AfNOG Chix

Blantyre, Malawi
Security & Cryptographic Methods
Exercises with openssl
By Marcus K. G. Adomey

To check the version of openssl
$ openssl version

To ask for help
$ openssl –help
or
$ openssl –h

CONFIDENTIALITY

NOTE: password for this exercise: ***afchix***

Create a text file called **confile.txt**
$ echo "Honey, I have arrived in Blantyre. I miss you dearly." > confile.txt

Encrypt the file **confile.txt** with to **confile.enc** using 256-bit AES in CBC mode
$openssl enc -aes-256-cbc -salt -in confile.txt -out confile.enc

View the content of the encrypted file **confile.enc**
$ vi confile.enc

Your comments

Decrypt binary confile.enc
$ openssl enc -d -aes-256-cbc -in confile.enc

INTEGRITY CHECK

To check the integrity of a file, follow the following steps:

Computation of the message digest of the file **confile.txt**
$ openssl dgst -sha1 confile.txt

Make a copy of **confile.txt** and name it **confilecp.txt**
*$ **cp confile.txt confilecp.txt***
$ vi confilecp.txt

Modify the content by adding at the end of the content the name - ***Cucu***
The content of the new file **confilecp.txt** will look like this¨
Honey, I have arrived in Nairobi. I miss you dearly. - Cucu

Go through the message digest computation with the file **confilecp.txt**
$ openssl dgst -sha1 confilecp.txt

Compare the two message digests computed. Your comments!

NON-REPUDIATION AND AUTHENTICATION

Generation of pair Private/Public key

Generate a 2048-bit private-key

$ openssl genrsa -out private.key 2048

To view the content of your private key

$ vi private.key

Generate a public key from the generated private-key

$ openssl rsa -in private.key -out public.key -pubout

To view the content of your public key

$ vi public.key

Digital Signature

To sign

$ openssl dgst -sha1 -sign private.key -out confile.sign confile.txt

To verify signature

$ openssl dgst -sha1 -verify public.key -signature confile.sign confile.txt

Digital certificate

Generate a Certificate Signing Request

$ openssl req -new -newkey rsa:1024 -keyout hostkey1.key -nodes -out hostcsr1.csr

Create a Self-Signed Certificate from a Certificate Signing Request

$ openssl req -x509 -days 365 -in hostcsr1.csr –key hostkey1.key -out hostcert1.crt

Generate a Self-Signed Certificate from Scratch

$ openssl req -x509 -days 365 -newkey rsa:1024 -keyout hostkey2.key -nodes -out hostcert2.crt

---

NOTE :

Country Name (2 letter code) [AU]:	**MW**
State or Province Name (full name) [Some-State]:	**Blantyre**
Locality Name (eg, city) []:	**Blantyre**
Organization Name (eg, company) [Internet Widgits Pty Ltd]:	**AfNOGChix**
Organizational Unit Name (eg, section) []:	**SA-E**
Common Name (eg, YOUR name) []:	**Marcus Adomey**
Email Address []:	**madomey@hotmail.com**

Please enter the following 'extra' attributes to be sent with your certificate request

A challenge password []:	**afchix**
An optional company name []:	**NOG**

To view the certificate

$ openssl x509 -text -in hostcert2.crt

To view the one who issued the certificate?

$ openssl x509 -noout -in hostcert2.crt -issuer

To view the one to whom was it issued?

$ openssl x509 -noout -in hostcert2.crt -subject

To view what dates is it valid?

$ openssl x509 -noout -in hostcert2.crt -dates

To view the above, all at once

$ openssl x509 -noout -in hostcert2.crt -issuer -subject -dates

To view the hash value?

$ openssl x509 -noout -in hostcert2.crt -hash

To view the fingerprint?

$ openssl x509 -noout -in hostcert2.crt -fingerprint

Do the same with the certificate ***hostcert1.key***
Generate the public key for ***hostkey1.key*** and name it ***clientkey1.key***
Generate the public key for ***hostkey2.key*** and name it ***clientkey2.key***