Firma digital de los trabajos con GnuPG

Jose Luis Díaz

2006

Resumen

Este año, de forma experimental, se pide a los alumnos que firmen digitalmente los trabajos que entregan. Este documento introduce el por qué de esta medida y muestra cómo utilizar la herramienta GnuPG (gpg) para este cometido.

Las primeras secciones de este documento se centran en detallar paso a paso los procedimientos elementales de uso de GnuPG, incidiendo en los conceptos básicos, y pueden ser útiles a cualquiera que desee comprender esta herramienta y comenzar a usarla. Quienes ya tengan experiencia en el uso de GnuPG, pueden omitir la lectura de este documento y leer directamente el titulado Pasos a seguir para la entrega[1], que detalla específicamente los procedimientos para la firma digital y entrega de las prácticas.

La descripción de los procedimientos asume un uso de `gpg` desde línea de comandos, y en particular desde una de las máquinas Linux de prácticas, para asegurar que todos los alumnos obtienen unos resultados como los mostrados en este documento. GnuPG está disponible también para Windows, y existen interfaces gráficos para simplificar su uso (como por ejemplo, WinPT), no descritos en este documento. Para más información y manuales (incluso en castellano) puedes visitar la página oficial de GnuPG[2].

En este documento se describe GnuPG como una herramienta útil "por si sola", sin necesidad de integrarse con otras herramientas. Sin embargo, el mayor potencial de GnuPG se obtiene cuando se integra con el programa de correo electrónico, pues en este caso permite enviar correos firmados y/o cifrados de forma casi transparente. Este documento tampoco describe este uso. El lector interesado puede probar a instalar el cliente de correo Thunderbird[3] y la extensión Enigmail[4].

A lo largo de este documento usaremos cuatro alumnos ficticios que llamaremos Alberto, Beatriz, Carlos y Damián (A, B, C, D). Alberto y Beatriz son dos buenos amigos. Carlos no es de fiar y siempre está maquinando jugarretas en contra de Alberto, aunque éste no lo sabe y le tiene por un buen amigo. Finalmente Damián es un buen amigo de Beatriz, y no conoce directamente a Alberto. Estos personajes servirán para plantear ejemplos de cómo la criptografía de clave pública y las herramientas GnuPG pueden ayudar a resolver muchos problemas.

[1] http://www.atc.uniovi.es/inf_superior/4atc/entregas/comun/pasos_a_seguir.pdf
[2] http://www.gnupg.org
[3] http://www.mozilla.com/thunderbird/
[4] http://enigmail.mozdev.org/
[5] http://creativecommons.org/licenses/by-nc-sa/2.5/

Índice

1. Introducción

La entrega de prácticas de ATC se realiza de forma electrónica a través de un formulario Web. En este formulario, la forma de identificar al alumno es mediante su NIF y su correo en Uniovi. Pero estos datos son fáciles de conocer por otras personas.

Esto no tendría por qué ser un problema de seguridad. De hecho, para algunos alumnos puede incluso resultar una ventaja el que la entrega de prácticas sea tan "flexible". Imaginemos que a Alberto se le ha estropeado el ordenador (o no tiene), y debe entregar la práctica con urgencia. Puede copiar el fichero a entregar en un disquete y dárselo a su amiga Beatriz, para que ella lo entregue desde su ordenador. Beatriz introducirá los datos de Alberto y su fichero en el formulario, por lo que a todos los efectos será como si Alberto mismo hubiera entregado la práctica.

Pero esto puede plantear otros problemas. Imaginemos que, en lugar de a Beatriz, Alberto confía su práctica a Carlos para que la entregue. Carlos odia secretamente a Alberto y decide gastarle una mala pasada. Antes de "subir" el archivo de Alberto a la Web, lo abre y lo modifica introduciendo graves errores. El profesor no tiene forma de saber quién ha enviado en realidad este fichero, ni si ha sido modificado.

Lo peor es que Alberto no podrá protestar ante el profesor para negar la autoría de esta práctica. O mejor dicho, el profesor no sabrá qué creer si ve una práctica entregada por Alberto, llena de errores, pero Alberto le dice que él no ha realizado esa práctica y que alguien la ha "cambiado". Podría ser una excusa para no reconocer los errores en su trabajo. En realidad ni siquiera hace falta que Alberto haya confiado la práctica a nadie para que esta situación pueda llegar a darse. Alberto podría "repudiar" su práctica (negar su autoría) con excusas como que tal vez alguien entró en el servidor Web y modificó las prácticas que los alumnos habían entregado. Ya puestos, incluso podría acusar al profesor de haber modificado su práctica. ¿Cómo podría defenderse el profesor de tal acusación? ¿O cómo podría Alberto demostrarla si fuera cierta?

Por otra parte, imaginemos otra posible preocupación que un alumno podría tener acerca de la seguridad. Carlos no quiere que nadie vea su práctica por miedo a que se la copien. En sus cuentas de usuario tiene todo protegido e incluso utiliza archivos ZIP protegidos con contraseña, por si alguien pudiera llegar a robárselos. Sin embargo en el momento de la entrega el archivo que "sube" al servidor no va protegido. Carlos tiene miedo de que la seguridad del servidor no sea suficiente y alguien pueda entrar en él para robar su práctica y copiarla.

Todos estos problemas se arreglan mediante la criptografía de clave pública. En este tipo de criptografía, cada alumno (y el profesor) tienen un par de claves (estas claves no las eligen libremente, son generadas automáticamente por un programa, como veremos). Una de estas claves es privada, y debe ser protegida con celo. La otra es pública y puede darse a conocer libremente a todo el mundo. Lo que se cifra con una clave sólo se puede descifrar con la otra.

Cuando Alberto cifra algo con su clave privada, cualquiera puede descifrarlo con la clave pública de Alberto. Esto no sirve para proteger los datos (cualquiera puede descifrarlos), pero sirve para que cualquiera pueda quedar convencido de que Alberto era el autor (ya que sólo él pudo tener acceso a la clave privada con la que fueron cifrados).

Por otro lado, los amigos de Alberto que tengan su clave pública podrán usar ésta para cifrar documentos. Ya que estos textos cifrados sólo se pueden descifrar con la correspondiente clave privada, sólo Alberto podrá descifrarlos. Por tanto esto proporciona un mecanismo para que cualquiera pueda enviar a Alberto información confidencial, sin necesidad de ponerse previamente de acuerdo en una contraseña.

2. Preparación de GnuPG

Las claves privada y pública de las que hablábamos en el punto anterior son en realidad datos binarios con un gran número de bits. No son claves en el sentido habitual de "contraseña". El usuario no elige estas claves (las genera aleatoriamente un programa) y tampoco debe memorizarlas ni teclearlas nunca, sino que se almacenan en archivos predeterminados. Los programas de cifrado acceden a estos archivos predeterminados cuando necesiten alguna de estas claves.

El archivo que contiene la clave privada, debe estar protegido. Si este archivo cayera en manos de otro, podría suplantar la personalidad de quien originalmente creó el par de claves, o descifrar los archivos dirigidos a él. Para evitar estos riesgos, el archivo que contiene la clave privada está a su vez cifrado mediante una frase de paso. Quien no conozca la frase de paso no podrá usar ese archivo, aún cuando consiguiera robárnoslo.

En cambio el archivo que contiene la clave pública no está sujeto a ninguna precaución. De hecho, es conveniente que este archivo sea lo más público posible. Hay quienes ponen este archivo en su página Web, por ejemplo. Otra posibilidad es subirlo a un "servidor de claves". Veremos cómo hacer esto último más adelante. Haciendo público este archivo posibilitamos que cualquiera pueda verificar nuestra firma.

