***Imagen que contiene Logotipo

Descripción generada automáticamente***

Alejandro Ortega Martínez

Grado en Ingeniería Informática

12/4/2021

Practica 1

Criptografía

Diseño y Administración de Sistemas y Redes

Contenido

[Cifrado de passwords 2](#_Toc69063491)

[Programas de Cifrado disponibles en Linux 4](#_Toc69063492)

[GPG: GNU Privacy Guard 6](#_Toc69063493)

# Cifrado de passwords

**1) Estudiar el manual de la llamada al sistema crypt(3).**

Crypt es la función de Linux para encriptar contraseñas. Esta basada en el algoritmo Data Encryption Standard (DES), a la que se le añaden variantes, como es la sal, para hacer mas complejo el romper una clave.

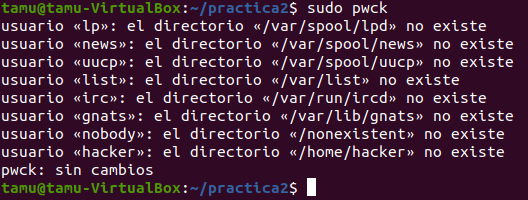
A esta función se le pasan dos parámetros, la contraseña del usuario, y su sal.

Este algoritmo coge los 7 primeros bits de los 8 primeros caracteres de la contraseña, consiguiendo una key de 56 bits. Con esta llave de 56 bits se encripta repetidamente un String de 0, y se compara con la contraseña encriptada, para ver si coincide.

Si la llamada ha ido como lo esperado, la función retorna un puntero a la contraseña encriptada. Si no es así, retorna NULL.

**2) Intentar localizar diferentes posibilidades de comprobación de contraseñas en nuestra máquina y tras entender su utilización por medio del man, probarlas en un entorno seguro. Comprobar la instalación o en su defecto instalar el paquete pwck (Verificación de la integridad del fichero de contraseñas) y pwgen (Password generator).**

El comando pwck lo que hace es comprobar si esta todo en orden en el fichero de contraseñas (fichero shadow).



Al ejecutarlo en mi maquina, vemos que en algunos usuarios faltan directorios. Sin mbargo, son solo “warnings”, no son errores, por lo que no nos ha solicitado hacer ningún cambio.

En cuanto al comando pwgen, se encarga de generar contraseñas que intentan ser fácilmente recordables por humanos. Estas contraseñas no son tan seguras como las aleatorias, pero se intenta maximizar su seguridad.

Se le pueden pasar varios parámetros, como la longitud de las contraseñas, la cantidad de contraseñas que queremos que genere, como queremos que nos las muestre, o su longitud.



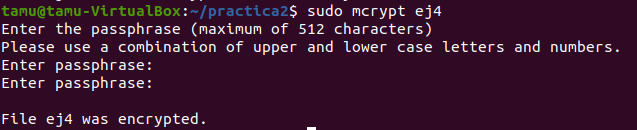
En este caso le hemos pedido que nos muestre las contraseñas de una en una (**-1**), que haya por lo menos una mayúscula en cada contraseña (**-C**), y que nos saque 10 contraseñas de 8 caracteres cada una.

**3) Instalar de igual forma el paquete mcrypt (Sustitución del comando crypt) y probar su utilización.**

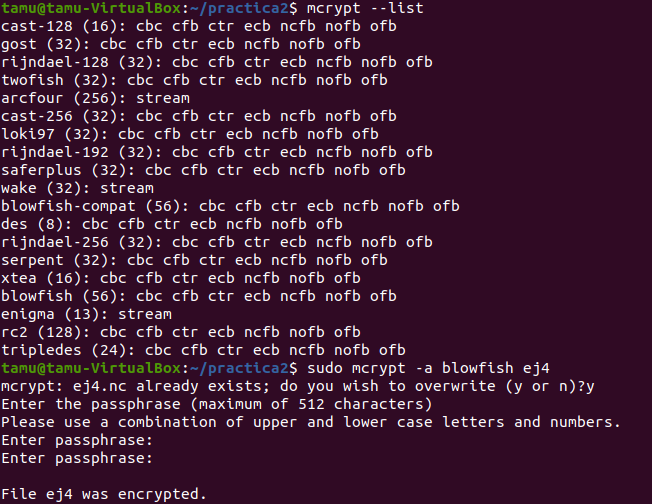
Este paquete viene ya instalado, y lo probaremos en ejercicios posteriores.

