Guía de “Gnu Privacy Guard”

Guía de “Gnu Privacy Guard”

Para cualquier duda, error, o sugerencia sobre este manual, diríjase al autor del mismo, Mike Ashley (<`jashley@acm.org`>). Para cualquier duda, corrección, o sugerencia sobre la versión en castellano, diríjase al traductor, Horacio (<`homega@ciberia.es`>). También han contribuido a la creación del manual Matthew Copeland, Joergen Grahn y David A. Wheeler.

Tabla de contenidos

1. Primeros Pasos 1
- Generar un nuevo par de claves 1
  - Generar un certificado de revocación 3
- Intercambiar claves 4
  - Exportar una clave pública 4
  - Importar una clave pública 4
- Cifrar y descifrar documentos 6
- Firmar y verificar firmas 7
  - Documentos con firmas ASCII 7
  - Firmas acompañantes 8

2. Conceptos 10
- Sistemas de cifrado simétrico 10
- Sistemas de cifrado asimétrico 11
- Sistemas de cifrado híbridos 12
- Firmas digitales 12

3. Gestión de Claves 14
- Gestionando nuestro par de claves 14
  - Integridad de claves 15
  - Añadir y eliminar componentes a las claves 16
  - Revocar componentes de una clave 17
  - Actualizar la fecha de caducidad de una clave 18
- Validar otras claves en nuestro anillo de claves públicas 19
  - Confianza en el propietario de una clave 19
  - Usar la confianza para validar las claves 21
- Distribución de claves 22

4. Uso diario de GnuPG 25
- Definiendo los requerimientos en seguridad 25
  - Selección del tamaño de la clave 25
  - Protección de la clave privada 26
  - Selección de las fechas de caducidad y uso de subclaves 27
  - Gestión del anillo de confianza 28
- Construcción del anillo de confianza 28
- Uso legal de GnuPG 29

5. Tópicos 31
- Escribir interfaces de usuario 31

Lista de figuras

3-1. Un anillo de confianza hipotético....................22

Capítulo 1. Primeros Pasos

GnuPG es una herramienta de seguridad en comunicaciones electrónicas. Este capítulo es una guía rápida que cubre las funciones básicas de GnuPG. Estas funciones incluyen generar un par de claves, intercambiar y comprobar la autenticidad de claves, cifrar y descifrar documentos, y firmar documentos y verificar firmas digitales. En este capítulo no se detallan los conceptos de la criptografía de clave pública, cifrado, y firmas digitales. Todo esto se cubrirá en detalle en el Capítulo 2. Tampoco se explica el uso avanzado the GnuPG. Esto se explica en los Capítulos 3 y 4.

GnuPG utiliza criptografía de clave pública para que los usuarios puedan comunicarse de un modo seguro. En un sistema de claves públicas cada usuario posee un par de claves, compuesto por una *clave privada* y una *clave pública*. Cada usuario debe mantener su clave privada secreta; no debe ser revelada nunca. La clave pública se puede entregar a cualquier persona con la que el usuario desee comunicarse. GnuPG implementa un esquema algo más sofisticado en el que un usuario tiene un par de claves primario, y ninguno o más de un par de claves adicionales subordinadas[1]. Los pares de claves primarios y subordinados se encuentran agrupados para facilitar la gestión de claves, y el grupo puede ser considerado como un sólo par de claves.

Generar un nuevo par de claves

La opción de la línea de órdenes `--gen-key` se usa para generar un nuevo par de claves primario.

```
javier:~$ gpg --gen-key
gpg (GnuPG) 0.9.8; Copyright (C) 1999 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.

Please select what kind of key you want:
   (1) DSA and ElGamal (default)
   (2) DSA (sign only)
   (4) ElGamal (sign and encrypt)
Your selection?
```

GnuPG es capaz de crear varios tipos diferentes de pares de claves, pero debe existir una clave primaria capaz de generar firmas. Por lo tanto, existen sólo tres opciones. La opción 1 genera dos pares de claves. Un par de claves DSA que es el par de claves primario que se usará sólo para firmar. Un par de claves subordinadas ElGamal que se usará para el cifrado. La opción 2 es parecida a la anterior, pero sólo genera un par de claves DSA. La opción 4[2] genera un único par de claves ElGamal, que se usará tanto para firmar como para cifrar. En todos los casos existe la posibilidad de añadir subclaves adicionales para cifrar y firmar «a posteriori». La mayoría de los usuarios tienen suficiente con la opción por definición.

También hay que escoger un tamaño para la clave. El tamaño de una clave DSA debe estar entre los 512 y 1024 bits, y una clave ElGamal puede ser de cualquier tamaño. Sin embargo, GnuPG requiere que las claves no sean menores de 768 bits. Por tanto, si se escogió la opción 1 y también un tamaño de claves mayor de 1024 bits, la clave ElGamal tendrá el tamaño deseado pero la DSA se limitará a 1024 bits.

```
DSA keypair will have 1024 bits.
About to generate a new ELG-E keypair.
              minimum keysize is  768 bits
      default keysize is 1024 bits
    highest suggested keysize is 2048 bits
```

```
What keysize do you want? (1024)
```

Cuanto más larga sea la clave, más segura será contra ataques de «fuerza bruta», pero por lo demás el tamaño de la clave que se da por definición es el adecuado, ya que sería más barato circunvalar el cifrado que intentar entrar mediante ataques de fuerza. Además, el cifrado y descifrado de mensajes se ralentizaría a medida que se incrementara el tamaño de la clave, y un tamaño de clave más grande podría afectar a la longitud de la firma digital. Una vez seleccionado, el tamaño de una clave no se puede cambiar nunca.

Para terminar, hay que escoger un fecha de caducidad. Si se escogió anteriormente la opción 1, la fecha de caducidad se usará para sendos pares de claves, ElGamal y DSA.

```
Requested keysize is 1024 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0)

Key does not expire at all
Is this correct (y/n)?
```

Para la mayoría de los usuarios, una clave sin fecha de caducidad es la adecuada. Sin embargo, si se escoge con fecha de caducidad, el tiempo para ésta debe ser escogido con cuidado, ya que, aunque es posible cambiar la fecha de caducidad posteriormente a la generación de la clave, puede ser difícil comunicar un cambio a aquellos usuarios que posean esta clave pública.

Además de los parámetros de la clave, el usuario debe dar un identificador. El identificador de usuario se usa para asociar la clave que se está creando con una usuario real.

```
You need a User-ID to identify your key; the software constructs the user id
from Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name:
Email address:
Comment:
```

Sólo se creará un identificador de usuario al generar una clave, pero es posible crear identificadores adicionales si se desea usar la clave en dos o más contextos, v.g., si se usa por una parte en la oficina como empleado y por otra parte en casa como activista político. Hay que tener cuidado al crear un identificador de usuario, ya que después éste no puede ser editado para introducir cambios.

Aunque los caracteres especiales en iso-8859-1 son aceptados, GnuPG nos avisa si los usamos para rellenar estos campos[3]. Por ejemplo, si rellenáramos los campos con los siguientes datos,

- Real name: Javier
- Email address: javier@mad.es
- Comment: Páramo S.L.

Veríamos lo siguiente: "Javier (P\xc3\xa1ramo S.L.) <javier@mad.es>". Por tanto es mejor evitar estos carácteres.

```
You are using the 'iso-8859-1' character set.
```

```
You selected this USER-ID:
    "Javier (Paramo S.L.) <javier@casa.es>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit?
```

Aún así, dependiendo de la versión que estemos usando, al listar las claves veremos una serie de carácteres extraños en lugar de *vocales acentuadas, ñ, ç,* etc...

GnuPG necesita una contraseña con el fin de proteger las claves privadas, primarias y secundarias, que posea el usuario[4].

```
You need a Passphrase to protect your private key.

Enter passphrase:
```

No hay límite para la longitud de una contraseña, y ésta debe ser escogida con sumo cuidado. Desde un punto de vista de seguridad, la contraseña que desbloquea la clave privada es uno de los puntos más débiles en GnuPG (y en otros sistemas de cifrado de clave pública), ya que es la única protección que tiene el usuario si alguien se apoderara de su clave privada. Para una contraseña lo ideal es que no se usen palabras de un diccionario, y que se mezclen mayúsculas y minúsculas, dígitos, y otros caracteres. Una buena contraseña es crucial para el uso seguro de GnuPG.

```
Repeat passphrase:
```

Como antes con los campos de identificación del usuario, las contraseñas aceptan caracteres especiales de iso-8859-1. No obstante, debemos tener en cuenta que si alguna vez tuviéramos que usar nuestra contraseña desde una máquina con un teclado distinto al nuestro, nos veríamos imposibilitados a menos que cambiáramos la configuración del sistema.

Generar un certificado de revocación

Después de haber generado un par de claves, el usuario debe, de forma inmediata, generar un certificado de revocación para la clave pública primaria, mediante el uso de la opción `--gen-revoke`. Si el usuario olvidara la contraseña, o si su clave privada estuviera en peligro o extraviada, este certificado de revocación podría ser hecho público para notificar a otros usuarios que la clave pública no debe ser usada nunca más. Una clave pública revocada puede ser usada para verificar firmas hechas por el usuario en el pasado, pero no puede ser usada para cifrar datos. Esto tampoco afecta a la capacidad de descifrar mensajes que hayan sido cifrados con la clave antes de su revocación, siempre y cuando el usuario todavía tenga acceso a la clave privada.

```
javier:~$ gpg --output D58711B7.asc --gen-revoke 0xD58711B7

sec  1024D/D58711B7 1999-09-24   Javier (Paramo S.L.) <javier@casa.es>
```

El argumento **`miclave`** debe ser un *especificador de clave*, ya sea éste el identificador de clave ("key ID") del par primario del usuario, o ya sea cualquier otra parte de un identificador de usuario ("user ID") que identifique el par de claves del susodicho usuario. El certificado que se genere se encontrará en el fichero *`revoke.asc`*. Si se omite la opción `--output`, el resultado se pondrá en la salida típica. Dado que el certificado es corto, es posible que el usuario desee imprimir una copia en papel del certificado para guardarla en algún sitio seguro, como por ejemplo una caja fuerte de seguridad. El certificado no debería ser guardado en lugares a los que otros puedan tener acceso, ya que cualquiera podría hacer público el certificado de revocación e inutilizar la correspondiente clave pública.

Intercambiar claves

Para poder comunicarse con otros, el usuario debe intercambiar las claves públicas. Para obtener una lista de las claves en el fichero («anillo») de claves públicas, se puede usar la opción de la línea de órdenes `--list-keys`.

```
javier:~$ gpg --list-keys

/home/javier/.gnupg/pubring.gpg
--------------------------------
pub  1024D/D58711B7 1999-09-24 Javier (Paramo S.L.) <javier@casa.es>
sub  1024g/92F6C9E3 1999-09-24
```

Exportar una clave pública

Para poder enviar una clave pública a un interlocutor, antes hay que exportarla. Para ello se usará la opción de la línea de órdenes `--export`. Es necesario un argumento adicional para poder identificar la clave pública que se va a exportar. Como en la opción anterior `--gen-revoke`, hay que usar el identificador de clave o cualquier parte del identificador de usuario para identificar la clave que se desea exportar.

```
javier:~$ gpg --output javi.gpg --export javier@casa.es
```

La clave se exporta en formato binario, y esto puede no ser conveniente cuando se envía la clave por correo electrónico o se publica en una página web. Por tanto, GnuPG ofrece una opción de la línea de órdenes `--armor`[5] que fuerza que la salida de la orden sea generada en formato armadura-ASCII, parecido a los documentos codificados con uuencode. Por regla general, cualquier salida de una orden de GnuPG, v.g.. claves, documentos cifrados y firmas, pueden ir en formato armadura-ASCII añadiendo a la orden la opción `--armor`.

```
javier:~$ gpg --armor --output javi.asc --export javier@casa.es
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v0.9.8 (GNU/Linux)
Comment: For info see http://www.gnupg.org

[...]
-----END PGP PUBLIC KEY BLOCK-----
```

Importar una clave pública

Se puede añadir una clave pública al anillo de claves públicas mediante la opción `--import`.

```
javier:~$ gpg --import arancha.gpg
gpg: key B63E132C: public key imported
gpg: Total number processed: 1
gpg:               imported: 1

javier:~$ gpg --list-keys
/home/javier/.gnupg/pubring.gpg
--------------------------------
pub  1024D/D58711B7 1999-09-24 Javier (Paramo S.L.) <javier@casa.es>
sub  1024g/92F6C9E3 1999-09-24

pub  1024D/B63E132C 1999-09-24 Aranzazu (A.G.deZ.) <arancha@nav.es>
```

```
sub  1024g/581A915F 1999-09-24
```

Una vez que la clave haya sido importada, es necesario validarla. GnuPG usa un potente y flexible modelo de confianza que no requiere que el usuario dé validez personalmente a cada clave que importe. Sin embargo, algunas claves pueden necesitar que el usuario les dé validez de forma personal. Una clave se valida verificando la huella digital de la clave, y firmando dicha clave para certificar su validez. La huella digital se puede ver con la opción de la línea de órdenes `--fingerprint`, pero para certificar la clave hay que editarla.

```
javier:~$ gpg --edit-key arancha@nav.es

pub  1024D/B63E132C  created: 1999-09-24 expires: never      trust: -/q
sub  1024g/581A915F  created: 1999-09-24 expires: never
(1)  Aranzazu (A.G.deZ.) <arancha@nav.es>

Command> fpr
pub  1024D/B63E132C 1999-09-24 Aranzazu (A.G.deZ.) <arancha@nav.es>
             Fingerprint: 4203 82E2 448C BD30 A36A  9644 0612 8A0F B63E 132C
```

La huella digital de una clave se verifica con el propietario de la clave. Esto puede hacerse en persona o por teléfono, o por medio de otras maneras, siempre y cuando el usuario pueda garantizar que la persona con la que se está comunicando sea el auténtico propietario de la clave. Si la huella digital que se obtiene por medio del propietario es la misma que la que se obtiene de la clave, entonces se puede estar seguro de que se está en posesión de una copia correcta de la clave.

Después de comprobar la huella digital ya se puede firmar la clave con el fin de validarla. Debido a que la verificación es un punto débil en criptografía de clave pública, es aconsejable ser cuidadoso en extremo y *siempre* comprobar la huella digital de una clave con la que nos dé el propietario antes de firmar dicha clave.

```
Command> sign

pub  1024D/B63E132C  created: 1999-09-24 expires: never      trust: -/q
             Fingerprint: 4203 82E2 448C BD30 A36A  9644 0612 8A0F B63E 132C

     Aranzazu (A.G.deZ.) <arancha@nav.es>

Are you really sure that you want to sign this key
with your key: "Javier (Paramo S.L.) <javier@casa.es>"

Really sign? y

You need a passphrase to unlock the secret key for
user: "Javier (Paramo S.L.) <javier@casa.es>"
1024-bit DSA key, ID D58711B7, created 1999-09-24

Enter passphrase:
```

Una vez firmada, el usuario puede comprobar la clave para obtener un listado de las firmas que lleva y para ver la firma que le acaba de añadir. Cada identificador de usuario tendrá una o más autofirmas, así como una firma por cada usuario que haya validado la clave en cuestión.

```
Command> check

uid  Aranzazu (A.G.deZ.) <arancha@nav.es>
sig!       B63E132C 1999-09-24   [self-signature]
sig!       D58711B7 1999-09-24   Javier (Paramo S.L.) <javier@casa.es>
```

```
Command> quit
```

Cifrar y descifrar documentos

Cada clave pública y privada tiene un papel específico en el cifrado y descifrado de documentos. Se puede pensar en una clave pública como en una caja fuerte de seguridad. Cuando un remitente cifra un documento usando una clave pública, ese documento se pone en la caja fuerte, la caja se cierra, y el bloqueo de la combinación de ésta se gira varias veces. La parte correspondiente a la clave privada, esto es, el destinatario, es la combinación que puede volver a abrir la caja y retirar el documento. Dicho de otro modo, sólo la persona que posee la clave privada puede recuperar un documento cifrado usando la clave pública asociada al cifrado.

Con este modelo mental se ha mostrado el procedimiento de cifrar y descifrar documentos de un modo muy simple. Si el usuario quisiera cifrar un mensaje para Javier, lo haría usando la clave pública de Javier, y él lo descifraría con su propia clave privada. Si Javier quisiera enviar un mensaje al usuario, lo haría con la clave pública del usuario, y éste lo descifraría con su propia clave privada.

Para cifrar un documento se usa la opción `--encrypt`. El usuario debe tener las claves públicas de los pretendidos destinatarios. El programa espera recibir como entrada el nombre del documento que se desea cifrar o, si éste se omite, una entrada típica. El resultado cifrado se coloca en la salida típica o donde se haya especificado mediante la opción `--output`. El documento se comprime como medida adicional de seguridad, aparte de cifrarlo.

```
javier:~$ gpg --output doc.gpg --encrypt --recipient arancha@nav.es doc
```

La opción `--recipient` se usa una vez para cada destinatario, y lleva un argumento extra que especifica la clave pública con la que será cifrado el documento. El documento cifrado sólo puede ser descifrado por alguien con una clave privada que complemente uno de las claves públicas de los destinatarios. El usuario, en este caso el remitente, no podrá descifrar un documento cifrado por sí mismo a menos que haya incluido su propia clave pública en la lista de destinatarios.

Para descifrar un mensaje se usa la opción `--decrypt`. Para ello es necesario poseer la clave privada para la que el mensaje ha sido cifrado. De igual modo que en el proceso de cifrado, el documento a descifrar es la entrada, y el resultado descifrado la salida.

```
arancha% gpg --output doc --decrypt doc.gpg

You need a passphrase to unlock the secret key for
user: "Aranzazu (A.G.deZ.) <arancha@nav.es>"
1024-bit ELG-E key, ID 581A915F, created 1999-09-24 (main key ID B63E132C)

Enter passphrase:
```

También es posible cifrar documentos sin usar criptografía de clave pública. En su lugar, se puede usar sólo una clave de cifrado simétrico para cifrar el documento. La clave que se usa para el cifrado simétrico deriva de la contraseña dada en el momento de cifrar el documento, y por razones de seguridad, no debe ser la misma contraseña que se esté usando para proteger la clave privada. El cifrado simétrico es útil para asegurar documentos cuando no sea necesario dar la contraseña a otros. Un documento puede ser cifrado con una clave simétrica usando la opción `--symmetric`.

```
javier:~$ gpg --output doc.gpg --symmetric doc
Enter passphrase:
```

Firmar y verificar firmas

Una firma digital certifica un documento y le añade una marca de tiempo. Si posteriormente el documento fuera modificado en cualquier modo, el intento de verificar la firma fallaría. La utilidad de una firma digital es la misma que la de una firma escrita a mano, sólo que la digital tiene una resistencia a la falsificación. Por ejemplo, la distribución del código fuente de GnuPG viene firmada con el fin de que los usuarios puedan verificar que no ha habido ninguna manipulación o modificación al código fuente desde que fue archivado.

Para la creación y verificación de firmas, se utiliza el par público y privado de claves en una operación que es diferente a la de cifrado y descifrado. Se genera una firma con la clave privada del firmante. La firma se verifica por medio de la clave pública correspondiente. Por ejemplo, Javier haría uso de su propia clave privada para firmar digitalmente la entrega de su última ponencia a la Revista de Química Inorgánica. El editor asociado que la recibiera, usaría la clave pública de Javier para comprobar la firma, verificando de este modo que el envío proviene realmente de Javier, y que no ha sido modificado desde el momento en que Javier lo firmó. Una consecuencia directa del uso de firmas digitales es la dificultad en negar que fue el propio usuario quien puso la firma digital, ya que ello implicaría que su clave privada ha sido puesta en peligro.

La opción de línea de órdenes `--sign` se usa para generar una firma digital. El documento que se desea firmar es la entrada, y la salida es el documento firmado.

```
javier:~$ gpg --output doc.sig --sign doc

You need a passphrase to unlock the private key for
user: "Javier (Paramo S.L.) <javier@casa.es>"
1024-bit DSA key, ID D58711B7, created 1999-09-24

Enter passphrase:
```

El documento se comprime antes de ser firmado, y la salida es en formato binario.

Con un documento con firma digital el usuario puede llevar a cabo dos acciones: comprobar sólo la firma o comprobar la firma y recuperar el documento original al mismo tiempo. Para comprobar la firma se usa la opción `--verify`. Para verificar la firma y extraer el documento se usa la opción `--decrypt`. El documento con la firma es la entrada, y el documento original recuperado es la salida.

```
arancha% gpg --output doc --decrypt doc.sig
gpg: Signature made Fri Sep  24 12:02:38 1999 CDT using DSA key ID D58711B7
gpg: Good signature from "Javier (Paramo S.L.) <javier@casa.es>"
```

Documentos con firmas ASCII

Las firmas digitales suelen usarse a menudo para firmar mensajes de correo electrónicos o en los grupos de noticias. En estas situaciones no se debe comprimir el documento al firmarlo, ya que para aquellos que no dispongan de un sistema para procesarlo sería ininteligible.

```
javier:~$ gpg --clearsign doc
```

```
You need a passphrase to unlock the secret key for
user: "Javier (Paramo S.L.) <javier@casa.es>"
1024-bit DSA key, ID D58711B7, created 1999-09-24

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

[...]
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v0.9.8 (GNU/Linux)
Comment: For info see http://www.gnupg.org

iEYEARECAAYFAjdYCQoACgkQJ9S6ULt1dqz6IwCfQ7wP6i/i8HhbcOSKF4ELyQB1
oCoAoOuqpRqEzr4kOkQqHRLE/b8/Rw2k
=y6kj
-----END PGP SIGNATURE-----
```

Firmas acompañantes

Un documento firmado tiene una utilidad limitada. Los otros usuarios deben recuperar la versión original del documento de la versión firmada, y aun en el caso de los documento firmados en ASCII, el documento firmado debe ser editado para poder recuperar el original. Por tanto, existe un tercer método para firmar un documento, que genera una firma acompañante. Para generar una firma acompañante se usa la opción `--detach-sig`.

```
javier:~$ gpg --output doc.sig --detach-sig doc

You need a passphrase to unlock the secret key for
user: "Javier (Paramo S.L.) <javier@casa.es>"
1024-bit DSA key, ID D58711B7, created 1999-09-24

Enter passphrase:
```

Tanto el documento como la firma acompañante son necesarios para poder verificar la firma. La opción `--verify` se usará para comprobar la firma.

```
arancha% gpg --verify doc.sig doc
gpg: Signature made Fri Sep  24 12:38:46 1999 CEST using DSA key ID D58711B7
gpg: Good signature from "Javier (Paramo S.L.) <javier@casa.es>"
```

Notas

1. N. de T. En este y otros documentos se intercambia el término «claves subordinadas» con el de «subclaves»
2. La opción 3 es para generar un par de claves ElGamal que no puede ser usado para firmar.
3. Se aceptan excepto en la dirección de correo electrónico. En el resto pueden dar problemas, así que se recomienda no usarlos.
4. N. de T. Esta contraseña adopta la forma de una «frase» ("passphrase"), no de una sola «palabra» ("password").

5. Muchas opciones de la línea de órdenes que se usan con frecuencia, también se pueden configurar en un fichero de configuración.

Capítulo 2. Conceptos

GnuPG utiliza varios conceptos criptológicos que incluyen *sistemas de cifrado simétrico*, *sistemas de cifrado de clave pública*, y *‘one-way hashing’*. Se puede usar GnuPG de un modo sencillo sin tener que entender estos conceptos en toda su amplitud, pero para el uso avanzado de GnuPG se hace necesario comprenderlos un poco.

Este capítulo es una introducción a los conceptos básicos de la criptografía usada en GnuPG. Para una lectura más detallada sobre esta materia, existen otros libros. Un buen libro para alcanzar un conocimiento más avanzado es Bruce Schneier (http://www.counterpane.com/schneier.html)’s “Applied Cryptography” (http://www.counterpane.com/applied.html).

Sistemas de cifrado simétrico

Un sistema de cifrado simétrico es un tipo de cifrado que usa una misma clave para cifrar y para descifrar. Las dos partes que se comunican mediante el cifrado simétrico deben estar de acuerdo en la clave a usar de antemano. Una vez de acuerdo, el remitente cifra un mensaje usando la clave, lo envía al destinatario, y éste lo descifra usando la misma clave. Como ejemplo de sistema simétrico está «Enigma»; Éste es un sistema que fue usado por Alemania, en el que las claves se distribuían a diario en forma de libros de códigos. Cada día, un operador de radio, receptor o transmisor, consultaba su copia del libro de códigos para encontrar la clave del día. Todo el tráfico enviado por ondas de radio durante aquel día era cifrado y descifrado usando las claves del día. Algunos ejemplos actuales de algoritmos simétricos son 3DES, Blowfish e IDEA.

Un buen sistema de cifrado pone toda la seguridad en la clave y ninguna en el algoritmo. En otras palabras, no debería ser de ninguna ayuda para un atacante conocer el algoritmo que se está usando. Sólo si el atacante obtuviera la clave, le serviría conocer el algoritmo. Los algoritmos de cifrado usados en GnuPG tienen estas propiedades.

Dado que toda la seguridad está en la clave, es importante que sea muy difícil adivinar el tipo de clave. Esto quiere decir que el abanico de claves posibles, o sea, el *espacio de posibilidades de claves*, debe ser amplio. Richard Feynman fue famoso en Los Álamos por su habilidad para abrir cajas de seguridad. Para alimentar la leyenda que había en torno a él, llevaba encima un juego de herramientas que incluían un estetoscopio. En realidad, utilizaba una gran variedad de trucos para reducir a un pequeño número la cantidad de combinaciones que debía probar, y a partir de ahí simplemente probaba hasta que adivinaba la combinación correcta. En otras palabras, reducía el tamaño de posibilidades de claves.

Inglaterra usó máquinas para adivinar las claves durante la Segunda Guerra Mundial. El sistema alemán Enigma estaba provisto de un amplio abanico de claves, pero los ingleses diseñaron máquinas de cómputo especializado, los Bombes, para probar las claves de un modo mecánico hasta que la clave del día era encontrada. Esto significaba que algunas veces encontraban la clave del día unas pocas horas después de que ésta fuera puesta en uso, pero también que otros días no podían encontrar la clave correcta. Los Bombes no fueron máquinas de cómputo general, sino los precursores de las computadoras (ordenadores) de hoy en día.

Hoy por hoy, los ordenadores pueden adivinar claves con extrema rapidez, y ésta es la razón por la cual el tamaño de la clave es importante en los «criptosistemas» modernos. El algoritmo de cifrado DES usa una clave de 56 bits, lo que significa que hay 2^{56} claves posibles. 2^{56} son 72.057.594.037.927.936 claves. Esto representa un número muy alto de claves, pero una máquina computadora de uso general puede comprobar todo el espacio posible de claves en cuestión de días. Un máquina especializada lo puede hacer en horas. Por otra parte, algoritmos de cifrado de diseño más reciente como 3DES, Blowfish e IDEA usan todos claves de 128 bits, lo que significa que existen 2^{128} claves posibles. Esto representa muchas, muchísimas más claves, y aun en el caso de que todas las máquinas del planeta estuvieran cooperando, todavía tardarían más tiempo que la misma edad del universo en encontrar la clave.

Sistemas de cifrado asimétrico

El principal problema con los sistemas de cifrado simétrico no está ligado a su seguridad, sino al intercambio de claves. Una vez que el remitente y el destinatario hayan intercambiado las claves pueden usarlas para comunicarse con seguridad, pero ¿qué canal de comunicación que sea seguro han usado para «comunicar» la clave entre ellos? Sería mucho más fácil para un atacante intentar interceptar una clave que probar las posibles combinaciones del espacio de claves. Otro problema es el número de claves que se necesitan. Si tenemos un número *n* de personas que necesitan comunicarse entre ellos, entonces se necesitan *n(n-1)/2* claves para cada pareja de personas que tengan que comunicarse de modo privado. Esto puede funcionar con un grupo reducido de personas, pero sería imposible llevarlo a cabo con grupos más grandes.

Los sistemas de cifrado de clave pública se inventaron con el fin de evitar por completo el problema del intercambio de claves. Un sistema de cifrado de clave pública usa un par de claves para el envío de mensajes. Las dos claves pertenecen a la misma persona a la que se ha enviado el mensaje. Una clave es *pública* y se puede entregar a cualquier persona. La otra clave es *privada* y el propietario debe guardarla para que nadie tenga acceso a ella. El remitente usa la clave pública del destinatario para cifrar el mensaje, y una vez cifrado, sólo la clave privada del destinatario podrá descifrar este mensaje.

Este protocolo resuelve el problema del intercambio de claves, que es inherente a los sistemas de cifrado simétricos. No hay necesidad de que el remitente y el destinatario tengan que ponerse de acuerdo en una clave. Todo lo que se requiere es que, antes de iniciar la comunicación secreta, el remitente consiga una copia de la clave pública del destinatario. Es más, esa misma clave pública puede ser usada por cualquiera que desee comunicarse con su propietario. Por tanto, se necesitarán sólo *n* pares de claves por cada *n* personas que deseen comunicarse entre ellas.

Los sistemas de cifrado de clave pública se basan en funciones-trampa de un sólo sentido. Una función de un sólo sentido es aquélla cuya computación es fácil, mientras que invertir la función es extremadamente difícil. Por ejemplo, es fácil multiplicar dos múmeros primos juntos para sacar uno compuesto, pero es difícil factorizar uno compuesto en sus componentes primos. Una función-trampa de un sentido es algo parecido, pero tiene una trampa. Esto quiere decir que si se conociera alguna pieza de la información, sería fácil computar el inverso. Por ejemplo, si tenemos un número compuesto por dos factores primarios y conocemos uno de los factores, es fácil computar el segundo. Dado un cifrado de clave pública basado en factorización de números primos, la clave pública contiene un número compuesto de dos factores primos grandes, y el algoritmo de cifrado usa ese compuesto para cifrar el mensaje. El algoritmo para descifrar el mensaje requiere el conocimiento de los factores primos, para que el descifrado sea fácil si poseemos la clave privada que contiene uno de los factores, pero extremadamente difícil en caso contrario.

Como con los sistemas de cifrado simétricos buenos, con un buen sistema de cifrado de clave pública toda la seguridad descansa en la clave. Por lo tanto el tamaño de la clave es una medida del seguridad del sistema, pero no se puede comparar el tamaño del cifrado simétrico con el de un cifrado de clave pública para medir la seguridad. En un ataque de fuerza bruta sobre un cifrado simétrico con una clave de un tamaño de 80 bits, el atacante debe enumerar hasta 2^{81}-1 claves para encontrar la clave correcta. En un ataque de fuerza bruta sobre un cifrado de clave pública con un clave de un tamaño de 512 bits, el atacante debe factorizar un número compuesto codificado en 512 bits (hasta 155 dígitos decimales). La cantidad de trabajo para el atacante será diferente dependiendo del cifrado que esté atacando. Mientras 128 bits es suficiente para cifrados simétricos, dada la tecnología de factorización de hoy en día, se recomienda el uso de claves públicas de 1024 bits para la mayoría de los casos.

Sistemas de cifrado híbridos

Los cifrados de clave pública no son ninguna panacea. Muchos cifrados simétricos son más fuertes desde un punto de vista de seguridad, y el cifrado y descifrado con clave pública son más caros que las correspondientes operaciones en sistemas simétricos. De todos modos, los cifrados de clave pública son una herramienta efectiva para distribuir claves de cifrado simétrico, y de esta manera es como se usan en sistemas de cifrado híbridos.

Un sistema de cifrado híbrido usa tanto un cifrado simétrico como uno asimétrico. Funciona mediante el uso de un cifrado de clave pública para compartir una clave para el cifrado simétrico. El mensaje que se esté enviando en el momento, se cifra usando la clave y enviándolo al destinatario. Ya que compartir una clave simétrica no es seguro, la clave usada es diferente para cada sesión.

Tanto PGP como GnuPG usan sistemas de cifrado híbridos. La clave de sesión es cifrada con la clave pública, y el mensaje saliente es cifrado con la clave simétrica, todo combinado automáticamente en un sólo paquete. El destinatario usa su clave privada para descifrar la clave de sesión y acto seguido usa la clave de sesión para descifrar el mensaje.

Un sistema de cifrado híbrido no es más fuerte que el de cifrado asimétrico o el de cifrado simétrico de los que hace uso, cualquiera que de los dos que sea el más débil. En PGP y GnuPG el sistema de clave pública es probablemente la parte más débil de la combinación. Sin embargo, si un atacante pudiera descifrar un clave de sesión, sólo sería útil para poder leer un mensaje, el cifrado con esa clave de sesión. El atacante tendría que volver a empezar y descifrar otra clave de sesión para poder leer cualquier otro mensaje.

Firmas digitales

Una función 'hash' es una función múltiple que asigna su entrada a un valor dentro de un grupo finito. Por regla general este grupo es un rango de números naturales. Un modelo simple de función 'hash' es $f(x) = 0$ para todo entero x. Una función hash más interesante es $f(x) = x \; mod \; 37$, que asigna x al resto de la división x entre 37.

Una firma digital en un documento es el resultado de aplicar una función 'hash' al documento. Para que sea de utilidad, la función 'hash' necesita satisfacer dos propiedades importantes. Primero, debería ser difícil encontrar dos documentos cuyo valor para una función 'hash' sea el mismo. Segundo, dado un valor 'hash' debería ser difícil de recuperar el documento que produjo es valor.

Algunos sistemas de cifrado de clave pública[1] se pueden usar para firmar documentos. El firmante cifra el documento con su clave *privada*. Cualquiera que quiera comprobar la firma y ver el documento, no tiene más que usar la clave pública del firmante para descifrarla. Este algoritmo satisface las dos propiedades necesarias para una buena función de ‘hash’, pero en la práctica este algoritmo es demasiado lento para que sea de utilidad.

Como alternativa está el uso de funciones ‘hash’ designadas para satisfacer estas dos importantes propiedades. SHA y MD5 son dos ejemplos de este tipo de algoritmos. Al usar uno de estos algoritmos, un documento se firma con una función ‘hash’, y el valor del ‘hash’ es la firma. Otra persona puede comprobar la firma aplicando también una función ‘hash’ a su copia del documento y comparando el valor ‘hash’ resultante con el del documento original. Si concuerdan, es casi seguro que los documentos son idénticos.

Claro que el problema está en usar una función ‘hash’ para firmas digitales que no permita que un atacante interfiera en la comprobación de la firma. Si el documento y la firma se enviaran descifrados, un atacante podría modificar el documento y generar una firma correspondiente sin que lo supiera el destinatario. Si sólo se cifrara el documento, un atacante podría manipular la firma y hacer que la comprobación de ésta fallara. Una tercera opción es usar un sistema de clave pública híbrido para cifrar tanto la firma como el documento. El firmante usa su clave pública, y cualquiera puede usar su clave pública para comprobar la firma y el documento. Esto suena bien, pero en realidad no tiene sentido. Si este algoritmo hiciera el documento seguro también lo aseguraría de manipulaciones, y no habría necesidad de firmarlo. El problema más serio es que esto no protege de manipulaciones ni a la firma, ni al documento. Con este algoritmo, sólo la clave de sesión del sistema de cifrado simétrico, es cifrada usando la clave privada del firmante. Cualquiera puede usar la clave pública y recuperar la clave de sesión. Por lo tanto, es sencillo para un atacante recuperar la clave de sesión y usarla para cifrar documentos substitutos y firmas para enviarlas a terceros en nombre del remitente.

Un algoritmo que funciona es aquél que hace uso de un algoritmo de clave pública para cifrar sólo la firma. En particular, el valor ‘hash’ se cifra mediante el uso de la clave privada del firmante, de modo que cualquiera pueda comprobar la firma usando la clave pública correspondiente. El documento firmado se puede enviar usando cualquier otro algoritmo de cifrado, o incluso ninguno si es un documento público. Si el documento se modifica, la comprobación de la firma fallará, pero esto es precisamente lo que la verificación se supone que debe descubrir. El "Digital Signature Standard" (DSA) es un algoritmo de firmado de clave pública que funciona como hemos descrito. DSA es el algoritmo principal de firmado que se usa en GnuPG.

Notas

1. El sistema de cifrado debe poseer la propiedad de que la clave pública o la privada puedan ser usadas por el algoritmo de cifrado como clave pública. RSA en un ejemplo de este tipo de algoritmo, mientras que ElGamal no.

Capítulo 3. Gestión de Claves

En criptografía de clave pública la manipulación de claves es un punto flaco a tener en cuenta. Un «escucha» podría manipular los ficheros con las claves, o falsificar la clave pública y hacerla pública para que la usaran otros como si fuera la auténtica. Por ejemplo, supongamos que Jordi quisiera monitorizar los mensajes que Javier envía a Arancha. Jordi podría montar lo que se conoce como un *intermediario* en el plan de ataque. En este ataque Jordi crea una nuevo par de claves pública y privada, reemplazando la clave pública de Arancha que posee Javier con la nueva clave pública. A partir de ahí intercepta los mensajes que Javier envía a Arancha. Cada mensaje que intercepte lo descifrará usando la nueva clave privada, lo volverá a cifrar con la clave auténtica de Arancha, y reenviará el mensaje cifrado a Arancha. Así, todos los mensajes que Javier envíe a Arancha pueden ser leídos por Jordi.

Una buena gestión de claves es crucial para asegurarse, no sólo de la integridad de nuestros ficheros de claves, sino también de la integridad de los anillos de claves de otros. El punto central en la gestión de claves en GnuPG es la noción que hay detrás de firmar las claves. Firmar las claves tiene dos propósitos principales: nos permite detectar una manipulación en nuestros anillos de claves, y nos permite certificar que una clave realmente pertenece a la persona cuyo nombre aparece en el identificador de usuario de la clave. Las firmas de las claves también se usan en un esquema conocido como *anillo de confianza* para hacer extensiva la certificación de claves que no han sido firmadas directamente por el usuario, sino que han sido firmadas por otros en los que él confía. Un usuario responsable que lleve a cabo una buena gestión de claves puede contrarrestar los efectos prácticos de la manipulación de claves con GnuPG.

Gestionando nuestro par de claves

Un par de claves se compone de una clave pública y otra privada. Una clave pública se compone de la parte pública de la clave de firmado maestra, las partes públicas de las subclaves de firmado y cifrado, y de un juego de identificadores de usuario que se usa para asociar la clave pública con una persona real. Cada una de estas partes contiene datos sobre sí misma. Para una clave estos datos constan de su propio identificador, fecha de creación, fecha de caducidad, etc... Para un identificador de usuario, estos datos constan del nombre de la persona a la que identifica, un comentario opcional, y una dirección de correo electrónico. La estructura de la claves privada es parecida, con la diferencia de que sólo contiene las partes privadas de las claves, y que no tiene la información del identificador de usuario.

La opción de la línea de órdenes `--edit-key` se puede usar para ver un par de claves. Por ejemplo,

```
jordi% gpg --edit-key jordi@cat.es
Secret key is available.

pub  1024D/1208DD4F  created: 1999-09-24 expires: never      trust: -/u
sub  1024g/B9688D9E  created: 1999-09-24 expires: never
sub  2016G/4D88ACC4  created: 1999-09-26 expires: 2002-09-25
sub   960D/412EAF0C  created: 1999-09-26 expires: 2002-09-25
(1)  Jordi (Castell S.L.) <jordi@cat.es>
(2)  Jordi (oficina) <jordi@castell.com>

Command>
```

La clave pública se muestra junto con una indicación sobre si la clave privada se encuentra disponible o no. La información sobre cada componente de la clave pública se da en una lista. La primera columna indica el tipo de la clave. La palabra clave `pub` identifica a la clave pública maestra de firmado, y la palabra clave `sub` identifica a una clave pública subordinada a la anterior. La segunda columna informa sobre el tamaño en "bits" de la clave, el tipo, y el identificador. El tipo es `D` para una clave DSA, `g` para una clave ElGamal que sólo se pueda usar para cifrar, y `G` para una clave ElGamal que se pueda usar tanto para cifrar como para firmar. La fecha de creación y la fecha de caducidad se muestran en las columnas tres y cuatro. Los identificadores de usuario aparecen en una lista a continuación de las claves.

Se puede obtener más información sobre la clave mediante órdenes interactivas. La orden **toggle** sirve para conmutar entre los componentes público y privado de un par de claves, siempre y cuando ambos componentes estén presentes.

```
Command> toggle

sec  1024D/1208DD4F  created: 1999-09-24 expires: never
sbb  1024g/B9688D9E  created: 1999-09-24 expires: never
sbb   960D/412EAF0C  created: 1999-09-26 expires: 2002-09-25
sbb  2016G/4D88ACC4  created: 1999-09-26 expires: 2002-09-25
(1)  Jordi (Castell S.L.) <jordi@cat.es>
(2)  Jordi (oficina) <jordi@castell.com>
```

Esta información es parecida a la del componente para la clave pública. La palabra clave `sec` identifica a la clave privada maestra de firmado, y la palabra clave `sbb` identifica las claves subordinadas privadas. Los identificadores de usuario de la clave pública también aparecen en este caso.

Integridad de claves

Al distribuir nuestra clave pública, estamos distribuyendo los componentes públicos de nuestras claves maestra y subordinada, así como los identificadores de usuario. Sin embargo, por sí sola, la distribución de este material constituye un riesgo para la seguridad ya que sería posible que un atacante manipulara la clave. Se puede modificar una clave pública añadiendo o substituyendo claves, o añadiendo o cambiando los identificadores de usuario. Al manipular un identificador de usuario, el atacante podría cambiar la dirección de correo del identificador de usuario y redireccionar así el correo a su propia dirección. Al cambiar una de las claves de cifrado el atacante también podría descifrar los mensajes que le llegaran redireccionados.

El uso de las firmas digitales es una solución a este problema. Cuando unos datos son firmados por una clave privada, la clave pública correspondiente queda ligada a los datos firmados. En otras palabras, sólo la clave pública correspondiente puede verificar la firma y asegurar que los datos no han sido modificados. Una clave pública se puede proteger de la manipulación usando su correspondiente clave privada maestra, y firmando con ésta los componentes de la clave pública y los identificadores de usuario, ligando de este modo los componentes a la clave pública maestra. La acción de firmar los componentes de la clave pública con la correspondiente clave privada maestra se conoce como *autofirmar*, y una clave pública que contenga identificadores de usuario autofirmados ligados a ella se llama *certificado*.

Como ejemplo, si Jordi tuviera dos identificadores de usuario y tres subclaves, las firmas en los identificadores de usuario se podrían comprobar con la orden **check** desde el menú de edición de la clave.

```
jordi% gpg --edit-key jordi
Secret key is available.

pub  1024D/1208DD4F  created: 1999-09-24 expires: never      trust: -/u
sub  1024g/B9688D9E  created: 1999-09-24 expires: never
sub   960D/412EAF0C  created: 1999-09-26 expires: 2002-09-25
sub  2016G/4D88ACC4  created: 1999-09-26 expires: 2002-09-25
(1)  Jordi (Castell S.L.) <jordi@cat.es>
```

```
(2)  Jordi (oficina) <jordi@castell.com>

Command> check
uid  Jordi (Castell S.L.) <jordi@cat.es>
sig!       1208DD4F 1999-09-24   [self-signature]
uid  Jordi (oficina) <jordi@castell.com>
sig!       1208DD4F 1999-09-26   [self-signature]
```

Como era de esperar, la clave firmante para cada firma es la clave de firmado maestra con el identificador de clave `0x26B6AAE1`. La autofirma en las subclaves se encuentra presente en la clave pública, pero en la interfaz de GnuPG no aparece.

Añadir y eliminar componentes a las claves

Tanto las nuevas subclaves como los nuevos identificadores de usuario, pueden ser añadidos a nuestro par de claves una vez que éste ya haya sido creado. Un identificador de usuario se añade mediante la orden **adduid**. A continuación se nos pedirá que introduzcamos un nombre real, una dirección de correo electrónico, y un comentario, del mismo modo que cuando generamos el par de claves inicial. Una subclave se añade usando la orden **addkey**. La interfaz es parecida a la usada cuando generamos un par de claves nuevo. La subclave puede ser una clave para firmado DSA y una clave sólo para cifrado ElGamal, o una clave para firmado y cifrado ElGamal. Cuando se genera una subclave o un identificador de usuario, éstos son autofirmados con nuestra clave de firmado maestra, por lo que se nos requerirá que introduzcamos la contraseña.

Los identificadores de usuario adicionales son útiles si necesitamos múltiples identidades. Por ejemplo, puede darse el caso de que utilicemos una identidad para el trabajo y otra para nuestras actividades políticas. Los colegas de trabajo nos conocerán por nuestro identificador de usuario del trabajo. Los correligionarios políticos nos conocerán por nuestra identidad de usuario política. Como es posible que estos grupos de personas no coincidan, cada grupo no se fiará del identificador que no corresponda. Por lo tanto, ambos identificadores de usuario son necesarios.

