

UT01: Adopción de pautas de seguridad informática – 3) – Técnicas De Cifrado.

Nombre: Francisco Jesús García – Uceda Díaz

Curso: 2º ASIR.

Índice

a) Cifrado: Historia.....	2
- Realizar CISCO CCNA Security 2.0. Laboratorio: Explorando métodos de cifrado.	2
b) Cifrado: Funciones HASH--> Integridad y Autenticidad.	3
- 1. Utiliza un programa en Windows o GNU/Linux para simular la integridad, utilizando MD5 y SHA1.....	3
- 2. Utiliza un programa en Windows o GNU/Linux para simular la autenticidad utilizando HMAC-MD5, HMAC-SHA1	6
c) Cifrado: Simétrico y Asimétrico--> Confidencialidad, Integridad y Autenticidad.	8
- 1. Utiliza un programa en Windows o GNU/Linux para simular la confidencialidad mediante “cifrado simétrico”.	8
- 2. Utiliza un programa en Windows o GNU/Linux para simular la confidencialidad mediante “cifrado asimétrico”.	10
- 3. Utiliza un programa en Windows o GNU/Linux para simular la autenticidad mediante “cifrado asimétrico”.....	17
- 4. Utiliza un programa en Windows o GNU/Linux para simular la “autenticidad”+“confidencialidad” mediante “cifrado asimétrico”.....	19
- 5. Utiliza un programa en Windows o GNU/Linux para simular una comunicación segura utilizando cifrados híbridos “autenticidad” + “confidencialidad”+“integridad”: asimétricos (clave pública) y simétricos (clave privada).	21
Conclusión	25

Introducción

En esta práctica aprenderemos sobre el cifrado, para ello realizaremos técnicas de cifrado y veremos la historia que tienen.

a) CIFRADO: Historia.

- Realizar CISCO CCNA Security 2.0. Laboratorio: Explorando métodos de cifrado.

[Link de cómo realizarlo \(Cisco\).](#)

El cifrado Vigenère es un cifrado basado en diferentes series de caracteres o letras del cifrado César formando estos caracteres una tabla, llamada tabla de Vigenère, que se usa como clave. El cifrado de Vigenère es un cifrado de sustitución simple poli alfabético. El cifrado Vigenère se ha reinventado muchas veces.

Encriptación

Cipher Keyword	F	R	A	N	V	I	G	S	A	D
Encrypted Message	R	O	U	T	E	R	R	O	U	T
Decrypted Message	W	F	U	G	Z	Z	X	G	U	W

Desencriptación

Cipher Keyword	W	F	U	G	Z	Z	X	G	U	W
Encrypted Message	R	O	U	T	E	R	R	O	U	T
Decrypted Message	F	R	A	N	V	I	G	S	A	D

Página que recomienda Cisco para probar esto: [Vignere Cipher](#)

The screenshot shows the 'Sharky's Vignere Cipher' web application interface. At the top, it says 'Sharky's Vignere Cipher' and '[Sharkysoft home]'. Below that, a note reads 'This page is for amusement only. Instructions are given below this form.' The main form has four sections: 'Input:' with a text area containing 'FRANVIGSAD' and a 'clear' button; 'Key:' with a text area containing 'Router' and a 'clear' button; 'Coding direction:' with two buttons, 'encode' (selected) and 'decode'; and 'Output:' with an empty text area and a 'clear' button. Below the form, the 'instructions:' section contains two numbered steps: '1. Enter the string to encrypt or decrypt in the Input field (you may copy and paste it from another text editor).' and '2. Enter the key in the Key field. (You may use any sequence of characters, but only alphabetic characters will actually be processed. The key "Hey, you!" is equivalent to the key "hEyYoU".)'

This screenshot shows the same 'Sharky's Vignere Cipher' web application interface, but with the 'decode' button selected under 'Coding direction:'. The 'Input:' field now contains the ciphertext 'hFUGZZXGLM'. The 'Output:' field is empty, indicating that the decryption result has not yet been displayed. The 'Key:' field still contains 'Router'. The 'instructions:' section remains the same as in the previous screenshot.

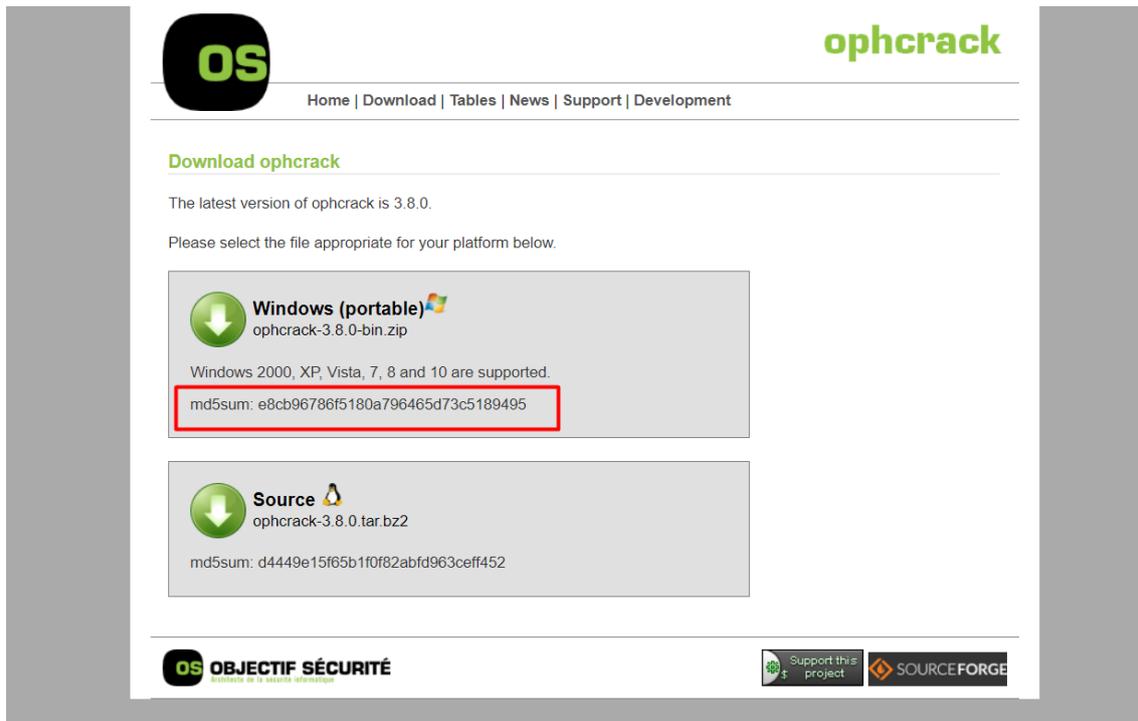
b) CIFRADO: Funciones HASH--> Integridad y Autenticidad.

- 1. Utiliza un programa en Windows o GNU/Linux para simular la integridad, utilizando MD5 y SHA1.

Para realizar esto usare PowerShell. PowerShell es una interfaz de consola (CLI con posibilidad de escritura y unión de comandos por medio de instrucciones (script)). Esta interfaz de consola está diseñada para su uso por parte de administradores de sistemas, con el propósito de automatizar tareas o realizarlas de forma más controlada.

Lo realizaré con PowerShell porque siempre viene instalado en los sistemas operativos Windows 7 en adelante la función que tenemos y no siempre podemos instalar un programa para comprobar la integridad de otro programa en MD5 o SHA1.

En mi caso lo usare comprobare la integridad de la versión portable de Ophcrack. Podemos ver el md5 desde su misma página web.



The screenshot shows the Ophcrack website interface. At the top left is the 'OS' logo, and at the top right is the 'ophcrack' logo. Below the logo is a navigation menu with links: Home | Download | Tables | News | Support | Development. The main heading is 'Download ophcrack'. Below this, it states 'The latest version of ophcrack is 3.8.0.' and 'Please select the file appropriate for your platform below.' There are two download options: 'Windows (portable)' with a download icon and the file name 'ophcrack-3.8.0-bin.zip', and 'Source' with a download icon and the file name 'ophcrack-3.8.0.tar.bz2'. The MD5 hash for the Windows (portable) version is highlighted with a red box: 'md5sum: e8cb96786f5180a796465d73c5189495'. The MD5 hash for the Source version is 'md5sum: d4449e15f65b1f0f82abfd963ceff452'. At the bottom, there are logos for 'OBJECTIF SÉCURITÉ' and 'SUPPORT THIS PROJECT SOURCEFORGE'.

Una vez guardado el programa simplemente abrimos PowerShell y ejecutamos el siguiente comando para sacar el hash del archivo:

```
Get-FileHash C:\ruta\al\archivo.iso -Algorithm MD5
```

También lo podremos hacer con multitud de algoritmos:

```
Get-FileHash C:\ruta\al\archivo.iso -Algorithm SHA1
```

```
Get-FileHash C:\ruta\al\archivo.iso -Algorithm SHA256
```

```
Get-FileHash C:\ruta\al\archivo.iso -Algorithm SHA384
```

```
Get-FileHash C:\ruta\al\archivo.iso -Algorithm SHA512
```

```
Get-FileHash C:\ruta\al\archivo.iso -Algorithm MACTripleDES
```

```
Get-FileHash C:\ruta\al\archivo.iso -Algorithm RIPEMD160
```

Vemos como nos genera un hash.

```
Administrador: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\WINDOWS\system32> Get-FileHash Z:\pruebas\ophcrack-3.8.0-bin.zip -Algorithm MD5

Algorithm      Hash                                                    Path
-----
MD5            E8CB96786F5180A796465D73C5189495                    Z:\pruebas\ophcrack-3.8.0-bin...
```

Podremos hacer lo mismo en SHA1.

```
PS C:\WINDOWS\system32> Get-FileHash Z:\pruebas\ophcrack-3.8.0-bin.zip -Algorithm SHA1

Algorithm      Hash                                                    Path
-----
SHA1          F7975E274BC4344080CAB23408F78E9AD659E857            Z:\pruebas\ophcrack-3.8.0-bin.zip
```

Ahora simplemente tenemos que comprar los dos hashes (el de PowerShell con el que ofrece la página Ophcrack. Vemos que son iguales por lo cual la integridad es correcta.

Home | Download | Tables | News | Support | Development

Download ophcrack

The latest version of ophcrack is 3.8.0.

Please select the file appropriate for your platform below.

Windows (portable)
ophcrack-3.8.0-bin.zip

Windows 2000, XP, Vista, 7, 8 and 10 are supported.

md5sum: e8cb96786f5180a796465d73c5189495

```
Administrador: Windows PowerShell
PS C:\WINDOWS\system32> Get-FileHash Z:\pruebas\ophcrack-3.8.0-bin.zip -Algorithm MD5

Algorithm      Hash                                                    Path
-----
MD5            E8CB96786F5180A796465D73C5189495                    Z:\pruebas\ophcrack-3.8.0-bin.zip
```

Para probar SHA1 al no encontrar programas que brinden en su página web el SHA1 para comprobar la integridad lo que haré es crear un archivo txt con algo escrito, generar su SHA1 y después modificarlo para que se vea cómo cambia.