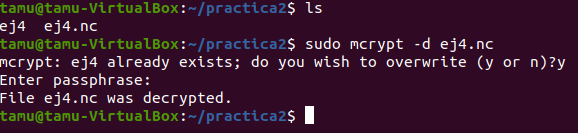
# Programas de Cifrado disponibles en Linux

**4) Utilizar la orden crypt para cifrar y descifrar ficheros.**

Para encriptar fichero usaremos el comando mcrypt. Si no especificamos nada (solo el nombre del fichero a encriptar), no pedirá la contraseña dos veces, y lo encriptará. 

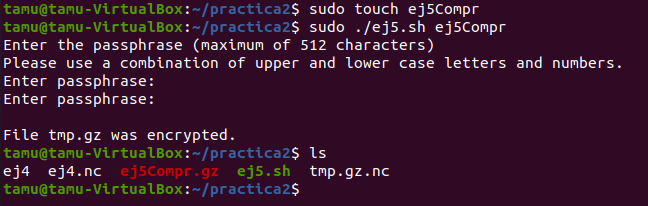
Si queremos encriptar un fichero usando un algoritmo concreto, usaremos el parámetro **-a** y el nombre del algoritmo (para ver la lista de algoritmos completos, usaremos la opción **mcrypt -list**).



Para desencriptarlo, tan solo tenemos que usar el parámetro **-d**.

**5) (Optativo a valorar) Crear un guió en shell que combine el cifrado y la compresión de textos para mejorar la orden crypt.**

****

****

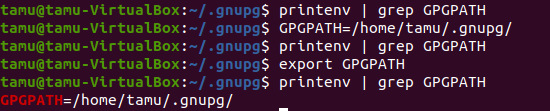
# GPG: GNU Privacy Guard

**6. Comprobar si existe un directorio llamado .gnupg en el directorio $HOME y asegurarse de que están instalados todos los archivos y librerías relacionados con gpg. En caso de no existir el directorio, crearlo.**

En mi caso si que existe, por lo que no hace falta crear dicho directorio.



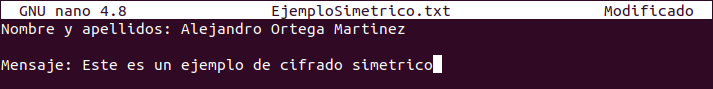
**7. Verificar si existe una variable GPGPATH apuntando al directorio .gnupg, en caso negativo, definirla.**



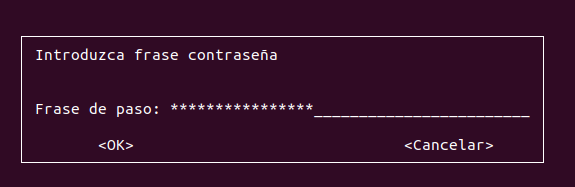
En mi caso esta variable no existía, por lo que la he tenido que crear, y hacerla variable global con el comando export.

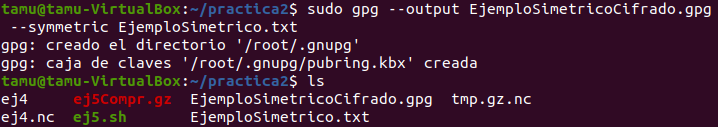
**8) Crear un fichero “EjemploSimetrico.txt” y cifrarlo mediante el sistema simétrico**

Primero creamos el fichero con nano:



Y posteriormente lo ciframos con gpg:

Primero nos pide la contraseña, tras introducirla, ya tendremos creado nuestro fichero cifrado:

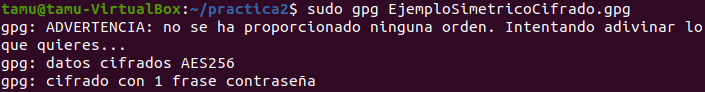


**9. Comprobar el contenido del fichero resultante y descifrarlo.**

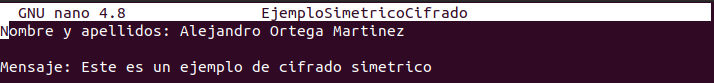
****

Como podemos observar, el fichero resultante esta lleno de caracteres ininteligibles, algunos de ellos no se pueden siquiera imprimir por pantalla.

Para descifrarlo, utilizamos el siguiente comando:



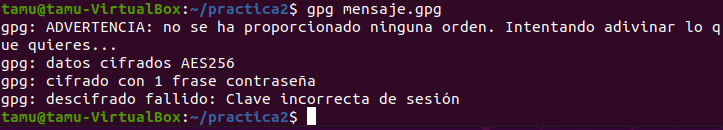
Tras introducir la contraseña, obtenemos el fichero EjemploSimetricoCifrado, con el siguiente contenido:



Como podemos observar, se ha cifrado y descifrado correctamente.