Las subclaves adicionales son también muy útiles. Los identificadores de usuario asociados con nuestra clave pública maestra son validados por las personas con quien nos comunicamos y por tanto, cambiar la clave maestra requiere «recertificación». Pero esto puede resultar difícil y una pérdida de tiempo si nos comunicamos con muchas personas. Por otra parte, es bueno cambiar las subclaves de cifrado periódicamente. Si una clave es descubierta, todos los datos cifrados con esa clave serán vulnerables. De todos modos, al cambiar las claves sólo los datos cifrados con la clave descubierta serán revelados.

También es posible eliminar las subclaves y los identificadores de usuario. Para eliminar una subclave o un identificador de usuario es preciso seleccionarlo primero con las órdenes **key** o **uid** respectivamente. Estas órdenes actúan como conmutadores. Por ejemplo, la orden **key** ***2*** selecciona la segunda subclave, e invocando **key** ***2*** de nuevo, desactiva la selección. Si no se da ningún otro argumento extra, todas las subclaves o identificadores de usuario son deseleccionados. Una vez que los identificadores de usuario que se van a eliminar son seleccionados, la orden **deluid** elimina los identificadores de usuario de la clave. De igual forma, la orden **delkey** elimina todas las subclaves previamente seleccionadas de nuestras claves pública y privada.

En la gestión local de anillos de claves, el eliminar componentes innecesarios de una clave es una buena práctica para aliviar los anillos de claves públicas de otros. Sin embargo, eliminar identificadores de usuarios y subclaves de nuestra propia clave no es siempre conveniente, ya que puede complicar la distribución de la clave. Por definición, cuando un usuario importa nuestra clave pública actualizada ésta se fusiona con la copia de la clave pública vieja en su anillo de claves, siempre que la vieja esté allí «a priori». Los componentes de ambas claves se combinan con la fusión, y el resultado de ésta es que se añade cualquier nuevo componente, pero también que se restaura cualquier componente eliminado. Para actualizar de manera correcta la clave es necesario que el usuario elimine primero la versión antigua de nuestra clave, y después que importe la versión nueva. Esto significa una carga extra para la gente con la que nos comunicamos. Es más, si enviáramos nuestra clave a un servidor de claves la fusión ocurriría a nuestro pesar, y cualquiera que se bajara nuestra clave del servidor no vería nunca nuestra clave con los componentes eliminados. En consecuencia, para actualizar nuestra propia clave es mejor revocar los componentes de la clave que eliminarlos.

Revocar componentes de una clave

Para revocar una subclave antes tenemos que seleccionarla. Una vez seleccionada, puede ser revocada con la orden **revkey**. La clave se revoca añadiéndole una autofirma de revocación. Al contrario que la opción de la línea de órdenes `--gen-revoke`, el efecto de revocación de una subclave es inmediato.

```
Command> key 4D88ACC4

pub  1024D/1208DD4F  created: 1999-09-24 expires: never      trust: -/u
sub  1024g/B9688D9E  created: 1999-09-24 expires: never
sub* 2016G/4D88ACC4  created: 1999-09-26 expires: 2002-09-25
sub   960D/412EAF0C  created: 1999-09-26 expires: 2002-09-25
(1)  Jordi (Castell S.L.) <jordi@cat.es>
(2)  Jordi (oficina) <jordi@castell.com>

Command> revkey
Do you really want to revoke this key? y

You need a passphrase to unlock the secret key for
user: "Jordi (Castell S.L.) <jordi@cat.es>"
1024-bit DSA key, ID 1208DD4F, created 1999-09-24

pub  1024D/1208DD4F  created: 1999-09-24 expires: never      trust: -/u
sub  1024g/B9688D9E  created: 1999-09-24 expires: never
sub  2016G/4D88ACC4  created: 1999-09-26 expires: 2002-09-25
rev! subkey has been revoked: 1999-09-26
sub   960D/412EAF0C  created: 1999-09-26 expires: 2002-09-25
(1)  Jordi (Castell S.L.) <jordi@cat.es>
(2)  Jordi (oficina) <jordi@castell.com>
```

La revocación de un identificador de usuario funciona de modo diferente. Generalmente, un identificador de usuario recolecta firmas que atestigüen que el identificador de usuario describe a la persona que sea la auténtica propietaria de la clave asociada. En teoría un identificador de usuario describe a una persona para siempre, ya que la identidad de esa persona no cambiará nunca. En la práctica los elementos del identificador de usuario, como la dirección de correo electrónico y el comentario, pueden cambiar con el tiempo, invalidando así el identificador de usuario.

Las especificaciones de OpenPGP

* *Primera referencia a OpenPGP*

no preveen la revocación del identificador de usuario, pero un identificador de usuario puede ser revocado de modo efectivo, revocando la autofirma en el identificador de usuario. Por razones de seguridad descritas anteriormente, los corresponsales no confiarán en un identificador de usuario con una autofirma no válida.

Una firma puede ser revocada mediante la orden **revsig**. Como es posible que hayamos firmado cualquier cantidad de identificadores de usuario, la interfaz nos pedirá que decidamos si queremos o no revocar cada una de las firmas.

```
Command> revsig
Command> revsig
You have signed these user IDs:
     Jordi (Castell S.L.) <jordi@cat.es>
   signed by 1208DD4F at 1999-09-24
     Jordi (oficina) <jordi@castell.com>
   signed by 1208DD4F at 1999-09-26
user ID: "Jordi (Castell S.L.) <jordi@cat.es>"
signed with your key 1208DD4F at 1999-09-24
Create a revocation certificate for this signature? (y/N)n
user ID: "Jordi (oficina) <jordi@castell.com>"
signed with your key 1208DD4F at 1999-09-26
Create a revocation certificate for this signature? (y/N)y
You are about to revoke these signatures:
     Jordi (oficina) <jordi@castell.com>
   signed by 1208DD4F at 1999-09-26
Really create the revocation certificates? (y/N)y

You need a passphrase to unlock the secret key for
user: "Jordi (Castell S.L.) <jordi@cat.es>"
1024-bit DSA key, ID 1208DD4F, created 1999-09-24

Enter passphrase:

pub  1024D/1208DD4F  created: 1999-09-24 expires: never      trust: -/u
sub  1024g/B9688D9E  created: 1999-09-24 expires: never
sub  2016G/4D88ACC4  created: 1999-09-26 expires: 2002-09-25
rev! subkey has been revoked: 1999-09-26
sub   960D/412EAF0C  created: 1999-09-26 expires: 2002-09-25
(1)  Jordi (Castell S.L.) <jordi@cat.es>
(2)  Jordi (oficina) <jordi@castell.com>
```

Un identificador de usuario revocado vendrá indicado por la firma de revocación en el identificador, y se podrá ver en las firmas en la lista de identificadores de usuario de las claves.

```
Command< check
uid  Jordi (Castell S.L.) <jordi@cat.es>
sig!       1208DD4F 1999-09-24   [self-signature]
uid  Jordi (oficina) <jordi@castell.com>
rev!       1208DD4F 1999-09-26   [revocation]
sig!       1208DD4F 1999-09-26   [self-signature]
sig!       B87DBA93 1999-06-28   [self-signature]
```

Al revocar tanto las subclaves como las autofirmas en los identificadores de usuarios, añadimos autofirmas de revocación a la clave. Ya que las firmas son añadidas a la clave y no eliminamos nada, una revocación siempre estará visible para otros usuarios desde el momento en el que la actualización de nuestra clave pública sea distribuida y fusionada con las copias más viejas de ésta. Por lo tanto, la revocación garantiza que todos los usuarios tengan una copia consistente de nuestra clave pública.

Actualizar la fecha de caducidad de una clave

La fecha de caducidad de una clave puede ser actualizada con la orden **expire** desde el menú de edición de la clave. Si no se selecciona ninguna clave, se actualizará la fecha de caducidad de la clave primaria. De otro modo, se actualizará la fecha de caducidad de la clave subordinada que hayamos seleccionado.

La fecha de caducidad de una clave está asociada con la autofirma de la clave. La fecha de caducidad se actualiza eliminando la autofirma antigua y añadiendo una nueva autofirma. Ya que los corresponsales no habrán eliminado la autofirma antigua, verán una autofirma adicional en la clave cuando actualicen su copia de nuestra clave. La última autofirma es la que tiene precedencia, así que todos los corresponsales pueden saber sin ambigüedad las fechas de caducidad de nuestras claves.

Validar otras claves en nuestro anillo de claves públicas

En el capítulo 1 se introdujo un procedimiento para validar las claves públicas de otros usuarios: la clave de otro usuario se valida comprobando personalmente la huella digital de su clave, y firmando su clave pública con nuestra clave privada. Comprobando personalmente la huella digital podemos estar seguros de que la clave pertenece realmente al supuesto usuario y, dado que hemos firmado la clave, podemos estar seguros de que detectaremos cualquier manipulación o falsificación en un futuro. Desafortunadamente este proceso se complicado cuando debemos validar un gran número de claves o cuando debemos comunicarnos con personas a las que no conocemos personalmente.

GnuPG trata este problema con un mecanismo conocido como *anillo de confianza*. En el modelo del anillo de confianza la responsabilidad de la validación de las claves públicas recae en las personas en las que confiamos. Por ejemplo, supongamos que

- Javier ha firmado la clave de Arancha y que,
- Arancha ha firmado las claves de Jordi y de Ignacio.

Si Javier confía en Arancha lo suficiente como para validar las claves que él firma, entonces Javier puede deducir que las claves de Jordi y de Ignacio son válidas sin llegar a comprobarlas personalmente. Javier se limitará a usar su copia validada de la clave pública de Arancha para comprobar que las firmas de Arancha sobre Jordi e Ignacio son auténticas. Por lo general, y asumiendo que Javier confíe plenamente en todos como para validar las claves que firmen, cualquier clave firmada por una clave válida también es considerada válida. El origen es la clave de Javier, que se asumirá axiomáticamente como válida.

Confianza en el propietario de una clave

En la práctica la confianza es algo subjetivo. Por ejemplo, la clave de Arancha es válida para Javier ya que ha sido firmada por ella, pero Javier puede desconfiar de otras claves que hayan sido validadas por la firma de Arancha. En este caso, Javier no aceptaría las claves de Jordi e Ignacio como válidas sólo porque hayan sido firmadas por Arancha. El modelo del anillo de confianza prevee este caso mediante un indicador que asocia nuestro nivel de confianza en el propieario de la clave, a cada clave pública en nuestro anillo de claves. Existen cuatro niveles de confianza:

unknown

Desconocido. No se sabe nada sobre el dueño de la clave firmante. Las claves en nuestro anillo de claves que no nos pertenezcan tendrán al principio este nivel de confianza.

none

Ninguno. Se sabe que el propietario firma otras claves de modo impropio.

marginal

Marginal. El propietario comprende las implicaciones de firmar una clave y valida las claves de forma correcta antes de firmarlas.

full

Absoluto. El propietario comprende perfectamente las implicaciones de firmar una clave y su firma sobre una clave es tan buena como la nuestra.

El nivel de confianza en una clave es algo que sólo nosotros podemos asignar a la clave, y se considera información privada. El nivel de confianza no se exporta con la clave, de hecho no se almacena en los anillos de claves sino en una base de datos aparte.