Crearemos un documento de texto y generamos su SHA1.

```
prueba1.txt: Bloc de notas
Archivo Edición Formato Ver A
DOCUMENTO SHA1

Administrador: Windows PowerShell
PS Z:\pruebas> ls

Directorio: Z:\pruebas

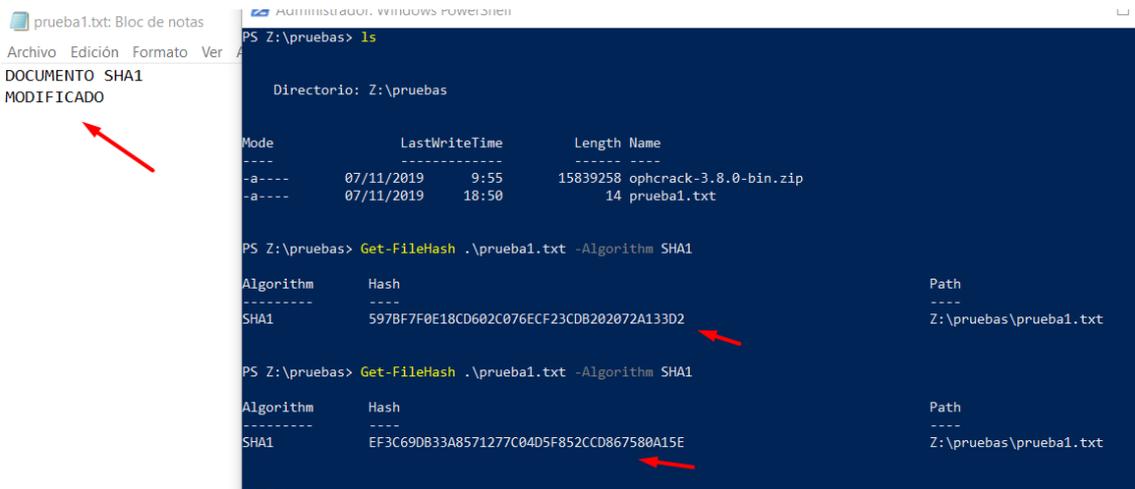
Mode                LastWriteTime         Length Name
-----
-a----             07/11/2019   9:55         15839258 ophcrack-3.8.0-bin.zip
-a----             07/11/2019  18:50             14 prueba1.txt

PS Z:\pruebas> Get-FileHash .\prueba1.txt -Algorithm SHA1

Algorithm      Hash                                                    Path
-----
SHA1          597BF7F0E18CD602C076ECF23CDB202072A133D2            Z:\pruebas\prueba1.txt

PS Z:\pruebas>
```

Si editamos el contenido vemos cómo cambia el SHA1.



2. Utiliza un programa en Windows o GNU/Linux para simular la autenticidad utilizando HMAC-MD5, HMAC-SHA1.

En términos criptográficos, es un tipo de código para autenticar mensajes en clave-hash (HMAC), es decir, es un diseño específico para generar un código de autenticación de mensaje (MAC) lo cual implica una función hash criptográfica junto con una llave criptográfica secreta.

También puede ser usada para verificar simultáneamente la autenticación de un mensaje y la integridad de los datos. El poder criptográfico del HMAC va a depender de la cantidad de caracteres de la clave secreta que se utilice.

Utilizaré dos páginas web conocidas para realizar esta tarea. La primera es la siguiente [\(Link\)](#). Introduciremos un mensaje con su clave y generaremos el HMAC-MD5.

The screenshot shows the 'Online HMAC Generator Tool' interface. The tool allows testing various HMAC hashing algorithms. The input fields are:

- The String to be Encrypted:** FRANZ-HMAC
- Secret Key:** SWITCH
- Select Hash Algorithm to Use:** md5

The **Computed HMAC Result** is: 449656050c25a4676edc60ac5c120a60

Comprobaremos con otra página que el resultado es realmente correcto. [\(Link\)](#)

The screenshot shows a web interface for generating HMACs. At the top, there is a dropdown menu labeled "Algorithms" with "HMAC-MD5" selected. Below this is a text input field containing "SWITCH" and another text input field containing "FRAN2-HMAC". A "Generate HMAC" button is centered below these fields. At the bottom, a text area displays the resulting HMAC value: "449656050c25a4676edc60ac5c120a60".

Podemos ver que los dos hmac-hashes son iguales por lo cual la autenticidad es correcta. Lo siguiente es hacer lo mismo, pero en **HMAC-SHA1**, lo realizaremos en las páginas web de antes.

Online HMAC Generator Tool

This tool will allow you to test a variety of HMAC Hashing algorithms for a variety of purposes.

The screenshot shows the "Online HMAC Generator Tool" interface. It features several input fields and buttons. The "The String to be Encrypted" field contains "FRAN2-SHA1". The "Secret Key" field contains "sad". The "Select Hash Algorithm to Use" dropdown menu is set to "sha1". Below these fields are "Generate" and "Clear" buttons. The "Computed HMAC Result" field displays the value "2a652d493d9d2a69f7225acf02dfb648c626bb0e".

Lo compararemos con la otra página web. Vemos que es igual que la anterior página por lo cual la autenticidad es correcta.

The screenshot shows a web form for generating an HMAC. The 'Algorithms' dropdown menu is set to 'HMAC-SHA1'. The input field contains the text 'sad'. Below the input field is a 'Generate HMAC' button. The output field displays the resulting HMAC value: '2a652d493d9d2a69f7225ac102dfb648c628bb0e'.

c) CIFRADO: Simétrico y Asimétrico--> Confidencialidad, Integridad y Autenticidad.

- 1. Utiliza un programa en Windows o GNU/Linux para simular la confidencialidad mediante “cifrado simétrico”.

La criptografía de clave simétrica, también llamada criptografía de clave secreta o criptografía de una clave, es un método criptográfico en el cual se usa una misma clave para cifrar y descifrar mensajes en el emisor y el receptor.

Utilizaremos para llevar acabo esta práctica **Linux con Ubuntu 18**. El programa que usare es **pgp**. Primero crearemos un documento y lo cifraremos con una clave simétrica.

```

franciscojesus@ubuntu18: ~/SAD-2
franciscojesus@ubuntu18: ~/SAD-2 139x24
franciscojesus@ubuntu18:~/SAD-2$ echo "contenido muy secreto" > simetrico_fran
franciscojesus@ubuntu18:~/SAD-2$ ls
simetrico_fran
franciscojesus@ubuntu18:~/SAD-2$ cat simetrico_fran
contenido muy secreto
franciscojesus@ubuntu18:~/SAD-2$ █

```

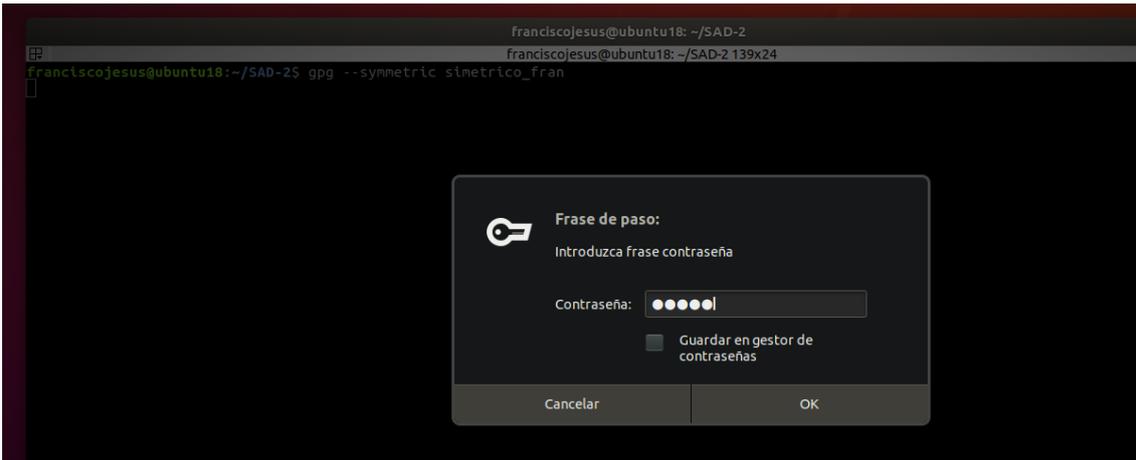
Ahora lo cifraremos de manera simétrica con **gpg**.

```

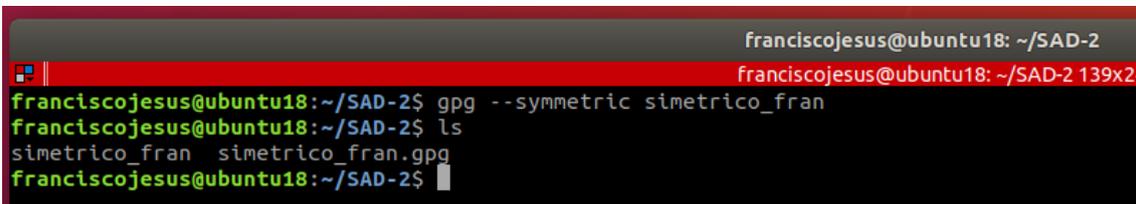
franciscojesus@ubuntu18: ~/SAD-2
franciscojesus@ubuntu18: ~/SAD-2 139x24
franciscojesus@ubuntu18:~/SAD-2$ gpg --symmetric simetrico_fran █

```

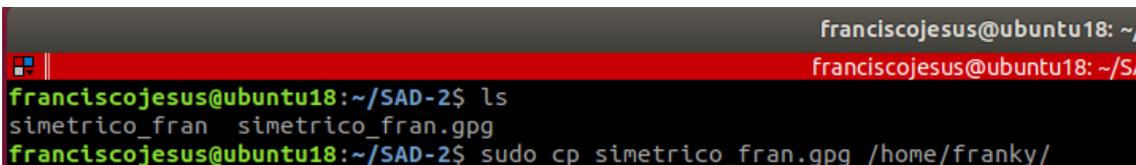
Ponemos una clave.



Vemos como nos generó el mismo archivo, pero con extensión. gpg



Ahora simplemente enviaremos el archivo cifrado con clave simétrica al /home del segundo usuario que tenemos.

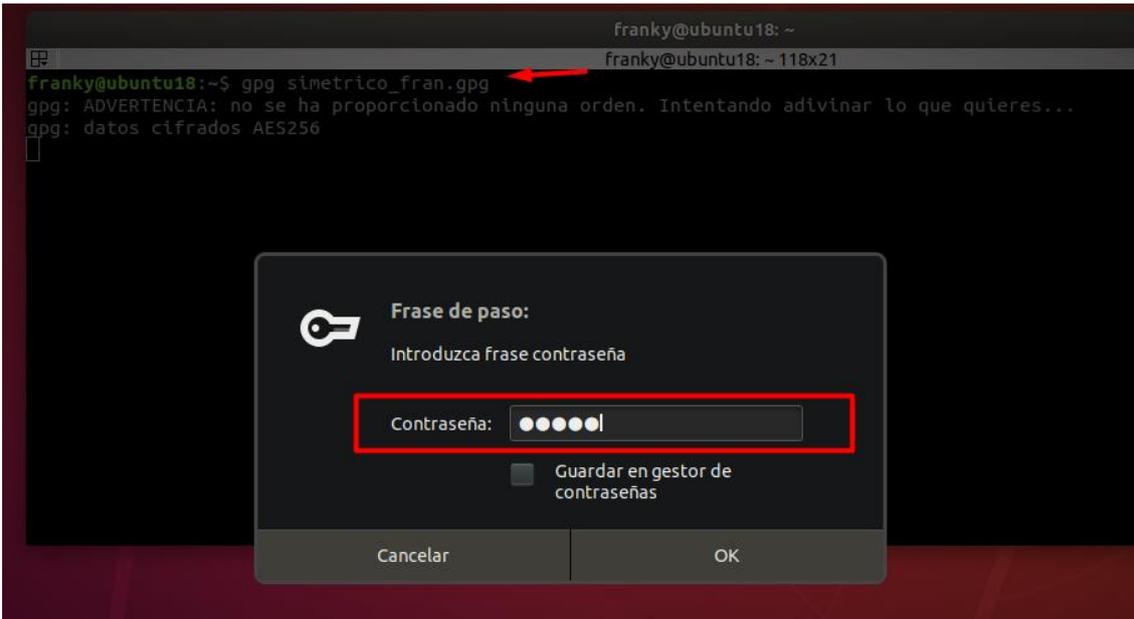


Iniciaremos sesión con el segundo usuario e intentaremos ver el contenido con un simple cat. Vemos como lógicamente no deja al estar cifrado.

