

CIFRADO SIMÉTRICO Y ASIMÉTRICO CON GPG

El programa **GnuPG** es una implementación del estándar **OpenPGP**, que deriva del software criptográfico PGP desarrollado por Phil Zimmermann. El objetivo de esta sesión de laboratorio es aprender a realizar las tareas más sencillas de manejo de PGP/GnuPG, a saber:

- Invocar el programa
- Usarlo para cifrar y descifrar un documento (de texto o binario) por medio de
- Criptografía simétrica.
- Intercambiar correo cifrado con un compañero
- Generar un par clave pública/privada
- Distribuir nuestra clave pública
- Emplear el mecanismo de clave pública para intercambiar correo de forma segura
- A través de un canal inseguro
- Firmar digitalmente un documento y comprobar la firma

1. Ejecutar gpg

Para poder utilizar este programa, escribiremos **gpg** en la consola. Podemos obtener ayuda con el comando **man gpg**. En la ayuda podemos obtener todas las opciones que debemos utilizar para realizar las diferentes tareas con gpg. Las opciones que utilizaremos en esta práctica serán las siguientes:

<i>Opción</i>	<i>Significado</i>
-c	Cifrar con un algoritmo simétrico
-a	Produce una salida ASCII codificada en BASE64
--gen-key	Genera un par de claves
--export	Exporta una o más claves públicas
--import	Importa las claves públicas a nuestro keyring
-kv	Verifica o comprueba nuestro keyring
-kvc	Lista el fingerprint (huella) del keyring
--encrypt	Cifrar con criptografía asimétrica
-r	Especificar destinatario
-s	Firma digitalmente un documento
-b	Firma separada
-clearsign	Firma sin cifrar

2. Cifrado simétrico

Para cifrar simétricamente un documento utilizamos la opción `-c`. Su invocación es como sigue:

```
$ gpg -c documento_que_cifrar
```

El programa **gpg** nos solicitará una contraseña con la que él cifrará el documento. Hecho esto, nos creará un documento cifrado con extensión `.gpg` que podemos enviar a un destinatario, éste podrá descifrarlo simplemente invocando:

```
$ gpg documento_cifrado.gpg
```

y el propio **gpg** reconocerá el formato del documento, solicitará la contraseña al destinatario y generará el documento descifrado si todo ha ido bien.

Ejercicio 1 *Cifrado simétrico de un documento.*

1. *Crea un documento de texto con cualquier editor o utiliza uno del que dispongas.*
2. *Cifra este documento con alguna contraseña acordada con el compañero de al lado.*
3. *Haz llegar por algún medio al compañero de al lado el documento que acabas de cifrar.*
4. *Descifra el documento que te ha hecho llegar tu compañero de al lado.*
5. *Repite el proceso anterior, pero añadiendo la opción `-a`. Observa el contenido del archivo generado con un editor de textos o con la orden `cat`.*
6. *Copia y pega el contenido del archivo cifrado anteriormente y envíalo por mail a tu compañero para que lo descifre.*
7. *Una vez has recibido el mensaje de tu compañero en tu mail, copialo en un archivo de texto para obtener el mensaje original.*

3. Creación de la pareja de claves pública-privada

Los algoritmos de cifrado asimétrico utilizan dos claves para el cifrado y descifrado de mensajes. Cada persona involucrada (receptor y emisor) debe disponer, por tanto, de una pareja de claves pública y privada.

Para generar nuestra pareja de claves con **gpg** utilizamos la opción `--gen-key`:

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

Tras ejecutar **gpg** con esta opción, empieza un proceso interactivo que va preguntando al usuario, que debe decidir entre una serie de opciones. Iremos explicando este proceso paso a paso.

Esto es lo que nos aparecerá tras ejecutar el comando:

```
gpg (GnuPG) 1.4.2.2; Copyright (C) 2005 Free Software Foundation, Inc.  
This program comes with ABSOLUTELY NO WARRANTY.
```

This is free software, and you are welcome to redistribute it under certain conditions. See the file COPYING for details.

Por favor seleccione tipo de clave deseado:

- (1) DSA y ElGamal (por defecto)
- (2) DSA (sólo firmar)
- (5) RSA (sólo firmar)

¿Su elección?:

Nos pide que seleccionemos el tipo de clave que queremos crear. Vamos a elegir el tipo de clave por defecto, opción 1.

A continuación nos pide que indiquemos la longitud de las claves, a mayor longitud de la clave, mayor seguridad obtendremos. Marcamos la opción por defecto, 2048 bits.

El par de claves DSA tendrá 1024 bits.

las claves ELG-E pueden tener entre 1024 y 4096 bits de longitud.

¿De qué tamaño quiere la clave? (2048)

El siguiente paso es indicar el periodo de validez de la clave. Esto es necesario para que pasado un cierto tiempo, la clave que vamos a crear deje de ser válida. El motivo para hacer esto es que, pasado un cierto tiempo, la seguridad de nuestras claves se ve comprometida puesto que alguien ha podido intentar descubrirlas. Por ello, es recomendable, establecer una fecha de caducidad a nuestras claves.

