FIRMA DIGITAL LINUX

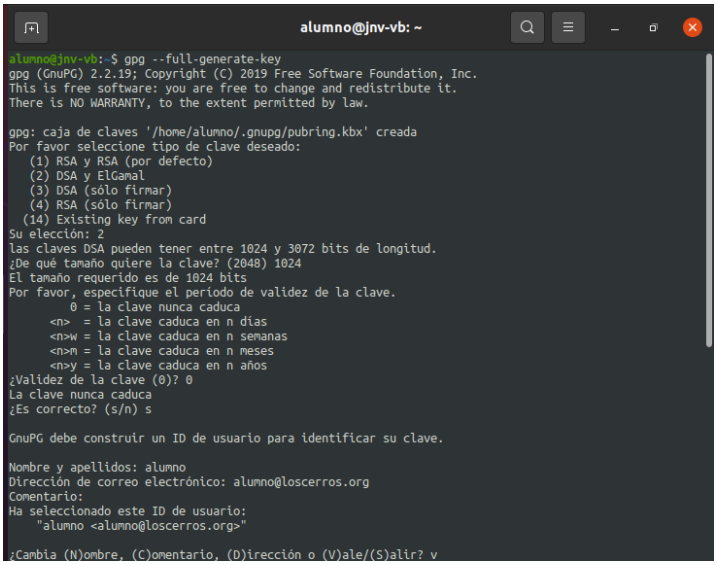
Juan Carlos Navidad García

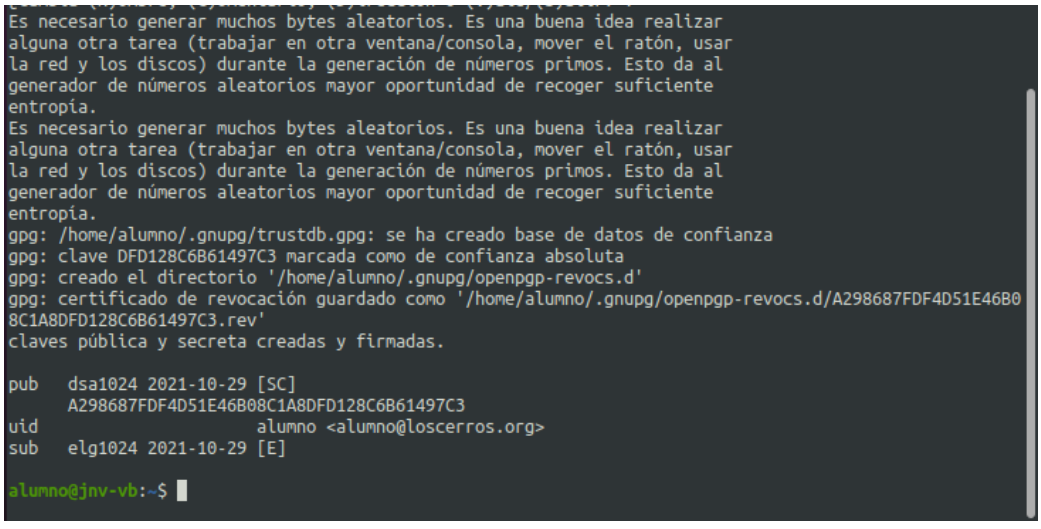
Seguridad Informática

1. Cifrar de manera asimétrica en Linux:

Entramos con un usuario de la máquina (en este ejemplo, alumno). Lo primero será generar un **par de claves de criptografía asimétrica**, nuestra **propia clave pública y clave privada.** El comando es:

alumno$ gpg –full-generate-key





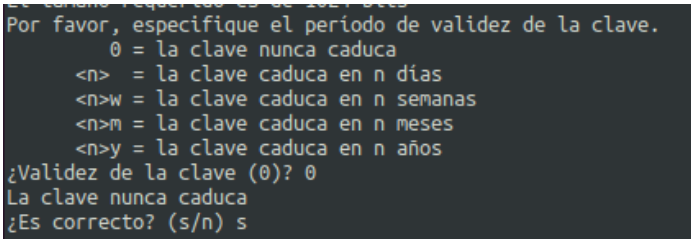
En el proceso de generación de la clave nos preguntarán varios detalles. El primero es el tipo de clave. La herramienta nos ofrece cuatro opciones. Los nombres se corresponden con el tipo de algoritmo asimétrico asociado (hay varios tipos, como también ocurría en criptografía simétrica: DES, AES, etc.). Es decir, una clave de tipo **DSA se utiliza en un algoritmo DSA, y una clave Elgamal, en un algoritmo Elgamal**. Las dos primeras opciones ofrecen dos algoritmos, luego generan dos pares de claves: en total, cuatro claves, dos públicas y dos privadas. **El motivo es que generalmente se utiliza una clave (un par) para cifrar y otra diferente (otro par) para firmar**. Elegimos la opción 2, que tiene algoritmos distintos, y así veremos claramente cuándo se utiliza cada clave.

```
gpg: caja de claves '/home/alumno/.gnupg/pubring.kbx' creada
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)
  (14) Existing key from card
Su elección: 2
```

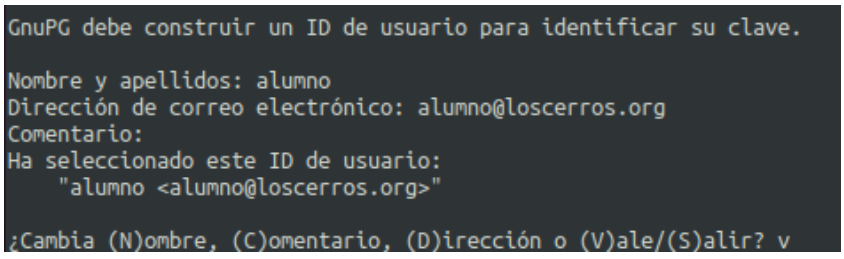
A continuación, nos pregunta el tamaño de la clave del **algoritmo DSA**. Por defecto ofrece **2048**, pero en este ejemplo elegimos el mínimo, **1024**:

```
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
```

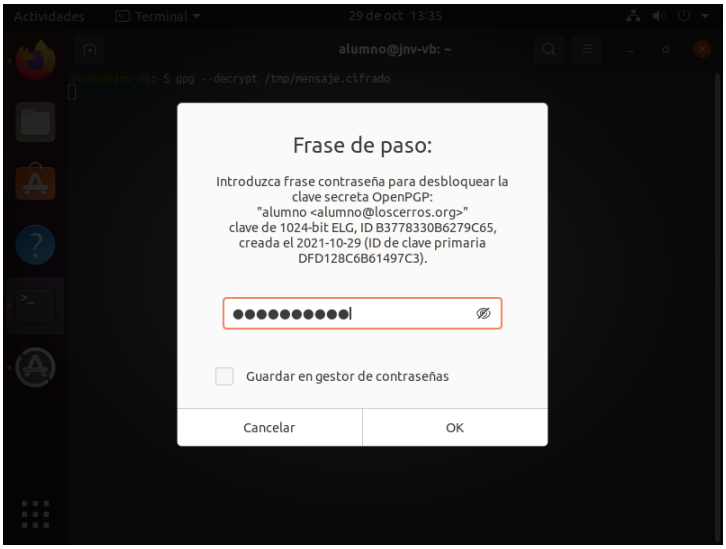
La siguiente pregunta es el **periodo de validez de la clave**. Ya estamos advertidos de los problemas que supone proteger la clave privada. Por si la perdemos, conviene fijar una fecha de caducidad para que no se pueda usar más allá de ese día. En nuestro ejemplo no hace falta tanta seguridad y elegiremos que nunca caduque.



A continuación, nos pide algunos datos para identificar la clave. Ya sabemos que en el llavero se pueden almacenar varias claves (de hecho, estamos generando dos pares ahora mismo). Para elegir una u otra en cada ocasión, tenemos que identificarlas fácilmente. En este caso nos pedirá un **nombre**, una **dirección de correo** y un **comentario**. La herramienta **gpg** nunca nos enviará un correo; pero es una forma de contactar con el dueño de la clave (entregaremos nuestra clave pública a mucha gente, y ellos tendrán las claves públicas de otras personas).