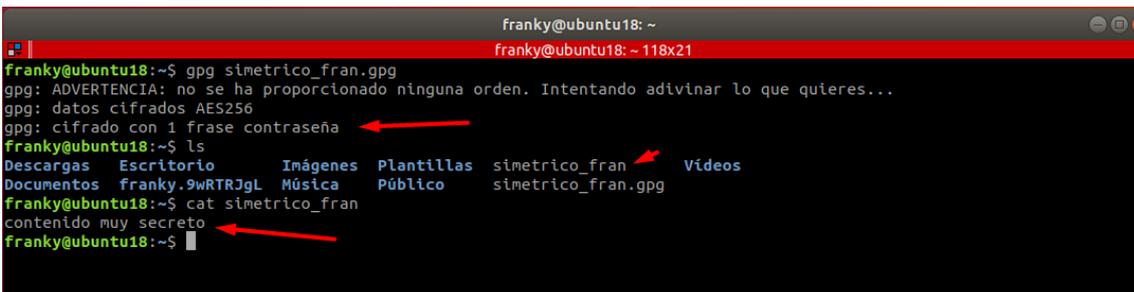


Simplemente lo descifraremos con gpg, nos pedirá la contraseña que pusimos al cifrar el archivo.

`$ gpg simetrico_fran.gpg`



Vemos como nos descifra el archivo y podemos ver su contenido.

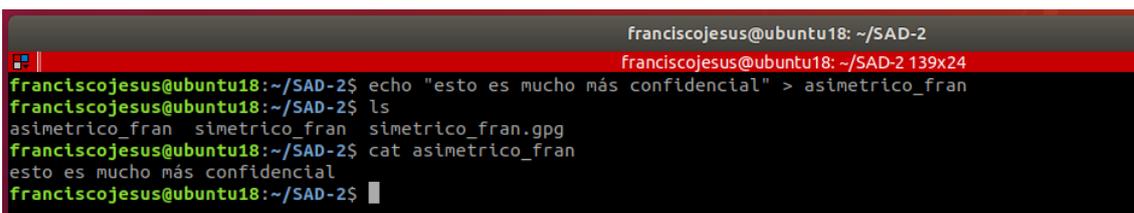


- 2. Utiliza un programa en Windows o GNU/Linux para simular la confidencialidad mediante “cifrado asimétrico”.

La criptografía asimétrica, también llamada criptografía de clave pública o criptografía de dos claves, es el método criptográfico que usa un par de claves para el envío de mensajes. Las dos claves pertenecen a la misma persona que recibirá el mensaje.

Para realizar esta práctica, igual que antes, utilizaremos gpg en Ubuntu 18.

Crearemos para ello un nuevo archivo con la cuenta franciscojesus.



Ahora generaremos las claves necesarias (dos, una publica y otra privada) para cada usuario.

Primero crearemos las claves del primer usuario.

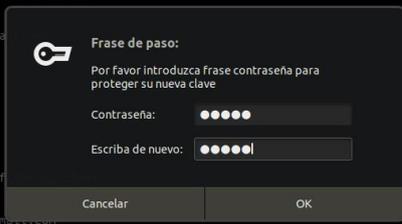
`$ gpg --full-generate-key`

```
Franciscojesus@ubuntu18: ~/SAD-2
Franciscojesus@ubuntu18: ~/SAD-2 141x33
Franciscojesus@ubuntu18:~/SAD-2$ gpg --full-generate-key
gpg (GnuPG) 2.2.4: Copyright (C) 2017 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Por favor seleccione tipo de clave deseado:
(1) RSA y RSA (por defecto)
(2) DSA y ElGamal
(3) DSA (sólo firmar)
(4) RSA (sólo firmar)
Su elección: 2
Las claves DSA pueden tener entre 1024 y 3072 bits de longitud.
¿De qué tamaño quiere la clave? (2048) 1024
El tamaño requerido es de 1024 bits
Por favor, especifique el periodo de validez de la clave.
  0 = la clave nunca caduca
  <n> = la clave caduca en n días
  <n>w = la clave caduca en n semanas
  <n>m = la clave caduca en n meses
  <n>y = la clave caduca en n años
¿Validez de la clave (0)? 0
La clave nunca caduca
¿Es correcto? (s/n) S
GnuPG debe construir un ID de usuario para identificar su clave.
Nombre y apellidos: franciscojesus
Dirección de correo electrónico: franciscojesus@gmail.com
Comentario:
Ha seleccionado este ID de usuario:
  "franciscojesus <franciscojesus@gmail.com>"
¿Cambia (N)ombre, (C)omentario, (D)irección o (V)ale/(S)alir? V
```

Le ponemos una contraseña.

```
Franciscojesus@ubuntu18: ~/SAD-2
Franciscojesus@ubuntu18: ~/SAD-2 141x33
(1) RSA y RSA (por defecto)
(2) DSA y ElGamal
(3) DSA (sólo firmar)
(4) RSA (sólo firmar)
Su elección: 2
Las claves DSA pueden tener entre 1024 y 3072 bits de longitud.
¿De qué tamaño quiere la clave? (2048) 1024
El tamaño requerido es de 1024 bits
Por favor, especifique el periodo de validez de la clave.
  0 = la clave nunca caduca
  <n> = la clave caduca en n días
  <n>w = la clave caduca en n semanas
  <n>m = la clave caduca en n meses
  <n>y = la clave caduca en n años
¿Validez de la clave (0)? 0
La clave nunca caduca
¿Es correcto? (s/n) S
GnuPG debe construir un ID de usuario para identificar su clave.
Nombre y apellidos: franciscojesus
Dirección de correo electrónico: franciscojesus@gmail.com
Comentario:
Ha seleccionado este ID de usuario:
  "franciscojesus <franciscojesus@gmail.com>"
¿Cambia (N)ombre, (C)omentario, (D)irección o (V)ale/(S)alir? V
Es necesario generar muchos bytes aleatorios. Es una buena idea realizar
alguna otra tarea (trabajar en otra ventana/console, mover el ratón, usar
la red y los discos) durante la generación de números primos. Esto da al
generador de números aleatorios mayor oportunidad de recoger suficiente
entropía.
```



```
Franciscojesus@ubuntu18: ~/SAD-2
Franciscojesus@ubuntu18: ~/SAD-2 141x33
GnuPG debe construir un ID de usuario para identificar su clave.
Nombre y apellidos: franciscojesus
Dirección de correo electrónico: franciscojesus@gmail.com
Comentario:
Ha seleccionado este ID de usuario:
  "franciscojesus <franciscojesus@gmail.com>"
¿Cambia (N)ombre, (C)omentario, (D)irección o (V)ale/(S)alir? V
Es necesario generar muchos bytes aleatorios. Es una buena idea realizar
alguna otra tarea (trabajar en otra ventana/console, mover el ratón, usar
la red y los discos) durante la generación de números primos. Esto da al
generador de números aleatorios mayor oportunidad de recoger suficiente
entropía.
Es necesario generar muchos bytes aleatorios. Es una buena idea realizar
alguna otra tarea (trabajar en otra ventana/console, mover el ratón, usar
la red y los discos) durante la generación de números primos. Esto da al
generador de números aleatorios mayor oportunidad de recoger suficiente
entropía.
gpg: clave 1D5F080A83FBCF27 marcada como de confianza absoluta
gpg: creado el directorio '/home/franciscojesus/.gnupg/openpgp-revocs.d'
gpg: certificado de revocación guardado como '/home/franciscojesus/.gnupg/openpgp-revocs.d/A98E84D2DB74640E4C2E416C1D5F080A83FBCF27.rev'
claves pública y secreta creadas y firmadas.

pub  dsa1024 2019-11-07 [SC]
     A98E84D2DB74640E4C2E416C1D5F080A83FBCF27
uid  franciscojesus <franciscojesus@gmail.com>
sub  elg1024 2019-11-07 [E]
```

Ahora, vamos al otro usuario (Franky) y generaremos su clave igual que hicimos con franciscojesus.

```
franky@ubuntu18: ~  
franky@ubuntu18:~$ gpg --full-generate-key  
gpg (GnuPG) 2.2.4; Copyright (C) 2017 Free Software Foundation, Inc.  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.  
  
Por favor seleccione tipo de clave deseado:  
  (1) RSA y RSA (por defecto)  
  (2) DSA y ElGamal  
  (3) DSA (sólo firmar)  
  (4) RSA (sólo firmar)  
Su elección: 2  
Las claves DSA pueden tener entre 1024 y 3072 bits de longitud.  
¿De qué tamaño quiere la clave? (2048) 1024  
El tamaño requerido es de 1024 bits  
Por favor, especifique el periodo de validez de la clave.  
  0 = la clave nunca caduca  
  <n> = la clave caduca en n días  
  <n>w = la clave caduca en n semanas  
  <n>m = la clave caduca en n meses  
  <n>y = la clave caduca en n años  
¿Validez de la clave (0)? 0  
La clave nunca caduca  
¿Es correcto? (s/n) s  
  
GnuPG debe construir un ID de usuario para identificar su clave.  
Nombre y apellidos: franky  
Dirección de correo electrónico: franky@gmail.com  
Comentario:  
Ha seleccionado este ID de usuario:  
"franky <franky@gmail.com>"  
  
¿Cambia (N)ombre, (C)omentario, (D)irección o (V)ale/(S)alir?
```

Le ponemos una contraseña.

```
(1) RSA y RSA (por defecto)  
(2) DSA y ElGamal  
(3) DSA (solo firmar)  
(4) RSA (solo firmar)  
Su elección: 2  
Las claves DSA pueden tener entre 1024 y 3072 bits de longitud.  
¿De qué tamaño quiere la clave? (2048) 1024  
El tamaño requerido es de 1024 bits  
Por favor, especifique el periodo de validez de la clave.  
  0 = la clave nunca caduca  
  <n> = la clave caduca en n días  
  <n>w = la clave caduca en n semanas  
  <n>m = la clave caduca en  
  <n>y = la clave caduca en  
¿Validez de la clave (0)? 0  
La clave nunca caduca  
¿ES correcto? (s/n) s  
  
GnuPG debe construir un ID de u  
Nombre y apellidos: franky  
Dirección de correo electrónico  
Comentario:  
Ha seleccionado este ID de usua  
"franky <franky@gmail.com>"  
  
¿Cambia (N)ombre, (C)omentario,  
Es necesario generar muchos bytes aleatorios. Es una buena idea realizar  
alguna otra tarea (trabajar en otra ventana/console, mover el ratón, usar  
la red y los discos) durante la generación de números primos. Esto da al  
generador de números aleatorios mayor oportunidad de recoger suficiente  
entropía.  
Es necesario generar muchos bytes aleatorios. Es una buena idea realizar  
alguna otra tarea (trabajar en otra ventana/console, mover el ratón, usar  
la red y los discos) durante la generación de números primos. Esto da al  
generador de números aleatorios mayor oportunidad de recoger suficiente  
entropía.  
gpg: clave 4A0FFB86042C99A marcada como de confianza absoluta  
gpg: creado el directorio '/home/franky/.gnupg/openpgp-revocs.d'  
gpg: certificado de revocación guardado como '/home/franky/.gnupg/openpgp-revocs.d/15E246097F0A19C24928E2064A0FFB86042C99A.rev'  
claves pública y secreta creadas y firmadas.  
  
pub  dsa1024 2019-11-07 [SC]  
    15E246097F0A19C24928E2064A0FFB86042C99A  
uid  franky <franky@gmail.com>  
sub  elg1024 2019-11-07 [E]  
  
franky@ubuntu18:~$
```