En esta sección veremos cómo utilizar el software GnuPG para crear las claves, y para intercambiar la clave pública con nuestros contactos. En las restantes secciones veremos cómo usar GnuPG y las claves que hemos creado para firmar digitalmente documentos o para cifrarlos de modo que sólo una persona concreta pueda leerlos. La descripción de la herramienta está orientada exclusivamente a su uso para firmar los trabajos de la asignatura y para cifrarlos si se desea que nadie excepto el profesor pueda leerlos. Otros usos de GnuPG no son descritos aquí. No obstante se insta al lector a que investigue por su cuenta cómo integrar GnuPG con su cliente de correo favorito, para el envío de email firmado y/o cifrado.

2.1. GnuPG

GnuPG es un sistema de criptografía de clave pública, de código abierto y gratuito. Se puede instalar en Windows y en Linux. De hecho ya está instalado en las máquinas Linux de prácticas. Usaremos este sistema para crear nuestra pareja de claves (privada y pública), así como para realizar la firma digital y la verificación de la misma.

2.2. Creación de la pareja de claves

Para crear una pareja de claves única y personal, basta teclear en la línea de comandos de una máquina Linux el comando `gpg --gen-key`, el cual nos realizará una serie de preguntas que detallaremos a continuación. Asumiremos que el alumno que está creando su pareja de claves se llama "Alberto Nónimo", y que habitualmente usa como dirección de correo electrónico `a.nonimo@example.com`. Más adelante será posible añadir más direcciones de correo a su clave, si fuera necesario. Estas son las respuestas recomendadas:

```
$ gpg --gen-key
gpg (GnuPG) 1.4.1; Copyright (C) 2005 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

Please select what kind of key you want:
   (1) DSA and Elgamal (default)
   (2) DSA (sign only)
   (5) RSA (sign only)
Your selection?
```

Si Alberto sólo quisiera usar `gpg` para firmar digitalmente los trabajos, puede seleccionar cualquiera de las opciones anteriores. Sin embargo, `gpg` también puede usarse para *cifrar* los trabajos entregados de modo que nadie excepto el profesor pueda leerlos. Si elegimos la opción (1), podremos darle cualquiera de estos dos usos, por tanto es la opción más recomendable. Además, es la opción que `gpg` propone por defecto, de modo que basta pulsar Enter.

Seguidamente nos preguntará el tamaño de la clave. El valor propuesto por defecto (2048) es adecuado. Basta pulsar Enter de nuevo.

```
DSA keypair will have 1024 bits.
ELG-E keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
```

Lo siguiente que nos pregunta es el periodo de validez de la clave. Especificaremos 0 para que no caduque nunca (de nuevo basta pulsar Enter). Nos pide confirmación de esta respuesta, por lo que debemos teclear “y” (*yes*).

```
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) y
```

Seguidamente nos pedirá datos personales. Es necesario dar datos correctos. Estos datos quedarán asociados a nuestra firma digital, y serán los que se mostrarán cuando cualquiera intente verificar nuestra firma. Por tanto no se deben usar seudónimos ni direcciones de correo falsas. Pon tu verdadero nombre y apellidos, y la dirección email que consultes habitualmente (se recomienda utilizar la dirección de Uniovi). También nos pide un comentario que podemos dejar en blanco. Si pones algo en el comentario, aparecerá entre paréntesis tras el nombre y antes de la dirección de correo. Podría ser un seudónimo por el que nos conozcan los amigos.

Es preferible no usar acentos en el nombre ni en el comentario. En principio `gpg` no asume ninguna codificación especial, y se limitaría a almacenar los códigos que tu terminal le envíe. Así que según la configuración de la terminal los acentos quedarían almacenados como ISO-8859-1, o como UTF-8, por ejemplo. Mientras revises tus datos desde esa misma codificación no habrá problema; pero cuando alguien acceda a tus datos (por ejemplo, para verificar tu firma digital), puede que esté usando una terminal con una codificación diferente, por lo que no vería bien tu nombre.

Tras responder, nos presentará la información introducida y pulsaremos “o” (de “okay”) para continuar si todo está bien. En nuestro alumno de ejemplo, las respuestas serán:

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

Real name: Alberto Nonimo
Email address: a.nonimo@example.com
Comment:
You selected this USER-ID:
    "Alberto Nonimo <a.nonimo@example.com>"

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

El siguiente paso es muy importante. Estamos a punto de crear los archivos con las claves pública y privada, y para evitar que el archivo con la clave privada pueda caer en malas manos, se nos pide una frase de paso. Esta frase protegerá este archivo. Cada vez que intentemos firmar algo digitalmente con el programa `gpg`, se nos preguntará la frase de paso. De este modo, aún si el archivo con la clave privada es obtenido por otra persona, si no conoce la frase de paso no podrá usarlo para firmar nada.

La frase de paso debe ser larga, difícil de adivinar por los demás, pero a la vez fácil de recordar por nosotros ya que si la olvidamos no podremos firmar nada. Nunca uses datos de tu nombre o correo como parte de la frase de paso, pues sería fácil de adivinar. La frase puede contener espacios, pero no recomiendo que introduzcas acentos por las mismas razones que ya se han expuesto para el caso de los datos personales.

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

Enter passphrase:
Repeat passphrase:
```

Recuerda que la frase de paso es tu única protección frente a posibles usos fraudulentos de tu firma. De alguna forma, tu frase de paso "eres tú". Naturalmente nunca debes confiarla a nadie. No debería haber nadie cerca mientras la tecleas. Quizás estas recomendaciones sean excesivamente paranoides para el uso que le vamos a dar a esta clave, una mera entrega de un trabajo de una asignatura, pero son recomendaciones válidas para otros usos más importantes de las herramientas criptográficas (por ejemplo, la Fabrica Nacional de Moneda y Timbre emite certificados electrónicos que podemos usar para firmar algunos documentos oficiales, como la declaración de la Renta).

Una vez has escrito la frase de paso dos veces, `gpg` comienza a generar las claves. Como hemos dicho, estas claves son en realidad enormes números, que se almacenarán en binario. Para generar estos números, necesita generar números primos y muchos números aleatorios. Para garantizar que los números generados son realmente aleatorios y por tanto impredecibles, el programa `gpg` utiliza información que esté cambiando en la máquina de forma impredecible (lo que están tecleando otros usuarios, las instrucciones que ejecutan otros programas...) Por tanto `gpg` necesita que haya suficiente "actividad" en el sistema. A menudo este requisito no se cumple y `gpg` te invitará a que abras otras terminales y hagas diferentes cosas desde ellas (por ejemplo, ver el contenido de carpetas, compilar programas, ejecutarlos, etc.)

```
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
++++++++++.++++++++++++++++++++..+++++.+++++.+++++..++++++++++++
++++++++.++++++++++++++++++..++++++++++..+++++.+++++++++++++++
+++++..+++++++++++++++....>..+++++.............................
............+++++
```

Un comando que genera bastante actividad es "`ls -lR /`" que muestra los contenidos de todas las carpetas del disco. Prueba a ejecutarlo desde otra terminal y comprueba que `gpg` sigue avanzando. El programa `gpg` va escribiendo en pantalla puntos y cruces cuando logra suficiente aleatoriedad. Al cabo de un tiempo, terminará la generación de las claves, y mostrará información sobre las mismas.

```
+++++++++++++++++++++++++++++++++.++++++++++++++++++.+++++.+
++++++++++++++..+++++.++++++++++.++++++++++.+++++...++++++++++++
++++++++++++++++++.+++++.+++++>+++++..+++++>+++++............>.+
++++.................................................+++++^^^^

gpg: key E0CA9E25 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   2  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 2u
pub   1024D/E0CA9E25 2005-12-22
      Key fingerprint = CEDB 1401 1B26 8379 D969  7BDA F95E 1F18 E0CA 9E25
uid                  Alberto Nonimo <a.nonimo@example.com>
sub   2048g/88885614 2005-12-22
```

En particular, puede interesarte apuntar el número identificado como *key*, que en este caso es `E0CA9E25`, aunque siempre puedes averiguarlo más adelante si lo necesitas, como veremos.

Los dos ficheros que `gpg` acaba de crear están en la carpeta `.gnupg` de tu carpeta raíz. Si entras en esta carpeta verás varios ficheros. Uno de ellos, llamado `secring.gpg` contiene tu clave secreta. Otro llamado `pubring.gpg` contiene de momento tu clave pública, pero también podrás añadir a él las claves públicas de tus amigos y compañeros, y de este modo podrás verificar sus firmas, o enviarles mensajes cifrados que sólo ellos podrán descifrar.

2.3. Creación del certificado de revocación

Es posible que en el futuro Alberto desee invalidar las claves que acaba de crear. Por ejemplo, porque tenga fundadas sospechas de que alguien le ha pillado la frase de paso y el fichero con la clave privada y podría usarlo para hacerse pasar por él. O un caso más trivial, que haya olvidado la frase de paso y ya no pueda usar esta clave para firmar nada.

Sea cual sea el caso, si en el futuro Alberto desea "dar de baja" esta clave, necesitará un certificado de revocación. El mejor momento para crear este certificado es ya mismo. Si lo deja para más adelante, cabe la posibilidad de que olvide su frase de paso ¡y entonces ya no podrá tampoco crear el certificado!

La creación del certificado es muy sencilla. Basta que escriba:

```
$ gpg --gen-revoke -a --output anulacion-clave.asc E0CA9E25
```

donde `E0CA9E25` es la *key* de Alberto, que puede averiguar con el comando `gpg --list-secret-keys`. GnuPG nos preguntará la razón de la revocación. Una buena respuesta puede ser que la clave ya no se usa (aunque todavía no sabemos cuál será la razón real, ésta es lo bastante genérica). Nos pedirá la frase de paso y generará el certificado solicitado.

El resultado será un fichero llamado `anulacion-clave.asc`, que Alberto inmediatamente deberá transferir a un almacén seguro (por ejemplo, un lápiz USB, un diskette o un CD-ROM). Hay quien recomienda incluso imprimir este archivo en papel (son unas pocas líneas); si el diskette o CD-ROM resultaran ilegibles en el momento en que se necesite el certificado, siempre se puede "copiar a mano" desde el papel. Una vez copiado en lugar seguro, debe borrarse de la cuenta. Este fichero no debería estar accesible para nadie, excepto para Alberto, ya que con él es posible anular sus claves.

En la sección 3.4 se explicará cómo usar este fichero para anular (*revocar*) una clave, y las consecuencias que tiene.

2.4. Difusión de la clave pública

La clave pública se la podemos comunicar a nuestros amigos, o incluso a nuestros enemigos, sin que ello entrañe ningún riesgo para nosotros. Cuando una persona cualquiera obtenga tu clave pública, lo único que podrá hacer con ella será:

- Verificar archivos que tú hayas firmado digitalmente (con tu clave privada). Es decir, comprobar que la firma es correcta, que eres tú el que has firmado y que el archivo no ha sido alterado desde su firma.
- Cifrar archivos, de modo que el resultado sólo podrá descifrarse con tu clave privada (y por tanto sólo tú podrás descifrar estos archivos, ya que eres el único que tiene acceso a esta clave).

Como vemos, ninguno de estos dos usos supone ningún riesgo para ti.

Para que un amigo pueda verificar tu firma, o bien enviarte mensajes cifrados que sólo tú podrás descifrar, debes comunicarle a este amigo tu clave pública.

Supongamos que Alberto quiere comunicar su clave pública a su amiga Beatriz. Los pasos a seguir serán:

1. Usando `gpg`, Alberto generará un fichero ASCII que contiene su clave pública.
2. Alberto le hará llegar de alguna forma este fichero a Beatriz (se lo puede dar en mano, enviarlo por email, "colgarlo" en una página Web, o "publicarlo" en un servidor de claves).

3. Beatriz añade la clave recibida de Alberto a su "anillo"[6].

4. Beatriz deberá verificar que el archivo que ha recibido es realmente el que Alberto ha creado en el paso 1. Si ha recibido el archivo directamente en un diskette de manos de Alberto, la verificación ya está hecha (asumimos que Beatriz puede reconocer físicamente a Alberto). Si lo ha recibido por cualquier otro medio, Beatriz deberá verificar las "huellas dactilares", como veremos más adelante.

5. Una vez verificado que la clave de Alberto es la buena, Beatriz podrá estampar su propia "firma" en esta clave, como garantía. Al firmar una clave, Beatriz está "certificando" que esa clave pertenece a quien dice pertenecer. Este certificado será útil a otras personas que no conozcan a Alberto, pero se fíen de Beatriz.

Detallaremos seguidamente estos cinco pasos:

2.4.1. Alberto genera un fichero ASCII conteniendo la clave pública

Alberto genera una versión ASCII de su clave con el comando `gpg -a --export`. Conviene usar la opción `--output` para indicar el nombre del fichero ASCII que queremos obtener, pues de lo contrario la clave será volcada en la pantalla, lo que no es muy útil. Además, si Alberto tiene varias claves, deberá especificar la *key* asociada a la clave que quiere exportar (de momento Alberto sólo tiene una clave, pero pronto tendrá muchas más a medida que añada a su anillo las claves de sus amigos). Alberto puede averiguar su *key* en cualquier momento con el comando `gpg --list-keys`.

Por tanto, el comando completo sería en este caso:

```
$ gpg -a --export --output clave-publica-alberto.asc E0CA9E25
```

De este modo Alberto obtiene el fichero llamado `clave-publica-alberto.asc` (el nombre lo ha elegido él, puede ser cualquier otro), y contiene la clave pública asociada a la *key* E0CA9E25 (que es la de Alberto). Este archivo contiene sólo caracteres ASCII, por lo que puede ser examinado con cualquier editor.

2.4.2. Alberto comunica a Beatriz la clave pública obtenida

Alberto tiene varias opciones:

- Copiar el fichero `clave-publica-alberto.asc`, que acaba de generar, en un diskette o CD-ROM y darle en mano este disco a Beatriz.
- Enviarle a Beatriz este fichero por correo electrónico (puede ser un adjunto, o incluso puede ir copiado y pegado en un mensaje, ya que es texto ASCII).
- Si Alberto tiene una página Web personal, subir este archivo a su Web y poner un enlace al mismo, con un texto como "Esta es mi clave pública GPG". Puede notificarlo por email u otros métodos a sus amigos para que se descarguen el archivo.
- Finalmente, Alberto puede "publicar" su clave en un servidor de claves, con lo que cualquiera podrá buscarla y descargarla cuando le sea necesario. Esto tiene el inconveniente de que su dirección de correo electrónico también se hace pública, y tal vez Alberto no quiera esto.

La última de estas opciones puede realizarse directamente con el programa `gpg`. Basta que Alberto escriba la orden siguiente:

[6] En GnuPG se denomina "anillo" al conjunto de claves que un usuario tiene. Normalmente en este anillo estará su clave privada y pública, y las claves públicas de todos sus amigos. El término "anillo" proviene de que en inglés, clave y llave se dice igual, y por tanto el anillo de claves es un "anillo de llaves" o "llavero"