El siguiente paso es elegir la **clave simétrica** que protegerá nuestras claves. Como siempre, debe ser una clave fácil de recordar para nosotros y difícil de averiguar para cualquier otra persona. Si la olvidamos, no podremos utilizar nuestras claves asimétricas. En mi caso, he utilizado la contraseña **Cerros2020**.



Ya no hay más preguntas. Ahora la herramienta ejecuta los procedimientos matemáticos para obtener las claves asimétricas que hemos solicitado. Estos procedimientos necesitan muchos datos aleatorios, por lo que nos pide que generemos actividad en el sistema para ayudarle.

```
claves pública y secreta creadas y firmadas.

pub   dsa1024 2021-10-29 [SC]
      A298687FDF4D51E46B08C1A8DFD128C6B61497C3
uid                      alumno <alumno@loscerros.org>
sub   elg1024 2021-10-29 [E]

alumno@jnv-vb:~$
```

Si nos fijamos, veremos una clave primaria (**pub**) de tipo **DSA** con tamaño de clave **1024** (1024D). Debajo hay una clave subordinada de tipo **Elgamal** con tamaño de clave **1024** (1024g).

En el directorio **HOME** del usuario se ha creado un directorio oculto llamado **.gnupg**, donde se guardan los ficheros internos que utiliza la herramienta. El fichero **pubring.kbx** contiene las claves públicas y el directorio **private-keys-v1.d**, las claves privadas. Por supuesto, ambos están cifrados.

```
alumno@jnv-vb:~$ ls -l .gnupg/
total 20
drwx------ 2 alumno alumno 4096 oct 29 13:17 openpgp-revocs.d
drwx------ 2 alumno alumno 4096 oct 29 13:17 private-keys-v1.d
-rw-rw-r-- 1 alumno alumno 1166 oct 29 13:17 pubring.kbx
-rw------- 1 alumno alumno   32 oct 29 13:17 pubring.kbx~
-rw------- 1 alumno alumno 1240 oct 29 13:17 trustdb.gpg
```

Para trabajar con las claves debemos utilizar la propia herramienta. La lista de claves la obtenemos con el parámetro **list-keys**. En pantalla aparecerá nuestra clave principal y la subordinada.

- ***alumno$ gpg --list-keys***

```
alumno@jnv-vb:~$ gpg --list-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/alumno/.gnupg/pubring.kbx
------------------------------
pub   dsa1024 2021-10-29 [SC]
      A298687FDF4D51E46B08C1A8DFD128C6B61497C3
uid        [  absoluta ] alumno <alumno@loscerros.org>
sub   elg1024 2021-10-29 [E]

alumno@jnv-vb:~$
```

Comparado con el caso del cifrado simétrico, estamos en el paso 13 y todavía no hemos cifrado nada: como ya suponíamos, la criptografía asimétrica es un poco más complicada. Ahora tenemos que comunicar nuestra clave pública a quien esté interesado en enviarnos un mensaje cifrado. Primero tenemos que sacarla del llavero. El parámetro es **export**:

- ***alumno$ gpg –a --export –o /tmp/alumno.pub alumno***

Hemos utilizado los parámetros **a** (**armor**) para que el resultado no sea binario y **o** (**output**) para guardarlo directamente en un fichero (si no, aparece por la salida estándar). El fichero lo ponemos tranquilamente en**/tmp** porque no nos importa que otros usuarios lo copien.

```
alumno@jnv-vb:~$ gpg -a --export -o /tmp/alumno.pub alumno
alumno@jnv-vb:~$
```

En la máquina tenemos un segundo usuario llamado ***profesor***. Entramos con este usuario para enviar un mensaje cifrado al usuario alumno con la misma herramienta **gpg**. Para coger las claves públicas de alumno utilizamos el parámetro **import**:

- ***profesor$ gpg --import /tmp/alumno.pub***

```
profesor@jnv-vb:~$ gpg --import /tmp/alumno.pub
gpg: creado el directorio '/home/profesor/.gnupg'
gpg: caja de claves '/home/profesor/.gnupg/pubring.kbx' creada
gpg: /home/profesor/.gnupg/trustdb.gpg: se ha creado base de datos de confianza
gpg: clave DFD128C6B61497C3: clave pública "alumno <alumno@loscerros.org>" importada
gpg: Cantidad total procesada: 1
gpg:               importadas: 1
```

Podemos consultar las claves disponibles en el llavero con el mismo parámetro **list-keys** que vimos anteriormente:

- ***profesor$ gpg --list-keys***

```
profesor@jnv-vb:~$ gpg --list-keys
/home/profesor/.gnupg/pubring.kbx
---------------------------------
pub   dsa1024 2021-10-29 [SC]
      A298687FDF4D51E46B08C1A8DFD128C6B61497C3
uid           [desconocida] alumno <alumno@loscerros.org>
sub   elg1024 2021-10-29 [E]
```

Vamos a crear un fichero llamado mensaje y lo cifraremos para enviárselo al usuario alumno. Los comandos serían:

- ***profesor$ fortune > mensaje***
- ***profesor$ gpg –v –a –o /tmp/mensaje.cifrado --encrypt --recipient alumno mensaje***

```
profesor@jnv-vb:~$ touch mensaje
profesor@jnv-vb:~$ fortune > mensaje
profesor@jnv-vb:~$ cat mensaje
question = ( to ) ? be : ! be;
                  -- Wm. Shakespeare
```

```
profesor@jnv-vb:~$ gpg -v -a -o /tmp/mensaje.cifrado --encrypt --recipient alumno mensaje
gpg: usando pgp como modelo de confianza
gpg: usando subclave B3778330B6279C65 en vez de clave primaria DFD128C6B61497C3
gpg: B3778330B6279C65: No hay seguridad de que esta clave pertenezca realmente
al usuario que se nombra

sub  elg1024/B3778330B6279C65 2021-10-29 alumno <alumno@loscerros.org>
 Huella clave primaria: A298 687F DF4D 51E4 6B08  C1A8 DFD1 28C6 B614 97C3
      Huella de subclave: 5D6B 84B8 D847 BD1A 30C1  6898 B377 8330 B627 9C65

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 'mensaje'
gpg: escribiendo en '/tmp/mensaje.cifrado'
gpg: ELG/AES256 cifrado para: "B3778330B6279C65 alumno <alumno@loscerros.org>"
profesor@jnv-vb:~$
```

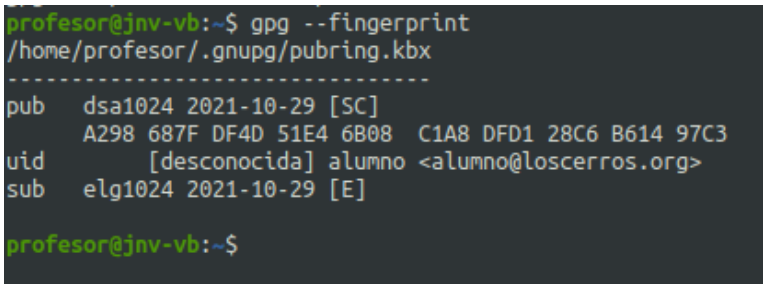
Los parámetros **a**y**o** ya los conocemos (dejamos el fichero en **/tmp** porque no nos importa que otros usuarios puedan leerlo: sin la clave privada, el contenido es ininteligible). Hemos añadido el parámetro **v** (**verbose**) para obtener más información. En este caso, nos sirve para conocer que utilizará la clave subordinada (**Elgamal**), en lugar de la clave primaria (**DSA**).

El parámetro **encrypt** indica que deseamos cifrado asimétrico y el parámetro **recipient** va seguido del identificador de la clave pública que queremos utilizar. Como ya sabemos, el cifrado utiliza la clave pública del receptor.

