

# A Practical Guide to GPG Part 5: Digital Signature - LinuxBabe

*Xiao Guoan (Admin)*

4-5 minutes

---

In [previous GPG tutorials](#), I explained using GPG for encryption. In this part, you will learn how to use GPG to add digital signatures on documents. GPG provides two modes of signing:

- Signing without encryption
- Signing with encryption

## Signing without Encryption

GPG can sign a document without encrypting it. Sometimes you may want to send someone a document and you don't care whether the content is confidential or not but you want to make sure that the person who receives it knows the document comes from you and also hasn't been tampered with.

We can do so by signing a document with our private key in the following syntax.

```
gpg --clearsign /path/to/file
```

Let's create a test txt file.

```
echo "This is a test text file" > test.txt
```

Then add a digital signature on this file.

```
gpg --clearsign test.txt
```

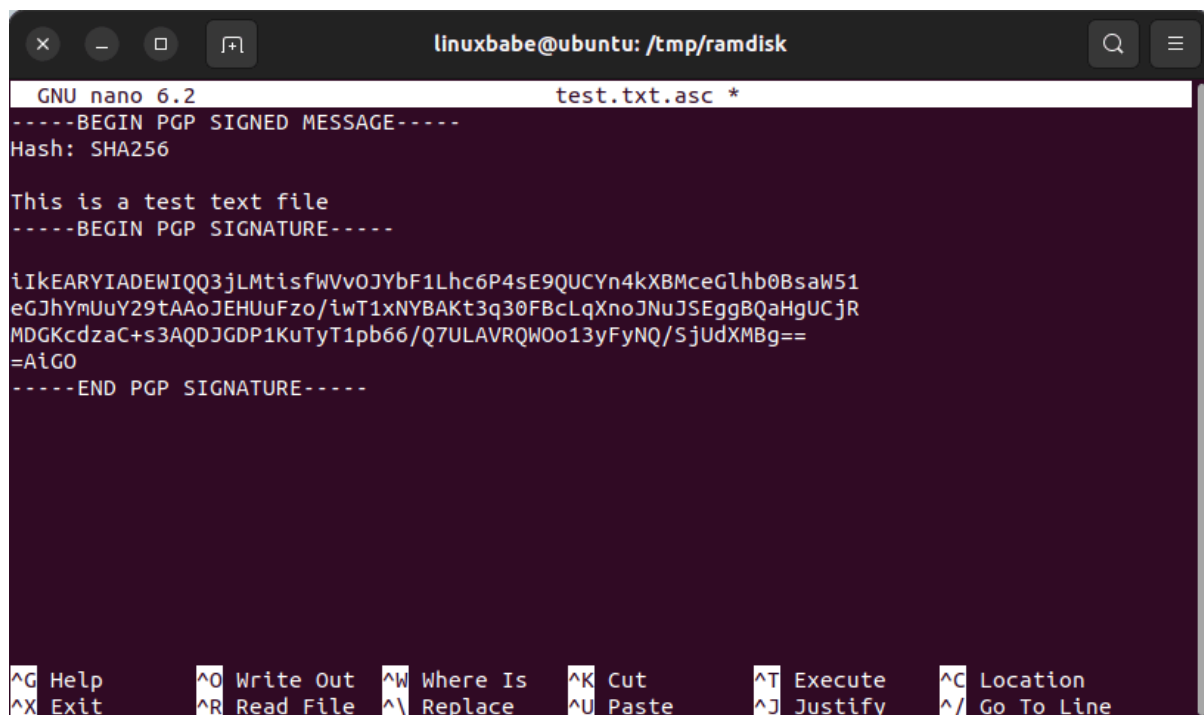
- The `--clearsign` option means to make a clear text signature. The content will still be readable without using any special software.

OpenPGP software is only needed to verify the signature.

- GPG uses your private key to create the digital signature so you will be prompted to enter your passphrase to unlock the private key. When the command is completed, a new file with .asc extension is created.
- If you have multiple GPG key pair, then you can select a specific private key with: `gpg --local-user user-id --clearsign test.txt`.