Ya la tendremos generada.

```
franky@ubuntu18: ~  
franky@ubuntu18:~$ gpg --full-generate-key  
gpg (GnuPG) 2.2.4; Copyright (C) 2017 Free Software Foundation, Inc.  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.  
  
Por favor seleccione tipo de clave deseado:  
  (1) RSA y RSA (por defecto)  
  (2) DSA y ElGamal  
  (3) DSA (sólo firmar)  
  (4) RSA (sólo firmar)  
Su elección: 2  
Las claves DSA pueden tener entre 1024 y 3072 bits de longitud.  
¿De qué tamaño quiere la clave? (2048) 1024  
El tamaño requerido es de 1024 bits  
Por favor, especifique el periodo de validez de la clave.  
  0 = la clave nunca caduca  
  <n> = la clave caduca en n días  
  <n>w = la clave caduca en n semanas  
  <n>m = la clave caduca en n meses  
  <n>y = la clave caduca en n años  
¿Validez de la clave (0)? 0  
La clave nunca caduca  
¿Es correcto? (s/n) s  
  
GnuPG debe construir un ID de usuario para identificar su clave.  
Nombre y apellidos: franky  
Dirección de correo electrónico: franky@gmail.com  
Comentario:  
Ha seleccionado este ID de usuario:  
"franky <franky@gmail.com>"  
  
¿Cambia (N)ombre, (C)omentario, (D)irección o (V)ale/(S)alir? V  
Es necesario generar muchos bytes aleatorios. Es una buena idea realizar  
alguna otra tarea (trabajar en otra ventana/console, mover el ratón, usar  
la red y los discos) durante la generación de números primos. Esto da al  
generador de números aleatorios mayor oportunidad de recoger suficiente  
entropía.  
Es necesario generar muchos bytes aleatorios. Es una buena idea realizar  
alguna otra tarea (trabajar en otra ventana/console, mover el ratón, usar  
la red y los discos) durante la generación de números primos. Esto da al  
generador de números aleatorios mayor oportunidad de recoger suficiente  
entropía.  
gpg: clave 4A0FFB86042C99A marcada como de confianza absoluta  
gpg: creado el directorio '/home/franky/.gnupg/openpgp-revocs.d'  
gpg: certificado de revocación guardado como '/home/franky/.gnupg/openpgp-revocs.d/15E246097F0A19C24928E2064A0FFB86042C99A.rev'  
claves pública y secreta creadas y firmadas.  
  
pub  dsa1024 2019-11-07 [SC]  
    15E246097F0A19C24928E2064A0FFB86042C99A  
uid  franky <franky@gmail.com>  
sub  elg1024 2019-11-07 [E]  
  
franky@ubuntu18:~$
```

Ahora simplemente exportaremos las claves publicas de los dos usuarios.

Franky:

```
$ gpg -a --export -o asimetrico_franky.pub Franky
```

```
franky@ubuntu18: ~
franky@ubuntu18: ~ - 137x33
franky@ubuntu18:~$ ls
Descargas Documentos Escritorio Imágenes Música Plantillas Público simetrico_fran simetrico_fran.gpg Videos
franky@ubuntu18:~$ gpg -a --export -o asimetrico_franky.pub Franky
franky@ubuntu18:~$ ls
asimetrico_franky.pub Documentos Imágenes Plantillas simetrico_fran Videos
Descargas Escritorio Música Público simetrico_fran.gpg
franky@ubuntu18:~$
```

franciscojesus:

```
$ gpg -a --export -o asimetrico_franciscojesus.pub franciscojesus
```

```
franciscojesus@ubuntu18: ~/SAD-2
franciscojesus@ubuntu18: ~/SAD-2 141x33
franciscojesus@ubuntu18:~/SAD-2$ gpg -a --export -o asimetrico_franciscojesus.pub franciscojesus
franciscojesus@ubuntu18:~/SAD-2$ ls
asimetrico_fran asimetrico_franciscojesus.pub simetrico_fran simetrico_fran.gpg
franciscojesus@ubuntu18:~/SAD-2$
```

Recordamos que con el comando `gpg --list-public-keys` podemos ver las claves públicas que tenemos.

```
franciscojesus@ubuntu18: ~/SAD-2
franciscojesus@ubuntu18: ~/SAD-2 141x33
franciscojesus@ubuntu18:~/SAD-2$ gpg --list-public-keys
gpg: comprobando base de datos de confianza
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: nivel: 0 validez: 1 firmada: 0 confianza: 0-, 0q, 0n, 0m, 0f, 1u
/home/franciscojesus/.gnupg/pubring.kbx
-----
pub   dsa1024 2019-11-07 [SC]
      A98E84D2DB74640E4C2E416C1D5F080A83FBCF27
uid   [ absoluta ] franciscojesus <franciscojesus@gmail.com>
sub   elg1024 2019-11-07 [E]

franciscojesus@ubuntu18:~/SAD-2$
```

Podemos de esa forma exportar la clave publica que queramos.

```
franciscojesus@ubuntu18: ~/SAD-2
franciscojesus@ubuntu18: ~/SAD-2 148x38
franciscojesus@ubuntu18:~/SAD-2$ gpg --list-public-keys
/home/franciscojesus/.gnupg/pubring.kbx
-----
pub   dsa1024 2019-11-07 [SC]
      A98E84D2DB74640E4C2E416C1D5F080A83FBCF27
uid   [ absoluta ] franciscojesus <franciscojesus@gmail.com>
sub   elg1024 2019-11-07 [E]

franciscojesus@ubuntu18:~/SAD-2$ gpg --export -a A98E84D2DB74640E4C2E416C1D5F080A83FBCF27 > export.pub
franciscojesus@ubuntu18:~/SAD-2$ cat export
cat: export: No existe el archivo o el directorio
franciscojesus@ubuntu18:~/SAD-2$ cat export.pub
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQG1BF3Ez2IRBADDCK4VtXyLM72QVEa/n0G01ZnJ0l7cuesE5YlThw07Jh3ehLc9
/znlUxpI3XJHTmY3YyWuzQldPhSjjxgoha7Zt6n18w3S0E/jQUJt9sHEKv23JeQ
FOFP4T3LHM4MlWRS+kNZNAoEKON4p7sNens9xKwqAL15AUhSD2H8kNtFwCgrgt0
95mndr60xa0c+SmsA+NBqBSAKbuua/Sn+j+kF7+/i3yl+VBbnS9AYNuR40SosgZ
LxdXC1GbsDrbuS1ONmLLJbXL0AAEFXf0MmQn/4ncvEHYcrRM080GX7zuLZADj
WrJwRmTha1L3L60AqR7EpLSJYkygNzY5CFh/q1bT6JWeAN7kL0mkBp69ogpeSuq
lnfBA/9PAwM1Z5F8mBeT+qGHYhJYIRNNAZh0TKV4n7sTVeEhd0GEnUnUSGcAc
Agz6dIzV57LCSqouN9JNG9hJ2TqySTVdLjx307nz8Lo+y2J5ZzPdP3rUL/PqE1T
OyWLGX9V6deJvNPAmQby6VVAazgFtZyAFKcJodIYnPLjsUUTurQpZnJhbnNpc2Nv
amVzdXVwPcZyYV5j3XNjB2pLc3vZQgdvYlSLmivbTOIEAQTEQIA0BvYhKmhNLb
dCQ0Tc5BbbifCAeD+8bn0JdxGcyAhsD08sCACcBhIK0gLA0MAGHBAH4BAHEA
AAQ0EB1fCAqD+8BndkAn0jd7KT3/U0U411jSoSbLynh37zoAKCFoqRkLqXSpYv6
C/ISwTA1VtmCmrkBDQRdxGcyEAQo7sCnps9/989IyCndrLd7LM8Gg8kt13+0E0j
VHnuS3I9f3hoBDBtGkyJ/ua/0ABna7vappIhwZkPq5Jcxyk2L9ua9bA/p0D5n
WgzY2sRFIFIBmWYun804Uml0JYRukXho1yxxdKC2rdwKfMI7PMDJNBuJbEz1B1L
BxnRVVMAAWUD/1coqLna10EmxnTsdYdVEoc/PjnzUGR1zD+2d0lJ10LYs93sHRW
jTbFL0unEeJYo96y/tZSRoE1g3aegC3zVwaby/onS+zjTNw0b0x3YLnXV5essZ0
QFVY598qz2pQ0Z23daYkWsHeBHFHW/EnvjSr499ewLChXlknvFR00tGAEGBEC
ACAWI0Spj0TS23RkdkwuQWdXwKg/vPjWUCXcRnMgIbDAAKCRAdXwKg/vPjX3n
AKCDVm1Rd/svhJtqX3l8c1hp+kzACFFA0QVCK60fucADiModI9ubdLk=
=RKvx
-----END PGP PUBLIC KEY BLOCK-----

franciscojesus@ubuntu18:~/SAD-2$
```

Ahora copiaremos las claves públicas de un usuario a otro. Clave publica de franciscojesus a Franky.

```
franciscojesus@ubuntu18: ~/SAD-2
franciscojesus@ubuntu18: ~/SAD-2 148x38
franciscojesus@ubuntu18:~/SAD-2$ sudo cp export.pub /home/franky/export.pub
franciscojesus@ubuntu18:~/SAD-2$ sudo ls /home/franky/
asimetrico_franky.pub Documentos export.pub Música Público simetrico_fran.gpg
Descargas Escritorio Imágenes Plantillas simetrico_fran Videos
franciscojesus@ubuntu18:~/SAD-2$
```

Clave publica de Franky a franciscojesus.

```
franky@ubuntu18: ~
franky@ubuntu18: ~ 137x33
franky@ubuntu18:~$ ls
asimetrico_franky.pub Documentos export.pub Música Público simetrico_fran.gpg
Descargas Escritorio Imágenes Plantillas simetrico_fran Videos
franky@ubuntu18:~$ sudo cp asimetrico_franky.pub /home/franciscojesus/asimetrico_franky.pub
franky@ubuntu18:~$ sudo ls /home/franciscojesus/
asimetrico_franky.pub contraseña Documentos Imágenes Plantillas Python-3.7.5 SAD-2
bin Descargas Escritorio Música Público Python-3.7.5.tgz Videos
franky@ubuntu18:~$
```

Ahora importaremos las claves copiadas a cada usuario.

```
franky@ubuntu18: ~
franky@ubuntu18: ~ 137x33
franky@ubuntu18:~$ gpg --import export.pub
gpg: clave 1D5F080A83FBCF27: clave pública "franciscojesus <franciscojesus@gmail.com>" importada
gpg: Cantidad total procesada: 1
gpg: importadas: 1
franky@ubuntu18:~$
```

