Crypto Lab - PKI

Monday, April 28 2014, 8:51 PM

Crypto Lab – Public-Key Cryptography and PKI

Task 1: Become a Certificate Authority (CA)

```
[04/22/2014 13:44] seed@ubuntu:~/Desktop/PKILab$ ls
ca.key  demoCA  dir  openssl.cnf
[04/22/2014 13:44] seed@ubuntu:~/Desktop/PKILab$ openssl req -new -x509 -keyout
ca.key -out ca.crt -config openssl.cnf
Generating a 1024 bit RSA private key
........++++++
.....................++++++
writing new private key to 'ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:NewYork
Locality Name (eg, city) []:Syracuse
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SEEDLabs
Organizational Unit Name (eg, section) []:.
Common Name (e.g. server FQDN or YOUR name) []:Mark
Email Address []:.
[04/22/2014 13:46] seed@ubuntu:~/Desktop/PKILab$
```

Task 2: Create a Certificate for PKILabServer.com

Step 1: Generate public/private key pair

```
[04/22/2014 13:48] seed@ubuntu:~/Desktop/PKILab$ openssl genrsa -des3 -out serve
r.key 1024
Generating RSA private key, 1024 bit long modulus
.++++++
..........................++++++
e is 65537 (0x10001)
Enter pass phrase for server.key:
Verifying - Enter pass phrase for server.key:
[04/22/2014 13:49] seed@ubuntu:~/Desktop/PKILab$
```

Step 2: Generate a Certificate Signing Request (CSR)

```
[04/22/2014 13:50] seed@ubuntu:~/Desktop/PKILab$ openssl req -new -key server.ke
y -out server.csr -config openssl.cnf
Enter pass phrase for server.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:NewYork
Locality Name (eg, city) []:Syracuse
Organization Name (eg, company) [Internet Widgits Pty Ltd]:PKILabServer
Organizational Unit Name (eg, section) []:.
Common Name (e.g. server FQDN or YOUR name) []:PKILabServer.com
Email Address []:.

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:123456
An optional company name []:.
[04/22/2014 13:52] seed@ubuntu:~/Desktop/PKILab$
```

Step 3: Generating Certificates

```
[04/22/2014 13:55] seed@ubuntu:~/Desktop/PKILab$ openssl ca -in server.csr -out
server.crt -cert ca.crt -keyfile ca.key -config openssl.cnf
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
The organizationName field needed to be the same in the
CA certificate (SEEDLabs) and the request (PKILabServer)
[04/22/2014 13:55] seed@ubuntu:~/Desktop/PKILab$
```

