GnuPG (The GNU Privacy Guard)

Fernando Reyero Noya
Universidad de León, España

camarlengo@yahoo.es

Sinopsis

GnuPG es un sustituto de PGP libre (en el sentido de libertad. Ver Proyecto GNU (http://www.gnu.org/philosophy/free-sw.es.html)), que no utiliza el algoritmo patentado IDEA y que, por tanto, puede ser usado sin ningún tipo de restricciones. Además GnuPG cumple con el RFC2440 (OpenPGP).

OpenPGP, aplicación en la que se basa GnuPG, fue desarrollado originalmente por Phillip Zimmermann.

Página oficial de GnuPG (http://www.gnupg.org)

Introducción

GnuPG es una aplicación que implementa un sistema de encriptación. Su utilidad principal es la su combinación con nuestro cliente de correo, sin embargo, su capacidad va mucho más allá.

Además, GnuPG evita el uso de algoritmos patentados para mantener la libertad e independencia del sistema.

Como el resto de artículos que acompañan este proyecto, la finalidad de este documento es presentar una introducción de esta aplicación dentro de entornos libres, en concreto, Debian GNU/Linux. Sin embargo, GnuPG se encuentra disponible para otros muchos sistemas como GNU/Hurd, *BSD, AIX, IRIX, SCO y MS Windows y varias arquitecturas.

Al contrario que en otras ocasiones, la documentación básica de GnuPG se encuentra perfectamente recogida en el GnuPG Handbook (http://www.gnupg.org/gph/es/manual.html), en su capítulo 1. Este artículo tan sólo reproduce esta documentación, por considerarla perfecta para tomar contacto con el sistema.

Instalación

Como en otros artículos, el sistema base empleado es Debian GNU/Linux (http://www.debian.org), por tanto, las referencias de esta sección son aplicables sólo en Debian. Como siempre, la instalación es tarea casi trivial.

Los paquetes instalados son los siguientes:

- g
nupg
- g
nupg-doc

Para ello, siendo superusuario, escribe en la consola:

potasio:~# apt-get install gnupg gnupg-doc

Debian se encargará de instalar la aplicación, realizar una configuración básica y situar cada fichero en el lugar adecuado. Visita */usr/share/doc/gnupg/* para obtener más detalles sobre GnuPG, su instalación, puesta a punto o su changelog.

Asimismo, puedes consultar el *GnuPG Handbook* completo y el *GnuPG mini-HOWTO* en */usr/share/doc/gnupg-doc/*.

Primeros pasos

En esta sección describiremos las operaciones básicas que se llevan a cabo con GnuPG. Entre ellas, están la generación de un par de claves, intercambiar y comprobar la autenticidad de claves, cifrar y descifrar documentos, firmar documentos y verificar firmas digitales.

GnuPGutiliza criptografía de clave pública para que los usuarios puedan comunicarse de modo seguro. En un sistema de claves públicas cada usuario posee un par de claves, compuesto por un clave privada y una clave pública. Cada usuario debe mantener su clave privada en secreto; 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 con la que el usuario tiene un par de claves primario, y ninguno o más pares de claves adicionales subordinadas. Los pares de claves primarios y subordinados se encuentran agrupados para facilitar la gestión de claves, y el grupo puede ser considerado como un solo para de claves.

Nota

En este artículo se intercambia el término claves subordinadas por el de subclaves.

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.

potasio:~$ gpg --gen-key

```
gpg (GnuPG) 1.0.7; 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)
    (3) 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 tanto para firmar como para cifrar. En todos los casos existe la posibilidad de añadir subclaves adicionales para cifrar y firmar posteriormente. La mayoría de los usuarios tienen suficiente con la opción por defecto.

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 a 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 ralentizará a medida que se incremente 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 una 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 al 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 esta 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 aquello 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 un 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:
        "Fernando Reyero Noya (Reggie) <camarlengo@yahoo.es>"

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 contexto, como por ejemplo, si se usa por una parte en la oficina como empleado y, por otra, en casa como dueño de un pequeño negocio empresarial. Hay que tener cuidado al crear un identificador de usuario, ya que después no se podrá editar para introducir nuevos cambios.

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

- Real name: Fernando Reyero Noya
- Email address: camarlengo@yahoo.es
- Comment: Universidad de León

Veríamos lo siguiente: `"Fernando Reyero Noya (Universidad de Le\xc3\xa1n) <camarlengo@yahoo.es>"`. Por lo tanto, es mejor evitar estos caracteres.

```
You are using the 'iso-8859-1' character set.
You selected this USER-ID:
        "Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.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 caracteres extraños en lugar de vocales acentuadas...

GnuPG necesita una contraseña con el fin de proteger las claves primarias y secundarias que posea el usuario.

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

Enter passphrase:
```

No hay límite para la longitud de una contraseña. Debes elegirla con sumo cuidado. Desde el punto de vista de la 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, debes tener en cuenta que si alguna vez tuvieramos que usar nuestra contraseña desde una máquina con un teclado distinto al nuestro, te verías imposibilitado a menos que cambiaras la configuración del sistema.

Generar un certificado de revocación

Después de haber generado un par de claves, el usuario debe, de forma inmediatea, 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 oextraviada, este certificado de revocación podría ser hecho público para notificar a otros usuario 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.

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

```
sec 1024D/D58711B7 2002-05-05    Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.es>
```

El argumento *miclave* debe ser un especificador 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 pondría 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`.

potasio:~$ gpg --list-keys

```
/home/camarlengo/.gnupg/pubring.gpg
```

```
--------------------------------
pub  1024D/D58711B7 2002-05-05 Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.es>
sub  1024g/92F6C9E3 2002-05-05
```

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 opció 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.

potasio:~$ gpg --output fernando.gpg --export camarlengo@yahoo.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* 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, por ejemplo, claves, documentos cifrados y firmas, pueden ir en formato armadura-ASCII añadiendo a la orden la opción *--armor*.

potasio:~$ gpg --armor --output fernando.asc --export camarlengo@yahoo.es

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.0.7 (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*.

potasio:~$ gpg --import carlos.gpg

```
gpg: key B63E132C: public key imported
gpg: Total number processed: 1
gpg:               imported: 1
```

potasio:~$ gpg --list-keys

```
/home/camarlengo/.gnupg/pubring.gpg
--------------------------------
pub  1024D/D58711B7 2002-05-05 Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.es>
sub  1024g/92F6C9E3 2002-05-05

pub  1024D/B63E132C 2002-06-24 Carlos Rodriguez Lorenzana (Universidad de Sevilla) <carrodlor@unisev.es>
sub  1024g/581A915F 2002-06-24
```

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.

potasio:~$ gpg --edit-key carrodlor@unisev.es

```
pub  1024D/B63E132C  created: 2002-06-24 expires: never      trust: -/q
sub  1024g/581A915F  created: 2002-06-24 expires: never
(1)  Carlos Rodriguez Lorenzana (Universidad de Sevilla) <carrodlor@unisev.es>
```

command> fpr

```
pub  1024D/B63E132C 2002-06-24 Carlos Rodríguez Lorenzana (Universidad de Sevilla) <carrodlor@unisev.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: 2002-06-24 expires: never      trust: -/q
             Fingerprint: 4203 82E2 448C BD30 A36A  9644 0612 8A0F B63E 132C

     Carlos Rodríguez Lorenzana (Universidad de Sevilla) <carrodlor@unisev.es>

Are you really sure that you want to sign this key
with your key: "Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.es>"

Really sign? y

You need a passphrase to unlock the secret key for
user: "Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.es>"
1024-bit DSA key, ID D58711B7, created 2002-05-05

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  Carlos Rodríguez Lorenzana (Universidad de Sevilla) <carrodlor@unisev.es>
sig!       B63E132C 2002-06-24   [self-signature]
sig!       D58711B7 2002-06-24   Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.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 esta 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 Fernando, lo haría usando la clave pública de Fernando, y él lo descifraría con su propia clave privada. Si Fernando quisiera enviar un mensaje al usuario, lo haría con la clave pública del usuario, y este 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 se omite, una entrada estándar. El resultado cifrado se coloca en la salida estándar o donde se haya especificado mediante la opción *--output*. El documento se comprime como medida adicional de seguridad, aparte de cifrarlo.

potasio:~$ gpg --output doc.gpg --encrypt --recipient carrodlor@unisev.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.

charly:~% gpg --output doc --decrypt AudreyTautou.gpg

```
You need a passphrase to unlock the secret key for
user: "Carlos Rodríguez Lorenzana (Universidad de Sevilla) <carrodlor@unisev.es>"
1024-bit ELG-E key, ID 581A915F, created 2002-06-24 (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 está 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`.

potasio:~$ 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, Fernando 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 Fernando para comprobar la firma, verificando de este modo que el envío proviene realmente de Fernando, y que no ha sido modificado desde el momento en que Fernando 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.

potasio:~$ gpg --output doc.sig --sign doc

```
You need a passphrase to unlock the private key for
user: "Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.es>"
1024-bit DSA key, ID D58711B7, created 2002-05-05

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.

charly:~% gpg --output doc --decrypt doc.sig

```
gpg: Signature made Fri Jun  24 12:02:38 2002 CDT using DSA key ID D58711B7
gpg: Good signature from "Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.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.

potasio:~$ gpg --clearsign doc

```
You need a passphrase to unlock the secret key for
user: "Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.es>"
1024-bit DSA key, ID D58711B7, created 2002-05-05

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

[...]
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.0.7 (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`.

potasio:~$ gpg --output doc.sig --detach-sig doc

```
You need a passphrase to unlock the secret key for
user: "Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.es>"
1024-bit DSA key, ID D58711B7, created 2002-05-05

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.

charly:~% gpg --verify doc.sig doc

```
gpg: Signature made Fri Jun  24 12:38:46 2002 CEST using DSA key ID D58711B7
gpg: Good signature from "Fernando Reyero Noya (Universidad de Leon) <camarlengo@yahoo.es>"
```

Generación del artículo

Este artículo ha sido producido usando DocBook XML 4.1.2

DocBook es una aplicación XML (también hay una versión SGML) que facilita los sistemas de documentación, al dotar de semática a los textos desde el punto de vista de los sistemas informáticos. Además permite la versatilidad de dar como salida casi cualquier formato documental: LaTeX, TeX, TeXinfo, PDF, RTF, xhtml,...

La edición del texto ha sido realizada con Emacs 21, en el major mode *xml-mode*. Los paquetes Debian usados han sido *psgml* (http://packages.debian.org/testing/text/psgml.html) y *xae (http://packages.debian.org/testing/text/xae.html)*. Las transfomaciones de prueba han sido realizadas con las hojas de estilo XSL mantenidas por Normal Walsh (http://sourceforge.net/projects/docbook), recogidas en el paquete *docbook-xsl (http://packages.debian.org/testing/text/docbook-xsl.html)* y con el procesador *xsltproc (http://packages.debian.org/testing/text/xsltproc.html)*, recogido en el paquete con el mismo nombre.

Sobre este documento

Este documento ha sido escrito en formato XML utilizando la DTD de DocBook (http://www.docbook.org). Mediante este sistema, puede ser fácilmente transformado a múltiples formatos (HTML, TXT, PDF, PostScript, LaTeX, DVI, ...). Se recomienda su utilización como herramienta de documentación potente y libre.

Bibliografía

Mike Ashley. *Guía de Gnu Privacy Guard (http://www.gnupg.org/gph/es/manual.html)*. 1999.

RFC 2440 y GnuPG (http://www.gnupg.org/rfc2440.html)

Mantenedor: Nils Ellmenreich. GnuPG Frequently Asked Questions (http://www.gnupg.org/faq.html). 24 de Septiembre del 2001. 1.5.6

Brenno de Winter, Michael Fischer Mollard, y Arjen Baart Gnu Privacy Guard (GnuPG) Mini Howto (http://webber.dewinter.com/gnupg_howto/english/GPGMiniHowto.html). 17 de Mayo del 2002. 0.1.3

GnuPG mailing lists (http://www.gnupg.org/docs-mls.html).