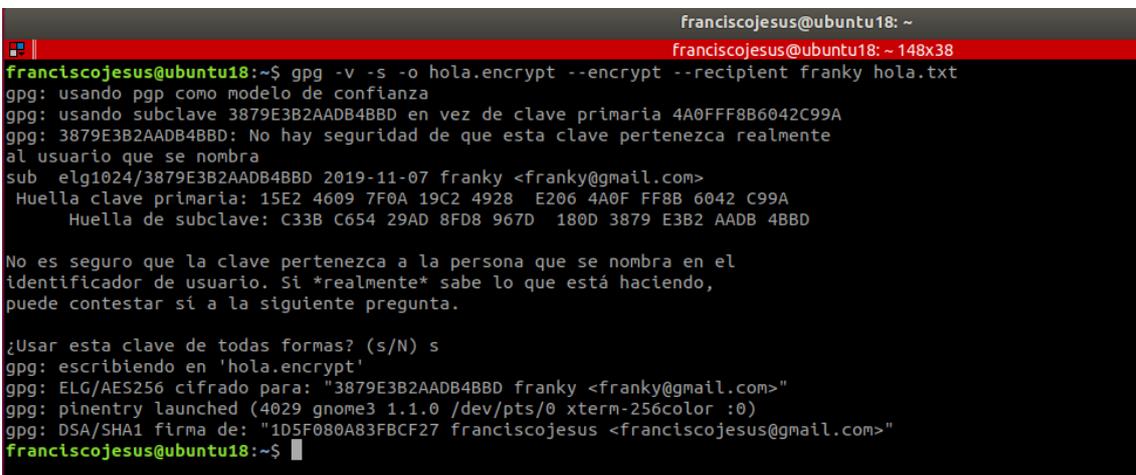
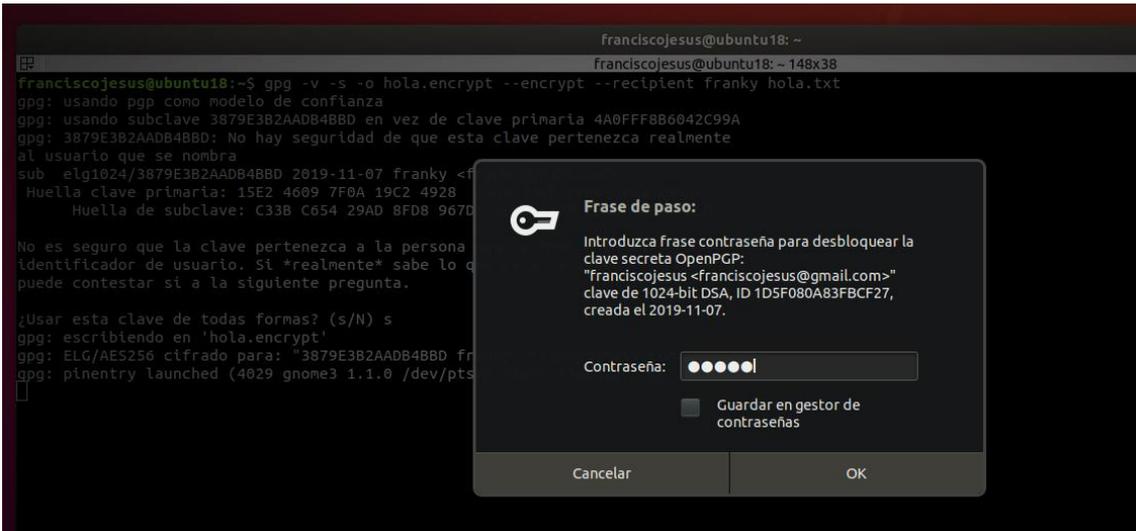
```
franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 148x38
franciscojesus@ubuntu18:~$ ls
asimetrico_franky.pub contraseña Documentos Imágenes Plantillas Python-3.7.5 SAD-2
bin Descargas Escritorio Música Público Python-3.7.5.tgz Videos
franciscojesus@ubuntu18:~$ gpg --import asimetrico_franky.pub
gpg: clave 4A0FFF8B6042C99A: clave pública "franky <franky@gmail.com>" importada
gpg: Cantidad total procesada: 1
gpg: importadas: 1
franciscojesus@ubuntu18:~$
```

Ahora creamos un archivo nuevo en la cuenta franciscojesus y lo encriptaremos con la clave publica del usuario Franky que acabamos de importar.

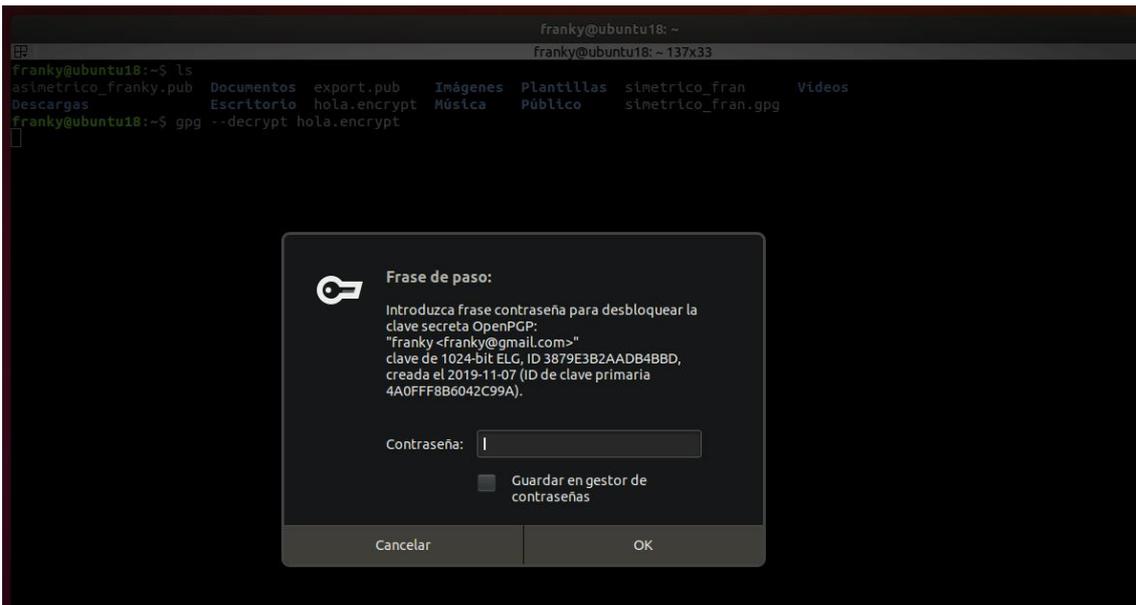
```
franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 148x38
franciscojesus@ubuntu18:~$ echo "ESTE ARCHIVO ES PARA FRANKY DE franciscojesus" > hola.txt
franciscojesus@ubuntu18:~$ ls
asimetrico_franky.pub contraseña Documentos hola.txt Música Público Python-3.7.5.tgz Videos
bin Descargas Escritorio Imágenes Plantillas Python-3.7.5 SAD-2
franciscojesus@ubuntu18:~$ cat hola.txt
ESTE ARCHIVO ES PARA FRANKY DE franciscojesus
franciscojesus@ubuntu18:~$
```

Lo encriptamos con la clave publica de Franky.

```
$ gpg -v -s -o hola.encrypt --encrypt --recipient franky hola.txt
```



Pasamos el archivo a Franky y lo desciframos con su clave privada (lo podemos realizar ya que el archivo lo ciframos con la clave publica de Franky).



```

franky@ubuntu18: ~
franky@ubuntu18: ~ 137x33
franky@ubuntu18:~$ ls
asimetrico_franky.pub  Documentos  export.pub  Imágenes  Plantillas  simetrico_fran  Videos
Descargas             Escritorio  hola.encrypt Música      Público      simetrico_fran.gpg
franky@ubuntu18:~$ gpg --decrypt hola.encrypt
gpg: cifrado con clave de 1024 bits ELG, ID 3879E3B2AADB4BBD, creada el 2019-11-07
"franky <franky@gmail.com>"
ESTE ARCHIVO ES PARA FRANKY DE franciscojesus
gpg: Firmado el jue 07 nov 2019 20:51:41 CET
gpg: usando DSA clave A98E84D2DB74640E4C2E416C1D5F080A83FBCF27
gpg: comprobando base de datos de confianza
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: nivel: 0  validez: 1  firmada: 0  confianza: 0-, 0q, 0n, 0m, 0f, 1u
gpg: Firma correcta de "franciscojesus <franciscojesus@gmail.com>" [desconocido]
gpg: ATENCIÓN: ¡Esta clave no está certificada por una firma de confianza!
gpg: No hay indicios de que la firma pertenezca al propietario.
Huellas dactilares de la clave primaria: A98E 84D2 DB74 640E 4C2E 416C 1D5F 080A 83FB CF27
franky@ubuntu18:~$

```

Podemos hacer lo mismo a la inversa (esta vez usaremos otro comando para cifrar con la clave publica, esta vez especificaremos con la clave).

```

franky@ubuntu18: ~
franky@ubuntu18: ~ 137x33
franky@ubuntu18:~$ echo "Esto es de franky para fcojesus" > de_franky.txt
franky@ubuntu18:~$ ls
asimetrico_franky.pub  Descargas  Escritorio  hola.encrypt  Música  Público  simetrico_fran.gpg
de_franky.txt          Documentos  export.pub  Imágenes     Plantillas  simetrico_fran  Videos
franky@ubuntu18:~$ cat de_franky.txt
Esto es de franky para fcojesus
franky@ubuntu18:~$

```

```

franky@ubuntu18: ~
franky@ubuntu18: ~ 137x33
franky@ubuntu18:~$ gpg --list-public-keys /home/franky/.gnupg/pubring.kbx
-----
pub   dsa1024 2019-11-07 [SC]
      15E246097F0A19C24928E2064A0FFF8B6042C99A
uid   [ absoluta ] franky <franky@gmail.com>
sub   elg1024 2019-11-07 [E]

pub   dsa1024 2019-11-07 [SC]
      A98E84D2DB74640E4C2E416C1D5F080A83FBCF27
uid   [desconocida] franciscojesus <franciscojesus@gmail.com>
sub   elg1024 2019-11-07 [E]

franky@ubuntu18:~$
franky@ubuntu18:~$ gpg -a -er A98E84D2DB74640E4C2E416C1D5F080A83FBCF27 de_franky.txt
gpg: D08FE431BD59DF45: No hay seguridad de que esta clave pertenezca realmente al usuario que se nombra
sub   elg1024/D08FE431BD59DF45 2019-11-07 franciscojesus <franciscojesus@gmail.com>
Huella clave primaria: A98E 84D2 DB74 640E 4C2E 416C 1D5F 080A 83FB CF27
Huella de subclave: 1702 8DA7 C5AD 198A 40C1 D719 D08F E431 BD59 DF45

No es seguro que la clave pertenezca a la persona que se nombra en el
identificador de usuario. Si *realmente* sabe lo que está haciendo,
puede contestar sí a la siguiente pregunta.

¿Usar esta clave de todas formas? (s/N) s
franky@ubuntu18:~$ ls
asimetrico_franky.pub  de_franky.txt.asc  Documentos  export.pub  Imágenes  Plantillas  simetrico_fran  Videos
de_franky.txt         Descargas          Escritorio  hola.encrypt  Música     Público      simetrico_fran.gpg
franky@ubuntu18:~$

```

Lo movemos al /home de franciscojesus y lo desciframos.

```

franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 148x38
franciscojesus@ubuntu18:~$ ls
asimetrico_franky.pub  contraseña  Descargas  Escritorio  hola.txt  Música  Público  Python-3.7.5.tgz  Videos
bin                   de_franky.txt.asc  Documentos  hola.encrypt  Imágenes  Plantillas  Python-3.7.5  SAD-2
franciscojesus@ubuntu18:~$ gpg -d de_franky.txt.asc > de_franky.desencrypt.txt

```

Frases de paso:

Introduzca frase contraseña para desbloquear la clave secreta OpenPGP:
"franciscojesus <franciscojesus@gmail.com>"
clave de 1024-bit ELG, ID D08FE431BD59DF45, creada el 2019-11-07 (ID de clave primaria 1D3F080A83FBCF27).

Contraseña:

Guardar en gestor de contraseñas

Cancelar OK

```

franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 148x38
franciscojesus@ubuntu18:~$ ls
asimetrico_franky.pub  contraseña  Descargas  Escritorio  hola.txt  Música  Público  Python-3.7.5.tgz  Videos
bin                    de_franky.txt.asc  Documentos  hola.encrypt  Imágenes  Plantillas  Python-3.7.5  SAD-2
franciscojesus@ubuntu18:~$ gpg -d de_franky.txt.asc > de_franky.desencrypt
gpg: cifrado con clave de 1024 bits ELG, ID D08FE431BD59DF45, creada el 2019-11-07
"Franciscojesus <franciscojesus@gmail.com>"
franciscojesus@ubuntu18:~$ ls
asimetrico_franky.pub  contraseña  de_franky.txt.asc  Documentos  hola.encrypt  Imágenes  Plantillas  Python-3.7.5  SAD-2
bin                    de_franky.desencrypt  Descargas  Escritorio  hola.txt  Música  Público  Python-3.7.5.tgz  Videos
franciscojesus@ubuntu18:~$ cat de_franky.desencrypt
Esto es de franky para fcojesus
franciscojesus@ubuntu18:~$

```

Hemos podido aprender a importar de dos modos y a descriptar de dos modos, como vemos es sencillo, se tarda más en comprender y leer que en hacer.

