GNU GPG

Introducción

Las comunicaciones mediante Internet en un principio no se diseñaron para ser seguras ni privadas. Si se manda un mensaje este puede ser interceptado en el camino por un tercero malicioso que puede leerlo e incluso modificarlo.
Para evitar este problema se ha creado una herramienta que usa criptografía simétrica y asimétrica (llave 🔑 pública y privada) para que los mensajes sean privados. En un principio la empresa Symantec creo un estándar patentado llamado PGP (Pretty Good Privacy). Su uso como herramienta es tan potente que se adoptó como un estándar la versión libre de PGP llamado GPG (GNU Privacy Guard). Los formatos a parte del nombre son prácticamente idénticos.

Concepto de funcionamiento

Supongamos una analogía con una carta ✉️.

Una carta tiene:

- Datos sobre a donde (país, ciudad) y a quien se debe entregar la carta junto con la estampilla.
- El cuerpo del mensaje, donde está el mensaje que se quiere comunicar a la otra parte.

La idea es utilizar técnicas de encriptación en el 'cuerpo del mensaje' de manera que si es interceptado en algún punto no se pueda acceder a la información. Únicamente el destinatario puede desencriptarlo y leerlo.

Encriptado Simétrico

Desde la antigüedad hasta el siglo XX se usaron técnicas simples para encriptar información como:

– Utilizar un corrimiento en el alfabeto por ejemplo con un corrimiento de 3, 'A' se corresponde con 'E', 'B' con 'F', 'C' con 'G', etc.
– La máquina enigma cifraba mecánicamente la información a partir de códigos numéricos.

Tiene dos desventajas:

- Con suficiente información y data cifrada se puede romper utilizando el poder computacional de hoy.
- Que tanto receptor como emisor tengan que conocer la misma clave para manejar un mensaje significa que en algún punto compartieron su clave. Esta es una brecha de seguridad.

> Las máquinas enigma del ejercito Nazi es un ejemplo de la aplicación de estas técnica de cifrado simétrico. Mecánicamente podía encriptar o desencriptar mensajes con la clave apropiada. Finalmente los británicos logran superar esta encriptación usando 'Enigma' un computador creado por el equipo de Alan Turing.

Cifrado Asimétrico.

Utilizando técnicas matemáticas (estudiadas a profundidad durante siglos) es que en la década de los 70s del siglo XX juntamente con la llegada de los ordenadores digitales se propone una nueva forma de encriptar mensajes usando no una sola clave, sino dos 🔑 llaves distintas:

- Una llave pública que puede ser conocida por cualquier persona en la red.
- Una lleve privada que solo una persona conoce y no comparte con nadie.

Para enviar un mensaje usando criptografía de llave pública:

- En el apartado de datos (que contiene información sobre a donde debe llegar el mensaje) debe estar visible. Se usa la llave pública (que fue previamente expuesta por el usuario al que

queremos enviar el mensaje) para encriptar el 'cuerpo del mensaje'.
- Se envía este mensaje encriptado de manera que solo quien tenga la llave privada podría desencriptarlo y leerlo.

Tanto la clave pública como la llave privada están relacionadas entre sí. La forma en que se relacionan dependerá del método de encriptación, por ejemplo,las dos más usadas:

- RSA. Donde ambas llaves son dos números muy grandes que multiplicados generan a otro: a*b=c. Este último es muy difícil de obtener solo teniendo un número para multiplicar a*x=c. Son números tan grandes que encontrar x solo puede hacerse por prueba y error. Y el tiempo que demoraría es tan grande (un superordenador demoraría millones de años) que no tiene sentido.
- Curva Elíptica. Ambas llaves representan puntos `(x,y)` de una curva y cumplen una relación geométrica que es virtualmente imposible de obtener a partir de solo un punto.

Firma digital / Verificación de similitud

Supongamos en ejemplo:

🤠 Bob tiene dos llaves:
1. 🔑 Llave pública.
2. 🔐 Llave privada.

Todos pueden ver la llave pública, pero solo Bob puede ver su llave privada. Si Bob encripta un mensaje con su llave privada, todos los que vean el mensaje pueden desencriptarlo con la llave pública de Bob.

Esto se hace a propósito con un mensaje cualquiera. Lo importante es demostrar que solo el propietario de la llave privada pudo cifrar tal mensaje.

> Como se puede ver la encriptación/desencriptación funciona en ambos sentidos.

Modo de uso

En Linux se puede instalar usando `sudo apt install gnupg` si es que no viene por defecto. En muchas distribuciones si viene pre instalado.

Generar llaves PGP

Se usan los comandos

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

Para crear un nuevo par de llaves 🔑 privada y pública.

Se puede configurar el método y la cantidad de bits para generar cada par de llaves.

Con `gpg --list-keys` podemos ver las llaves que tenemos almacenadas como su tag.
De igual forma con `gpg --fingerprint jp.cr3spo@pm.me` tenemos la información de una llave en específico.

Podemos usar este tag para guardar un backup online de la firma digital

```
gpg --send-keys B1390CA2AAE27C173616DC6CA0F6C59A33AD52DF
```

Exportar una Public Key

Con el comando

```
gpg --export --armor youremail@example.com > mypubkey.asc
```

Se genera un archivo `.asc` con texto plano que contiene la llave pública.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQGNBGOP/6QBDACo7MH97zzkXCsd69NuBZHEDbmxApu0WXEl6HaDMff62hf3BBF5
3rkt1IR2Lbw1xKa6tjb+gyg/pS7AtXAN/xNtWNDcLeRgmuo45j2SzPdzftmUlROP
```

```
0upy0L+Ol0cFxslUKmUJJHUMHzaV791i6mZ7dP2WIJhOgCO+gKDDTULKCMk8GD+2
CH3DGHVDBA3amRPjYUz6gExINdArS6vT+WUnQJ1TNscuO4UP//o02ca1gHM6NYVc
6Htnix23RnakKFUHE8ueQA4FynAriY/EoyGQxb0DRSRYhvMtiDNFjwxK1D0xbOmy
gOxIIrYFFRrFFHUQi/ERHlaF0K5qauSMeBOUmsAluAXZTCTSzaXk/NlYGAdivCJf
qnnUnvEiHpXaRdEdkAnnUeHIkm2tMhtv34PSNUxZSCqunDA3/RGmFjgBudjKe7V8
9g1Jx6puxDAjq1rRt2XUHRGXnGl7GQvY0u+1l2WOCPi9cTFj+maduWT1PcCgV41w
q7D23kH7IzzB0VkAEQEAAbQqSnVhbiBQYWJsbyBDcmVzcG8gVmFyZ2FzIDxqcC5j
cjNzcG9AcG0ubWU+iQHUBBMBCgA+FiEEsTkMoqrifBc2FtxsoPbFmjOtUt8FAmOP
/6QCGwMFCQPCZwAFCwkIBwIGFQoJCAsCBBYCAwECHgECF4AACgkQoPbFmjOtUt+u
+Av+O/l9w7W5toJ3JVkYcRJZhCAwST95WczlRzoKrZMSuVGmWTHGbPRfSzCk58G2
T34otTznSgq8ODv/zlzUWG2Kih7uWuXX2/zOpAXvNeqyPUsK5A1p0rxrzOu7Z4hC
V5kvI6UJazMMPqVVrQl9whp+IbI2RKVbSBlQ7jIthrHJlP2D0bEMjjNcVaBV7yeF
7Wncg4QMnPvjcLYHx7S2cAxQe6wKpgN4SBiWZzRe8HvuVkoAIenbD2JJDXn45Cie
eXWtfVG4zvFyxu0kN1qo39Vi6uYjcfP9yj5uqp0SpSiWdfZv/9b7fPU+VMdD6jOK
EhMauoubm/+0X/2KXfkRW2jSsjAnOhWpQlplSXswmPBDxh1nYATv0Vghe6QSaDGz
EIK405fHd4z729jSI1FoiRyUw3Vb9z2pyRFbphh0A6NVYWl72HYrTldRaBC+9bjq
9MhD3/szupvlJXAho9moW3kShLTRQC1tlM0O+zr/OBa/sEG4oaFkPuWplbRQdNj6
iPdAuQGNBGOP/6QBDADi0HHsgmLGTPeo/Ap9hOuDc1noAeX+hIu3gfnueDx+i7GI
NGEwhtrEC+YOlZOilPo+u9VmnIUwErkRu51yyVa4GnC4vAg435tI72Vp0+Gwvio7
/QVf3BIQNRoye+qXCKGQOiFs1m45CJa1Yfi1voab2vbCJKAhGe7WDhAAivCJSrSK
d7rtKfaFII7Acdxsl8h3/kx/CjwZB/G0hjovCne4I82QyGAqPXtLQQiz0TLeDBj6
Uotj1Co0ivXW1W3I4iameeH9nuaKq3ofo64aKJaC85cmV78/mBOmNLzd2upbY442
uNf41fCCrWOlDZ2riz/x++iT6lcvX6XhI9hn938pTAYQhiffJlIhQDzuXM8vw4yU
K9izYWrdXTKbBGUHEkBjGAqoV8pKOXdz6LjYsjPclJH+hkc/c0GrYYVCCDmYt0aJ
hy337vpCQvuDwdFWn/uzKFL7pob/iauzVSuSCODAUfnvbGzz1pQbgclfua+IKXk8
mhKwdyV8Ms08ByzoabEAEQEAAYkBvAQYAQoAJhYhBLE5DKKq4nwXNhbcbKD2xZoz
rVLfBQJjj/+kAhsMBQkDwmcAAAoJEKD2xZozrVLffRcMAJokG5NO/y9KDAgU3zp1
NyU4zzDQv001J2Si7DjZtNyiaGzR25Vf5cmkjr0GmBjo1o4icyKPwWnT3aEl1tkQ
Nb8FnoYt0ShbcIvXDZsG/x7uUkysgQPkSCxSo+VSbhVyxdODnF/5UXuUB7P4aKgF
x7nty57HzTSNvIinpjjs09oEJk5r1DlH4De/3XqhEzw51/AqozPoV95sZbMw3eIx
M3T1G4Ec/IVz39+cWf4mP1Rfg8ZtNy7cNBA9YAJUePS6AriHK9WKhJ4FiKnUC5nw
4U5D7XbLA4oPmYXYZuDzVTLscFDaOxwuffxDI7URtKA+1kezSLHkO39DzLDuhqox
xQTfKaX5a27VXOaneCFOovJYJZOo5reAVyJsDPrMoNQx31Ej6mkGZ7jpCpcYu4Wm
Odrbj9AGRBtkrv0XI/00Gj4PdmeR29JnH4GEBCq+2WtlorA26xmbXCDnMLu4vMGQ
utAs6J0pBenV79OZ3G5POPH0qfBvLNm70csLtJpFNQEWaA==
=mkuN
-----END PGP PUBLIC KEY BLOCK-----
```

Importar una Public Key

Se usa el comando

```
pgp --import <input>.asc
```
Copiar

Se pueden listar las llaves públicas registradas en el ordenador:

```
gpg --list-keys
gpg --list-secret-keys
```

Encriptar un Archivo

Podemos crear un mensaje.

```
touch text.txt && echo 'Hola mundo secreto' > text.txt
```

Este comando genera un archivo `.gpg` que es un archivo binario ilegible. Si se quisiera tener una salida con un archivo en texto plano ascii se usa el flag `--armor o -a`

```
gpg --output mensaje_encriptado.txt --encrypt --armor --recipient 'jp.cr3spo@pm.me' text.txt
```

El mensaje original es un archivo de texto:

```
Hola mundo secreto
```

Y el encriptado en un archivo de texto también:

```
-----BEGIN PGP MESSAGE-----

hQGMA1W6MQgpVxhMAQwAz9LNZgg6U+HacB/7k7e57cmqf1dqb9HjICha2CobRxvS
id1kYUJvTxKENKrWS+1F32q2U/g21uGvXKOExtSWun+wu/peeMdk5w7AxOlsfvWl
kNgGCWlg9vzQmo9+6OGyw030xQoUtGVDbGu5ufEOvdJbDCDw2a4p54G/l/JTCIpR
V0NaVbsxRYCa+HgPsJvqrwGJL2NFKjPrAgrjlOKTXT3h5pujpXHTZEaIceqCa7BW
mMXJLmOxXg3M69+c0le67q5OuqXE0hZjN9nzxT8oZCo5TwkQoVJ6Pm0K+sn/7bli
3evqsUHfmF8yo2j9Bz0pysCzwBl4EghOLpozy3kZBP96AbgX8m18I/TRImtqE5Dj
U86g77rFqdiAe+uVeAr6ZsQKHyZlt08X0cvzvcq8yKo4WM8sNvkPf/zAB40vECmm
zTDDR2UwCQPN+6908vcyg6RXAlLmvJT01PEh3OgpzUKqCFPHe2GDCN9T0BPCkX+k
3cvk7LqKe9YNcUhCYQRB0lYBWwW64Ll7Rf11E7XREMmAmW9qwVICkunoZaCHDIjf
LqAy5BnqODv1C2oa3tGjlyxs1d90TEARUKshBk+L1ZftqpOobyL4LqlHTvIeSccf
tmSsiKVraQ==
=cE6k
-----END PGP MESSAGE-----
```

Desencriptar un Archivo

Usamos el comando

```
gpg --decrypt text.txt.gpg > texto_desencr.txt
```

Para desencriptar un mensaje con la llave privada que esta almacenada en el ordenador.

Backup de llaves

Con

```
 gpg -a --export-secret-keys 'jp.cr3spo@pm.me'
```

Se puede tener una copia de la llave privada.

GPG simétrico

Se puede encriptar usando una frase o contraseña

```
gpg -c text.txt
```

Esta se desencripta proporcionando la misma contraseña:

```
gpg -d text.txt.gpg > text_dec.txt
```

Firma digital

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.

Sign

Con el comando siguiente se puede generar una firma a partir de un archivo de entrada

```
gpg --output doc.sig --sign text.txt
```

Deja como salida un archivo comprimido en formato binario.

Para SOLO verificar que la firma es válida usamos el comando

```
gpg --verify doc.sig
>gpg: Signature made Wed 07 Dec 2022 19:01:18 -04
>gpg:                using RSA key
>B1390CA2AAE27C173616DC6CA0F6C59A33AD52DF
>gpg: Good signature from "Juan Pablo Crespo Vargas <jp.cr3spo@pm.me>"
[ultimate]
```

Para verificar que la firma es válida y extraer el archivo del binario.

```
gpg --output doctext.txt --decrypt doc.sig
```

Recupera el archivo original.

Clearsign

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.

Para tener el mensaje integro y además se añada la firma pues usamos:

```
gpg --sign text.txt
```

Crea un archivo `text.txt.asc` que contiene lo siguiente.

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512
```

```
Hola mundo secreto
-----BEGIN PGP SIGNATURE-----

iQGzBAEBCgAdFiEEsTkMoqrifBc2FtxsoPbFmjOtUt8FAmORHY4ACgkQoPbFmjOt
Ut9Cxwv7BTn9/PydWXj/GV8zjc/ETP3reyYtXY8fHgQ7Df36tXpeVwtwBMbsGUkX
F0ItXq93Lx7PZK/wHwRMclA+9ZOe88zNIislwGrSjwmPNu7xe4ei0dLcKs7pg1uf
XP9oQNudMFjs0ecxGzPfjrV3bnRHraps3QLyKPfIynBhd5eVKY99TsAIPgHRfSeK
F5w4412KuyJv4Qb69Fa/C/jxBAO86dyNtPtYtT3ji2oCrhEEmKAbXot2K3wvN3rF
CBp2sFWLoY0mVjpbZVDplgunlj24BRPSDywrXG8bpjQ7PeLHW9jSYtLogKPyJ/KS
JxkRER3VnsoXciLpA84B8cPHg6jHmtqsLhA5iwzsyTJMGA0pR/KOHKaj1Ll+SPrG
53iWe7wJd9zz7FOrBMplIIZF+T1RFQRVpyIUlEBGfEhw7D/yzZ3czVYeCh93xKmj
VaUyxiCHmwHzVZ5E60thFweT+T7yFS9MiySYHrw8eFbvl1Y+OKZSNR3SBOt/3BYZ
uPgLHxzl
=X14v
-----END PGP SIGNATURE-----
```

Firma acompañante

Para firmar un archivo pero no se busca que se añada la firma editando el mensaje se puede usar `--detach-sig` para tener el archivo intacto.

```
gpg --output doc.sig --detach-sig text.txt
```
Copiar

Esto genera dos archivos:

1. Archivo original.
2. Firma `doc.sig`.

Si se quisiera verificar que el Archivo original no se modificado podríamos verificarlo con la firma en conjunto:

```
gpg --verify doc.sig text.txt
>gpg: Signature made Wed 07 Dec 2022 21:44:56 -04
>gpg:                using RSA key
B1390CA2AAE27C173616DC6CA0F6C59A33AD52DF
>gpg: Good signature from "Juan Pablo Crespo Vargas <jp.cr3spo@pm.me>"
[ultimate]
```
Copiar

Si cambiamos el archivo `text.txt` la verificación da un error.

```
gpg --verify doc.sig text.txt
>gpg: Signature made Wed 07 Dec 2022 21:44:56 -04
>gpg:                using RSA key
B1390CA2AAE27C173616DC6CA0F6C59A33AD52DF
>gpg: BAD signature from "Juan Pablo Crespo Vargas <jp.cr3spo@pm.me>"
[ultimate]
```

Copiar

Copia/Recupera keys del Cloud

Una vez que se tiene un par de llaves, se quiere dejar la llave pública visible para todos. Se puede usar cualquier medio para compartirlo. Existen servidores específicos que almacenan llaves públicas para que cualquiera pueda verlas.

```
gpg --send-keys id_key
> gpg: sending key 6C3B8DB995484D72 to hkps://keys.openpgp.org
```

Copiar

Y si se desea enviarlo a un servidor específico como el del MIT usamos:

```
gpg --send-keys --keyserver pgp.mit.edu id_key
```

Copiar

Para recuperar una lleve pública de un servidor:

```
gpg --keyserver pgp.mit.edu --recv-keys id_key
```

Copiar

Extra: Keybase

Keybase is secure messaging and file-sharing.

Se puede guardar la llave gpg y puede identificar propiedad de redes. Es parecido al uso de gpg con un par de pasos extra.