Information Security

Asymmetric encryption - LAB

Lecturer: Nguyễn Thị Thanh Vân – FIT - HCMUTE

Objective

℠ Openssl

℠ Practice

Introduction

ઠ The openssl application that ships with the OpenSSL libraries can perform a wide range of crypto operations.

ઠ Download and install on Linux

ઠ Practice crypto operations

Commands

ઠ Version: openssl version

ઠ Performance: openssl speed

ઠ Digests: MD5, SHA1

ઠ Encryption/ Decryption

ઠ Keys

ઠ Password hashes

ઠ Prime numbers

ઠ Random data

Digests: MD5, SHA1

- ✍ Digests: MD5, SHA1
  - ○ openssl dgst -md5 filename
  - ○ openssl dgst -sha1 filename
  - ○ openssl dgst -sha256 filename
  - ○ md5sum filename
  - ○ sha1sum filename

11/14/2018   5

Encryption using different ciphers and modes

- ✍ The algorithm seems to follow the pattern:
  (Algorithm name)-(key size)-(encryption mode)
  Noted: If the key size is omitted or excluded then it means there is only one key-size for that algorithm.
- ✍ **Algorithm name:**
  - ○ RC2 and RC4.
  - ○ There are several encryption algorithm in OpenSSL, to see, use openssl enc –help
- ✍ **Key size:** key size is in bit. The longer the key the stronger your encryption is, but the slower operation it takes.
- ✍ **Encryption mode:**
  - ○ Electronic Codebook (ECB),
  - ○ Cipher Block Chaining (CBC),
  - ○ Cipher Feedback (CFB),
  - ○ Output Feedback (OFB), and
  - ○ Counter (CTR)

11/14/2018   6

3

Encrypt and Decrypt Text file

ജ Create a sub directory named "crypto_lab" in your home dir

ജ Create text file named plain.txt with whatever content that you like. For example "My name is …."

ജ Encrypt plain.txt, view the encrypted file with xxd then decrypt it with different aes cipher, in different modes.

ജ For example: to encrypt file plain.txt in aes-128 bit with cbc cipher with key –K and initialization vector -iv

```
openssl enc -aes-128-cbc -e -in plain.txt -out
cipher-aes-128-cbc.bin \ -K
00112233445566778889aabbccddeeff \ -iv
0102030405060708
```

ജ Try at least 3 different cipher modes and compare the results.

Encryption Mode – ECB vs. CBC

ജ Download a bitmap file from here to the crypto_lab directory, save the file name as origin.bmp

ജ Encrypt the file using the ECB (Electronic Code Book) and CBC (Cipher Block Chaining) modes, and then do the following:

   o Treat the encrypted picture as a picture, and use a picture viewing software to display it.

   o However, For the .bmp file, the first 54 bytes contain the header information about the picture, we have to set it correctly, so the encrypted file can be treated as a legitimate .bmp file.

   o We will replace the header of the encrypted picture with that of the original picture. You can use linux dd command to directly modify binary files.

ജ Display the encrypted picture using any picture viewing software. Can you derive any useful information about the original picture from the encrypted picture? Please explain your observations

Encryption/ Decryption, ex

- ಠ get a long list, one cipher per line
  - ○ openssl list-cipher-commands
- ಠ encrypt file.txt to file.enc using 256-bit AES in CBC mode
  - ○ openssl enc -aes-256-cbc -salt -in file.txt -out file.enc
- ಠ decrypt binary file.enc
  - ○ openssl enc -d -aes-256-cbc -in file.enc
- ಠ # decrypt base64-encoded version
  - ○ openssl enc -d -aes-256-cbc -a -in file.enc
- ಠ # provide password on command line
  - ○ openssl enc -aes-256-cbc -salt -in file.txt \ -out file.enc -pass pass:mySillyPassword
- ಠ # provide password in a file
  - ○ openssl enc -aes-256-cbc -salt -in file.txt \ -out file.enc -pass file:/path/to/secret/password.txt

11/14/2018

9

Generate keys

- ಠ Generate an RSA key
  - ○ openssl genrsa
- ಠ # 2048-bit key, saved to file named mykey.pem
  - ○ openssl genrsa -out mykey.pem 2048
- ಠ # same as above, but encrypted with a passphrase
  - ○ openssl genrsa -des3 -out mykey.pem 2048
- ಠ produce a public version of your private RSA key.
  - ○ openssl rsa -in mykey.pem -pubout

11/14/2018

10

sign a digest, verify a signed digest

- ✏ If you want to ensure that the digest you create doesn't get modified without your permission, you can sign it using your <u>private key</u>.
- ✏ # signed digest will be foo-1.23.tar.gz.sha1
    - ○ openssl dgst -sha256 \ -sign mykey.pem -out foo-1.23.tar.gz.sha1 \ foo-1.23.tar.gz
- ✏ To verify a signed digest you'll need the file from which the digest was derived, the signed digest, and the signer's <u>public key</u>.
- ✏ # to verify foo-1.23.tar.gz using foo-1.23.tar.gz.sha1 and pubkey.pem
    - ○ openssl dgst -sha256 \ -verify pubkey.pem \ -signature foo-1.23.tar.gz.sha1 \ foo-1.23.tar.gz

11/14/2018   11

Password hashes

- ✏ Ex:
    - ○ openssl passwd MySecret
      8E4vqBR4UOYF.
- ✏ generate a shadow-style password hash
    - ○ openssl passwd -1 MySecret
      1sXiKzkus$haDZ9JpVrRHBznY5OxB82.

11/14/2018   12

Prime numbers

- ഐ test whether a number is prime?
  - ○ openssl prime 119054759245460753
    1A6F7AC39A53511 is not prime
- You can also pass hex numbers directly.
  - ○ openssl prime -hex 2f
    2F is prime
- ഐ generate a set of prime numbers?
  - ○ openssl prime -generate -bits 64
    16148891040401035823
  - ○ openssl prime -generate -bits 64 -hex
    E207F23B9AE52181

Generate random data

- ഐ Use the rand option to generate binary or base64-encoded data.
- ഐ # write 128 random bytes of base64-encoded data to stdout
  - ○ openssl rand -base64 128
- ഐ # write 1024 bytes of binary random data to a file
  - ○ openssl rand -out random-data.bin 1024

Encryption Mode – Corrupted Cipher Text

~ Do exercise:
- o Create a text file that is at least 64 bytes long.
- o Encrypt the file using the AES-128 cipher.
- o Unfortunately, a single bit of the 30th byte in the encrypted file got corrupted. You can achieve this corruption using dd command
- o Decrypt the corrupted file (encrypted) using the correct key and IV.

~ Please answer the following questions:
- o (1) How much information can you recover by decrypting the corrupted file, if the encryption mode is ECB, CBC, CFB, or OFB, respectively? Please answer this question before you conduct this task, and then find out whether your answer is correct or wrong after you finish this task.
- o (2) Please explain why.
- o (3) What are the implication of these differences?