**10.Tratar de descifrar el texto con una frase de paso errónea**

Si intentamos descifrarlo con una clave errónea, nos salta el siguiente mensaje:

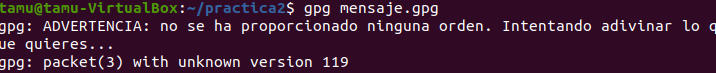


**11. Modificar el contenido del fichero cifrado e intentar descifrarlo.**

Hemos modificado el contenido del fichero de la siguiente manera:



Al intentar descifrarlo, da el siguiente error:



**12.¿Puedes enviar mensajes cifrados mediante este sistema por correo?. ¿Cuál es el principal problema que se plantea?**

No se pueden enviar estos mensajes cifrados, ya que no todos los caracteres que contienen son ASCII, por lo que a la hora de copiarlos en el cliente de correo, este no va a entender la mayoría.

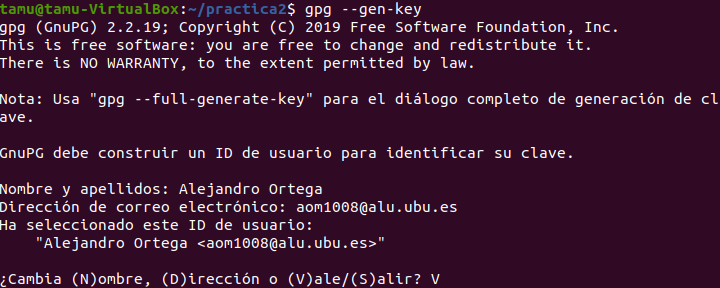
Por eso hay una opción particular que nos permite encriptar datos con caracteres ASCII para enviar por correo (**-ca**).

A parte, para descifrarlos, tendríamos que pasar la clave para descifrarlo por correo; por tanto, cualquiera que intercepte el mensaje cifrado, también tendrá acceso a la clave.

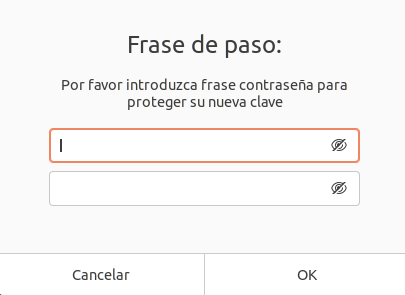
**13.Consultar las opciones -gk o –gen-key en el manual de gpg y utilízalas para generar tus claves.**

Según el manual, esta opción genera un nuevo par de claves usando los parámetros por defecto. Este es el comando estándar para crear una nueva clave.

Primero nos pide introducir los datos:

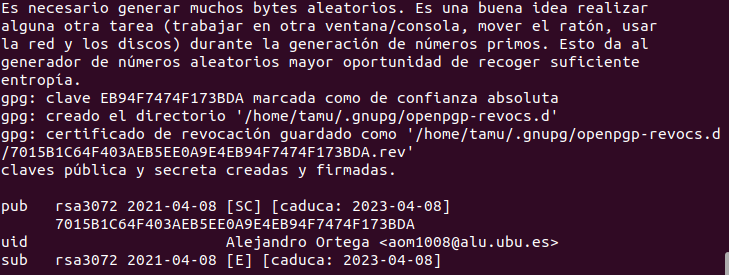


Mas adelante, nos pide escribir la contraseña para proteger nuestra clave.



Posteriormente, se pondrá a crear la clave, por lo que nos pedirá que movamos el ratón, trabajemos en otras cosas, etc, para conseguir mas entropía, y encontrar números primos más aleatorios.

Por último, nos creara nuestra clave.



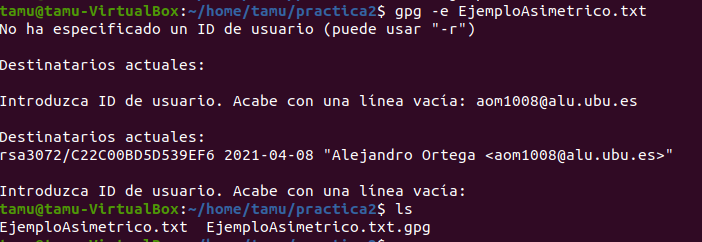
**14.Consulta los ficheros creados en tu directorio .gnupg**



Se han creado la semilla de aleatoriedad, los ficheros con la clave publica, y otros ficheros mas.