Una vez introducido ese comando, la respuesta es una advertencia: no hay seguridad de que esa clave pública sea realmente la clave pública de alumno. Cualquiera podría haber cambiado el fichero **/tmp/alumno.pub** antes de que profesor hiciera el **import** de las claves. Como ayuda, el comando nos ofrece la huella (**fingerprint**) de la clave y nos pide la confirmación de que es la clave que queremos utilizar.

Podemos volver a la sesión del usuario alumno, obtener la huella de su clave pública y compararla con la que aparece en la sesión de profesor. El parámetro para obtener la huella es **fingerprint**

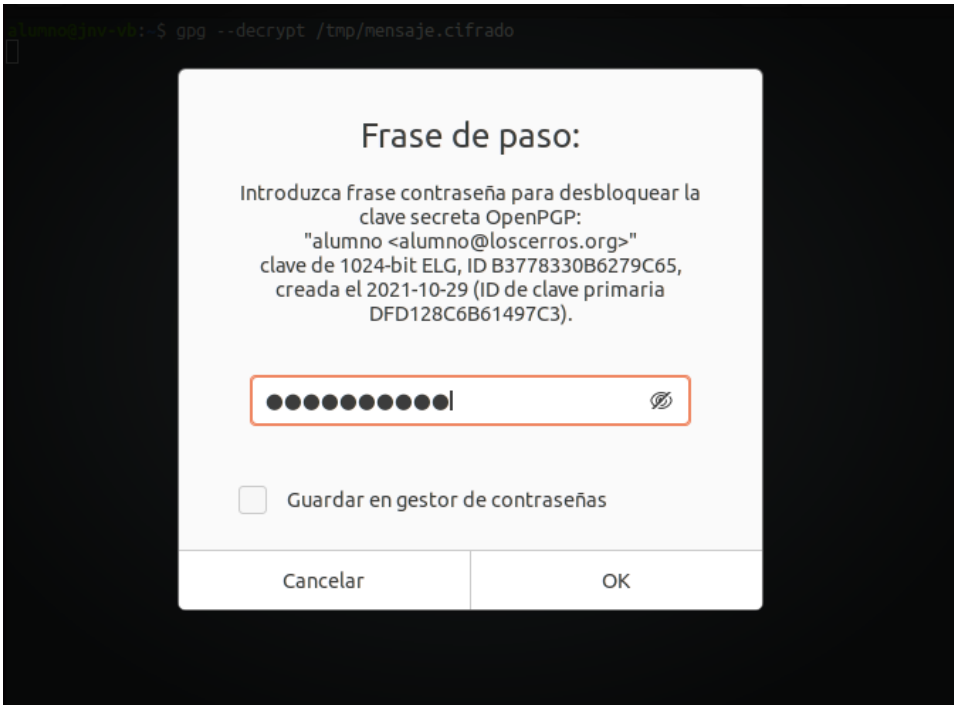
- ***alumno$ gpg --fingerprint***



Efectivamente, las huellas coinciden y podemos confiar en que la clave importada en profesor es correcta. Confirmamos que queremos usar esa clave y se crea el fichero cifrado. Ahora podemos volver a la sesión del usuario alumno e intentar descifrarlo. El parámetro es **decrypt**:

- ***alumno$ gpg --decrypt /tmp/mensaje.cifrado***

Como esperábamos, el comando nos solicita la contraseña que da acceso a la clave privada. En la ventana aparece qué clave necesita (en nuestro caso, la clave del algoritmo **Elgamal**), con toda su identificación.



Aquí se puede ver el mensaje ya descifrado:

```
alumno@jnv-vb:~$ gpg --decrypt /tmp/mensaje.cifrado
gpg: cifrado con clave de 1024 bits ELG, ID B3778330B6279C65, creada el 2021-10-29
      "alumno <alumno@loscerros.org>"
question = ( to ) ? be : ! be;
                -- Wm. Shakespeare
alumno@jnv-vb:~$
```