Por favor, especifique el período 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)?

A continuación crearemos el identificador de nuestra clave, que será lo que tengamos que indicar cuando queramos utilizar nuestra clave primaria. GPG crea el identificador de la clave utilizando los datos introducidos: nombre, apellidos, email... Esto se puede ver en el siguiente cuadro:

Necesita un identificador de usuario para identificar su clave. El programa construye el identificador a partir del Nombre Real, Comentario y Dirección de Correo Electrónico de esta forma:

"Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Nombre y apellidos: Jose L. Berenguel

Dirección de correo electrónico: profesorjose@gmail.com

Comentario:

Ha seleccionado este ID de usuario:

"Jose L. Berenguel <profesorjose@gmail.com>"

¿Cambia (N)ombre, (C)omentario, (D)irección o (V)ale/(S)alir? V

Dirección de correo electrónico: profesorjose@gmail.com

Ha seleccionado este ID de usuario:

¿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.

gpg: clave 699AB38D marcada como de confianza absoluta
claves pública y secreta creadas y firmadas.

gpg: 3 dudosa(s) necesarias, 1 completa(s) necesarias,
modelo de confianza PGP

pub 1024D/699AB38D 2007-04-25

uid Jose L. Berenguel <profesorjose@gmail.com>

sub 2048g/331B42E9 2007-04-25

Ejercicio 2 Creación de nuestro par de claves pública-privada.

1. Siguiendo las indicaciones de este epígrafe, crea tu par de claves pública y privada. La clave que vas a crear tendrá una validez de 1 mes.
2. Recuerda el ID de usuario de tu clave y la contraseña de paso utilizada. Anotala en un lugar seguro si lo consideras necesario.

4. Importar y exportar claves públicas

Para enviar archivos cifrados a otras personas, necesitamos disponer de sus claves públicas. De la misma manera, si queremos que cierta persona pueda enviarnos datos cifrados, ésta necesita conocer nuestra clave pública. Para ello, podemos hacérsela llegar por email por ejemplo.

Si queremos ver cómo es nuestra clave pública utilizaremos el siguiente comando:

```
$ gpg -a --export key_id
```

donde *key_id* es el ID de la clave que queremos visualizar:

```
gpg -a --export Jose L. Berenguel
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.2.2 (GNU/Linux)

mQGibEYvJykRBADKPmbaNyMZAeUBRmAxzYQZbHVH4Cnt517SJ41CEnWz/U69DoG
T
+4N62dTTa40yLSEZrJmCv+s5dsorhRRjjZQcasKAE80V/k32Lvi9CZsAwaqKpOBW
86qf64OvbWpYy9TdXdtWow+41qKN2+x13X3mq3uVhp2iZsZaKGSFVekmqwCgx0
Ry
gBSp5WLg0aPyJ/LR4x7W9DsD/3J9hyIYriWfqT/AqvDPuxSjrlCijnZiwOEtlp15
dSNJ9CKysvK6HG3K+8/T37tgqOpKH+5fM5ZscPxqCskkwYg8T1AOMHyGZ0yXLjL
5089mDdIN2ZHqcpBfX/0OSOBypPE5IpUFueIHJMT7JQMIAHt6S485lnlr0xWYPhs
MkCSA/0VuFfhRlmz3m/qAniFufr8/wwjAO8xVZN60OrEJCCx2ri9kE9qTXGokCGo
kp3t1UXjgXGZ7WOCPI3Ba4syHqQoakLAqrqwfSqsXY7ut0lBPff/UfCrdNfiq43p
6bF6rSCd0FNdus1Gg1kynjBJny75V65rIk989xSyROLK+/xPBrQqSm9zZSBMLiBC
ZXJlbmd1ZWwgPHByb2Zlc29yam9zZUBnbWFpbc5jb20+iGAEExECACAFakYvJykC
GwMGCwkIBwMCBBUCCAMEFgIDAQIeAQIXgAAKCRA54q94aZqzjWYfAJ0XIy+4ILI3
OYt+I2rINkORmIc6DQCgv7LbBWuOxtwNPgkVbOjFGWlv7du5Ag0ERi8nMBAIALv9
5yIErQhJPpSb5a/HUXPnZ3Y38O77Kt6qs3hOD9cTSkltxKDYnQs0eoGrvMrgyoS/
4I4WfX81N5enJSEYyyBbYIf8/LS0vMuiWijTIYYGwWbNblaGVsh6OKhCjDE/95B
xjP+0BdUDj73TWNU4+NEF+LOSoiCt8wZPQRdyovs0ZmMRJ+PlzBBWkuVsXgXIJW
d0Iq+2OjHOZvEhgfsBa5blGoFv059VBhSZT1Q8soj6o0Sz0JxkiWhcvDUjRiFMw
0wb6VZXkVO0aCwUYLZeZl4LV87UY/tD+sepbQC51/u62yCAUx42qdR8bS3a83FX
```



```
zXR50t0E9Elks9jfNeMAAwUH/2pardNrCxPHeZsvOvB45fHZoH7ap57qmRRAWsCE
IvoCKtv3Rn2iceYfUag5YYBFtnoLyfAmiT6lhNdfqdZOWAFEa6e/IRpoA7PTnwh5
HbWZTRHRKBZ1O6WM3ZOr45NMAY2VFbRO8Kng/MrcdGYfNUq459fhIponl4VwByl
a
loaYCMvgsShglU4lq702222ZKBnqTe+00RGBnN/90ZAwqth5YCtoAv02WDaGFKNn
c/CBhKzNvaK5o8PzTmOQB5Tr0e2UHynQldVml3UIUtBA2Mn4JJbqh9DYFO4SE+ie
4kgI+MPaDCfUQ6vMfGkfl8vWMXar2HMYwzKkkIrYWYoJk7iISQQYEQIACQUCRi8n
MAIbDAAKCRA54q94aZqzjbRyAKCidWun3ccVSxmPj8oRQWet9VvRTQCgptqpxvO+
4XbDcCh0Rp0+xvjdkAE=
=L9iE
-----END PGP PUBLIC KEY BLOCK-----
```