There're some interesting tidbits you can get from this .asc file, so open it with your text editor such as Nano.

`nano test.txt.asc`

A screenshot of a terminal window titled 'linuxbabe@ubuntu: /tmp/ramdisk'. The window shows the nano 6.2 text editor editing a file named 'test.txt.asc'. The content of the file is a PGP signed message. It starts with '-----BEGIN PGP SIGNED MESSAGE-----', followed by 'Hash: SHA256'. Then, the plaintext 'This is a test text file' is shown, followed by '-----BEGIN PGP SIGNATURE-----'. The signature itself is a long block of base64-encoded text: 'iIkEARYIADEWIQQ3jLMtisfWVvOJYbF1Lhc6P4sE9QUCYn4kXBMceG1hb0Bsaw51eGJhYmUuY29tAAoJEHUuFzo/iwT1xNYBAkt3q30FBcLqXnoJNuJSEggBQaHgUCjRMDGKcdzaC+s3AQDJGDP1KuTyT1pb66/Q7ULAVRQWoo13yFyNQ/SjUdXMBg=='. This is followed by '=AiGO' and '-----END PGP SIGNATURE-----'. At the bottom of the terminal, the nano editor's command shortcuts are displayed: ^G Help, ^O Write Out, ^W Where Is, ^K Cut, ^T Execute, ^C Location, ^X Exit, ^R Read File, ^\_ Replace, ^U Paste, ^J Justify, and ^/ Go To Line.

You will find the hash algorithm used. In my case GPG used SHA256 as the hash algorithm which is strong enough. Note that SHA1 is obsolete. The PGP signature is placed right after the original content. The plaintext file is first hashed, then your private key is used to sign the hash.

Exit the Nano text editor by pressing `Ctrl+X`.

## Verify the Signature

In part 3, we created a test user account, and imported the public

key, so let's move the signature file to the test user's home directory.

```
sudo mv test.txt.asc /home/test/
```

Then switch to the test user account.

```
su - test
```

Verify the signature with the following command:

```
gpg --verify test.txt.asc
```

The verification does two things.

- It makes sure that the file has not been changed somehow in transit. There's a hash calculated and then the hash is signed with the private key. By signing with the private key, it allows the recipient to determine the authenticity of the sender.
- During the verification process, GPG determines what key (key ID) is used to sign the document and then use the corresponding public key from public keyring to verify the signature.

If it say "Good signature", then the verification is successful, so you know this file hasn't been tampered with and is really come from the sender.

```
test@ubuntu:~$ gpg --verify test.txt.asc
gpg: Signature made Fri 13 May 2022 05:26:52 PM +08
gpg:      using EDDSA key 378CB32D8AC7D656F38961B1752E173A3F8B04F5
gpg:      issuer "xiao@linuxbabe.com"
gpg: Good signature from "Xiao Guoan <xiao@linuxbabe.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:      There is no indication that the signature belongs to the owner.
Primary key fingerprint: 378C B32D 8AC7 D656 F389 61B1 752E 173A 3F8B 04F5
test@ubuntu:~$
```

## Signing with Encryption

If we want to make sure that the authenticity of the sender is checked as well as the integrity of the encrypted message and we want to provide confidentiality, use the below command:

```
gpg --armor --recipient user-id -e --sign /path/to/file
```

Where

- - - armor: Create ASCII armored output. The default is to create the

binary OpenPGP format.

- `user-id` is the recipient's email address.
- `-e`: Encrypt the file.

You need to enter your passphrase to unlock the private key. It combines the commands to sign and encrypt in one step. It first signs the document with your private key, then encrypts it with the recipient's public key.

## Decrypt and Verify

The decrypt process will automatically verify the signature.

```
gpg --decrypt filename.asc
```

## Next Step

In the next part, we will learn how to verify software downloads on Linux.

- [How to Verify PGP Signature of Downloaded Software on Linux](#)