* **Pubring.kbx**: La clave publica usando el formato keybox
* **Trustdb.gpg:** El fichero candado para la base de datos trust
* **Random\_seed:** Un fichero usado para mantener el estado de la pool aleatoria interna
* **Opnegpg-revocs.d:** Directorio dondegpg guarda los certificados de revocación pre-generados.

**15.Crear el fichero “EjemploAsimetrico.txt” y cifralo mediante el método asimétrico (cifrado con tu clave pública).**



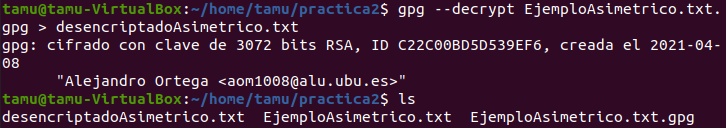
He probado a hacer el cifrado sin introducir el id de usuario, y te lo pide de forma interactiva al final.

**16.Comprobar el contenido del fichero resultante y descifrarlo**

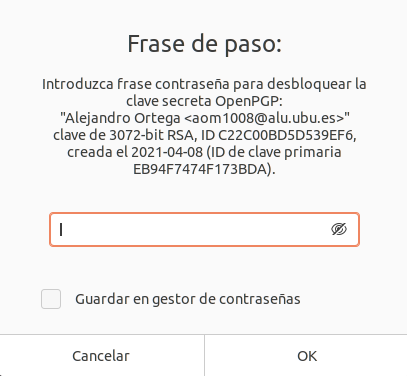
****

Como podemos observar, el resultado es un fichero ilegible.

Para desencriptarlo, usaremos los siguientes comandos.

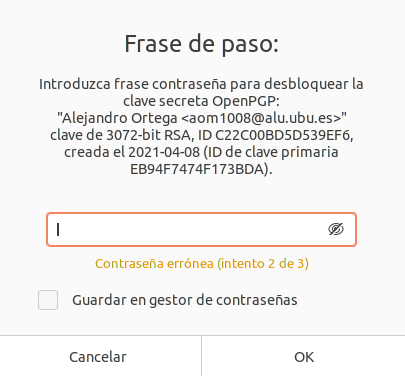


Para ello, nos pide la frase que introducimos para nuestra clave privada.

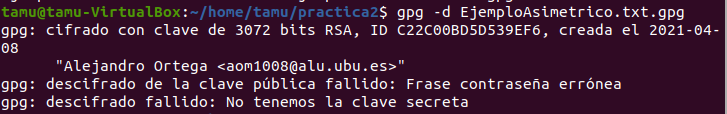


Tras introducirla, tendremos nuestro fichero desencriptado.

**17.Tratar de descifrar el texto con una frase de paso errónea.**

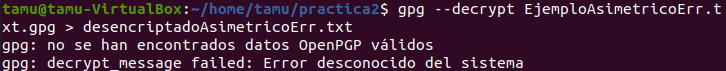
****

Como podemos observar, nos da tres intentos para introducir la contraseña y, si fallamos los tres, nos da el siguiente error.



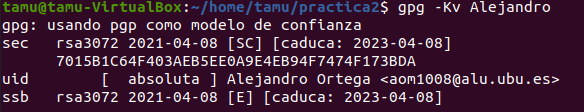
**18.Modificar el contenido del fichero cifrado e intentar descifrarlo.**

Hemos introducido lo mismo que con el ejemplo simétrico, y el error que nos da es el mostrado a continuación.



**19.Visualiza primero y extrae después tu clave pública sobre un fichero en ASCII. Comprueba el contenido del fichero.**

El contenido de mi clave publica es el siguiente:

****

Para exportarla, he hecho lo siguiente:

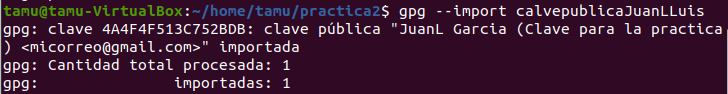


**20.Envía el fichero con tu clave a algunos compañeros con los que quieras intercambiar posteriormente ficheros cifrados y/o firmados (mediante mail), y que ellos también te envíen sus claves públicas.**

Mi clave la intercambiare con Juan Luis Garcia ([jgg1009@alu.ubu.es](mailto:jgg1009@alu.ubu.es))