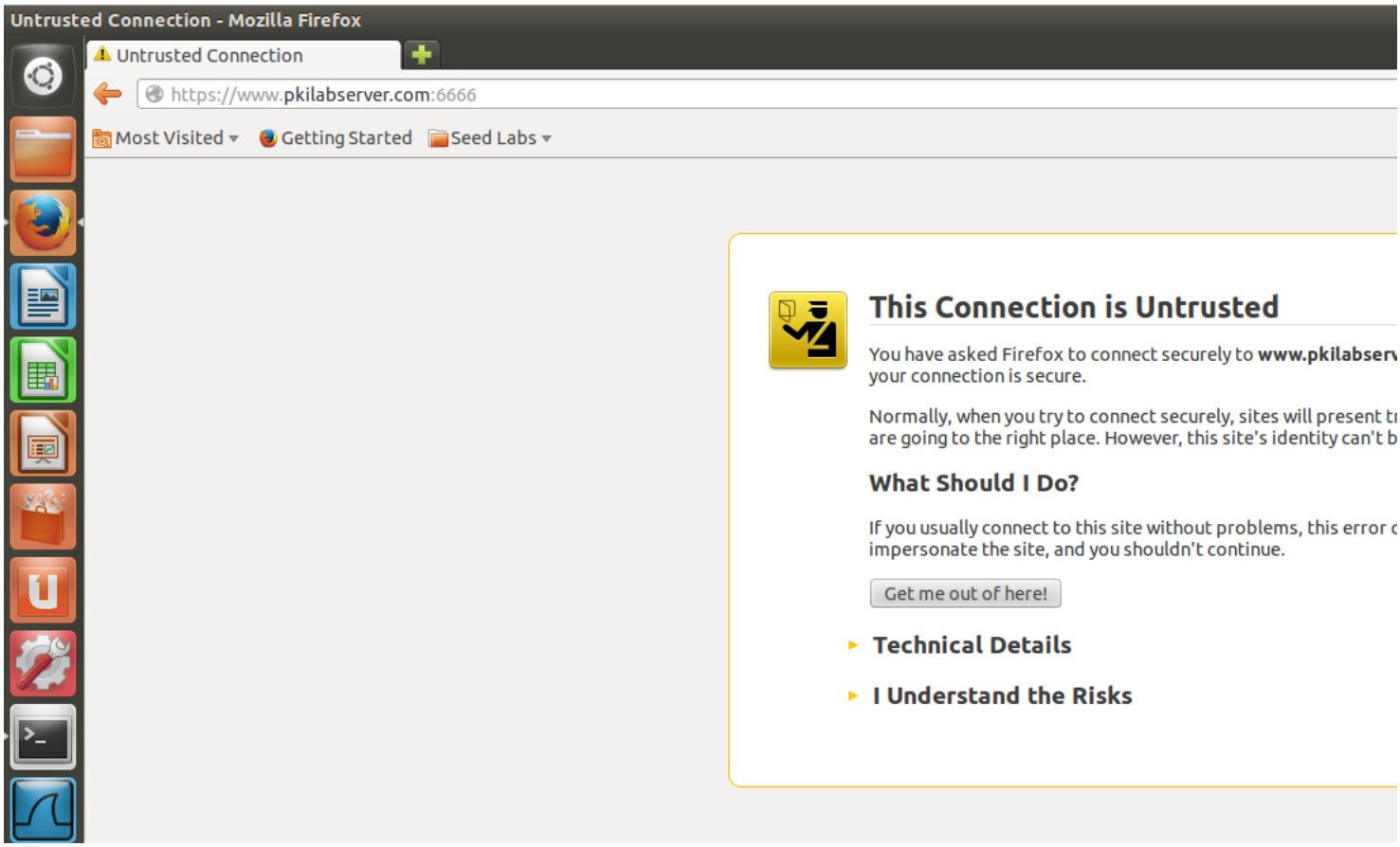
```
# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy          = policy_match

# For the CA policy
[ policy_match ]
countryName             = match
stateOrProvinceName     = match
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional
```

```
Terminal
[04/22/2014 14:11] seed@ubuntu:~/Desktop/PKILab$ sudo chmod 777 openssl.cnf
[04/22/2014 14:15] seed@ubuntu:~/Desktop/PKILab$ vim openssl.cnf
[04/22/2014 14:18] seed@ubuntu:~/Desktop/PKILab$ openssl ca -in server.csr -out
server.crt -cert ca.crt -keyfile ca.key -config openssl.cnf
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4096 (0x1000)
        Validity
            Not Before: Apr 22 01:19:15 2014 GMT
            Not After : Apr 22 01:19:15 2015 GMT
        Subject:
            countryName               = US
            stateOrProvinceName       = NewYork
            organizationName          = PKILabServer
            commonName                = PKILabServer.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                24:4C:89:9D:14:72:1E:63:27:7F:35:A1:64:18:34:46:E3:1A:CF:48
            X509v3 Authority Key Identifier:
                keyid:C9:87:24:7A:B3:B1:14:5D:BA:A2:16:B1:3F:F7:D7:91:13:7C:91:8
B

Certificate is to be certified until Apr 22 01:19:15 2015 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
[04/22/2014 14:19] seed@ubuntu:~/Desktop/PKILab$
```

