GNUPG

- Flavio Ramirez
-

Encritpación Simétrica

- Antes de abordar el tema de la suite de programas GNUPG ,aclararemos algunos conceptos.
- En el esquema de encriptación asimétrica todo usuario o sistema deberá crear un par de llaves de tamaño prefijado : una privada que se guardará en un repositorio llamado llavero privado y otra llave pública que se guardará en un llavero público, este llavero contendrá también las llaves públicas de los usuarios o sistemas con los que deseemos comunicación segura.

- El esquema es sencillo todo lo que se encripta con la llave privada del destinatario, este lo puede desencriptar con su privada,hay pues una relación matemático logica entre las llaves.

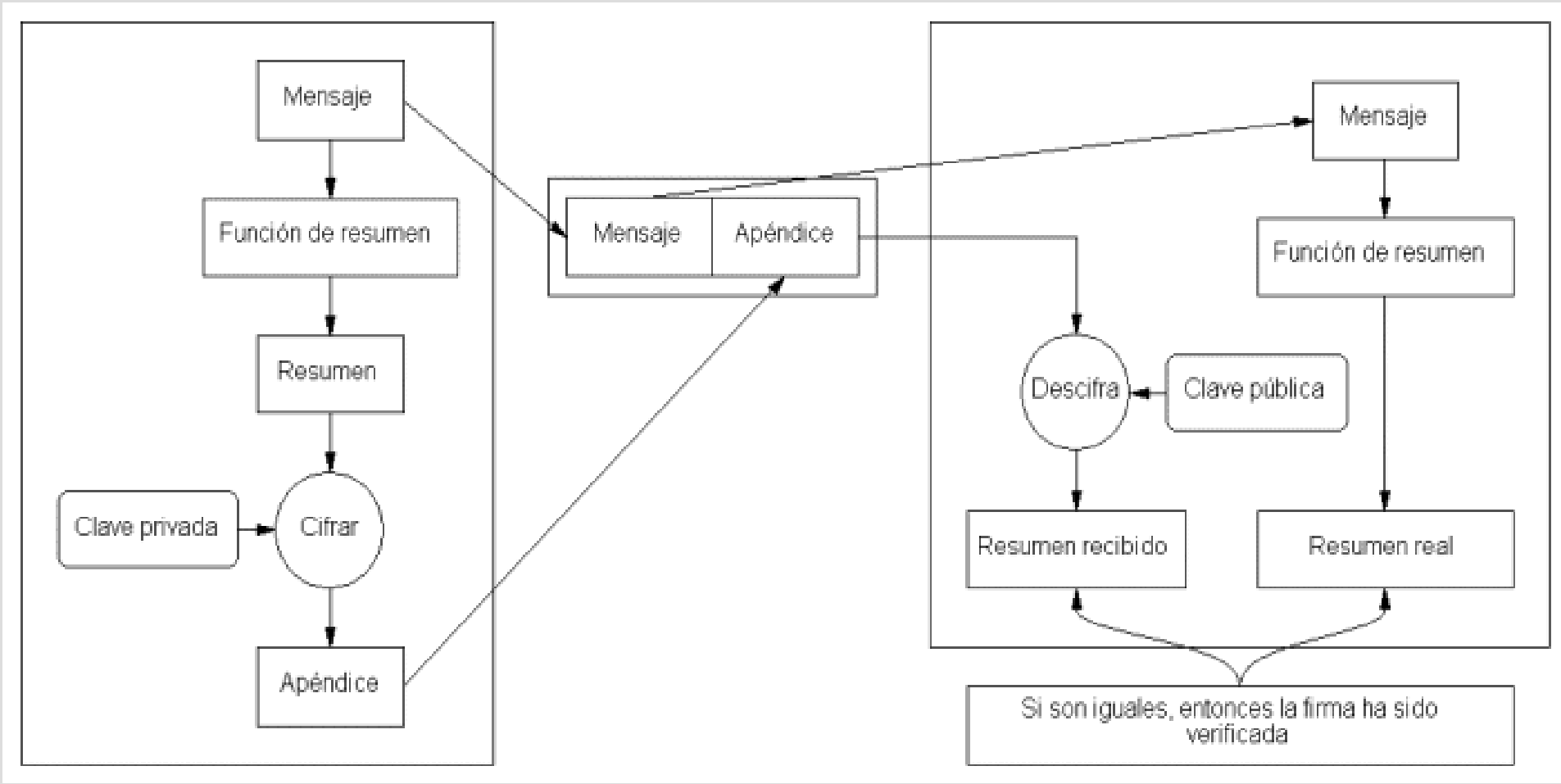
Firmas Digitales

- Este esquema permite verificar la autenticidad del emisor de un mensage de correo,se basa en sacar un resumen de un archivo usando una función HASH, de tal manera que este resumen es de un tamaño fijo independientemente del tamaño del archivo,despues ese resumen es encriptado con la llave privada del emisor.