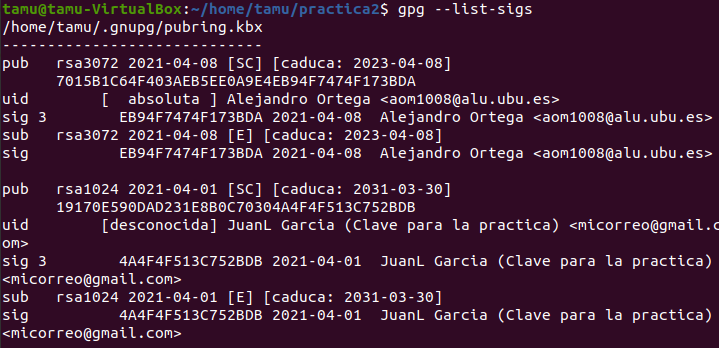
**21.Añade las claves públicas recibidas en tu propio llavero público.**

Con el comando –import podremos añadir el fichero que mi compañero me ha enviado, el cual contiene su clave pública.



**22.Comprueba el contenido de tu llavero público.**

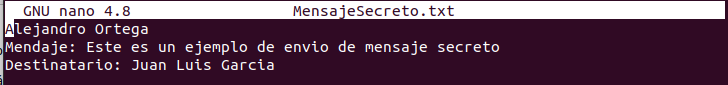
Con el comando –list-sigs podemos ver nuestro anilli de claves:



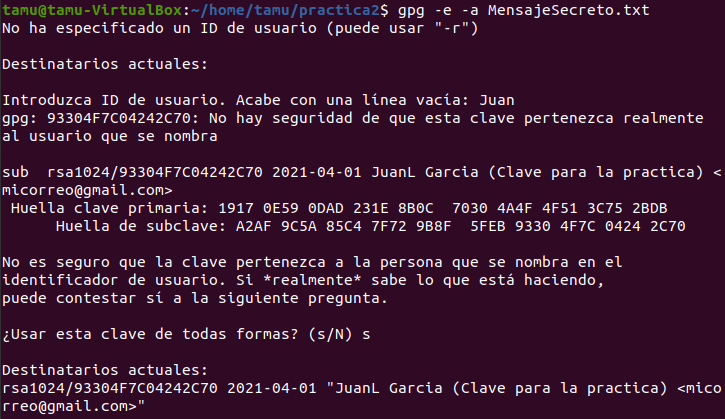
Como podemos observar, en los parámetros sig 3, esta tanto mi clave (Alejandro ortega), como la de mi compañero (JuanL Garcia)

**23.Crea el fichero “MensajeSecreto.txt” y cifralo con la clave pública de un compañero y guárdalo como ASCII.**

Primero creamos el fichero:



Y lo ciframos con la clave pública de mi compañero (guardándolo como ascii) Nos pide una confirmación, pero escribiendo s conseguiremos cifrarlo:

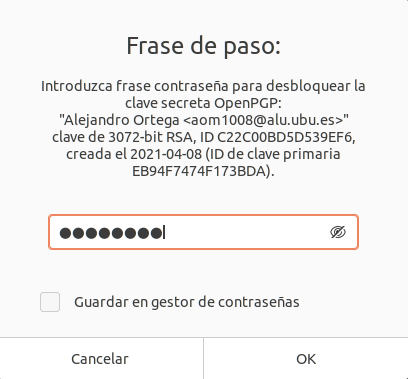


**24.Envíaselo por correo (mediante mail o cualquier otra aplicación existente) y que él te envíe el suyo a ti. Guarda el mensaje recibido en un fichero.**

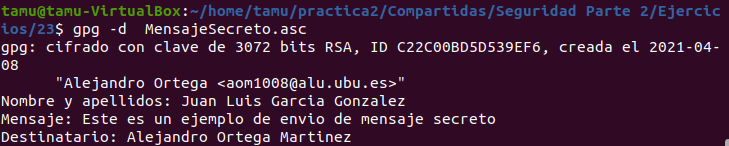
Le he enviado el fichero a mi compañero Juan Luis Garcia

**25.Descífralo con tu clave privada.**

Este proceso es igual qe anteriormente, nos pide nuestra contraseña, y si lo hemos escrito bien, nos descifra el mensaje.

