Cryptographic Methods with GPG

Reading

15m - 35m

Status | Incomplete

Introduction

GnuPG, popularly known as GPG, is a highly versatile tool, being widely used as the industry standard for encrypting things like emails, messages, files, or anything you need to send to someone securely.

In this reading, you will learn how to encrypt and decrypt files with GPG. What you will try is a simple lesson you can practice in the Linux VM, which will help you understand the GPG commands.

How Does GPG Work for Encryption?

GPG keys work with two files, a private key and a public key. These two keys are linked to each other and can use GPG's functionality to encrypt and decrypt files. When encrypting a file with GPG, it uses the private key. The newly encrypted file can later be decrypted.

The private key must be stored directly on your behalf, privately, and not given to anyone else. The public key, on the other hand, must be given to others or anyone you want to be able to decrypt your files. This is where GPG's primary approach to encryption comes into play. It allows you to encrypt files locally and then assure others that the files received were sent from you. As the only way for them to decrypt the file is with their public key, that would only work if the file is encrypted using their private key in the first place.

It also works in the opposite direction. Other people can encrypt files using your public key; the only way it can be decrypted is with your private key, and allows others to post files publicly without worrying about people other than you being able to read them.

Encrypting and Decrypting Files with GPG

To start using GPG, you must first have a GPG key. You'll use a GPG key to encrypt (or decrypt) files. It's also what's used to identify you, with things like your name and email linked to the key.

TIP: remember, the "$" signal just informs you that you have to use the Linux terminal. You don't have to type it together with the command.

Step 1 - Install GPG

GPG can be found in the repositories of most Linux distributions out of the box.

In the Linux VM, install the GPG package using the command in the MATE Terminal.

```
$ sudo apt install gpg
```

Step 2 - Generating a GPG Key

After installing the package, you will generate the GPG key on your system, a simple one-command procedure (you can use defaults for most questions).

```
$ gpg --full-generate-key
```

This will take you through the process of creating a GPG key. It will ask you for some information. The first thing will ask which kind of key we want. In this case we will select (1) RSA and RSA (Default). Then, we will leave the key size at 3072 bits. Let's also set it to never expire (0). Enter your information now such as name and email address when prompted. Then it will say it is generating the key. You can speed it up by typing on your keyboard or moving your mouse around.

After creating your key, you can then see that the private key and public key are both linked to each other by this ID shown in the pub using the parameters `--list-secret-keys` and `--list-public-keys`, respectively.

```
$ gpg --list-secret-keys
$ gpg --list-public-keys
```

Step 3 - Encrypt a File with GPG

Now that you've set up your GPG keys, you can start encrypting files.

First, create your test text file with the command `ECHO`.

```
$ echo "LightHouse Labs Cyber Security BootCamp" > file.txt
$ gpg --encrypt --output encrypted_file.gpg --recipient student@email.com file.txt
```

So, what does this command do?

First, you specify the `--encrypt` option and simply tell GPG that you will encrypt a file.

Then you specified `--output` encrypted_file.gpg. The name can be anything, although it will typically be the name of the file you are encrypting plus a `.gpg` extension (so `message.txt` would become `message.txt.gpg`).

Then you type `--recipient student@email.com`, which specifies the email for a corresponding GPG key that doesn't exist on this system.

A bit confusing?

It works because the email you specify here must be linked to a public key on your local system. Typically, this will be a different person's public GPG key, which you will encrypt your file with. After that, the file can only be decrypted with that user's private key. You would only have the public key if you were encrypting a file for someone else, but since you are encrypting the file, you have both keys on your system.

Finally, you simply specify the file you are going to encrypt. For this example, use a file called `file.txt` with the following content:

LightHouse Labs Cyber Security BootCamp

You can verify the information inside the file you created before using the command CAT.

```
$ cat file.txt
```

And after you finish step 3, you can verify that you have two different files. You can use the command LS to list all files inside the directory.

```
$ ls
```

If you try to read the file `encrypted_file.gpg` with CAT, you'll see that it looks all messed up with binary information.

```
$ cat encrypted_file.gpg
```

It is expected because the file is now encrypted.

Now, delete the unencrypted `file.txt` file so you can see that the `encrypted_file.gpg` file decrypts just fine without the original file:

```
$ rm file.txt
```

Step 4 - Decrypting the GPG Encrypted File

Lastly, decrypt the encrypted message. You can do this using the following command:

```
$ gpg --decrypt --output decrypt_file.txt encrypted_file.gpg
```

About the arguments, you first specify `--decrypt`, which tells GPG that you're going to decrypt a file. Then you type the `--output` file, which tells GPG which allows you to specify an output file to put your decrypted text into.

Finally, you type `encrypted_file.gpg`, which is the path to your encrypted file that you are decrypting.

You can check that the `decrypt_file.txt` file has the same content as the original `file.txt`.

```
$ cat decrypt_file.txt
```

Reference

[The GNU Privacy Handbook - Encrypting and decrypting documents]

✓ Mark Completed

Previous
Cryptographic Encryption

How well did this activity help you to understand the content?

Let us know how we're doing

☆ ☆ ☆ ☆ ☆

W06D5

Fri Aug 2

Outline & Notes (1)

Lectures (1)

Work (10)

7 hrs

- Cryptanalysis
- The Use of the Historical vs. Modern Encryption
- Cryptanalysis Quiz
- Cryptography Features and Objectives
- Typical Levels of Cryptography
- The Use of Cryptography
- Code a Python Playfair Cipher
- Code a Python Playfair Cipher
- Cryptographic Encryption
- Cryptographic Methods with GPG

W06D5 Schedule »