Firmas Digitales

- Finalmente el mensage en texto claro y el resumen encriptado (Firma digital), son enviado juntos.
- El receptor del mensage extrae el mensage en texto claro le saca nuevamente un resumen , lo compara con el resumen encriptado ,para lo cual desencripta este con la clave pública del emisor.
- Si ambos resumenes son iguales se puede aseverar que el emisor fue realmente quien enviò el mensage,ademas el mensage no fue alterado.

Proceso de la firma digital



Función HASH

- Las funciones HASH son fundamentales para las firmas digitales, estas funciones generan un resumen de un archivo de tamaño fijo, independiente del tamaño del archivo.Si variamos por lo menos un bit del archivo,el resumen variará en por lo menos la mitad de los bits.

Funciones Hash

- Este resumen es usado para comprobar la integridad de un archivo y si lo ciframos con la clave privada del emisor,nos permite autenticar al usuario,asi como agregar la caracteristica del NO REPUDIO.Existen algunos algoritmos para las funciones HASH como:
-

Funciones Hash

- MD2
- MD4
- MD5
- SHA-1 (Secure Hash Algoritm)
- RIPEMD-160

Ejemplo : md5sum archivo genera el resumen:

e234af43fd0b9f019e0b6853fd7380b1 de 128 bits de longitud ,usando el algoritmo MD5

GNUPG

PGP (**Pretty Good Privacy**) de Phil Zimmermann, es un paquete completo de seguridad de correo electrónico que proporciona **confidencialidad, validación de identificación, firmas digitales, usando algoritmos criptográficos existentes, y compresión**. El paquete completo, incluido código fuente, se distribuía (*ahora ya no*) por Internet de forma **gratuita**, por lo que es el de más amplio uso hoy en día. **NO ES UN ESTANDAR. Además, hay versiones incompatibles dado que algunos programadores han modificado el código fuente.**

Existe una versión completamente gratuita "GNU PG"(GNU Privacy Guard RFC 2440***) equivalente a PGP.***

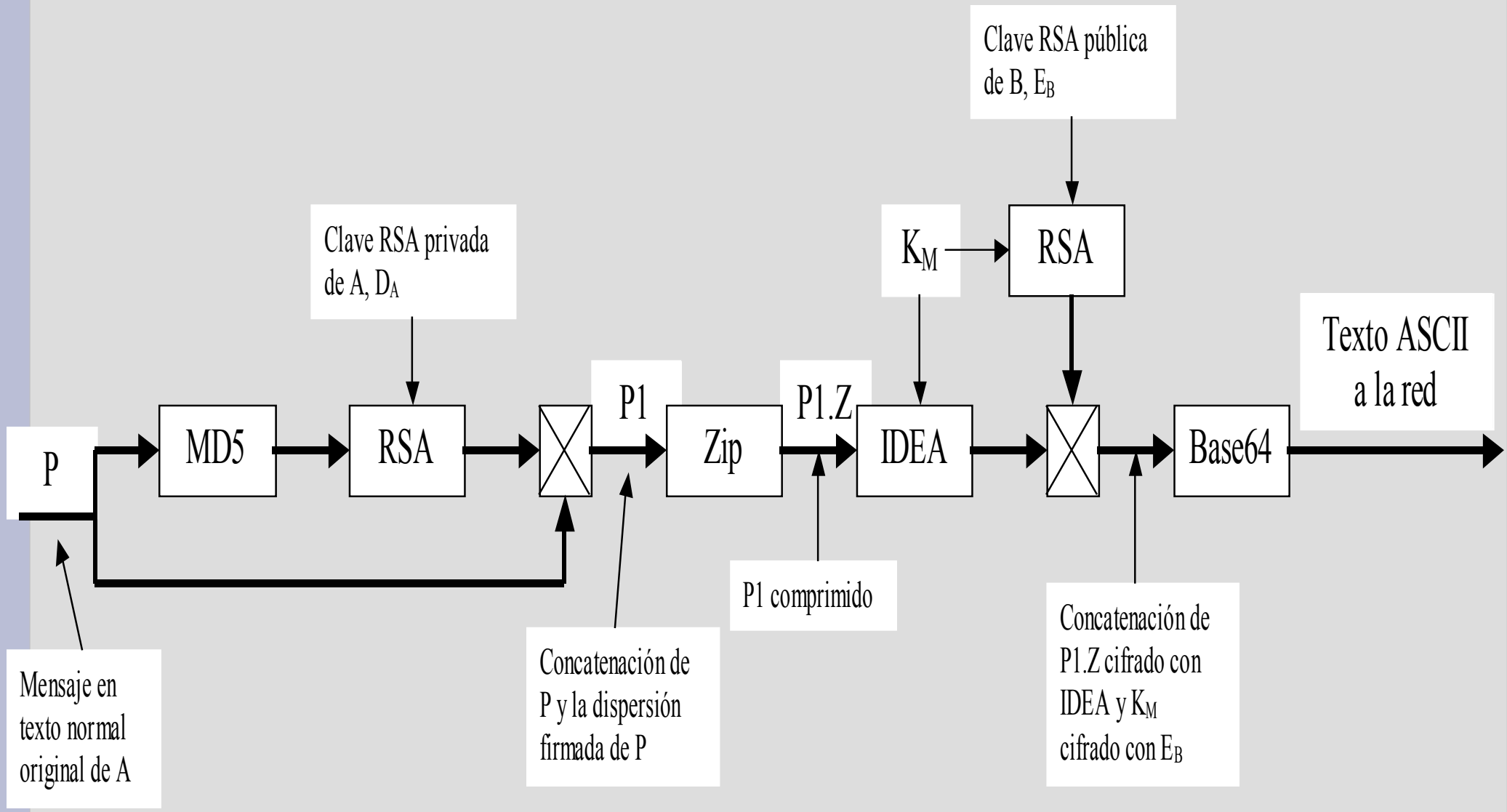
http://www.gnupg.org

Se conoce como sistema de firma digital para correo electrónico.

Se puede utilizar localmente para proteger ficheros (como adjunto de correo) sin necesidad de enviar el correo, para certificar usuarios, como repositorio de llaves (*key ring*), ...y también puede ser soporte para *e-commerce.*

PGP se basa en los algoritmos de encriptación RSA e IDEA y el compendio de mensaje MD5, *nació con la idea de que cada usuario era su propia CA*, cualquier usuario podía firmar a otro y decir cuál era el nivel de confianza en las firmas de otros usuarios.

Proceso GNUPG



Proceso GNUPG

Tanto A como B tienen claves RSA privadas (D_X) y públicas (E_X).

A invoca el programa PGP para mandar P:

Procedimiento:

1.- lo primero que se hace es dispersar P usando MD5 (o SHA), cifrando la dispersión resultante con su clave RSA privada, D_A

2.- la dispersión cifrada y el mensaje original se concatenan en un solo mensaje P1, y son comprimidos mediante el programa ZIP, obteniendo a la salida P1.Z

Proceso GNUPG

PGP solicita a A una entrada al azar, que junto con el contenido y la velocidad de tecleo se usan para generar una clave de mensaje IDEA de 128 bits, K_M (clave de sesión) que se utiliza para cifrar P1.Z con IDEA. **K_M se cifra con la clave pública de B, E_B.**

Las claves públicas solicitadas se guardan en CLAVEROS (*o también llamados* ***anillos de llaves,*** key rings***).***

Las claves pueden tener 512 (32 dígitos hexadecimales), 768 o 1024 bits en RSA.

4.- estos dos componentes (P1.Z y K_M) se concatenan y convierten a **base64 (codificación MIME)** que puede ponerse en un cuerpo RFC 822 y esperar que llegue sin modificación.

