Key Generation and Encryption Examples using OpenSSL

By Steven Gordon on Sat, 22/01/2011 - 5:48pm

1. Generating RSA Keys

Generate a 1024-bit RSA private key using a public exponent (e) of 65537 (-F4 option). The key is NOT encrypted with DES (or other ciphers).

```
$ openssl genrsa -out privkeyA.pem -F4 1024
```

The output file (`privkeyA.pem`) is plaintext. It contains the private key, encoded as Base64 [3], in between two lines indicating the begin and end of the key.

2. Setting up the Certificate Authority (CA Only)

The CA also has a certificate, which is created from a private key (same as using `genrsa` above), and a self-signed certificate. Since we don't have a higher-level CA to sign the certificate of our CA, we must sign the CAs certificate ourselve, as follows:

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

CA information can be stored in a set of directories. By default the openssl configration in `/usr/lib/ssl/openssl.cnf` assumes some directory structure. Hence we need to create it:

```
$ mkdir ./demoCA
$ mkdir ./demoCA/certs
$ mkdir ./demoCA/crl
$ mkdir ./demoCA/newcerts
$ mkdir ./demoCA/private
$ touch ./demoCA/index.txt
$ echo 02 > demoCA/serial
$ mv cacert.pem ./demoCA
$ mv cakey.pem ./demoCA/private
```

The CA's certificate (`cacert.pem`) is made available to all users.

3. Requesting a Certificate from CA

Create a certificate request that will be sent to the Certificate Authority (CA). This takes a private key as input (i.e. the file generated above) and produces a .csr certificate request file as output. This is a new certificate request. When prompted for information give the following responses:

- Country Name: TH
- State or Province: Pathumthani

- Locality: Bangkadi
- Organisation: SIIT
- Unit: ICT
- Common Name: FIRST_NAME LAST_NAME
- Email Address: EMAIL_FROM_MAILIST
- A challenge password: *(nothing, just press enter)*
- Optionaly company: *(nothing, just press enter)*

```
$ openssl req -new -key privkeyA.pem -out certA.csr
```

Send your certificate request to the CA.

4. Creating a Certificate (CA Only)

The CA will create a certificate (for the following to work, the appropriate directory structure at the CA must exist; see above):

```
$ openssl ca -in certA.csr -out cert-A.pem
```

and then return your certificate to you.

5. Verifying a Certificate

Upon receipt of the certificate, you should validate it:

```
$ openssl verify -CAfile cacert.pem cert-A.pem
cert-A.pem: OK
```

If it is OK (and you trust the CA) then you are sure the received certificate came from the CA and was not modified along the way.

6. Encrypting with RSA

Now lets assume a set of users (A, B, C, ...) have completed the above steps and have their certificates (e.g. cert-A.pem, cert-B.pem). RSA can be used for encrypting (small) files. Consider a message that A wants to confidentially send to B:

```
$ cat message.txt
A private message
```

Now user A encrypts the message using B's public key which is stored in B's certificate:

```
$ openssl rsautl -encrypt -certin -inkey cert-B.pem -in message.txt -out ciphertext.s
```

The output file `ciphertext.ssl` contains the encrypted message in binary:

```
$ ls -l ciphertext.ssl
-rw-r--r-- 1 sgordon sgordon 128 2011-01-22 12:22 ciphertext.ssl
```

Now user B decrypts using their private key:

```
$ openssl rsautl -decrypt -inkey privkeyB.pem -in ciphertext.ssl -out decrypted.txt
$ cat decrypted.txt
A private message
```

7. Signing with RSA

The most common application of RSA is to sign messages (as opposed to encrypt them). User A signs a message with their private key:

```
$ openssl rsautl -sign -inkey privkeyA.pem -in message.txt -out message.sgn
```

User B that receives the signed message can verify the signature using A's certificate:

```
$ openssl rsautl -verify -in message.sgn -certin -inkey cert-B.pem
A private message
```

If an incorrect key is used or the message has been modified an error should be reported. First consider a case where the message was signed with

Source URL: http://sandilands.info/sgordon/key-generation-and-encryption-examples-using-openssl

Links:
[1] http://sandilands.info/sgordon/key-generation-and-encryption-examples-using-openssl
[2] http://sandilands.info/sgordon/user/2
[3] http://en.wikipedia.org/wiki/Base64