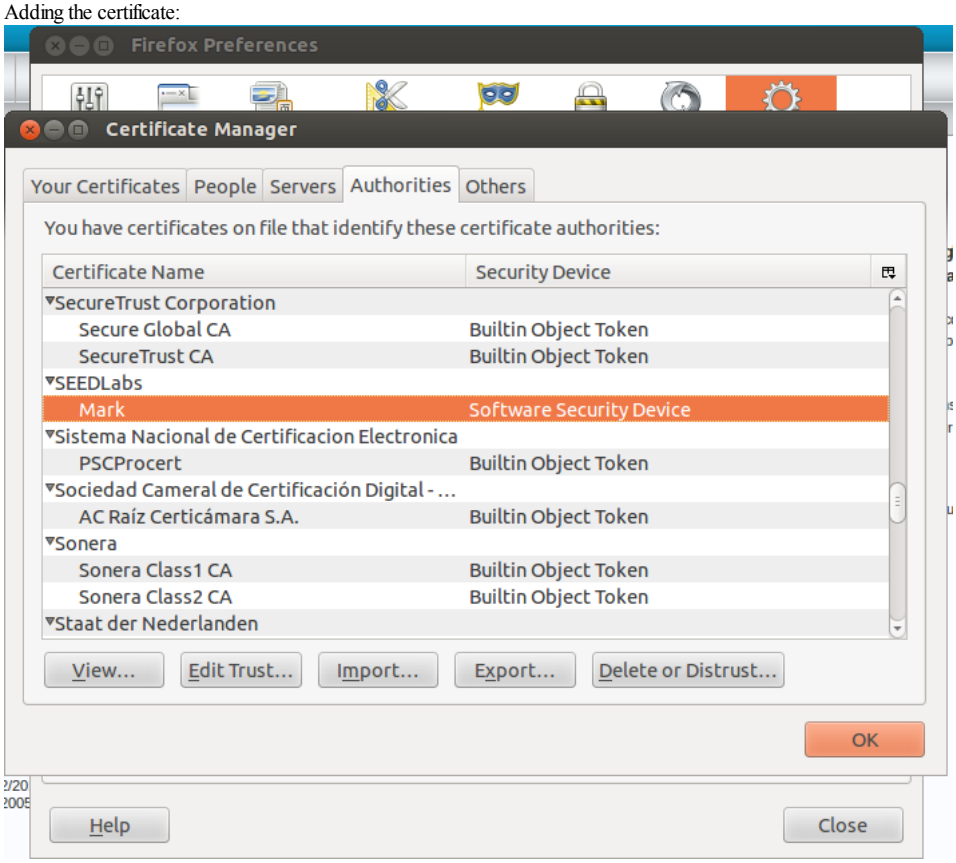
Task 3: Use PKI for Web Sites

```
[04/22/2014 14:31] seed@ubuntu:~/Desktop/PKILab$ openssl s_server -cert server.p
em -www -accept 6666
Enter pass phrase for server.pem:
Using default temp DH parameters
Using default temp ECDH parameters
ACCEPT
ACCEPT
ACCEPT
```

Mozilla without the certificate:



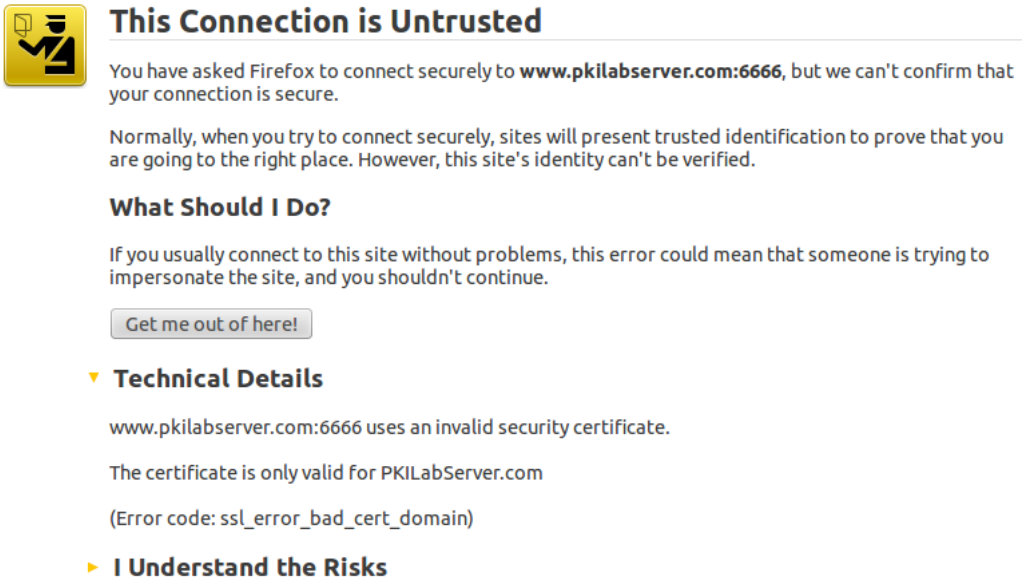
Adding the certificate:



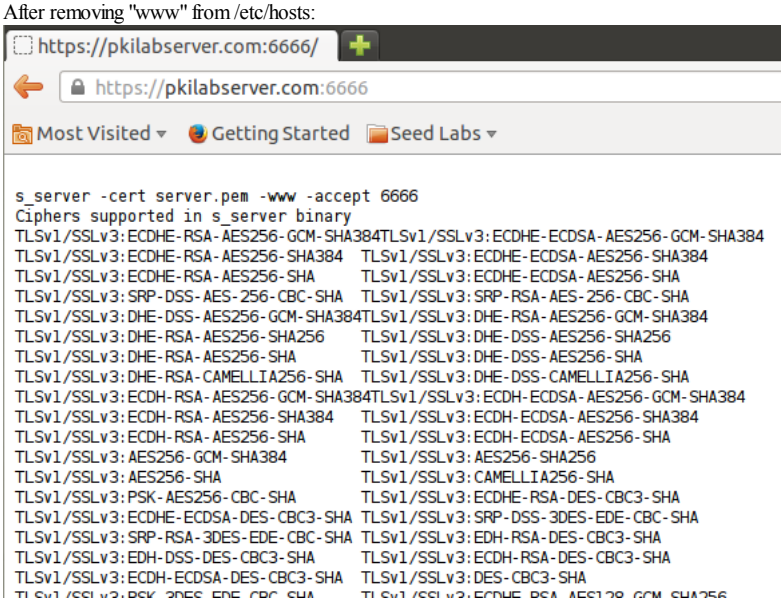
The certificate seems to be valid for 1 month by default:

Validity	
Issued On	04/22/2014
Expires On	05/22/2014

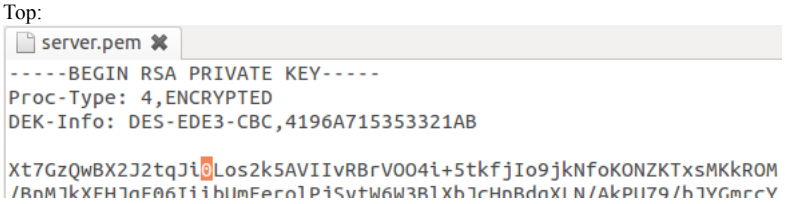
The certificate was still not authorized. This is because in /etc/hosts, I added www.pkilabserver.com, but to be correct, we should not prepend "www" to the domain name in this case.



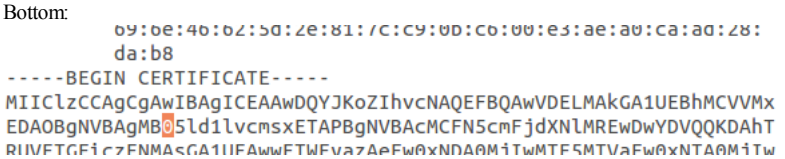
After removing "www" from /etc/hosts:



Modifying a single byre of server.pem:
There are 2 parts to edit: RSA private key (top) and the certificate part (bottom). To change a single bit easily, we find an ASCII character of '0' and change it to '1', effectively changing a single bit in the binary format of the file.
Top:



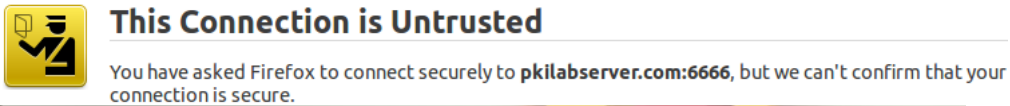
Bottom:



if we change the first part, it invalidates the private key. OpenSSL will throw this error:

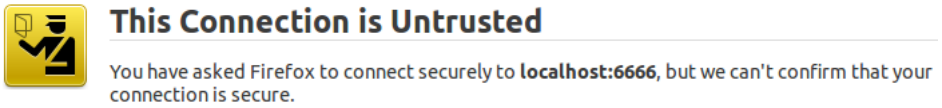
```
[04/29/2014 13:56] seed@ubuntu:~/Desktop/PKILab$ openssl s_server -cert server.p
em -www -accept 6666
Enter pass phrase for server.pem:
Using default temp DH parameters
Using default temp ECDH parameters
error setting private key
3073849544:error:0B080074:x509 certificate routines:X509_check_private_key:key v
alues mismatch:x509_cmp.c:331:
[04/29/2014 13:56] seed@ubuntu:~/Desktop/PKILab$
```

if we change the second part, it runs the server, but the certificate is practically different from the unmodified one:




```
[04/29/2014 13:56] seed@ubuntu:~/Desktop/PKILab$ openssl s_server -cert server.p
em -www -accept 6666
Enter pass phrase for server.pem:
Using default temp DH parameters
Using default temp ECDH parameters
ACCEPT
ACCEPT
ACCEPT
```

Using localhost instead of the domain name:
This will fail obviously since there is no way for the browser to know that pkilabserver was added in the /etc/hosts list as localhost, and the certificate was not signed for localhost, but for that specific domain, therefore it makes sense to act as if no certificate for that new domain (localhost) is present.