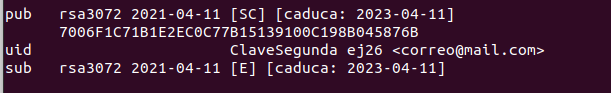


Como resultado obtenemos lo siguiente:

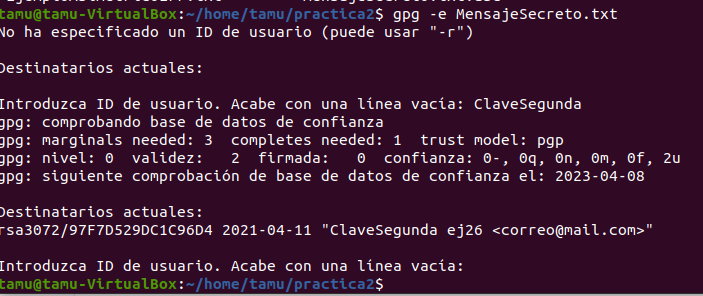


**26.Cifra el mismo mensaje con otra clave pública y envíaselo al mismo compañero**

Para esto, crearemos un nuevo par de claves pública-privada:



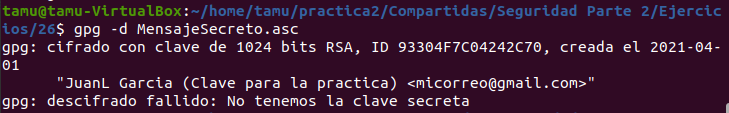
Y encriptamos el mensaje con esta clave:



**27.Recibe el mensaje de tu compañero y guárdalo en un fichero.**

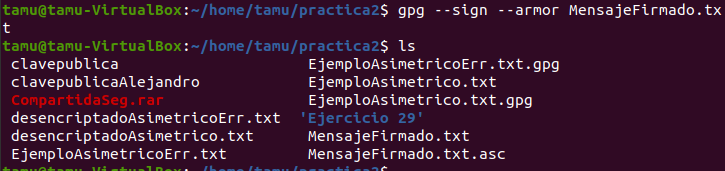


**28.Intenta descifrarlo con tu clave privada**



Como era de esperar, no se puede descifrar porque no tenemos la clave con la que se cifro.

**29.Crea el fichero “MensajeFirmado.txt” fírmalo con tu clave privada y guárdalo como ASCII.**

****

Como podemos observar, se ha creado en .asc, para poder mandarlo por correo

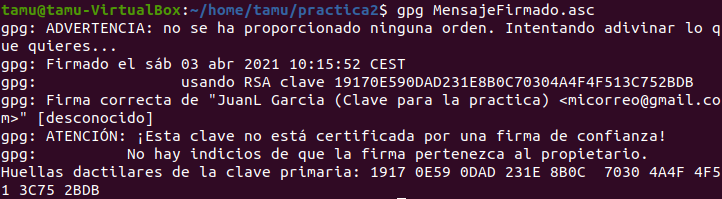
**30.Envía el fichero a un compañero que posea tu clave pública.**

Se lo he enviado a mi compañero Juan Luis García.

**31.Recibe algún o algunos mensaje firmados de tus compañeros y guárdalos en sendos ficheros.**

He recibido el mensaje firmado de mi compañero Juan Luis Garcia.

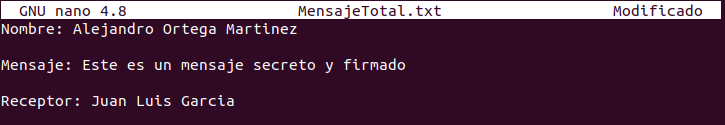
**32.Verifica la firma de los mensajes recibidos y comprueba el contenido de los mismos.**

****

Podemos verificar que la firma es, en efecto, de mi compañero Juan Luis García, y su contenido es el siguiente:



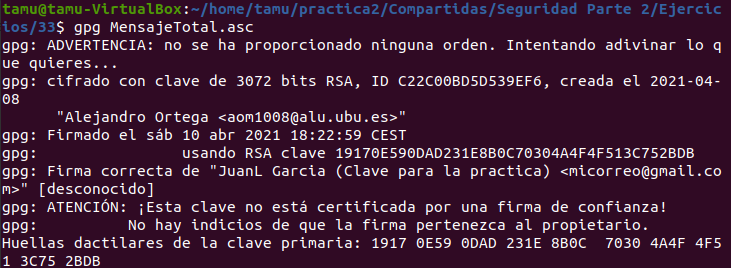
**33.Crea el fichero “MensajeTotal.txt” y genera un mensaje secreto y auténtico y envíaselo a un compañero.**



También se lo enviare a mi compañero juan Luis garcia



**34.Recibe algún mensaje con estas características, verifícalo y descífralo.**



Podemos ver que la firma es correcta, y que nos ha generado el mensaje:

