Asymmetric encryption - LAB

Lecturer: Nguyễn Thị Thanh Vân – FIT - HCMUTE

Objective

- Openssl
- Practice

Introduction

- The openssl application that ships with the OpenSSL libraries can perform a wide range of crypto operations.
- Download and install on Linux
    - yum install openssl
- Practice crypto operations

Commands

- Version: openssl version
- Performance: openssl speed
- Digests: MD5, SHA1
- Encryption/ Decryption
- Keys
- Password hashes
- Prime numbers
- Random data

Digests: MD5, SHA1

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

Encryption/ Decryption

- get a long list, one cipher per line

  -
    ```
    openssl list-cipher-commands
    ```
- 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
    ```
- decrypt binary file.enc

  -
    ```
    openssl enc -d -aes-256-cbc -in file.enc
    ```
- # decrypt base64-encoded version

  -
    ```
    openssl enc -d -aes-256-cbc -a -in file.enc
    ```
- # provide password on command line

  -
    ```
    openssl enc -aes-256-cbc -salt -in file.txt \ -out file.enc -pass pass:mySillyPassword
    ```
- # provide password in a file

  -
    ```
    openssl enc -aes-256-cbc -salt -in file.txt \ -out file.enc -pass file:/path/to/secret/password.txt
    ```

Generate keys

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

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 private key.
- # 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 public key.
- # 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

Password hashes

- Ex:
  - openssl passwd MySecret

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

Prime numbers

- 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
    ```

- generate a set of prime numbers?

  -
    ```
    openssl prime -generate -bits 64
    16148891040401035823
    ```

  -
    ```
    openssl prime -generate -bits 64 -hex
    E207F23B9AE52181
    ```

Generate random data

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

Q & A