```
$ gpg --keyserver pgp.mit.edu --send-keys E0CA9E25
gpg: sending key E0CA9E25 to hkp server pgp.mit.edu
```

Como vemos, la clave es enviada a una máquina llamada `pgp.mit.edu` (que hemos especificado con la opción `--keyserver`). Esta máquina también es accesible desde un navegador Web y en este caso se nos presenta un interfaz en el que podemos buscar las claves públicas de otras personas, si conocemos su *key*, su nombre, o su email. La mayoría de los servidores de claves importantes están conectados entre sí, de modo que enviando la clave a uno cualquiera de ellos, en poco tiempo estará replicada en todos. Antes de decidirte a subir tu clave pública a un servidor de claves, ten en cuenta que una vez subida ya no se puede borrar del servidor. Podrás *revocarla* (ver sección 3.4), lo que impedirá que se pueda seguir usando la clave, pero aún así seguirá estando pública en el servidor para siempre.

2.4.3. Beatriz importa la clave recibida de Alberto

Beatriz recibe una copia de `clave-publica-alberto.asc` por cualquiera de los métodos anteriores, y ejecuta el siguiente comando:

```
$ gpg --import clave-publica-alberto.asc
gpg: key E0CA9E25: public key "Alberto Nonimo <a.nonimo@example.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1
```

Como vemos en la salida de `gpg`, el archivo `.asc` contenía una clave pública, correspondiente a "Alberto Nonimo", y con *key* igual a `E0CA9E25`. Ahora Beatriz tiene esta clave almacenada en su "anillo de claves" (fichero `pubring.gpg`), y por tanto ya puede verificar documentos firmados por Alberto (y también, como veremos más adelante, puede generar documentos que sólo Alberto podrá leer).

Si Alberto no le ha hecho llegar la clave, sino que la ha "subido" a un servidor de claves, Beatriz podrá obtener una copia de forma muy sencilla, ejecutando el comando:

```
$ gpg --keyserver pgp.mit.edu --search-key 0xE0CA9E25
gpg: searching for "0xE0CA9E25" from hkp server pgp.mit.edu
(1)     Alberto Nonimo <a.nonimo@example.com>
          1024 bit DSA key E0CA9E25, created: 2005-12-22
Enter number(s), N)ext, or Q)uit >
```

Vemos cómo efectivamente la *key* es encontrada en el servidor. Nos muestra la lista de los posibles usuarios (buscando por la *key* sólo puede aparecer uno, pero si buscamos por el nombre, apellidos o email pueden aparecer muchas respuestas). Pulsando el número de una de ellas, esta clave se "importará", como si Alberto nos hubiese dado el fichero con su clave pública. En este caso, se pulsaría el 1:

```
Enter number(s), N)ext, or Q)uit >
gpg: requesting key E0CA9E25 from hkp server pgp.mit.edu
gpg: key E0CA9E25: public key "Alberto Nonimo <a.nonimo@example.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1
```

Pero queda un **paso importante** por dar antes de que Beatriz pueda usar la clave pública de Alberto. De hecho es el paso más importante, y un posible punto débil de todo el sistema GnuPG (o de cualquier otro basado en claves públicas): Beatriz debe estar absolutamente segura de que la clave que acaba de añadir a su anillo es exactamente la que Alberto había exportado. ¿Por qué no habría de ser así? podrías preguntarte. En el siguiente punto abordamos este tema.

2.4.4. Beatriz verifica con Alberto la clave recibida

Imagina que Carlos quiere hacerse pasar por Alberto ante Beatriz. Carlos usaría `gpg` para crear una pareja de claves, pero daría los datos de Alberto. Así crearía un fichero `clave-publica-alberto.asc` que, naturalmente, es diferente del que Alberto había creado (pues `gpg` nunca creará claves iguales), pero que no obstante contiene información acerca de "Alberto Nonimo". Seguidamente Carlos envía este archivo por email a Beatriz, y falsifica el remite del correo para que aparentemente provenga de `a.nonimo@example.com` (la dirección de Alberto). Beatriz recibe este correo y, confiada, añade la clave que acaba de recibir a un lista de claves públicas (usando `gpg --import` como hemos visto).

Entonces ocurrirá que Carlos ya podrá firmar documentos con la clave falsa que había creado y enviárselos a Beatriz, la cual usará `gpg` para verificar la firma y verá que todo es correcto, ya que en realidad se están usando las parejas de claves privada/pública adecuadas. Además, los mensajes que Beatriz envíe cifrados con la clave pública de Alberto (o la que ella cree de Alberto), podrán descifrarse por Carlos, pues tiene la clave privada correcta (es decir, la pareja de la clave pública que Beatriz ha usado).

¿Cómo puede estar segura Beatriz de que la clave que acaba de recibir proviene de Alberto? Sólo si Alberto se la ha entregado "en mano" podrá estar absolutamente segura. Si lo recibe por email, o se lo "baja" de la Web de Alberto, o de un servidor de claves, nunca podrá estar del todo segura de si es Alberto quien realmente ha "subido" esa clave. Ni siquiera llamando a Alberto por teléfono y preguntándoselo, ya que cabe la posibilidad de que, después de que Alberto haya "subido" el archivo, Carlos haya subido otra copia que sobreescriba la de Alberto.

Por suerte GnuPG preve otro método para verificar la clave recibida, que no obliga a entregarla "en mano". Consiste en comprobar si las "huellas dactilares" de las claves coinciden.

Para ello, una vez que Beatriz ha aceptado la clave con el comando `gpg --import`, ejecuta el comando `gpg --fingerprint` en la forma siguiente:

```
$ gpg --fingerprint E0CA9E25
pub   1024D/E0CA9E25 2005-12-22
      Key fingerprint = CEDB 1401 1B26 8379 D969  7BDA F95E 1F18 E0CA 9E25
uid                  Alberto Nonimo <a.nonimo@example.com>
sub   2048g/88885614 2005-12-22
```

Como vemos, al comando `gpg` hay que darle en este caso la *key* que pretendo comprobar. Alternativamente, en lugar de la *key* se le puede dar cualquier campo del nombre, por ejemplo *Alberto*, pero en este caso si hay más de una clave importada con ese nombre, se mostrarán todas las que concuerden.

El caso es que el comando me muestra lo que se conoce como la "huella dactilar" (*fingerprint*) de la clave pública, que es una serie de 40 cifras hexadecimales. Esta huella le servirá a Beatriz para convencerse de que el archivo se lo he enviado realmente Alberto. Para ello Beatriz llamará por teléfono a Alberto, o quedará con él en persona (canales de comunicación seguros), y le preguntará por la "huella dactilar" de su clave.

Alberto ejecutará en su ordenador el mismo comando `gpg --fingerprint` que acabamos de ver, para encontrar su propia huella dactilar, y le leerá el resultado por teléfono a Beatriz, o lo imprimirá en un papel y se lo dará a Beatriz en mano (canales seguros). Si ambas huellas dactilares concuerdan, es que se trata de la misma clave y por tanto Beatriz puede estar segura de que tiene la clave pública de Alberto y no la de un impostor[7].

2.4.5. Beatriz firma la clave de Alberto, una vez verificada

Una vez que Beatriz ha quedado convencida de la autenticidad de la clave que acaba de recibir (¡pero no antes!), la puede "firmar". Para que Beatriz pueda firmar las claves públicas que recibe de sus amigos, necesita a su vez tener su propio par de claves privada/pública, que habrá generado igual que explicamos para Alberto.

[7] En el fondo, lo mismo puede lograrse si Alberto lee por teléfono el contenido del fichero `clave-publica-alberto.asc`, pero esto es mucho más largo e incómodo. La "huella dactilar" no es más que un *hash* de ese fichero.

Cuando Beatriz firma una clave, significa que se ha ocupado personalmente de verificar la autenticidad de la misma (ya sea porque ha recibido "en persona" la clave pública, o porque ha verificado su "huella dactilar"). Uno no debe firmar claves a la ligera. Al firmar una clave estás "garantizando" personalmente que esa clave pública pertenece a la persona que dice pertenecer.

Beatriz usará el siguiente comando para firmar la clave de Alberto:

```
$ gpg --sign-key E0CA9E25

pub  1024D/E0CA9E25  created: 2005-12-22  expires: never       usage: CS
                     trust: unknown       validity: unknown
sub  2048g/88885614  created: 2005-12-22  expires: never       usage: E
[ unknown] (1). Alberto Nonimo <a.nonimo@example.com>


pub  1024D/E0CA9E25  created: 2005-12-22  expires: never       usage: CS
                     trust: unknown       validity: unknown
 Primary key fingerprint: CEDB 1401 1B26 8379 D969  7BDA F95E 1F18 E0CA 9E25

     Alberto Nonimo <a.nonimo@example.com>

Are you sure that you want to sign this key with your
key "Beatriz Nina <b.nina@example.com>" (21431905)

Really sign? (y/N)
```

Como vemos, `gpg` nos muestra información sobre la clave que Beatriz está a punto de firmar y le pide confirmación. Insistimos, sólo si Beatriz está completamente segura de que esa es la clave real de Alberto, responderá "y".

En ese momento, `gpg` necesitará acceder a la clave privada de Beatriz, puesto que ésta es la que se usa cuando Beatriz firma cualquier cosa. Cuando Beatriz creó las claves, asignó una frase de paso para proteger su clave privada. Cada vez que `gpg` necesite acceder a esta clave (cada vez que Beatriz firme algo), le preguntará esta frase.

```
You need a passphrase to unlock the secret key for
user: "Beatriz Nina <b.nina@example.com>"
1024-bit DSA key, ID 21431905, created 2005-12-22

Enter passphrase:
```

Si Beatriz responde con la frase de paso correcta, su firma se añadirá a la copia que Beatriz tiene de la clave privada de Alberto. Si Beatriz usa el comando `gpg --export -a` para crear una versión ASCII de la clave de Alberto, esta versión llevará incorporada la firma de Beatriz. Ahora Beatriz puede enviarle la clave firmada a otras personas, por ejemplo a Damián. Damián apenas conoce a Alberto, por lo que prefiere no llamarle por teléfono para pedirle la "huella dactilar". Sin embargo es un buen amigo de Beatriz, y ya tiene la clave pública de ésta, y confía en ella. Por tanto, al recibir la clave de Alberto firmada por Beatriz, podrá confiar también en que esta clave es correcta sin necesidad de verificarlo personalmente con Alberto.

3. Uso de GnuPG

3.1. Firma digital

Alberto puede ya "firmar" digitalmente cualquier documento. Por ejemplo, supongamos que ha escrito un poema y se lo quiere enviar a Beatriz. Quiere asegurarse de que el poema llega íntegro, y no es manipulado en el camino (no se fía demasiado del correo electrónico), y quiere que Beatriz pueda verificar que él es el verdadero autor del poema (o al menos, el que lo ha firmado).

Basta que Alberto ejecute el siguiente comando (asumiendo que el poema está almacenado en un fichero llamado poema.doc, que pudo haber sido creado con Microsoft Word, por ejemplo):

```
$ gpg --sign poema.doc
```

Para firmar el documento, gpg necesita acceso a la clave privada de Alberto, y ya que este ha protegido esta clave un una frase de paso (como debe ser, para evitar que otro pueda firmar en su lugar), gpg le pedirá en ese momento esta frase. Si Alberto responde correctamente, se creará un nuevo fichero, de igual nombre que el original, pero al que se le ha añadido la extensión .gpg. En este caso el nuevo fichero se llama poema.doc.gpg.

Este archivo ya no es directamente legible. Si el original poema.doc contenía texto visible (ASCII), el resultado poema.doc.gpg contiene caracteres aparentemente aleatorios, muchos de ellos no ASCII. Si el original había sido escrito con Microsoft Word, el resultado ya no se puede abrir en Word (ni siquiera aunque le quitáramos la extensión .gpg). La única forma de "reconstruir" este archivo es disponiendo del programa gpg y de la clave pública de Alberto.

Alberto envía este documento por email a Beatriz, y ésta al recibirlo tan solo debe ejecutar el comando gpg sobre el fichero recibido. No es necesario especificar parámetros. gpg restaurará el fichero original poema.doc, y además nos informará de quién ha firmado el documento y de si la firma es correcta.

```
$ gpg poema.doc.gpg
gpg: Signature made vie 23 dic 2005 17:21:43 CET using DSA key ID E0CA9E25
gpg: Good signature from "Alberto Nonimo <a.nonimo@example.com>"
```

Beatriz tendrá ahora en su carpeta un nuevo fichero poema.doc que es el originalmente escrito por Alberto, además de la certeza de que Alberto es el firmante y que el fichero poema.doc no ha sido modificado por nadie desde que Alberto lo firmó.

Alberto puede incluso poner el archivo poema.doc.gpg en su página Web. Cualquiera puede descargarlo, y usar gpg para reconstruir el archivo original, incluso sin tener la clave pública de Alberto. Supongamos que Emilio, que no conoce de nada a Alberto, se descarga de su página el fichero poema.doc.gpg, e intenta ejecutar gpg sobre él. Encontrará la respuesta siguiente:

```
$ gpg poema.doc.gpg
gpg: Signature made vie 23 dic 2005 17:21:43 CET using DSA key ID E0CA9E25
gpg: Imposible comprobar la firma: Clave pública no encontrada
```

En este caso, el archivo original poema.doc también es de todas formas reconstruido, y se informa al usuario de que hay una firma correspondiente a la *key* E0CA9E25, lo cual ya es un indicio de que el archivo no ha sido manipulado desde que se firmó (pues cualquier manipulación haría imposible la reconstrucción posterior). Sin embargo, ya que Emilio no dispone de la clave pública de Alberto, gpg no puede determinar a quién corresponde la *key* E0CA9E25, ni tampoco si el archivo fue alterado. Emilio puede tratar de importar la clave del servidor, por si Alberto la hubiera subido allí. Intentará el comando siguiente:

```
$ gpg --import-key 0xE0CA9E25
```

Si Alberto había subido su clave pública al servidor de claves, el comando anterior la encontrará y tratará de importarla en el anillo de claves de Emilio. Una vez Emilio acepte la importación, podrá verificar de nuevo la firma de poema.doc.gpg. En este caso gpg le informará de quién ha firmado el archivo (le dará su nombre y email), pero, ya que la clave pública que acaba de ser importada, no está firmada por nadie conocido, gpg también le dará un aviso:

```
$ gpg poema.doc.gpg
El fichero ‘poema.doc’ ya existe. Overwrite? (y/N) y
gpg: Signature made vie 23 dic 2005 17:21:43 CET using DSA key ID E0CA9E25
gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: Good signature from "Alberto Nonimo <a.nonimo@example.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: CEDB 1401 1B26 8379 D969  7BDA F95E 1F18 E0CA 9E25
```

Dicho en otras palabras, “Este archivo no ha sido alterado desde que fue firmado. La firma dice pertenecer a Alberto Nónimo, pero no estamos seguros de si será realmente él quien ha firmado”. Si Emilio estuviera muy interesado en confirmar este punto, debería tratar de ponerse en contacto con Alberto (quizás a través del email) para concertar una cita en la que poder validar las huellas dactilares (no por email).

3.1.1. Firma en archivo aparte

El método de firmar que acabamos de explicar tiene un inconveniente. Supongamos que Alberto ha escrito un manual (digamos, por ejemplo, que se llama `GnuPG-para-dummies.pdf` que quiere publicar en su Web. Quiere que este texto sea legible para el mayor número posible de personas y a la vez quiere que cualquier persona interesada en comprobar la autoría del documento, pueda hacerlo usando GnuPG.

Si se limita a firmar el documento con la opción `--sign` explicada en el punto anterior, el resultado será un nuevo archivo llamado `GnuPG-para-dummies.pdf.gpg` que muchos usuarios no sabrán manejar. Para reconstruir el archivo original, necesitarán tener instalado GnuPG, y saber usarlo (lo que, como estamos viendo, no es trivial). Alberto teme que debido a estas trabas pocas personas sean capaces de leer su documento.

Para este caso existe otra solución. Alberto “colgará” en su Web el documento PDF sin firmar, y la firma la pondrá en un archivo aparte. Para ello ejecuta el siguiente comando:

```
$ gpg -a --detach-sign GnuPG-para-dummies.pdf
```

Tras pedirle su frase de paso, `gpg` creará un nuevo fichero llamado `GnuPG-para-dummies.pdf.asc` que contiene la firma digital del primero (la opción `-a` es para que el archivo que contiene esta firma tenga formato ASCII, y así evitar posibles problemas al subirlo con ftp). Ahora Alberto subirá a su Web el archivo `pdf`, que todos podrán leer sin problemas, y el archivo `.pdf.asc` con instrucciones de cómo usarlo, para que los usuarios que quieran puedan verificar la autoría del documento. Para esto basta que ejecuten:

```
$ gpg GnuPG-para-dummies.pdf.asc
```

Si en la misma carpeta tienen un fichero llamado `GnuPG-para-dummies.pdf`, se asumirá que éste es el nombre del fichero cuya firma estamos comprobando. Si no existe, se nos pedirá el nombre del fichero original. El tipo de resultado que podemos obtener es el mismo que el visto en la sección anterior (es decir, si no tenemos la clave pública de Alberto, se nos indicará tan solo la *key* del firmante, si la tenemos se nos indicará el nombre y email del firmante y se nos avisará de que puede no ser fiable si su clave la tenemos sin firmar).

3.2. Cifrado de datos con GnuPG

3.2.1. Cifrado para un destinatario concreto

Si Alberto quiere enviar un archivo a Beatriz de forma que sólo ella pueda leerlo, GnuPG también lo hace posible. Sin GnuPG posiblemente Alberto intentaría enviarle a Beatriz un archivo `.zip` protegido con contraseña, pero también tendrá que hacerle llegar de algún modo la contraseña. En cambio usando GnuPG, no es necesario que Alberto y Beatriz compartan secreto alguno.

Basta que Alberto cifre el archivo usando la clave pública de Beatriz, para lo que usa el comando siguiente:

```
$ gpg --encrypt -r Beatriz poema.doc
```

La opción `--encrypt` es la que solicita el cifrado del documento `poema.doc`, y la opción `-r` permite especificar quién podrá leer el documento. En este caso Alberto ha especificado "Beatriz" por su nombre, ya que es la única clave pública que tiene perteneciente a alguna persona llamada Beatriz. Si hubiera varias, habría sido mejor usar la *key* o el email de esa persona. Observa que para realizar esta operación no se pide ninguna frase de paso, ya que no se está usando la clave privada de Alberto, sino la clave pública de Beatriz. Cualquiera puede cifrar un archivo de esta forma, pero sólo Beatriz podrá descifrarlos.

El resultado será un archivo llamado `poema.doc.gpg`. Este archivo puede hacérselo llegar a Beatriz usando cualquier canal, aunque sea inseguro (correo electrónico, publicarlo en una Web, etc.) Para descifrar el archivo y recuperar el documento original será necesario tener el software GnuPG y la clave privada de Beatriz. En teoría sólo ella tiene ambas cosas y por tanto sólo ella podrá acceder al documento.

El comando que Beatriz debe usar para descifrar el documento es simplemente `gpg` sobre el nombre del fichero recibido. Beatriz sí que tendrá que dar su frase de paso para poder continuar, ya que para descifrar se usa su clave privada.

```
$ gpg poema.doc.gpg
You need a passphrase to unlock the secret key for
user: "Beatriz Nina <b.nina@example.com>"
2048-bit ELG-E key, ID 84C60FD5, created 2005-12-22 (main key ID 21431905)

Enter passphrase: **********************
gpg: encrypted with 2048-bit ELG-E key, ID 84C60FD5, created 2005-12-22
      "Beatriz Nina <b.nina@example.com>"
```

Si una persona distinta de Beatriz intenta descifrar el archivo, se encontrará la siguiente respuesta:

```
gpg: encrypted with ELG-E key, ID 84C60FD5
gpg: descifrado fallido: clave secreta no disponible
```

3.2.2. Cifrado para varios destinatarios concretos

Es posible cifrar un documento para varios destinatarios, de modo que cualquiera de ellos podría descifrarlo. Basta especificar cada posible destinatario con la opción `-r`. Por ejemplo, Beatriz usaría el siguiente comando para cifrar un archivo que Alberto y Damián puedan leer:

```
$ gpg --encrypt -r Alberto -r Damian fichero.txt
```

El archivo resultante será mayor que si se cifra para un solo destinatario, pues en el fondo lo que contiene son dos versiones cifradas, una para Alberto y otra para Damián. Por otro lado, ten en cuenta que Beatriz no podrá descifrar el archivo que acaba de cifrar ella misma, pues no está incluida en la lista de "destinatarios".

3.2.3. Cifrado con clave simétrica

GnuPG también soporta un método de cifrado que no requiere claves privadas ni públicas, sino que se basa en suministrar una contraseña en el momento de cifrar el archivo. Cualquiera que conozca esta contraseña podrá descifrarlo después.

Esto puede ser conveniente si quieres que el archivo pueda ser descifrado por muchas personas, algunas de las cuales ni siquiera conoces aún. Cuando las conozcas, les harás llegar la contraseña de alguna forma segura (diciéndosela en persona, o enviándola en un mensaje cifrado por el método de clave privada/pública). Para cifrar un archivo por este método, debes usar `gpg` con la opción `--symmetric` (el nombre proviene del hecho de que la misma contraseña se usa para cifrar y para descifrar). La idea en el fondo es la misma que la que se usa para proteger un archivo ZIP con contraseña, pero los métodos de cifrado usados por GnuPG son especialmente fuertes (aunque, por supuesto, una contraseña demasiado simple y fácil de adivinar, da al traste con cualquier método de cifrado simétrico).

3.3. Cifrar y firmar

Las acciones de cifrar y firmar pueden combinarse sobre un mismo fichero. Por ejemplo, Alberto puede ejecutar el siguiente comando:

```
$ gpg --sign --encrypt -r Beatriz poema.doc
```

Alberto deberá introducir su frase de paso, ya que al firmar un archivo se está usando su clave privada. El resultado `poema.doc.gpg` estará a la vez firmado por Alberto y cifrado para Beatriz.

Cuando Beatriz lo reciba, simplemente ejecutando `gpg` sobre él (sin parámetros), podrá descifrarlo y a la vez verificar que la firma es de Alberto. Para descifrarlo se requiere la clave privada de Beatriz, por lo que ésta deberá introducir también su frase de paso. Para verificar la firma sólo se requiere la clave pública de Alberto, que Beatriz ya tiene.

Si cualquier otra persona (por ejemplo Damián o Carlos) obtienen una copia de este archivo, no podrán hacer nada con él. Al ejecutar `gpg` lo único que averiguarán es que el archivo está cifrado para Beatriz, pero no lo podrán descifrar y ni siquiera podrán saber quién lo firma.

3.4. Revocación

Si Alberto olvida la frase de paso con la que protegió su clave privada, ya no podrá firmar nada más con esa clave. Y tampoco podrá descifrar los mensajes cifrados que le envíen sus amigos. En este caso lo mejor para él será crear un par de claves nuevas y tener más cuidado la próxima vez.

También puede darse el caso de que alguien robe la clave a Alberto y se las arregle para averiguar la frase de paso (por ejemplo, espiando a Alberto mientras la teclea). Si Alberto tiene sospechas de que esto ha ocurrido, no debería seguir usando esas claves y también debería generar un par nuevo.

En todo caso ¿qué pasa con las claves antiguas? Sus amigos seguirán utilizándolas para enviarle mensajes cifrados. Terceras personas que no conocen directamente a Alberto seguirían utilizándolas para verificar documentos firmados por Alberto. Alberto debería poder notificar a todas estas personas que estas claves ya no son válidas. O mejor dicho, que no son válidas a partir de una fecha (puesto que los documentos firmados por Alberto antes de esa fecha sí que siguen siendo válidos).

GnuPG prevé un mecanismo para tratar con este problema, que se denomina la “revocación” de claves. El propietario de una clave puede revocarla en cualquier momento. Esta operación afecta a la clave pública, a la que se añade la información de que “ha sido revocada”. La clave puede seguir siendo pública y puede usarse para verificar firmas, pero ya no podrá usarse para cifrar datos, y además se considerarán no válidas todas las firmas realizadas en fecha posterior a la revocación.

Estos son los pasos que debe seguir Alberto para revocar una clave y advertir a sus amigos de este hecho.

- Alberto debe tener un certificado de revocación. Idealmente este certificado lo habría generado a la vez que la pareja de claves, como se explicó en la sección 2.3. Si no lo hizo entonces, aún puede hacerlo ahora, siempre que recuerde la frase de paso. Si tampoco recuerda la frase de paso, no hay solución posible. Supongamos que Alberto tiene el certificado en un fichero llamado `anulacion-clave.asc`.
- Este certificado contiene datos que en castellano dirían “la clave pública de Alberto Nónimo ha sido revocada y no debe usarse más para cifrar, ni para verificar firmas a partir de la fecha de su publicación”. Para hacer efectiva la revocación, el certificado debe añadirse a la clave pública de Alberto. Alberto lo hará sobre su propia copia mediante el comando:

  ```
  $ gpg --import anulacion-clave.asc
  ```

- El problema es que hay muchas copias de esta clave pública (distribuidas entre todos sus amigos), y todas deben ser actualizadas de la misma forma. Alberto debe enviar a cada uno de sus amigos una copia del certificado de revocación y pedir a sus amigos que ejecuten el comando anterior.
- Finalmente, si Alberto había "subido" en su momento su clave pública a un servidor de claves, deberá ahora subir otra vez la versión revocada. Así, cuando terceras personas quieran descargar del servidor la clave pública de Alberto, encontrarán una versión revocada, que sólo servirá para verificar firmas anteriores a la revocación (esto es, a la fecha en que Alberto "subió" la versión revocada).

3.5. Temas avanzados

3.5.1. Añadir o quitar correos electrónicos de la clave

Cuando Alberto creó su clave, tuvo que dar una dirección de correo electrónico, que queda asociada a su firma. Esto no es imprescindible, pero es útil para que alguien que lea un documento firmado por Alberto pueda ponerse en contacto con él fácilmente. Pero la principal utilidad de incorporar el email en la clave pública aparece cuando GnuPG se integra con el gestor de correo. En este caso, cuando Beatriz quiera enviar un correo cifrado a Alberto, simplemente deberá elegir en su cliente la opción "Enviar cifrado" y no necesitará especificar con qué clave lo ha de cifrar. El gestor de correo mirará quién es el destinatario del mensaje que Beatriz acaba de escribir, y obtendrá el email de destino. Seguidamente buscará en el anillo de claves públicas de Beatriz hasta encontrar una en la que figure ese email, y ésta será la clave pública que usará para cifrar el mensaje. Esta automatización es especialmente útil si el correo tiene varios destinatarios. En ese caso se cifrará el mensaje con las claves públicas de todos ellos, sin necesidad de especificar cuáles son.

Si Alberto tiene varias direcciones de email, sería deseable que todas ellas aparecieran listadas en su clave pública. De este modo, cuando Beatriz le escriba un correo a cualquiera de ellas, el método automático de "Enviar cifrado" seguirá funcionando.

Para añadir más direcciones de correo a una clave pública, se ha de usar el comando `gpg --edit-key`, seguido de la *key* correspondiente a la clave pública que se quiere editar. Esto nos llevará a un interfaz de comandos desde el cual se pueden dar diferentes órdenes que afectarán a dicha clave. En particular, el comando `adduid` permite añadir nuevas direcciones de correo (y "alias" para el nombre de Alberto).

También es posible "dar de baja" una dirección de correo electrónico que figure en la clave pública pero ya no se use, aunque esto no tan sencillo. No basta con eliminar el "uid" correspondiente, sino que hay que "revocarlo". Tanto el añadido de nuevos *uid* como su revocación se sale de los objetivos de este documento. El lector interesado puede consultar los detalles en el manual de GnuPG[8].

3.5.2. La red de confianza

Imaginemos el siguiente escenario: Beatriz ha creado un par de claves (pública y privada) y ha enviado la clave pública a sus dos amigos Alberto y Damián. Tanto Alberto como Damián conocen pesonalmente a Beatriz (aunque no se conocen entre sí), y cada uno por su lado se ocupa de verificar la huella dactilar de clave que Beatriz les envió, tras lo cual cada uno firma la copia de la clave pública de Beatriz que tiene en su poder.

Tenemos ahora tres versiones de la clave pública de Beatriz. Una, la que posee la propia Beatriz, que no está firmada por nadie[9]. Otra la que posee Alberto, que lleva la firma de Alberto, y otra la que posee Damián, que lleva la firma de Damián.

Sería interesante "juntar" estas tres versiones. Para ello tanto Alberto como Damián deberían "exportar" la clave de Beatriz que tienen en su poder, y enviársela a Beatriz (por ejemplo, por correo electrónico). Beatriz mezclaría estas copias con la suya propia, mediante el comando `--import`, y seguidamente exportaría esta

[8] `http://www.gnupg.org/gph/es/manual.html`

[9] En realidad está firmada por Beatriz, pero esto no debería ser una garantía para nadie, salvo para ella misma

nueva clave (que lleva ahora las firmas de Damián y Alberto), y se las reenviaría a sus amigos. Todos actualizarían su anillo con el comando `--import` y finalmente todos tendrían una copia de la clave pública de Beatriz que lleva estampada las firmas de Damián y Alberto.

Si ahora una tercera persona (Emilio) obtiene la clave pública de Beatriz, podrá ver que lleva estampadas dos firmas. Pueden darse ahora tres casos:

- Emilio no conoce ni a Alberto ni a Damián. Sus firmas no le darían muchas garantías, y en este caso Emilio debería verificar personalmente la huella dactilar de la clave recibida para seguidamente firmarla a su vez (tras lo que se podría repetir la ronda anterior para actualizar la clave de Beatriz que ahora tendría tres firmas).
- Emilio conoce a Alberto (o a Damián, para el caso es lo mismo) y sabe que es extraordinariamente escrupuloso a la hora de verificar firmas digitales. Tiene tanta confianza en Alberto como en sí mismo. En este caso Emilio no necesitaría verificar personalmente la clave que ha recibido de Beatriz. El hecho de que lleve la firma de Alberto es garantía suficiente para él.

  En este caso Emilio podría decirle a GnuPG que "confía ciegamente en Alberto". De este modo, todas las claves que vengan firmadas por Alberto serán dadas por buenas. Aún así, *Emilio no debería firmar la clave de Beatriz*, ya que no ha comprobado personalmente la huella dactilar.
- Emilio conoce a Alberto (o a Damián, o a ambos) y cree que con bastante probabilidad habrán verificado la clave de Beatriz, pero no está del todo seguro. En este caso Emilio puede decirle a GnuPG que "confía marginalmente en Alberto" (y/o Damián). Cuando GnuPG encuentre claves firmadas por Alberto, no las dará por buenas inmediatamente. Necesita que la clave lleve al menos tres firmas de personas en las que Emilio "confíe marginalmente".

Esto nos introduce un nuevo nivel de complejidad en GnuPG. No sólo debemos aceptar y validar claves públicas de nuestros amigos, sino también emitir un juicio sobre si estos amigos son escrupulosos o no verificando las claves que ellos a su vez reciben.

Para especificar esta "confianza" en los amigos, es necesario lanzar `gpg` con la opción `--edit-key` seguido de la *key* del amigo. Entramos en un menú interactivo en el que pondremos el comando `trust` y responderemos a la pregunta que `gpg` nos hará sobre la confianza que ponemos en ese amigo.

Observa que no es lo mismo la confianza que tenemos en su clave (que es algo objetivo, que podemos contrastar con la huella dactilar y que certificamos después con nuestra firma), que la confianza que tenemos en el amigo (que es algo subjetivo). Como la valoración de la confianza es subjetiva, la "puntuación" que le demos a cada amigo no se almacena en su clave pública, sino en un fichero separado. Esta información nunca se exporta.

A. Anexos

A.1. Compartir claves públicas con tus compañeros

Comparte tu clave pública con tanta gente como puedas. Recuerda que esto no supone ningún riesgo para ti, sino tan sólo la posibilidad de que las personas con las que la hayas compartido puedan enviarte mensajes cifrados, o verificar tus firmas.

No olvides que cuando recibas la clave pública de alguien, deberás verificar que es la correcta (mediante su *fingerprint*) a través de un canal seguro, como "en persona" o por teléfono. No obstante, si la clave que has recibido ya viene firmada por alguien que habías verificado previamente, este paso no será necesario (siempre que confíes en que ese alguien es tan escrupuloso como tú a la hora de verificar las claves).

En particular, el profesor verificará todas las claves públicas de sus alumnos, y las firmará digitalmente, tras lo cual enviará cada clave de nuevo a su propietario. Una clave firmada por el profesor debería ser fiable

para cualquiera, por lo que puedes pasar esta clave a tus compañeros y éstos podrán aceptarla sin necesidad de verificar huellas dactilares (el profesor ya lo ha hecho). Si lo deseas, también puedes subir la clave firmada por el profesor a un servidor de claves.

Algunos usuarios de GnuPG intentan que su clave pública esté firmada por el mayor número posible de personas. Una clave firmada por mucha gente debería ser de fiar, ya que hay que suponer que todos los que la han firmado han verificado la huella dactilar. Si recibes una clave firmada por muchas personas, incluso si no conoces de nada a estas personas, tienes buenos indicios para creer que la clave es válida[10].

Hay quienes incluso organizan “quedadas de firmas”, en las que varias personas quedan físicamente en un lugar, y todas acuden con varias copias de su “huella dactilar” impresa, para repartir entre los asistentes. De este modo, al volver a sus casas, quienes hayan recibido una copia de esta huella dactilar, podrán verificar la clave y firmarla.

A.2. Clave pública del autor

A continuación se muestra la clave pública del autor de este documento. Esta información está también disponible en la página Web de la asignatura.

