Ghost in the Machines

silently passing through the ethernet …

How To Use GPG on the Command Line

Posted on **March 1, 2015**

Introduction

I use GPG (also known as GnuPG) software for encrypting files that contain sensitive information (mostly passwords). As a systems engineer, I do most of my work on remote servers, accessible via command line interface. Naturally, I find it easier to use the command line version of GPG to directly encrypt and decrypt documents.

GPG (GNU Privacy Guard) is a free open source version of PGP (Pretty Good Privacy) encryption software. Conceptually, both use the same approach to cryptography (i.e. encryption and decryption). However, each is uniquely different in its implementation.

What follows is a quick primer on how to install the GPG command line tools, as well as a list of basic commands you are most likely to need.

Installing GPG

GPG can be installed in a number of different ways. The instructions here will install the core GPG command line tools, which are intended to be used in a terminal.

If, on the other hand, you prefer a graphical user interface (or GUI) for accessing GPG functionality (e.g. encrypting email communications, or encrypting documents in a GUI text editor), refer to the links at the end of this article.

Red Hat / CentOS

```
yum install gnupg
```

Ubuntu / Debian

```
apt-get install gnupg
```

Mac OS X

The easiest way to install the GPG command line tools on your Mac is to first install Homebrew, a package management system that makes thousands of software packages available for install on your Mac.

Open a Terminal window (Applications > Utilities menu), then enter the following command.

```
y -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/maste
```

When that's complete, install the GPG software package with the following command.

```
brew install gnupg
```

GPG Quick How To

What follows is a very brief introduction to command line usage of GPG. Think of it as a "quick reference" or a "cheat sheet." You should certainly learn more about GPG than what is explained within this post. It is intended only to get you started. If you expect to use GPG more extensively, I strongly advise you to read more documentation (see the Links section below).

GPG is powerful encryption software, but it can also be easy to learn — once you understand some basics. GPG uses a method of encryption known as public key cryptography, which provides a number of advantages and benefits. However, to obtain these advantages, a minimal level of complexity is required to make it all work. For an overview of how public key cryptography works, read the Introduction to Cryptography (link at the bottom of this post).

Typographical conventions used in commands:

In all examples below, text that you will need to replace with your own values (e.g. usernames, email addresses, filenames) is shown in *"gray italic"*. Text that you will type literally (unchanged) is indicated with "black constant width".

```
"gray italic"
"black constant width"
```

Create your GPG key:

To get started with GPG, you first need to generate your key pair. That is, you will generate both a private and a public key with a single command. Enter your name and email address at the prompts, but accept the default options otherwise.

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

The first key is your private (or secret) key. You must keep this private key safe at all times, and you must not share it with anyone. The private key is protected with a password. Try to make the password as long as possible, but something you will not forget. If you forget the password, there's no way to recover it. For the same reason, you should also make a backup copy of your private key. (Consider using Time Machine for backups on Mac OS X.)

The second key is your public key, which you can safely share with other people.

The relationship of the private and public key is actually very simple. Anything that is encrypted using the public key can only be decrypted with the related private key. Therefore, you will provide your public key to another person, and they will provide you with their public key. Anything encrypted to your public key can only be decrypted by you. Anything encrypted to the other person's public key can only be decrypted by the other person.

Export your public key:

The next step is to export your public key and share it with another person. That person should do the same, and export their public key.

```
gpg --export --armor youremail@example.com > mypubkey.asc
```

Import another person's public key:

When you import a public key, you are placing it into what is commonly referred to as your GPG "keyring."

```
gpg --import theirpubkey.asc
```

List the public keys in your keyring:

You can now view a list of public keys in your keyring, as well as the name and email address associated with each key.

```
gpg --list-keys
```

List private keys in your keyring:

The following command will list the private keys in your keyring. This will show your own private key, which you created earlier.

```
gpg --list-secret-keys
```

Trust a public key:

Once you have imported the other person's public key, you must now set the trust level of the key. This prevents GPG from warning you every time you encrypt something with that public key.