- 3. Utiliza un programa en Windows o GNU/Linux para simular la autenticidad mediante “cifrado asimétrico”.

Una firma digital certifica el contenido, fecha y hora de un documento. Si el documento luego es modificado de cualquier forma, la verificación de la firma fallará. Una firma digital funciona entonces como una firma escrita a mano en papel, con la ventaja adicional de resistir cualquier tipo de falsificación o alteración.

Crearemos primero un archivo con algo escrito en el.

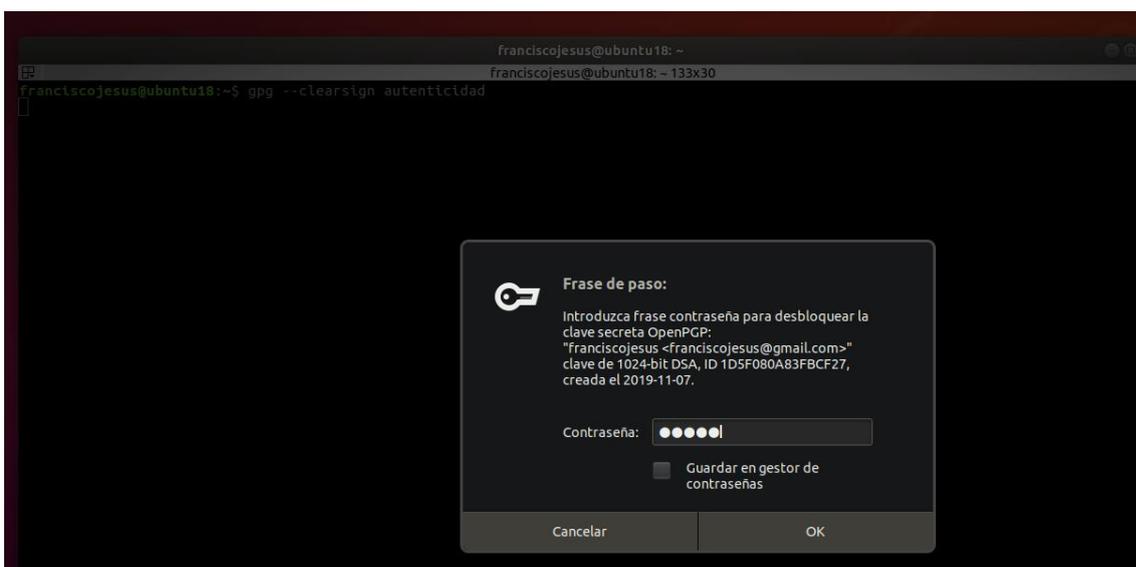
```

franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 133x30
franciscojesus@ubuntu18:~$ echo "Hola, me llamo franciscojesus" > autenticidad
franciscojesus@ubuntu18:~$ cat autenticidad
Hola, me llamo franciscojesus
franciscojesus@ubuntu18:~$

```

Lo siguiente que haremos es firmar el documento con la clave privada.

`$ gpg --clearsign autenticidad`



```
franciscojesus@ubuntu18:~$ gpg --clearsign autenticidad
franciscojesus@ubuntu18:~$ file autenticidad*
autenticidad:      ASCII text
autenticidad.asc:  ASCII text
franciscojesus@ubuntu18:~$
```

Podemos ver como se ha creado el mismo archivo con extensión .asc Si hacemos un cat podemos ver el contenido con la clave SHA1.

```
franciscojesus@ubuntu18:~$ cat autenticidad.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

Hola, me llamo franciscojesus
-----BEGIN PGP SIGNATURE-----

iF0EARECAB0WlQSpjoTS23RkDkwuQWwdXwgKg/vPJwUCXcU1UQAKCRAdXwgKg/vP
J9M1AJ9jxK293pBPByPJS207oS0EDNob5ACfboJgiXH1R3IAMxBVzr+v6QXAhfw=
=4+/d
-----END PGP SIGNATURE-----
franciscojesus@ubuntu18:~$
```

Ahora este mismo archivo se lo pasamos al usuario franky. Verificamos el archivo que este correctamente y no se haya alterado.

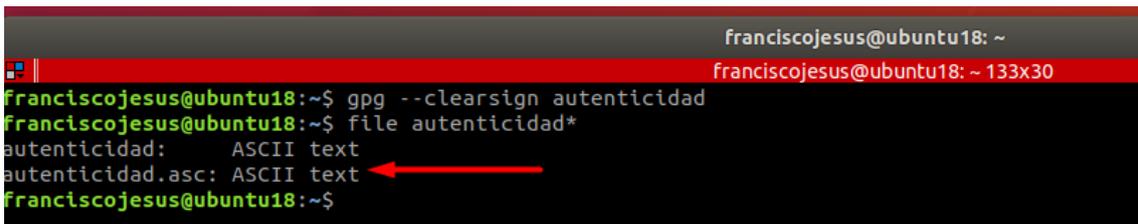
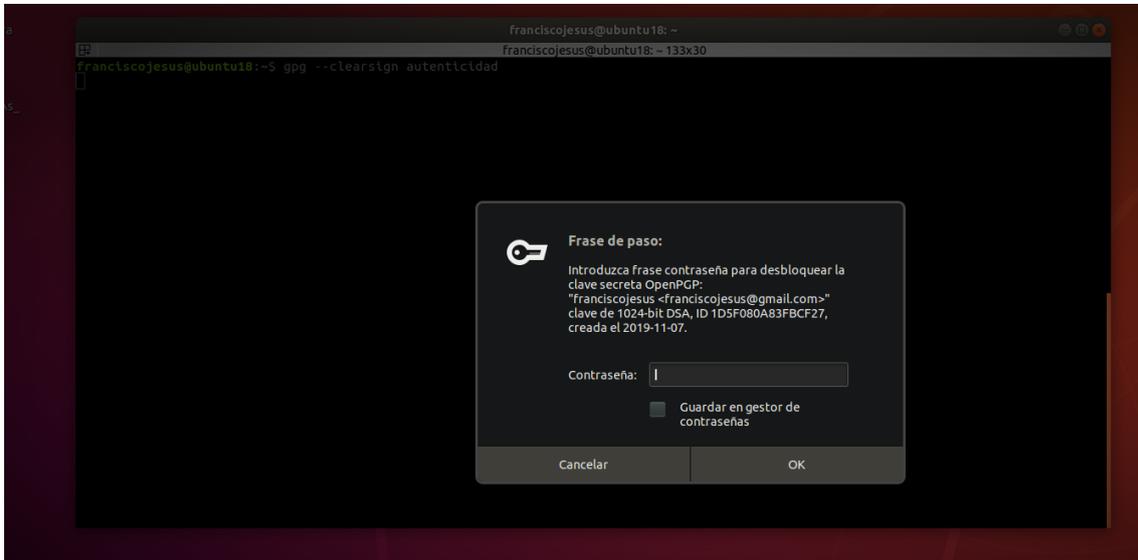
```
franky@ubuntu18:~$ gpg --verify autenticidad.asc
gpg: Firmado el vie 08 nov 2019 10:28:49 CET
gpg: usando DSA clave A98E84D2DB74640E4C2E416C1D5F080A83FBCF27
gpg: Firma correcta de "franciscojesus <franciscojesus@gmail.com>" [desconocido]
gpg: ATENCIÓN: ¡Esta clave no está certificada por una firma de confianza!
gpg: No hay indicios de que la firma pertenezca al propietario.
Huellas dactilares de la clave primaria: A98E 84D2 DB74 640E 4C2E 416C 1D5F 080A 83FB CF27
franky@ubuntu18:~$
```

Lo desciframos con la clave publica de franciscojesus. Vemos tanto el mensaje como la firma.

```
franky@ubuntu18:~$ gpg --decrypt autenticidad.asc > autenticidad.descifrada
gpg: Firmado el vie 08 nov 2019 10:28:49 CET
gpg: usando DSA clave A98E84D2DB74640E4C2E416C1D5F080A83FBCF27
gpg: Firma correcta de "franciscojesus <franciscojesus@gmail.com>" [desconocido]
gpg: ATENCIÓN: ¡Esta clave no está certificada por una firma de confianza!
gpg: No hay indicios de que la firma pertenezca al propietario.
Huellas dactilares de la clave primaria: A98E 84D2 DB74 640E 4C2E 416C 1D5F 080A 83FB CF27
franky@ubuntu18:~$ cat autenticidad.descifrada
Hola, me llamo franciscojesus
franky@ubuntu18:~$
```

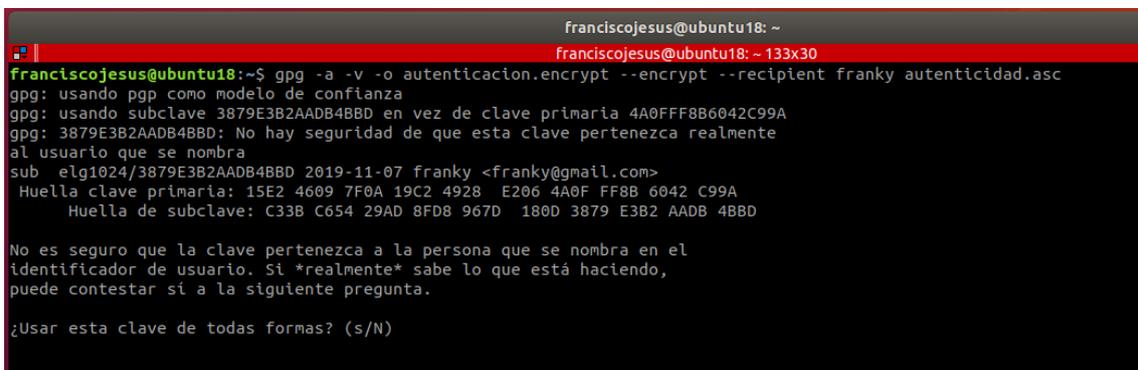
4. Utiliza un programa en Windows o GNU/Linux para simular la “autenticidad”+”confidencialidad” mediante “cifrado asimétrico”.

Primero firmaremos el documento *autenticidad* otra vez con la clave privada del usuario franciscojesus.



Ahora ciframos el documento firmado con la clave privada de franky para que este pueda descifrarlo.