El comando anterior muestra en pantalla nuestra clave pública en formato ASCII, si queremos exportar la clave directamente a un archivo, utilizamos alguno de los comandos siguientes:

```
$ gpg -a --export -o miclave.asc key_id
$ gpg -a --export --output miclave.asc key_id
$ gpg -a --export key_id > miclave.asc
```

donde *miclave.asc* es el nombre del archivo en el que se guardará la clave.

Cuando recibamos una clave pública de otra persona, ésta deberemos incluirla en nuestro **keyring** o anillo de claves, que es el lugar donde se almacenan todas las claves públicas de las que disponemos. Para incluir estas claves públicas de otras personas, lo haremos de la siguiente forma:

```
$ gpg --import clavepublica.asc
```

donde *clavepublica.asc* es el nombre del archivo recibido con la clave pública.

Una vez hemos importado la clave, podemos verificar el contenido de nuestro **keyring** con la opción *-kv*:

```
$ gpg -kv
```

Ejercicio 3 Exportar e importar claves públicas.

1. Exporta tu clave pública en formato ASCII y guárdalo en un archivo ***nombre_apellido.asc*** y envíalo a un compañero y al profesor.
2. Importa las claves públicas recibidas de vuestros compañeros.
3. Comprueba que las claves se han incluido correctamente en vuestro **keyring**.

5. Cifrado asimétrico con claves públicas

Tras realizar el ejercicio anterior, podemos enviar ya documentos cifrados utilizando la

clave pública de los destinatarios del mensaje. Por ejemplo, si queremos enviar un archivo cifrado a *Paco* y *Pepe*, escribiríamos lo siguiente:

```
$ gpg -a -r Paco -r Pepe --encrypt documento
```

La orden anterior crearía el archivo *documento.asc*, que es el que enviaríamos por correo a los destinatarios. Posteriormente, estos destinatarios tras recibir el archivo cifrado, podrán descifrarlo puesto que ha sido cifrado con su clave pública, y ellos dispondrán de la clave privada para poder descifrarlo. Con la orden siguiente podrán descifrar el archivo:

```
$ gpg documento.asc
```

Ejercicio 4 Cifrado y descifrado de un documento.

1. *Cifraremos un archivo cualquiera y lo remitiremos por email a uno de nuestros compañeros que nos proporcionó su clave pública.*
2. *Nuestro compañero, a su vez, nos remitirá un archivo cifrado para que nosotros lo descifremos.*
3. *Tanto nosotros como nuestro compañero comprobaremos que hemos podido descifrar los mensajes recibidos respectivamente.*
4. *Por último, enviaremos el documento cifrado a alguien que no estaba en la lista de destinatarios y comprobaremos que este usuario no podrá descifrar este archivo.*

6. Firma digital de un documento

Con la firma de un documento, nos aseguramos de que los destinatarios de éste, no tengan duda de que el autor del mensaje es quien dice ser (**autenticidad**), y de que el documento no ha sido modificado por nadie (**integridad**). Para firmar un documento lo haremos de la siguiente forma:

```
$ gpg -sb -a documentoafirmar
```

El programa nos pedirá la contraseña de nuestra clave privada puesto que ésta es necesaria para firmar el documento, y el resultado será un archivo *documentoafirmar.asc* que contiene la firma digital. Para verificar que la firma es correcta, debemos poseer tanto el documento original como el archivo de firma, y bastará con ejecutar **gpg** sobre el archivo de firma:

```
$ gpg documentoafirmar.asc
```

Ejercicio 5 Firma digital de un documento.

1. *Crea la firma digital de un archivo de texto cualquiera y envíale éste junto al documento con la firma a un compañero.*
2. *Verifica que la firma recibida del documento es correcta.*
3. *Modifica el archivo ligeramente, insertando un carácter o un espacio en blanco, y vuelve a comprobar si la firma se verifica.*