Specify the other person's name or email in the command.

```
gpg --edit-key glenn

trust (invoke trust subcommand on the key)
5 (ultimate trust)
y (if prompted)
quit
```

Useful GPG Commands

GPG has many options, most of which you will never need. Here's a quick list of the most useful commands you are likely to need.

Encrypt a file:

To encrypt a file named *filename.txt* for a single individual, specify that individual as a recipient.

```
gpg --encrypt --recipient glenn filename.txt
```

This will create a new encrypted file named *filename.txt.gpg*.

If you want to encrypt a file so that only you yourself can decrypt it, then specify yourself as the recipient.

```
gpg --encrypt --recipient 'my_name' filename.txt
```

If you want to encrypt a file so that both you and another person can decrypt the file, specify both you and the other person as recipients.

```
gpg --encrypt --recipient glenn --recipient 'my_name' filename.txt
```

If you want to encrypt a file for a group of people, define the group in your gpg.conf file (see section below), and then specify the group as a recipient.

```
gpg --encrypt --recipient journalists filename.txt
```

After a while, you'll want to be more concise and use the short version of the command line options. Here's the same command.

```
gpg -e -r journalists filename.txt
```

Decrypt a file to terminal (standard output):

The first version of this command will display the content of a file within the terminal window itself.

```
gpg --decrypt filename.txt.gpg
```

Use the –decrypt option only if the file is an ASCII text file. If it's a binary file, then omit the –decrypt option, which will write the decrypted file to disk. At that point, you can open the binary file in whatever application is used to view the file.

Decrypt a file to disk:

Whether the file is ASCII or binary, if you want to make changes to the content of an encrypted file, you must first decrypt it, make your changes, then re-encrypt the file. As I mentioned in the previous paragraph, you write the decrypted version of a file to disk, by omitting the –decrypt option from the command.

```
gpg filename.txt.gpg
```

If the encrypted file was named *filename.txt.gpg*, the above command will create a decrypted version named *filename.txt* (with the .gpg extension removed).

Create Groups of People in Your GPG Configuration File

For convenience, you can pre-define a group of people in your GPG configuration file. This has the benefit of allowing you to encrypt a file to every member of the group by specifying only the group name as the recipient, rather than tediously specifying every individual member of the group.

Your GPG software configuration is stored in your home directory within the ~/.gnupg/gpg.conf file. Edit this file using your favorite command line text editor (vim, nano, pico, emacs, etc). While there are numerous settings available in the configuration file, go to the section pertinent to defining groups.

When defining a group, you list the members of the group. Each member is referenced by some attribute of their public key found in your GPG keyring — typically a person's name (or partial name, such as first or last name) or an email address (or partial email address).

If you are a member of the group, remember to include yourself in the group! If you do not list yourself in the group, you won't be able to decrypt any files you encrypt to the group.

Here's an example of a group named "journalists", listing the first name of each person.

```
group journalists = glenn laura ewan barton
```

Where To Go From Here

I encourage you to learn more about GPG. See the Links below.

You may also want to learn about secure methods to erase files from your computer hard drive. Mac OS X has the "Secure Empty Trash" option within Finder. There are also numerous third-party tools you can install.

Since we're on the theme of learning how to use GPG in the command line, you may want to try "bcwipe" — a program to securely erase files within the command line.

On Mac OS X, you can install bcwipe via Homebrew.

```
brew install bcwipe
```

Links

General

- Introduction to Cryptography (PDF)
- GnuPG Software Website
- GnuPG on Wikipedia
- Homebrew (Package Manager for Mac OS X)

GUI Tools

- GPG Suite (GUI for Mac OS X)
- How To Use GPG Suite on Mac OS X (Electronic Frontier Foundation)
- Gpg4win (GUI for Windows)
- How To Use Gpg4Win on Windows (Electronic Frontier Foundation)

This entry was posted in **software**, **sysadmin** by **james**. Bookmark the **permalink [http://blog.ghostinthemachines.com/2015/03/01/how-to-use-gpg-command-line/]** .