`$ gpg -v -a -o autenticacion.encrypted --encrypt --recipient franky autenticidad.asc`



```

franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 133x30
franciscojesus@ubuntu18:~$ gpg -a -v -o autenticacion.encrypt --encrypt --recipient franky autenticidad.asc
gpg: usando gpg como modelo de confianza
gpg: usando subclave 3879E3B2AADB48BD en vez de clave primaria 4A0FF8B6042C99A
gpg: 3879E3B2AADB48BD: No hay seguridad de que esta clave pertenezca realmente
al usuario que se nombra
sub elg1024/3879E3B2AADB48BD 2019-11-07 franky <franky@gmail.com>
Huella clave primaria: 15E2 4609 7F0A 19C2 4928 E206 4A0F FF8B 6042 C99A
Huella de subclave: C33B C654 29AD 8FD8 967D 180D 3879 E3B2 AADB 48BD

No es seguro que la clave pertenezca a la persona que se nombra en el
identificador de usuario. Si *realmente* sabe lo que está haciendo,
puede contestar sí a la siguiente pregunta.

¿Usar esta clave de todas formas? (s/N) s
gpg: leyendo desde 'autenticidad.asc'
gpg: escribiendo en 'autenticacion.encrypt'
gpg: ELG/AES256 cifrado para: "3879E3B2AADB48BD franky <franky@gmail.com>"
franciscojesus@ubuntu18:~$

```

```

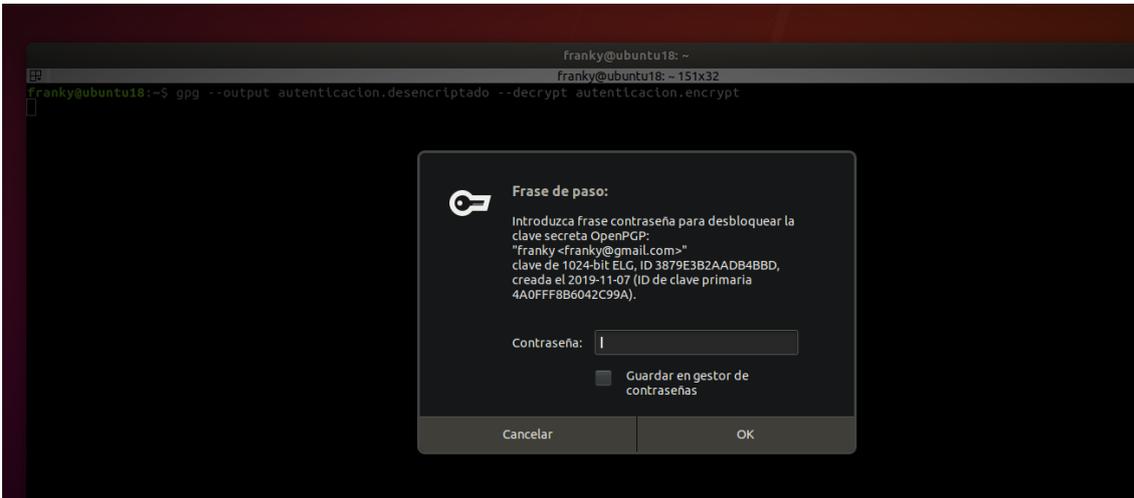
franciscojesus@ubuntu18:~$ cat autenticacion.encrypt
-----BEGIN PGP MESSAGE-----

hQE0Azh547Kq20u9EAP/UFL4/z2pJZxUlgqpVWVPsIuqsXRz1Fl/Si4e9ksa8KsW
YKr2lJFkeKvAG/uwgGLYn+VYftY0qu5wrXcH/szExndV8gkGVaf5GfjoArhqY7pZ
VdUWge4y07bcwsdGiF8TaSkjiJnpeXysytrtw337jRelAyJCpQyLLDsai3sZIKoD
/2haoycIC8rMJckJCDKHSqDo6XdQDt7us9Hye5GbnVjeimXCzk1hkQ/b992kTf/Z
s79YSBn7vMD8YEQao68Ha9eqYYZRSsdKkyBWJpsa4Cw7jFUwdvD8ry68n+4evXMP
TwCzRLGpx0zchEWZ76NIK5kzp7/ZnYjE5H/6wqATqBaY0sBdAYuCn3QiotpX0F3Ry
ieQoWHuXYhU5ywwbC10rLacCLzRcrOz9BwEFqY0541rXJ2H2uTTloJcHgsRnxJAA
UmZBSHIg4/u6tTUoZy8UBSgu3Tt+HhJg390wwvV/GgsT0EcK9c8YjiE75+mYjU6Y
BAvcnmxpEXn6B8wDqvseW9WoyR5xqo/x7Nj5tJPKcLqXL5Zc4myJ2zvgtyUmIarm
U080HzTrI8QF2/bRTmPbAe1mDDl6DNjKa8hgZ/w3LssBFHyyswZrrzRHl7PFnt1Y
dxtswvGst/y45x/2cZS2k3VLlg5v0otpcF/hTBp9azLrIf5ss1hSx5NaqnLAKU/T
BW4pRz5u2kHZQidbPNebYAGwFTSlzAxceuLX0JT/ok+u
=Mhss
-----END PGP MESSAGE-----

```

Le mandamos el documento a franky y lo desciframos con este.

`$ gpg --output autenticidad.desencriptado --decrypt autenticacion.encrypt`



`$ gpg --decrypt autenticidad.desencriptado`

```
franky@ubuntu18: ~  
franky@ubuntu18: ~ 151x32  
franky@ubuntu18:~$ gpg --output autenticacion.desencriptado --decrypt autenticacion.encrypt  
gpg: cifrado con clave de 1024 bits ELG, ID 3879E3B2AADB4BBD, creada el 2019-11-07  
"franky <franky@gmail.com>"  
franky@ubuntu18:~$
```

Podemos ver si hacemos un `cat autenticacion.desencriptado` como sigue aún la firma y el documento de desencripto.

```
franky@ubuntu18:~$ cat autenticacion.desencriptado  
-----BEGIN PGP SIGNED MESSAGE-----  
Hash: SHA1  
  
Hola, me llamo franciscojesus  
-----BEGIN PGP SIGNATURE-----  
  
iF0EARECAB0WIQSpjoTS23RkDkwuQWwdXwgKg/vPJwUCXcU7twAKCRAdXwgKg/vP  
J0jTAKCQENS4AELnH5qJ421kWllpvxjDCgCfRyNCE0pMdoamh9/mdggc+wbSd4w=  
=dCgP  
-----END PGP SIGNATURE-----
```

Si hacemos un `gpg --verify` podremos ver la firma la firma del documento y como está firmado por franciscojesus.

```
franky@ubuntu18: ~  
franky@ubuntu18: ~ 151x32  
franky@ubuntu18:~$ gpg --verify autenticacion.desencriptado  
gpg: Firmado el vie 08 nov 2019 10:56:07 CET  
gpg: usando DSA clave A98E84D2DB74640E4C2E416C1D5F080A83FBCF27  
gpg: Firma correcta de "franciscojesus <franciscojesus@gmail.com>" [desconocido]  
gpg: ATENCIÓN: ¡Esta clave no está certificada por una firma de confianza!  
gpg: No hay indicios de que la firma pertenezca al propietario.  
Huellas dactilares de la clave primaria: A98E 84D2 DB74 640E 4C2E 416C 1D5F 080A 83FB CF27  
franky@ubuntu18:~$
```

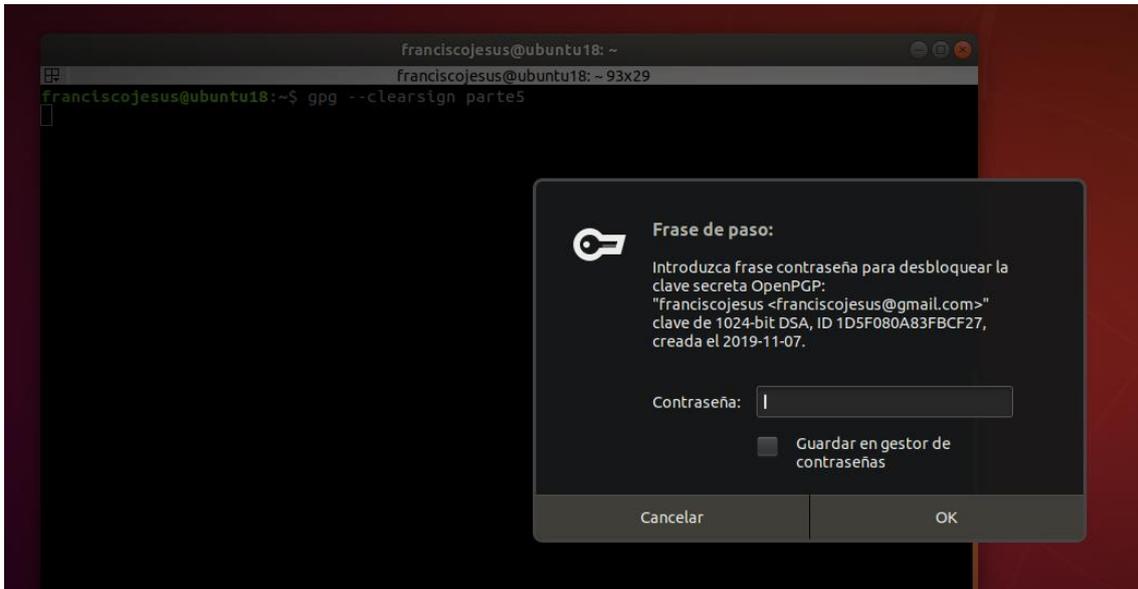
5. Utiliza un programa en Windows o GNU/Linux para simular una comunicación segura utilizando cifrados híbridos “autenticidad” + “confidencialidad”+“integridad”: asimétricos (clave pública) y simétricos (clave privada).

Lo primero es crear un archivo, en mi caso se llamará parte5

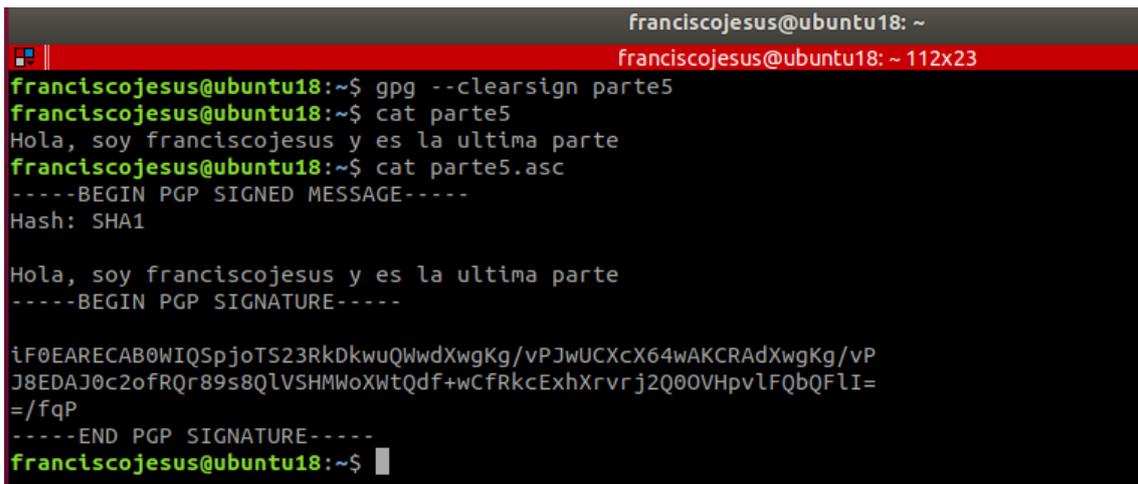
```
franciscojesus@ubuntu18: ~  
franciscojesus@ubuntu18: ~ 93x29  
franciscojesus@ubuntu18:~$ echo "Hola, soy franciscojesus y es la ultima parte" > parte5  
franciscojesus@ubuntu18:~$ cat parte5  
Hola, soy franciscojesus y es la ultima parte  
franciscojesus@ubuntu18:~$
```

Lo firmaremos con el usuario *franciscojesus* :

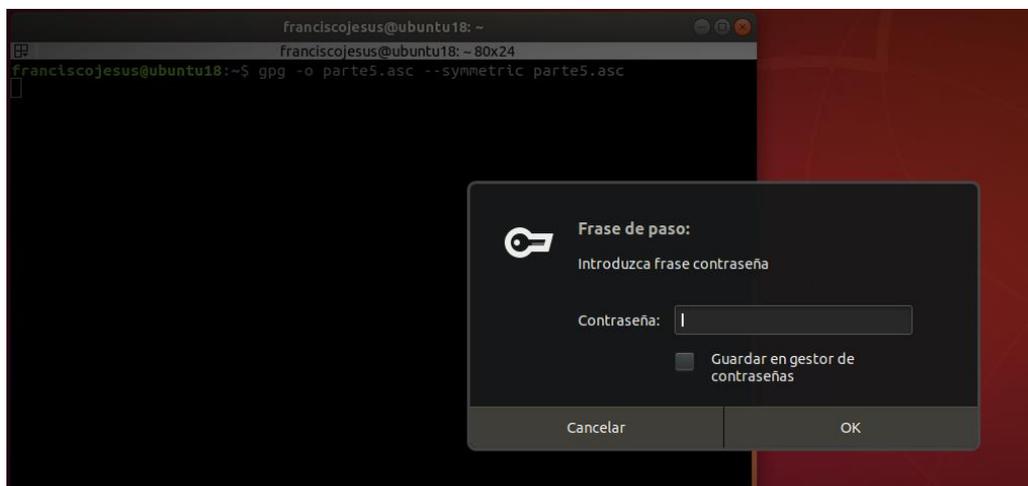
\$ gpg --clearsign parte5



Podemos ver el archivo firmado con un SHA1 para verificar la integridad del contenido y la firma.



