organization>
CN=<fully qualified domain name>

[ req_ext ]
subjectAltName = @alt_names

[ alt_names ]
DNS.1 = <fully qualified domain name>
DNS.2 = <next domain>
DNS.3 = <next domain>
EOF
))
```

List available EC curves

```
openssl ecparam -list_curves
```

Generate ECDSA private key

```
# _curve: prime256v1, secp521r1, secp384r1
( _fd="private.key" ; _curve="prime256v1" ; \
openssl ecparam -out ${_fd} -name ${_curve} -genkey )

# _curve: X25519
( _fd="private.key" ; _curve="x25519" ; \
openssl genpkey -algorithm ${_curve} -out ${_fd} )
```

Print ECDSA private and public keys

```
( _fd="private.key" ; \
openssl ec -in ${_fd} -noout -text )

# For x25519 only extracting public key
( _fd="private.key" ; _fd_pub="public.key" ; \
openssl pkey -in ${_fd} -pubout -out ${_fd_pub} )
```

Generate private key with csr (ECC)

```
# _curve: prime256v1, secp521r1, secp384r1
( _fd="domain.com.key" ; _fd_csr="domain.com.csr" ; _curve="prime256v1" ; \
openssl ecparam -out ${_fd} -name ${_curve} -genkey ; \
openssl req -new -key ${_fd} -out ${_fd_csr} -sha256 )
```