El editor de claves de GnuPG puede ser usado para ajustar nuestra confianza en el propietario de una clave. La orden es **trust**. En el siguiente ejemplo Javier edita su confianza en Arancha y actualiza la base de datos para recomputar qué claves son válidas, basándose en su nuevo nivel de confianza en Arancha.

```
javier:~$ gpg --edit-key Aranzanzu

pub  1024D/B63E132C  created: 1999-09-24 expires: never      trust: q/f
sub  1024g/581A915F  created: 1999-09-24 expires: never
(1)  Ar\xc3\xa1nzazu (A.G.deZ.) <arancha@nav.es>

Command> trust
pub  1024D/B63E132C  created: 1999-09-24 expires: never      trust: q/f
sub  1024g/581A915F  created: 1999-09-24 expires: never
(1)  Ar\xc3\xa1nzazu (A.G.deZ.) <arancha@nav.es>

Please decide how far you trust this user to correctly
verify other users' keys (by looking at passports,
checking fingerprints from different sources...)?

 1 = Don't know
 2 = I do NOT trust
 3 = I trust marginally
 4 = I trust fully
 s = please show me more information
 m = back to the main menu

Your decision? 3

pub  1024D/B63E132C  created: 1999-09-24 expires: never      trust: m/f
sub  1024g/581A915F  created: 1999-09-24 expires: never
(1)  Ar\xc3\xa1nzazu (A.G.deZ.) <arancha@nav.es>

Command> quit
[...]
```

La confianza en el propietario de la clave y la validez de la clave se indican a la derecha al mostrar la clave. La confianza en el propietario se muestra primero, y la validez después[1]. Los cuatro niveles de confianza/validez se encuentran abreviados: unknown (`q`), none (`n`), marginal (`m`), y full (`f`). O sea, *desconocido*, *ninguno*, *marginal* y *absoluto*. En este caso, la clave de Arancha es totalmente válida ya que está firmada por Javier. Al principio el nivel de confianza que Javier tiene en Arancha es desconocido, por lo que no procede validar otras claves; pero luego decide confiar en ella de modo marginal.

Usar la confianza para validar las claves

El anillo de confianza permite usar un algoritmo más elaborado para validar una clave. Anteriormente, una clave sólo se consideraba válida si la firmábamos nosotros personalmente. Ahora es posible usar un algoritmo más flexible: una clave *K* se considera válida si cumple dos condiciones:

1. si viene firmada por las suficientes claves válidas, lo que quiere decir

   - que la hemos firmado nosotros personalmente,
   - o que ha sido firmada por una clave de plena confianza,
   - o que ha sido firmada por tres claves de confianza marginal;

2. y si el camino de claves firmadas que nos lleva desde *K* hasta nuestra propia clave es de cinco pasos o menos.

La longitud del camino, en número de claves con confianza marginal requeridas, y el número de claves con confianza plena requeridas se pueden cambiar. Los números dados arriba son los valores por definición usados por GnuPG.

Figura 3-1 muestra un anillo de confianza cuyo origen está en Javier. El gráfico ilustra quién ha firmado las claves de quién. La tabla muestra qué claves son consideradas válidas por Javier en base a su confianza en otros miembros del anillo.

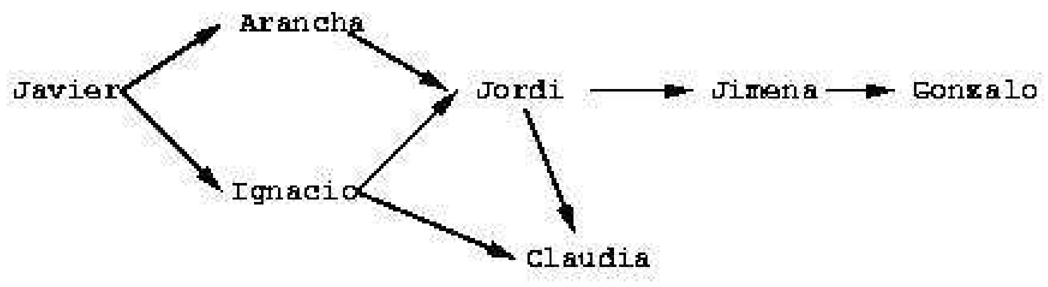
* *Un error potential:* `--completes-needed` *en la línea de órdenes, parece que es ignorado al combinarlo con* `--update-trustdb`*. Sin embargo, el valor se toma correctamente si se pone en el fichero "options".*

Este ejemplo asume que se necesitan dos claves de confianza marginal o una de confianza plena para validar otra clave. La longitud máxima del camino es tres.

Al computar claves válidas en el ejemplo, las de Arancha e Ignacio siempre son consideradas como totalmente válidas, ya que están directamente firmadas por Javier. La validez de las otras claves depende de la confianza. En el primer caso la confianza en Ignacio es plena, lo que implica que las claves de Jordi y Claudia se considerarán válidas. En el segundo ejemplo la confianza en Arancha e Ignacio es marginal. Ya que son necesarias dos claves de confianza marginal para dar validez total a una clave, la clave de Jordi será considerada como totalmente válida, pero la clave de Claudia será considerada sólo como marginalmente válida. En el caso en el que Jordi e Ignacio tuvieran confianza marginal, la clave de Jordi sería marginalmente válida ya que la clave de Ignacio es totalmente válida. Sin embargo, la clave de Claudia será considerada marginalmente válida ya que sólo se puede usar una clave completamente válida para validar otras claves, y la clave de Ignacio es la única clave válida que se ha usado para firmar la clave de Claudia. Al añadir un nivel de confianza marginal a Arancha, la clave de Jordi se convierte en totalmente válida y por tanto puede ser usada para validar totalmente la clave de Claudia, y validar marginalmente la clave de Jimena. Por último, una confianza plena en Arancha, Jordi y Jimena es todavía insuficiente para validar la clave de Gonzalo ya que el camino máximo de certificación es tres, pero la longitud del camino desde Gonzalo hasta Javier es cuatro.

El modelo del anillo de confianza es una aproximación flexible al problema del intercambio seguro de claves públicas. Nos permite poner a punto GnuPG para que refleje el modo en que lo usamos. Es posible llegar a insistir en múltiples caminos cortos desde nuestra clave hasta otra clave *K* para cambiar el nivel de confianza. Por otra parte, puede ser que caminos más largos nos satisfagan e incluso un sólo camino desde nuestra clave hasta la otra clave *K*. El requerimiento de múltiples caminos cortos es una fuerte garantía de que *K* pertenece a quien nosotros creemos. El precio, por supuesto, es la dificultad añadida para validar claves, ya que debemos firmar personalmente más claves que si aceptáramos menos y más largos caminos.

Figura 3-1. Un anillo de confianza hipotético



trust/confianza		**validity/validez**	
marginal	**full/plena**	**marginal**	**full/plena**
	Ignacio		Arancha, Jordi, Ignacio, Claudia
Arancha, Ignacio		Claudia	Arancha, Jordi, Ignacio
Jordi, Ignacio		Jordi, Claudia	Arancha, Ignacio
Arancha, Jordi, Ignacio		Jimena	Arancha, Jordi, Ignacio, Claudia
	Arancha, Jordi, Jimena		Arancha, Jordi, Jimena, Claudia

Distribución de claves

Lo ideal sería que pudiéramos distribuir nuestra clave entregándosela en persona a nuestros corresponsales. Sin embargo, en la práctica las claves se distribuyen a menudo por correo electrónico o algún otro medio de comunicación electrónica. La distribución por correo electrónico es una buena práctica sólo cuando tengamos unos pocos corresponsales, e incluso si tuviéramos muchos corresponsales, podríamos usar un medio alternativo como puede ser publicar nuestra clave pública en nuestra página en internet. Pero esto es inútil si las personas que necesitan nuestra clave pública no saben dónde encontrar nuestra página.

Para solventar este problema existen los servidores de claves públicas que recolectan y distribuyen las claves públicas. Cuando un servidor recibe una clave pública, bien la añade a la base de datos o bien la fusiona con una copia de la clave. Cuando alguien requiere al servidor una clave pública, éste la busca en la base de datos y, si la encuentra, la envía a quien se la haya solicitado.

Los servidores de claves también son útiles cuando hay muchas personas que firman las claves de otras con frecuencia. Sin un servidor de claves, cuando Arancha firmara la clave de Javier, debería enviar a éste una copia de la clave firmada por ella de manera que Javier pudiera añadir la clave firmada a su anillo de claves, así como distribuirla a todos sus corresponsales. Mediante este proceso Javier y Arancha sirven a la totalidad de la comunidad construyendo lazos en forma de anillos de confianza o lo que es lo mismo, mejorando la seguridad de PGP. De todos modos esto es una molestia si se firman las claves con frecuencia.

El uso de un servidor de claves facilita este proceso. Después de firmar la clave de Javier, Arancha puede enviar la copia firmada por ella al servidor de claves. El servidor de claves añade la firma de Arancha a la copia que ya existente de la clave pública de Javier. Las personas que estén interesadas en actualizar su copia de la clave de Javier consultan al servidor por propia iniciativa para obtener la clave actualizada. Javier no necesita distribuir la clave y puede obtener las firmas en su clave requiriéndolas al servidor.

* *La opción `--keyserver` debe preceder a las opciones `--send-key` o `--recv-key`. Esto es un error.*

Se puede enviar una o más claves usando la opción de la línea de órdenes `--send-keys`. Esta opción toma uno o más especificadores de clave, y envía las claves especificadas al servidor de claves. El servidor al que se envían las claves es especificado con la opción de la línea de orden `--keyserver`. Paralelamente, la opción `--recv-keys` se usa para obtener claves desde un servidor de claves, pero la opción `--recv-keys` requiere el uso de un identificador de clave para poder especificar la clave deseada. En el siguiente ejemplo Javier actualiza su clave pública con nuevas firmas desde el servidor de claves *certserver.pgp.com* y acto seguido envía su copia de la clave pública de Arancha al mismo servidor de claves para que se actualice con las claves que él mismo pueda haber añadido.

```
javier:~$ gpg --keyserver certserver.pgp.com
--recv-key D58711B7
gpg: requesting key D58711B7 from certserver.pgp.com ...
gpg: key D58711B7: 1 new signature

gpg: Total number processed: 1
gpg:         new signatures: 1
javier:~$ gpg --keyserver certserver.pgp.com
--send-key arancha@nav.es
gpg: success sending to ’certserver.pgp.com’ (status=200)
```

Existen varios servidores de claves en funcionamiento en todo el mundo. Los servidores más importantes están sincronizados de modo que es posible elegir un servidor de claves cercano a nosotros en Internet y usarlo de forma regular para enviar y recibir claves.

Notas

1. N. de T.: GnuPG hace un uso excesivo de la palabra “trust”, utilizándola en el sentido de «confianza en el propietario y confianza en la clave». Esto puede llevar a confusión. Algunas veces la confianza en un propietario viene referida como *owner-trust* para distinquirla de la confianza en una clave. Sin embargo, a lo largo de este manual “trust” se usa en el sentido de «confianza en el propietario de la

clave», y “validity” en el sentido de «confianza en que una clave pertenece a la persona asociada con el identificador de clave».

Capítulo 4. Uso diario de GnuPG

GnuPG es una compleja herramienta que no está exenta de polémica técnica, social y legal. En el plano técnico, ha sido diseñada para su uso en situaciones con diversos requerimientos de seguridad. Esto complica la gestión de claves. En el plano social, el uso de GnuPG no es estricatamente una decisión de tipo personal. Para que su uso tenga efectividad, GnuPG requiere que todas las partes en una comunicación sean usuarios del programa. En el plano legal, desde 1.999, las leyes que contemplan el uso de productos informáticos criptológicos, y en particular si el uso de GnuPG es legal, son diferentes según los países y están siendo actualmente debatidas por muchos gobiernos.

