INFO1112 Week 9 Tutorial

Security and cryptography

1: Really silly encryption

Let’s create a very simple (and not very secure) encryption method for encrypting text.

ROT13 is the name of a simple encryption scheme. ROT13 simply stands for Rotate-by-13. To encrypt a message you replace each letter with the letter 13 places further on in the alphabet, wrapping to the beginning if necessary.

Using the English alphabet, create a python program that will encrypt a file using ROT13.

```
$ python3 rot13.py
welcome to info1112
jrypbzr gb vasb1112
```

2: Symmetric Encryption

Most linux distributions provide a copy of OpenSSL. OpenSSL is a software library and application suite for secure communications over computer networks. With OpenSSL we are able to encrypt and decrypt files using the openssl command.

Using the openssl command, encrypt a file with aes-256-cbc encryption method and send it to your friend, you will need to provide your friend the decryption key so they are able to read the file.

To encrypt, use the following command string template:

```
openssl <encryption method> -in <input file> -out <output file>.
```

You are able to decrypt the file using the -d flag and changing the -in and -out flags.

3: Using GPG and Asymmetric Encryption

Similar to openssl, the gpg command can be used to generate a public-private key pair.

Generate a public-private key pair and distribute your public key to your friend to use. Once you have exchanged public keys, send the encrypted message over email and decrypt it.

Use the following command to generate a public-private key pair

```
gpg --gen-key
```

Afterwards, list the keys you have generated and export the public key

```
gpg --list-keys
gpg --output public_key.gpg --export <your key id here>
```

Exchange public keys with your friend, encrypt a message to send to them and encrypt it with the following command

```
gpg --encrypt-files --recipient-file <your friend’s public key>.gpg
<file to send>
```

Alternatively, you may be using an older version of gpg where you will need to import the gpg key. You can use gpg --import <public key file> and use their email as an identifier to encrypt like so gpg --encrypt-files --recipient <email associated with key> <file to send>.

Afterwards, you can send the file to your friend and decrypt the file they have sent to you with the following command

```
gpg <file your friend has sent>
```

- What is the advantage of asymmetric encryption vs symmetric encryption?
- What applications would symmetric encryption and asymmetric encryption?

4: Hashes and more hashes

In the previous questions we have seen the use of encryption. However we can utilize hashes within our communication and storage mechanisms.

Hashes allows for generating a signature of variable sized data to fixed sized data known as a hash. Hashes are used for a variety of areas such as data structures, password storage, data integrity verification.

Use the firmware file from canvas (firmware.bin) and the sha256 or sha256sum command to generate the hash for the firmware.bin file.

5: Comparing file integrity

You are working for a software company and you are attempting to track down an issue that your users have with downloaded firmware.

There are a number of company mirrors that host this file and you need to figure out what mirror is currently hosting a broken/invalid version of the firmware.

You have been given the file firmware.bin which acts as your source of truth. Write a script that compares the hash between firmware.bin and the firmware files on the mirrored endpoints.

You can access the mirrored files on canvas which are under the name firmware1.bin, firmware2.bin, firmware3.bin, firmware4.bin. Use the sha256 or sha256sum command to extract the hash of the file.

- What is the difference between encryption and hashing?

Extension

6: Password storage

Inspect the `/etc/passwd` or `/etc/shadow` file on your system (or the one we have provided). You may notice that a user is associated with a hash which is the hash of their password.

Recreate a simple login system using python by allowing the user to login with a username and password, comparing it to an existing password file with their user and hash.

Use the bcrypt module.

- What do you observe between sha256 and bcrypt?
- What domains would you use sha256 and bcrypt?