```
$ gpg --export -a 0xBFF51D6C
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.1 (GNU/Linux)

mQGiBEOyfWYRBAC1twduZGwC1wBuDTiz0GfXboShsV1EJM3o9IDsP8Dqa8JcS+3H
GJJJU2EoIQVVVK4tTRvRnznL+z/577aiq52ifONB/y47vH2GYjBdzCfOg6wc4/kr
p6zhNTiWKAI9pMEMAVFwe5DC1rcx2mxxO1KhYvcQ3PYiDs/0TNaUTxYtAwCg8Yzj
w2XP8FPei09jFsqFZYB5HekEAIjXY+lKnEtXJqphfP3XdsKww1faDjuDEvzr8Hvg
wXT3zGKzz5rbYpecWxkjt6XktXa1P2Pq+0qAl0HfEcQtL+FR5L6XxA3r1dakvz+N
QDCrg64vRliWo4sdGX/R3R4oZ1CvqC52pSMPbF5V7TQ2ZXfpp9v6J1uVUcbcU4Q6
Y6YhA/9wQslbZOEfVf3N3lq9d2L2kUv46Dvyt+++P8t9NjPcvRQS8PMFq5REWEDv
XU2I+gzoGVE8OY2tyymu3HdZGjhAfvrKcKXKuLsXq69aE8B8VrlXUSnmuBlgE405
KHjHRgpw6xo3p4k7A6uw79NlgHLgtm/sgzqXE4DnemMxTk0z17QrSm9zZSBMdWlz
IERpYXogKERvY2VudGUpIDxqbGRpYXpAdW5pb3ZpLmVzPoheBBMRAgAeBQJDsn1m
AhsDBgsJCAcDAgMVAgMDFgIBAh4BAheAAAoJEGlGLmC/9R1stBkAnRWv0VUUXmVC
z/QdBdMbhPCEusDtAKCqnuKHUQiDr0GWgbE3kxDMTVRJ9bkCDQRDsn3XEAgAt23X
ZkZFPEpnni/NC7bmZjz7w9r5GapEONpsom4w6iI/J+raP39aNjnpyOx5oE7rkEJi
6ocvr5gV461w5ss00eJY/kFDjy8srMr5veuuG4L4RoubHonhtk9FDiI+/kkTpdsT
yPN1oQaPmAAnXXx+JkKVoELpiXEr3dHA2PkVF8pA1X0oepNkPWmJcS+eb4O/WI/7
D16LP5R+MsXOJYvKmg+C7KgM3mb84HIDnM26zoM+5lhDkz/gS7iRDcWj0s6DupZF
8ZZkoRXoKmbcfLjzIyK4ccpSOpvNc/2dqGlCVQuEihZ2iyZbwmxRuE/HJw5gRfTl
mKiUS/zpGEn/sPy5qwADBgf+LOlVJ4RZDkAx+4QTMXvdun+ZJvmxFszjJcGGI1YR
zKBtUVCFCthvq7ARyjmUWXmKGpckhCz/qba65UdWC3AQIX8PBJ9DHQlFU/LdgcbS
LIkf6ZSsXq6wABGM0vqe0eyB3Ot/XHIuilz9igqxfnaCZWMPkVvWo6xV24AlemMJ
6epeFC7CDFMOm+H5rCyxCgd9QBd0PcyOXjQdYDnT6G3AKAsTYzzhO7bpJNK0UnIJ
lGaGxbZ9pc/O2Y03qSAzj64OS4M+xEtretEDlfa04I7D5ubJ9+JzM1ULy4RKnjr1
4hWaauU391zzudYPIYCGQ2dwwPaXgllYCOPfWtE6pBmEnYhJBBgRAgAJBQJDsn3X
AhsMAAoJEGlGLmC/9R1sogkAn24bDJ9dzbnjvkV+uR4yxochDqczAKDR03SmqHzk
nLdf1jteNiHvMHFe4A==
=6BI0
-----END PGP PUBLIC KEY BLOCK-----
```

La huella dactilar de esta clave es la siguiente:

```
$ gpg --fingerprint 0xBFF51D6C
pub   1024D/BFF51D6C 2005-12-28
      Key fingerprint = C655 417B FF58 4B98 8495  595F 6946 2E60 BFF5 1D6C
uid                  Jose Luis Diaz (Docente) <jldiaz@uniovi.es>
sub   2048g/734C241D 2005-12-28
```

[10] Creer otra cosa implicaría que todas esas personas estarían conspirando para hacerte aceptar una clave no válida

A.3. Este documento está firmado

Para garantizar que lo que estás leyendo es exactamente lo que el autor ha escrito, y que no ha sido alterado desde que ha sido firmado por éste, puedes descargarte su firma digital[11], y copiarlo a la misma carpeta en la que tienes el documento.

Ejecutando el comando:

```
$ gpg --verify GnuPG.pdf.asc
```

comprobarás quién firma este documento y si todo está correcto (asumiendo que tengas en tu anillo la clave del autor, convenientemente verificada).

[11] http://www.atc.uniovi.es/inf_superior/4atc/entregas/comun/GnuPG.pdf.asc