Este capítulo tratará sobre estos temas. Se intentará dar consejos prácticos sobre el uso de GnuPG de cara a los requerimientos de seguridad del usuario. También se sugerirán vías para promocionar el uso de GnuPG con vistas a la comunicación segura entre el usuario y las personas con las que éste se comunique, aun cuando éstos no sean usuarios de GnuPG. Finalmente, se resaltará el estado legal de GnuPG conforme con el estado actual de las leyes sobre criptología y cifrado en el mundo.

Definiendo los requerimientos en seguridad

GnuPG es una herramienta que utiliza el usuario para proteger su privacidad. La protección existe sólo si el usuario puede comunicarse con otros sin «escuchas» que puedan leer los mensajes.

El modo en que se deba usar GnuPG dependerá de la determinación y de los recursos de aquéllos que intenten, o puedan intentar, leer los mensajes cifrados del usuario. Un «escucha» podría ser un administrador de sistemas sin escrúpulos, que se encuentre, «por casualidad», escaneando el correo del usuario, o podría ser un espía industrial que intentara acceder a los secretos de una compañía, o incluso podría ser un organismo legal que quisiera llevarnos a juicio. El uso de GnuPG para protegernos contra «intromisiones casuales», será diferente de su uso para protegernos contra un adversario determinado. Nuestro fin último debe ser el de que el conseguir los datos no cifrados resulte más caro que el valor de los datos en sí.

La personalización del uso de GnuPG se basa en los siguientes aspectos:

- la elección del tamaño del par de claves público y privado.
- la protección de la clave privada,
- la selección de la fecha de caducidad y el uso de subclaves, y
- la gestión del anillo de confianza.

Una buena elección del tamaño de la clave nos protegerá contra ataques de fuerza bruta en los mensajes cifrados. La protección de nuestra clave privada ayudará a prevenir que un atacante pueda llegar a usarla para descifrar mensajes o firmar mensajes en nuestro nombre. Una gestión correcta de nuestro anillo de confianza, ayudará a evitar que cualquier atacante pueda hacerse pasar por una persona de nuestra confianza. La consecución de estos aspectos según las necesidades de nuestra propia seguiridad, es el modo de equilibrar el trabajo extra que se requiere para usar GnuPG con la privacidad que ofrece.

Selección del tamaño de la clave

La selección del tamaño de la clave depende de la clave en sí. En OpenPGP, un par de claves pública y privada contienen generaralmente claves múltiples. Como mínimo tiene una clave de firmado maestra, y probablemente una o más subclaves de cifrado adicionales. Si usamos para la generación de claves los parámetros por defecto en GnuPG, la clave maestra será una clave DSA, y las subclaves serán claves ElGamal.

DSA nos permite un tamaño de clave de hasta 1024 bits. Dada la tecnología de factorización de hoy en día, esto no es demasiado bueno, pero es lo que especifica el estándar. Sin duda alguna, debemos usar claves DSA de 1024 bits.

Por otra parte, las claves ElGamal pueden ser de cualquier tamaño. Ya que GnuPG es un sistema de clave pública híbrido, la clave pública se usa para cifrar una clave de sesión de 128 bits, y la clave privada se usa para descifrarla. Sin embargo el tamaño de la clave afecta a la velocidad del cifrado y descifrado, ya que el valor de estos algoritmos lleva como exponente el tamaño de la clave. Una clave de mayor tamaño también tarda más en ser generada, y ocupa más espacio. Además, cuanto mayor tamaño tenga una clave, la seguridad extra que nos ofrece, aumenta pero con una marcha decreciente. También hay que tener en cuenta que un «escucha» que se encuentre con una clave lo suficientemente grande como para resistir un ataque de fuerza bruta, se limitará a cambiar de método para poder obtener los datos no cifrados. Por lo tanto, 1024 bits es el tamaño de clave recomendado. Si nuestros requerimientos de seguridad fueran tan grandes como para necesitar claves de gran tamaño, entonces deberíamos consultar a un experto en seguridad de datos.

Protección de la clave privada

La protección de la clave privada es la parte más importante en el uso de GnuPG. Si alguien obtuviera nuestra clave privada, todos los datos que hubieran sido cifrados para esa clave podrían ser descifrados y se podría firmar documentos bajo nuestro nombre. Si perdiéramos la clave privada, ya no podríamos descifrar los documentos cifrados que nos hubieran enviado o que nos enviaran en un futuro, y no podríamos firmar los documentos. La pérdida de posesión de nuestra clave privada podría ser una catástofre.

Sea cual fuere el uso que hagamos de GnuPG, deberíamos guardar siempre un certificado de revocación de nuestras claves públicas, y una copia de seguridad de nuestra clave privada en un disco protegido contra la escritura, y en un lugar seguro. Por ejemplo, podríamos grabarlo en un CD-ROM y guardar éste en un cofre de depósito de un banco, dentro de un sobre sellado. Alternativamente, podríamos guardarlo en un disquete y esconderlo en nuestra casa. Hagamos lo que hagamos, los certificados de revocación de las claves públicas y las copias de seguridad de la clave privada deberíamos tenerlos guardados en un medio lo suficientemente seguro, mucho más que la clave privada que utilizamos a diario.

Como medida de seguridad, GnuPG no guarda nuestra clave privada «en bruto» en el disco duro, sino que la cifra mediante un algoritmo de cifrado simétrico. Por esta razón, para acceder a nuestra clave, necesitamos una contraseña. Por lo tanto, existen dos barreras que un atacante debe cruzar para poder acceder a nuestra clave privada: (1) debe conseguir la clave; (2) debe ser capaz de descifrarla.

Esconder nuestra clave privada en un sitio seguro es importante, pero todo tiene un precio. Lo ideal sería que guardáramos la clave en un disco que fuera portátil y que tuviera protección contra la escritura, como un disquete, y que sólo lo usáramos en una máquina con un solo usuario que no estuviera conectada a una red. Esto puede que no sea convenient o posible para nosotros. Por ejemplo, es posible que no tengamos nuestra propia máquina, y que debamos usar la de la universidad o la de la oficina, o puede significar que para ello tengamos que desconectar el modem cada vez que queramos usar GnuPG.

Esto no quiere decir que no podamos o no debamos usar GnuPG. Tan sólo significa que debemos decidir si los datos que estamos protegiendo son lo suficientemente importantes para cifrarlos, pero no tan importantes como para tomar medidas extra de seguridad. La elección es nuestra.

Una buena contraseña es absolutamente crítica para el uso de GnuPG. Cualquier atacante que logre acceder a nuestra clave privada, deberá sobrepasar el cifrado de nuestra clave privada. En lugar de usar un ataque de fuerza bruta, es casi seguro que un atacante intentará adivinar la contraseña.

El motivo por el que intentaría adivinarla, es que la mayoría de personas escogen contraseñas que son más fáciles de adivinar que de sobrepasar una clave aleatoria de 128 bits. Si la contraseña es una palabra ("password"), es mucho más fácil probar con todas las palabras existentes en los diccionarios de todas las lenguas del mundo. Aun cuando la palabra sea permutada, v.g. k3wldood, será más fácil probar palabras de diccionario con un catálogo de permutaciones. Lo mismo ocurre con las citas. Aunque sea una contraseña formada por frases ("passphrase"), si ésta tiene como base un lenguaje natural, será una contraseña débil, ya que existirá poca aleatoriedad y muchas redundancias. Si es posible, debemos evitar contraseñas basadas en lenguajes naturales.

Una buena contraseña es aquélla que podemos recordar, pero que es difícli que otro pueda adivinar. Debería incluir todos los carácteres imprimibles de nuestro teclado posibles. Esto incluye carácteres alfabéticos en mayúsculas y minúsculas, números, y carácteres especiales como } o ¦. Debemos ser creativos y perder un poco de tiempo inventando una contraseña; una buena elección es importante para asegurar nuestra privacidad.

Selección de las fechas de caducidad y uso de subclaves

Por defecto, una clave de firmado maestra DSA y una subclave de cifrado ElGamal son generadas al crear un nuevo par de claves. Esto es conveniente porque cada clave juega un papel diferente, y por lo tanto es posible que necesitemos que las claves tengan fechas de caducidad diferentes. La clave de firmado maestra se usa para las firmas digitales, y también recolectan las firmas de otras personas que hayan confirmado nuestra identidad. La clave de cifrado se usa sólo para descifrar los documentos cifrados que nos hayan enviado. Por lo general, una firma digital tiene una larga vida, v.g., para siempre, y tampoco queremos perder las firmas que tanto nos ha costado recolectar en nuestra clave. Por otra parte, la subclave de cifrado puede cambiar periódicamente por cuestiones de seguridad, ya que si una clave de cifrado es descifrada, el atacante puede leer todos los documentos que sean cifrados para esa clave.

Suele ocurrir que no queramos que nuestra clave maestra caduque. Existen dos razones por las que debamos escoger una fecha de caducidad. La primera es que tengamos la intención de que la clave tenga una vida limitada. Por ejemplo, si la usamos para una campaña política y no nos será útil una vez pasada la campaña. La segunda es que si perdemos el control de la clave, y no tenemos un certificado de revocación con el que revocarla, una fecha de caducidad sobre la clave maestra asegura que la clave caerá finalmente en desuso.

Cambiar las subclaves de cifrado puede ser sencillo, pero puede que no sea del todo conveniente. Si generamos un nuevo par de claves con una fecha de caducidad en la subclave, ésta caducará en su momento. Poco antes de la caducidad añadiremos una nueva subclave y publicaremos nuestra clave pública actualizada. Una vez que la subclave haya caducado, aquellos que deseen comunicarse con nosotros deberán encontrar antes la clave actualizada, pues ya no podrán enviar datos cifrados para esa clave. El inconveniente depende de cómo distribuyamos la clave. Por fortuna, no es necesario firmas extras ya que la nueva clave habrá sido firmada con con nuestra clave de firmado maestra, la cual ya había sido validada por nuestros corresponsales.

Todo depende de que queramos o no tener una seguridad extra. Al igual que nosotros, un atacante todavía podrá leer todos los documentos que hayan sido cifrados para una clave caducada. Cambiar las subclaves sólo protege los futuros documentos. Para poder leer documentos cifrados para la nueva subclave, el atacante necesitaría organizar un nuevo ataque, usando cualquier técnica que hubiera usado la primera vez.

Al final sólo tiene sentido tener una sola subclave de cifrado en un anillo de claves. No hay se gana seguridad adicional teniendo dos o más subclaves activas. Por supuesto, puede existir cualquier número de claves caducadas en un anillo de claves, para que los datos que hubieran sido cifrados en el pasado puedan todavía ser descifrados, pero sólo es necesario activar una sola subclave en un momento dado.

Gestión del anillo de confianza

Al igual que con la protección de nuestra clave privada, la gestión de nuestro anillo de confianza es otro aspecto del uso de GnuPG que requiere equilibrar la seguridad y la facilidad de uso. Si estamos usando GnuPG para protegernos contra la posibilidad de una simple interceptación casual o de una falsificación, entonces nos podemos permitir ser algo confiados con las firmas de otros. Si, por el contrario, nos preocupa que pueda haber un atacante determinado interesado en invadir nuestra privacidad, entonces deberíamos confiar mucho menos en otras firmas, e invertir más tiempo en verificarlas.