Task 4: Using PKI to establish secure TCP connections with PKILabServer.com

There are only 12 lines to add to make the client verify the CN of the server.
In this case I hardcoded the intended server, and added a check to retrieve the CN from the certificate, and compare it with the intended server.

```
#define HOSTNAME "PKILabServer.com"
```

```
[04/29/2014 17:27] seed@ubuntu:~/Desktop/PKILab/task4/demo_openssl_api$ ./cli
Enter PEM pass phrase:
SSL connection using AES256-GCM-SHA384
Server certificate:
         subject: /C=US/ST=New York/L=Syracuse/O=SEED Labs Inc./OU=PKI Lab/CN=PK
ILabServer.com/emailAddress=admin@PKILabServer.com
         issuer: /C=US/ST=New York/L=Syracuse/O=SEED Labs Inc./OU=PKI Lab/CN=PKI
LabCA.com/emailAddress=admin@PKILabCA.com
         CN: PKILabServer.com


 CN VERIFIED!
Got 11 chars:'I hear you.'
[04/29/2014 17:27] seed@ubuntu:~/Desktop/PKILab/task4/demo_openssl_api$
```

```
  X509_NAME *subjName;
  char subjCN[256];

  subjName = X509_get_subject_name(server_cert);
  X509_NAME_get_text_by_NID(subjName, NID_commonName, subjCN, sizeof(subjCN));
  printf("\t CN: %s\n", subjCN);

  if(strcmp (subjCN, HOSTNAME) == 0)
    printf("\n\n CN VERIFIED!\n");
  else {
    printf("\n\n CN ERROR!\n");
    return -1;
  }
```

Task 5: Performance Comparison: RSA versus AES

There is no accurate way to see time cause it prompts for pass phrase, and the time it takes is almost instantaneous.

```
[04/29/2014 15:38] seed@ubuntu:~/Desktop/PKILab/task5$ ls -l -h
total 12K
-rw-rw-r-- 1 seed seed 128 Apr 29 15:37 message_enc.txt
-rw-rw-r-- 1 seed seed  16 Apr 29 15:18 message.txt
-rw-rw-r-- 1 seed seed 963 Apr 29 15:17 server.key
[04/29/2014 15:38] seed@ubuntu:~/Desktop/PKILab/task5$ time openssl rsautl -encr
ypt -in message.txt -out message_enc.txt -inkey server.key
Enter pass phrase for server.key:

real    0m2.673s
user    0m0.000s
sys     0m0.000s
```

```
[04/29/2014 15:39] seed@ubuntu:~/Desktop/PKILab/task5$ time openssl rsautl -decr
ypt -in message_enc.txt -out message_dec.txt -inkey server.key
Enter pass phrase for server.key:

real    0m1.905s
user    0m0.000s
sys     0m0.004s
[04/29/2014 15:47] seed@ubuntu:~/Desktop/PKILab/task5$ cat message_dec.txt
000000000000000
[04/29/2014 15:47] seed@ubuntu:~/Desktop/PKILab/task5$
```

```
[04/29/2014 15:47] seed@ubuntu:~/Desktop/PKILab/task5$ time openssl aes-128-cbc
-in message.txt -out message_dec.txt
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:

real    0m10.736s
user    0m0.000s
sys     0m0.000s
```