Lo siguiente que realizaremos es un cifrado simétrico. Podemos ver como coge la extensión gpg.



```
franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 112x23
franciscojesus@ubuntu18:~$ file parte5*
parte5:          ASCII text
parte5.asc:      ASCII text
parte5.asc.gpg:  GPG symmetrically encrypted data (AES256 cipher)
franciscojesus@ubuntu18:~$
```

Ahora realizaremos el cifrado asimétrico, utilizaremos la clave publica de franky para que así este pueda descifrarlo con su clave privada cuando se lo pasemos. El nuevo archivo generado se llamara parte5.asc.gpg.as (lo redirigimos con -o).

```
gpg -v -a -o parte5.asc.gpg.as --encrypt --recipient franky parte5.asc.gpg
```

```
franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 112x23
franciscojesus@ubuntu18:~$ gpg -v -a -o parte5.asc.gpg.as --encrypt --recipient franky parte5.asc.gpg
gpg: usando gpg como modelo de confianza
gpg: usando subclave 3879E3B2AADB4BBD en vez de clave primaria 4A0FFF8B6042C99A
gpg: 3879E3B2AADB4BBD: No hay seguridad de que esta clave pertenezca realmente al usuario que se nombra
sub  e1g1024/3879E3B2AADB4BBD 2019-11-07 franky <franky@gmail.com>
Hue1la clave primaria: 15E2 4609 7F0A 19C2 4928  E206 4A0F FF8B 6042 C99A
Hue1la de subclave:  C33B C654 29AD 8FD8 967D  180D 3879 E3B2 AADB 4BBD

No es seguro que la clave pertenezca a la persona que se nombra en el
identificador de usuario. Si *realmente* sabe lo que está haciendo,
puede contestar sí a la siguiente pregunta.

¿Usar esta clave de todas formas? (s/N) s
gpg: leyendo desde 'parte5.asc.gpg'
gpg: escribiendo en 'parte5.asc.gpg.as'
gpg: ELG/AES256 cifrado para: "3879E3B2AADB4BBD franky <franky@gmail.com>"
franciscojesus@ubuntu18:~$
```

Podemos ver cómo está totalmente cifrado.

```
franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 112x23
franciscojesus@ubuntu18:~$ cat parte5.asc.gpg.as
-----BEGIN PGP MESSAGE-----

hQE0AZh547Kq20u9EAQAnHv7lr02rrZG89KwZWs/nyB7hXjt/j5dAV3/WzYWbnb9
Hr80v8zXpGrY1dVSDVHdAZKg6XR8EpKZmh9up+ij0SavRFNHISk3+AQow+krITp4
oqu5x+KDTrx8WRtttKJWPE+lR1+6i4FZo5cgKuh8b+ywqFnQylwF/YAhSZy0GbJUE
ALsws7/2jKZjG9XibpDVyGu0BrN1auBDypwL2zb5vbxLs94/G64NPtcbl29pj50k
hZM+yLziJnu57L2wsU2dJwKLOwPyItBTo8CCp4pw6MMYiVc6wqJjqYzcy4nAuEM2
N0K0+5J+5VbIRFgHn+pSwGv3C6/3RuwwDTQJdVwdbfGv0sC7AQ9NjRy9POS7IjHq
scdTqjDaZKB7g+XNrj6iowOz/eXcHa5jT5ggGoXvDN3W8TM5b3UbCsIwBuZKbyY
47INX/Ct8+hEnz9vudTMqKWTCeE1JS8VP0jN4DenlS9irjN/9GBUo4YjY6ERU/Yn
6j5c0gN1Q0r00zjP0mT49xyuo+hjUv5Ek50yOB3KHQFqqt0NDAPkVuc72HUxwie
0VKHqNx5zjbl9aIBBV21+aP4tpM/32ttM+UpAnlDeQ6F80ZY5CZT0nxW7br+Pzr0
H2DtMXi0HcMIOv2eRTaGxIvJ2FDM0w3SwX+CDDV1AvvMECdTgeLFuIv5TTXnEOal
4ekj7adGXgthpHeAln3uCK9A/hbdh0Uh5XzZ2HK0iInqpd9GH8gy0GAvDUJqpdT
Ica2mAZX7wkoknX+60mRixA75Et/14cbxRrzQILmTnpd7ajleUbn+jkZQZ92tuh
033b4moJzsRIiijdlJjs34cz4LpPgPH/bU0NVv0GgBw==
=n75z
-----END PGP MESSAGE-----
franciscojesus@ubuntu18:~$
```

Lo siguiente que haremos es crear el SHA1 del archivo anterior cifrado (parte5.asc.gpg.as). Lo que hare es redirigir el sha1 al archivo *parte5.sha1* para después compararlo y verificar la integridad. Copiare tanto el archivo cifrado *parte5.asc.gpg.as* como el archivo *parte5.sha1* al usuario franky.

```
$ sha1sum parte5.asc.gpg.as > parte5.sha1
```

```
franciscojesus@ubuntu18: ~
franciscojesus@ubuntu18: ~ 126x31
franciscojesus@ubuntu18:~$ sha1sum parte5.asc.gpg.as > parte5.sha1
franciscojesus@ubuntu18:~$ cat parte5.sha1
da0d85b2396127c8ca46f9826e19e7a267eceff4 parte5.asc.gpg.as
franciscojesus@ubuntu18:~$ cp parte5.sha1 /home/franky/
cp: no se puede efectuar `stat' sobre '/home/franky/parte5.sha1': Permiso denegado
franciscojesus@ubuntu18:~$ sudo cp parte5.sha1 /home/franky/
franciscojesus@ubuntu18:~$
```

Ya en el usuario franky podemos hacer un cat a parte5.sha1 para ver el SHA1 generado, después, con sha1sum generamos el sha1 del archivo parte5.asc.gpg.as y compararemos que coinciden los SHA1 para verificar la integridad del archivo. Vemos que son iguales por lo cual no fue alterado.

```
franky@ubuntu18: ~
franky@ubuntu18: ~ 126x31
franky@ubuntu18:~$ cat parte5.sha1
da0d85b2396127c8ca46f9826e19e7a267eceff4 parte5.asc.gpg.as
franky@ubuntu18:~$ sha1sum parte5.asc.gpg.as
da0d85b2396127c8ca46f9826e19e7a267eceff4 parte5.asc.gpg.as
franky@ubuntu18:~$
```

Si no queremos compararlo 1 a 1 podemos ejecutar el comando sha1sum con -c para comprar el sha1 que le hemos pasado al usuario franky con el archivo cifrado que ha recibido. Vemos que la suma coincide por lo cual la integridad es correcta.

```
franky@ubuntu18: ~
franky@ubuntu18: ~ 126x31
franky@ubuntu18:~$ sha1sum parte5.sha1 -c parte5.asc.gpg.as
parte5.asc.gpg.as: La suma coincide
```

Ahora empezaremos a descifrarlo. Primero descifraremos el cifrado asimétrico con la clave privada del usuario franky. Lo redirigimos al archivo parte5.decrypt

```
franky@ubuntu18: ~
franky@ubuntu18: ~ 122x23
franky@ubuntu18:~$ gpg --decrypt parte5.asc.gpg.as > parte5.decrypt
gpg: cifrado con clave de 1024 bits ELG, ID 3879E3B2AADB4BBB, creada el 2019-11-07
"franky <franky@gmail.com>"
```

Después desciframos el cifrado simétrico con la clave secreta (la contraseña que le pusimos al archivo cuando hicimos el cifrado simétrico). Lo redirigimos al archivo parte5.

```
franky@ubuntu18: ~
franky@ubuntu18: ~ 122x23
franky@ubuntu18:~$ gpg --decrypt parte5.decrypt > parte5
gpg: datos cifrados AES256
gpg: cifrado con 1 frase contraseña
franky@ubuntu18:~$
```

Podemos ver como se descripto y tenemos el mensaje visible con el SHA1 (este SHA1 es para verificar la integridad de la firma y mensaje cuando lo firmamos).

```
franky@ubuntu18: ~
franky@ubuntu18: ~ 122x23
franky@ubuntu18:~$ gpg --decrypt parte5.decrypt > parte5
gpg: datos cifrados AES256
gpg: cifrado con 1 frase contraseña
franky@ubuntu18:~$ cat parte5
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

Hola, soy franciscojesus y es la ultima parte
-----BEGIN PGP SIGNATURE-----

iF0EARECAB0WIQSpjoTS23RkDkwuQWwdXwgKg/vPJwUCXcX64wAKCRAdXwgKg/vP
J8EADAJ0c2ofRQr89s8QlVSHMwoXWtQdf+wCfRkcExhXrvrj2Q00VHpvLFQbQFLI=
=/fqP
-----END PGP SIGNATURE-----
franky@ubuntu18:~$
```

Si ejecutamos el comando `gpg --verify parte5` podemos ver que está firmado correctamente por *franciscojesus* y no se ha alterado en ningún momento. Ya tenemos el mensaje descifrado correctamente y visible.

```
franky@ubuntu18: ~
franky@ubuntu18: ~ 122x23
franky@ubuntu18:~$ gpg --verify parte5
gpg: Firmado el sáb 09 nov 2019 00:31:47 CET
gpg: usando DSA clave A98E84D2DB74640E4C2E416C1D5F080A83FB CF27
gpg: Firma correcta de "franciscojesus <franciscojesus@gmail.com>" [desconocido]
gpg: ATENCIÓN: ¡Esta clave no está certificada por una firma de confianza!
gpg: No hay indicios de que la firma pertenezca al propietario.
Huellas dactilares de la clave primaria: A98E 84D2 DB74 640E 4C2E 416C 1D5F 080A 83FB CF27
franky@ubuntu18:~$
```

```
franky@ubuntu18: ~
franky@ubuntu18: ~ 122x23
franky@ubuntu18:~$ gpg --verify parte5
gpg: Firmado el sáb 09 nov 2019 00:31:47 CET
gpg: usando DSA clave A98E84D2DB74640E4C2E416C1D5F080A83FB CF27
gpg: Firma correcta de "franciscojesus <franciscojesus@gmail.com>" [desconocido]
gpg: ATENCIÓN: ¡Esta clave no está certificada por una firma de confianza!
gpg: No hay indicios de que la firma pertenezca al propietario.
Huellas dactilares de la clave primaria: A98E 84D2 DB74 640E 4C2E 416C 1D5F 080A 83FB CF27
franky@ubuntu18:~$ cat parte5
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

Hola, soy franciscojesus y es la ultima parte
-----BEGIN PGP SIGNATURE-----

iF0EARECAB0WIQSpjoTS23RkDkwuQWwdXwgKg/vPJwUCXcX64wAKCRAdXwgKg/vP
J8EADAJ0c2ofRQr89s8QlVSHMwoXWtQdf+wCfRkcExhXrvrj2Q00VHpvLFQbQFLI=
=/fqP
-----END PGP SIGNATURE-----
franky@ubuntu18:~$
```

Conclusión

De las mejores prácticas que podemos hacer. Me ha divertido mucho aprender sobre los cifrados, las últimas partes me han costado entenderlas ya que no conseguía saber como juntar todo lo realizado en un solo archivo, pero con la explicación de algunos compañeros y comprendiendo lo que pedía el ejercicio me conseguí aclarar y pude comprender cada cosa perfectamente.