Aparte de nuestras propias necesidades de seguridad, deberíamos tener *siempre cuidado* al firmar las claves de otras personas. Firmar una clave sin el suficiente grado de confianza en la validez de la clave, sólo para satisfacer nuestras propias necesidades de seguridad, es una actitud egoísta. Otros, con otras necesidades de seguridad más grandes, podrían fiarse de una clave que llevara nuestra firma. Si no pueden confiar en nosotros, entonces se debilita el anillo de confianza y se hace más difícil la comunicación para todos los usuarios de GnuPG. Así pues, debemos poner el mismo cuidado al firmar una clave que el que nos gustaría que pusieran otras personas cuando dependamos de sus firmas.

En la práctica, la gestión de nuestro anillo de claves evita el tener que ajustar las opciones `--marginals-needed` y `--completes-needed`. Cualquier clave que firmemos personalmente será considerada válida, pero con la excepción de grupos reducidos, no es práctico firmar la clave de cada persona con la que nos comuniquemos. Por lo tanto, tendremos que dejar que otros asignen el nivel de confianza.

Es aconsejable ser precisos cuando asignemos el nivel de confianza y cuando usemos las opciones para configurar el cuidado con que GnuPG debe ir con la validación de claves. Por ejemplo, es posible que tengamos plena confianza con unos pocos amigos íntimos, que sabemos que serán lo suficientemente cuidadosos cuando firmen una clave, y que otorguemos un nivel de confianza marginal al resto en nuestro anillo de claves. Desde esta perspectiva, podemos tener `--completes-needed` como `1` y `--marginals-needed` como `2`. Si estuviéramos más preocupados por la seguridad, podríamos escoger valores de `1` y `3`, o `2` y `3` respectivamente. Pero si no estamos tan preocupados por los posibles ataques sobre la privacidad, y simplemente queremos una fiabilidad razonable sobre la validez, escogeremos los valores `1` y `1`. Por regla general, los números más altos para estas opciones suponen que sería necesario que hubiera más gente conspirando contra nosotros para poder validar una clave que no pertenezca a la persona que nosotros creemos.

Construcción del anillo de confianza

No basta con que una persona quiera usar GnuPG. Para poder usarlo para comunicarse en modo seguro con otros, es necesario que tengamos un anillo de confianza. A primera vista, construir un anillo de confianza es una tarea desalentadora. Las personas con las que nos comunicamos necesitan usar GnuPG[1], y es necesario que haya las suficientes firmas para que las clave sean consideradas válidas. Estos no son problemas de tipo técnico; son problemas de tipo social. Debemos superar estos problemas si queremos usar GnuPG.

En nuestros primeros pasos con GnuPG, es importante darse cuenta de que no necesitamos comunicarnos de modo seguro con todas las personas. Empecemos con un pequeño círculo de amigos, tal vez sólo nosotros y uno ó dos más que también quieran ejercitarse en su derecho a la privacidad. Generemos nuestras claves, y firmémoslas recíprocamente. Éste es nuestro anillo de confianza inicial. Al hacerlo así, apreciaremos el valor de un pequeño, pero robusto anillo de confianza, y seremos más cautos a medida que vaya creciendo nuestro anillo de confianza con el tiempo.

Además de aquéllos en nuestro anillo de confianza iniciático, es posible que deseemos comunicarnos en modo seguro con otros que también están usando GnuPG. Sin embargo, esto puede ser problemático por dos razones: (1) no siempre sabemos cuándo alguien usa o estaría dispuesto a usar GnuPG; (2) si sabemos de alguien que lo usa, todavía es posible que tengamos problemas para validar su clave. El primer motivo sucede porque las personas no siempre hacen público que usan GnuPG. La forma de cambiar ese comportamiento es sentar ejemplo y hacer público que usamos GnuPG. Existen al menos tres maneras de hacer esto: firmar digitalmente los mensajes que enviamos a otros[2], publicar nuestra clave pública en nuestra página en Internet, o, si la subimos a un servidor de claves, poner nuestro identificador de clave como parte de nuestra firma. Haciendo pública nuestra clave, ayudamos a que sea más aceptable para otras personas hacerla también pública. Además, hacemos más fácil para otros el que puedan comenzar a comunicarse con nosotros en modo seguro, ya que habremos tomado la iniciativa, dejando claro que usamos GnuPG.

La validación de la clave es más difícil. Si no conocemos personalmente a la persona cuya clave queremos firmar, no es posible que podamos firmar su clave personalmente. Debemos fiarnos de las firmas de otras personas y esperar que encontraremos una cadena de firmas que nos lleven desde la clave en cuestión hasta la nuestra. Para poder encontrar una cadena, debemos tomar la iniciativa y conseguir que nuestra clave sea firmada por otras personas fuera de nuestro anillo de confianza inicial. Un modo efectivo de conseguir esto es participando en «reuniones de firmas».

De todos modos, debemos tener en cuenta que todo esto es opcional. No existe ningún tipo de obligación de hacer públicas nuestras claves o de firmar las claves de otros. GnuPG es lo suficientemente flexible como para adaptarse a nuestros requerimientos de seguridad, sean éstos los que sean. Sin embargo, la realidad social es que necesitaremos tomar la iniciativa si queremos engrosar nuestro anillo de confianza, y hacer todo el uso posible de GnuPG.

Uso legal de GnuPG

El estado legal de los programas criptológicos varía según países, y las leyes que rigen estos programas evolucionan con rapidez. Bert-Japp Koops (http://cwis.kub.nl/~frw/people/koops/bertjaap.htm) mantiene unos recursos sobre leyes y criptografía, Crypto Law Survey (http://cwis.kub.nl/~frw/people/koops/lawsurvy.htm), que podemos tomar como referencia para el status legal en cada país.

Notas

1. En esta sección, GnuPG se refiere a la implementación OpenPGP de GnuPG, así como a otras implementaciones como el producto PGP de NAI.
2. Se recomienda no hacerlo nunca en listas de correo de discusión pública, ya que esto podría molestar a algunas personas.

Capítulo 5. Tópicos

Este capítulo trata temas misceláneos que no se ajustan a otros capítulos del manual de usuario. A medida que se vayan añadiendo nuevos temas, se pueden ir creando nuevos capítulos con éstos. Sugerencias sobre temas que no han sido tratados son bienvenidas. ¡Mejor aún si estas sugerencias vienen acompañadas por un boceto o esquema sobre el tema sugerido!

Escribir interfaces de usuario

Alma Whitten (http://www.cs.cmu.edu/~alma) y Doug Tygar (http://www.cs.berkeley.edu/~tygar) han hecho un estudio (http://reports-archive.adm.cs.cmu.edu/anon/1998/abstracts/98-155.html) sobre la interfaz de usuario de PGP 5.0 de NAI, llegando a la conclusión de que los usuarios nuevos encuentran PGP confuso y frustrante. En su estudio sobre factores humanos, sólo cuatro de cada doce de los sujetos en estudio consiguieron enviar correctamente correo cifrado a los miembros de su equipo, y tres de cada doce enviaron el correo secreto sin ningún tipo de cifrado. Más aún, la mitad de los sujetos del estudio poseían conocimientos técnicos.

Estos resultados no son sorprendentes. PGP 5.0 tiene una bonita interfaz de usuario que es excelente si ya comprendemos el funcionamiento de la criptografía de clave pública, y si estamos familiarizados con el modelo de gestión de claves del anillo de confianza, especificado por OpenPGP. Por desgracia, los usuarios nuevos no entienden ni la criptología de clave pública, ni la gestión de claves, y la interfaz de usuario no es de ninguna ayuda en este caso.

Alguien que estuviera diseñando una interfaz de usuario, debería leer el estudio de Whitten y Tygar sin lugar a dudas. En él se dan detalles específicos sobre cada uno de los sujetos sometidos a estudio, y esos comentarios son interesantes. Por ejemplo, parece ser que muchos de los sujetos creían que un mensaje enviado a otra persona, debía ser cifrado con la clave pública del remitente. Reflexionemos un minuto sobre esto, y nos daremos cuenta de que es un fallo muy fácil de cometer. Por lo general, los nuevos usuarios tienen dificultades en comprender los propósitos diferentes de las claves públicas y de las privadas en GnuPG. Un diseñador de una interfaz de usuario debería intentar aclarar en todo momento cuándo una de las dos claves debe ser usada. Se podría usar ayudas guiadas, u otras técnicas de interfaz gráfica de usuario (GUI), con el fin de guiar al usuario en tareas tan comunes como la generación de claves, donde pasos extras como la generación de un certificado de revocación y hacer una copia de seguridad, son esenciales para el uso correcto de GnuPG. Otros comentarios sobre el estudio se incluyen a continuación.

- La seguridad es, generalmente, un objetivo secundario; los usuarios quieren enviar correo, enviarlo, y demás... No se debe asumir que los usuarios tienen ningún tipo de motivación para la lectura de manuales, o que buscarán controles de seguridad.
- La seguridad de una máquina en red es tan fuerte como el más débil de sus componentes. Los usuarios necesitan ser guiados para que presten atención a todos los aspectos de su seguridad, no se les puede dejar que procedan en exploraciones aleatorias, como podrían hacer con un procesador de texto o un hoja de cálculo.
- Se debe ser consistente en el uso de los mismos términos para las mismas operaciones. No se deben alternar sinónimos como «cifrar» y «encriptar»[1].

- Se debe simplificar la vista en pantalla para los usuarios inexpertos. Un exceso de información oculta la información importante. La configuración inicial podría concentrarse en dar al usuario el modelo correcto de la relación entre claves públicas y privadas, y una clara explicación de las funciones para la adquisición y distribución de claves.

Diseñar una interfaz de usuario efectiva para la gestión de claves es todavía más difícil. El modelo de anillos de confianza de OpenPGP es, por desgracia, bastante rígido. Por ejemplo, la especificación impones el uso de tres niveles de confianza arbitrarios: none (ninguna), marginal (idem), y complete (total). Todos los grados de confianza que pueda sentir el usuario deben ajustarse a esos tres. La validación del algoritmo también es difícil de comprender para aquellos que no sean muy expertos, en particular las nociones de “marginals needed” y “completes needed”[2]. Pero ya que el modelo de anillos de confianza está bien especificado y no se puede cambiar, hay que hacerlo lo mejor posible y diseñar una interfaz de usuario que ayude a clarificárselo de cara al usuario. Por ejemplo, una mejora sería generar un diagrama de cómo ha sido validada una clave, al ser requerida por el usuario. A continuación algunos comentarios relevantes del estudio.

- Los usuarios tienden a tener incerteza sobre cuándo y cómo pueden conceder accesos.
- Asegurarse que los usuarios comprenden su propia seguridad lo suficientemente bien como para prevenirlos de que cometan errores que puedan suponer un alto coste, debe ser un máxima prioridad. Tales errores incluyen: eliminar la clave privada por accidente; hacer pública una clave por error; revocar una clave por error; olvidar la contraseña; y no hacer copias de seguridad de sus anillos de claves.

Notas

1. Del documento original en inglés, ya que en castellano la palabra «encriptar» es incorrecta, mientras que la correcta es «cifrar». Así pues, en este caso y en castellano, estas dos palabras *no* son sinónimos.
2. Cuántas firmas que tengan asignada «confianza marginal» y cuántas firmas que tengan asignada «confianza completa» son necesarias