When we run OpenSSL's speed argument, we get a big number of iterations in 10s for RSA and 3s for AES, which means that they are both fast, but the number of AES iterations in a shorter time is significantly higher, therefore this confirms that AES 128-bit is significantly faster than RSA 1024-bit, which was expected.

```
[04/29/2014 15:55] seed@ubuntu:~/Desktop/PKILab/task5$ openssl speed rsa
Doing 512 bit private rsa's for 10s: 90688 512 bit private RSA's in 9.90s
Doing 512 bit public rsa's for 10s: 994033 512 bit public RSA's in 9.91s
Doing 1024 bit private rsa's for 10s: 16594 1024 bit private RSA's in 9.91s
Doing 1024 bit public rsa's for 10s: 305670 1024 bit public RSA's in 9.89s
Doing 2048 bit private rsa's for 10s: 2472 2048 bit private RSA's in 9.90s
Doing 2048 bit public rsa's for 10s: 79810 2048 bit public RSA's in 9.91s
Doing 4096 bit private rsa's for 10s: ^C
[04/29/2014 15:56] seed@ubuntu:~/Desktop/PKILab/task5$ openssl speed aes
Doing aes-128 cbc for 3s on 16 size blocks: 22463563 aes-128 cbc's in 2.96s
Doing aes-128 cbc for 3s on 64 size blocks: 6344671 aes-128 cbc's in 2.97s
Doing aes-128 cbc for 3s on 256 size blocks: 1633730 aes-128 cbc's in 2.97s
Doing aes-128 cbc for 3s on 1024 size blocks: 800464 aes-128 cbc's in 2.97s
Doing aes-128 cbc for 3s on 8192 size blocks: 101566 aes-128 cbc's in 2.98s
Doing aes-192 cbc for 3s on 16 size blocks: ^C
[04/29/2014 15:56] seed@ubuntu:~/Desktop/PKILab/task5$
```

Task 6: Create Digital Signature

1. Sign the SHA256 hash of example.txt; save the output in example.sha256

```
[04/29/2014 16:05] seed@ubuntu:~/Desktop/PKILab/task6$ man dgst
[04/29/2014 16:06] seed@ubuntu:~/Desktop/PKILab/task6$ openssl dgst -sha256 -out
 example.sha256 -sign rsa.key example.txt
Enter pass phrase for rsa.key:
```

2. Verify the digital signature in example.sha256

```
[04/29/2014 16:25] seed@ubuntu:~/Desktop/PKILab/task6$ openssl rsautl -in exampl
e.sha256 -verify -asn1parse -inkey rsa.key
Enter pass phrase for rsa.key:
    0:d=0  hl=2 l=  49 cons: SEQUENCE
    2:d=1  hl=2 l=  13 cons:  SEQUENCE
    4:d=2  hl=2 l=   9 prim:   OBJECT            :sha256
   15:d=2  hl=2 l=   0 prim:   NULL
   17:d=1  hl=2 l=  32 prim:  OCTET STRING
      0000 - 6f 92 60 13 44 e1 68 51-31 6e 7c da 90 d0 53 c0   o.`.D.hQ1n|...S.
      0010 - ad 23 4e 04 7c cf 81 ce-6f e8 9e 78 bb db 11 1e   .#N.|...o..x....
[04/29/2014 16:25] seed@ubuntu:~/Desktop/PKILab/task6$ openssl dgst -sha256 exam
ple.txt
SHA256(example.txt)= 6f92601344e16851316e7cda90d053c0ad234e047ccf81ce6fe89e78bbd
b111e
[04/29/2014 16:25] seed@ubuntu:~/Desktop/PKILab/task6$
```

3. Slightly modify example.txt, and verify the digital signature again

The hash value in this case is naturally different. Digital signatures can therefore be used to verify file integrity.

```
[04/29/2014 16:25] seed@ubuntu:~/Desktop/PKILab/task6$ openssl rsautl -in exampl
e.sha256 -verify -asn1parse -inkey rsa.key
Enter pass phrase for rsa.key:
    0:d=0  hl=2 l=  49 cons: SEQUENCE
    2:d=1  hl=2 l=  13 cons:  SEQUENCE
    4:d=2  hl=2 l=   9 prim:   OBJECT            :sha256
   15:d=2  hl=2 l=   0 prim:   NULL
   17:d=1  hl=2 l=  32 prim:  OCTET STRING
      0000 - 6f 92 60 13 44 e1 68 51-31 6e 7c da 90 d0 53 c0   o.`.D.hQ1n|...S.
      0010 - ad 23 4e 04 7c cf 81 ce-6f e8 9e 78 bb db 11 1e   .#N.|...o..x....
[04/29/2014 16:28] seed@ubuntu:~/Desktop/PKILab/task6$ openssl dgst -sha256 exam
ple.txt
SHA256(example.txt)= f106f9b345c8968371ea0c167961e26d40c2262cddb2e3daf91d8f9744d
a80d2
[04/29/2014 16:28] seed@ubuntu:~/Desktop/PKILab/task6$
```