Cuando B recibe el mensaje, **invierte** la codificación **base64** y descifra la **clave IDEA** usando su **clave RSA** privada. Con dicha clave, B puede descifrar el mensaje para obtener **P1.Z.**

Tras descomprimir, B separa el texto normal de la **dispersión cifrada** y descifra la dispersión usando la clave pública de A. Si la dispersión del texto normal concuerda con su propio cálculo **MD5**, sabe que P es el mensaje correcto y que vino de A.

gpg-linea de comando

- Existe en la suite de programas un comando gpg el cual es una herramienta de encriptación y firma digital entre muchas de sus funcionalidades están la creación y administración de las llaves públicas y privadas.A continuación revisaremos algunas de sus funcionalidades

Instalacion GnuPG

-

Para instalar el aplicativo en distribuciones derivadas de RedHat puede usar YUM escribiendo lo siguiente.

[root@mineas ~]# yum install gnupg.i386

Para instalar el aplicativo en distribuciones derivadas de Debian puedes usar APT escribiendo lo siguiente

[root@mineas ~]# apt-get install gnupg

Creación de Claves

- GnuPG utiliza criptografía de clave pública (PKI) para que los usuarios puedan comunicarse de un modo seguro. PKI es un sistema de claves públicas donde cada usuario posee un par de claves, compuesto por una clave privada y una clave pública. GnuPG implementa un esquema algo más sofisticado en el que un usuario tiene un par de claves primario, y ninguno o más de un par de claves adicionales subordinadas. Para esto ejecutamos el comando:

Creación de Claves

- root@mineas ~]# gpg --gen-key
-
- Una vez ejecutado, si todo esta bien, nos pedirá que indiquemos el tipo de clave que deseamos crear, para esta experiencia elegiremos **1**. Se crearan un par de claves DSA, que es el par de claves primario que se usará sólo para firmar. Un par de claves subordinadas ElGamal que se usará para el cifrado
-
- Please select what kind of key you want:
- *(1) DSA and ElGamal (default)*
- *(2) DSA (sign only)*
- *(4) RSA (sign only)*
- *Your selection? **1***

Creación de Claves

También hay que escoger un tamaño para la clave y una fecha de caducidad. Estos parámetros deben escogerse con mucho cuidado al momento de implementar Sistemas. Informar por ejemplo de un cambio de clave vencida a gran número de usuarios puede ser complicado, o también tener miles de transacciones por minutos implica la ejecución de algoritmos miles de veces que afectan directamente al sistema si el hardware no es adecuado.

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) ***1024***

Creación de Claves

- 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) ***60***
- Key expires at Thu 19 Apr 2007 05:01:22 PM PET
- Is this correct (y/n)? y

Creación de Claves

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

Real name: ***flavio***
Email address: ***flavio@flavionet.com***
Comment: ***Mi clave PGP***
You selected this USER-ID:
"flavio (Mi clave PGP) < ***flavio@flavionet.com*** *>"*

Creación de Claves

Seguidamente confirme todo escribiendo O, y Lugo escriba una contraseña que servirá para proteger sus dos claves.

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? ***o***
You need a Passphrase to protect your secret key. **********

Importación exportación de claves

Para poder comunicarse con otros, el usuario debe intercambiar las claves públicas. Antes que nada hay que **exportar** las claves a un archivo que pueda ser compartido directamente o por un tercero. Hay que usar el identificador de clave o cualquier parte del identificador de usuario para identificar la clave que se desea exportar.

[root@mineas ~]# gpg --output flavio.gpg --export flavio@flavionet.com

Importación y Exportación de claves

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. GnuPG ofrece una opción --armor que fuerza que la salida del comando sea generada en formato ASCII.

[root@mineas ~]# gpg --armor gpg --output flavio.gpg --export flavio@flavionet.com

Veamos el contenido de este archivo

[root@mineas ~]#cat flavio.gpg

Importación Exportación de claves

- Intercambie por correo electrónico su clave con otro compañero. Recuerde que el archivo exportado fue creado en el directorio de trabajo del usuario que el que se logeo.
- Una ves reciba la clave hay que importarla al sistema

```
[root@mineas ~]# gpg --import ernesto.gpg
```

Encriptación de archivos

Para cifrar un documento se usa la opción --encrypt.
El usuario debe tener las claves públicas de los pretendidos destinatarios. Usaremos la clave publica de ernesto para cifrar, para descifrar lo haremos con su clave privada.

[root@mineas ~]# gpg --output pruebax.txt --encrypt --recipient ernesto@tl.com pruebay.txt

Leamos que tiene el documento creado
[root@mineas ~]#cat pruebax.txt

Encritación de Archivos

Enviamos por correo este archivo a su destinatario. El destinatario los desencriptara usando su clave privada y su passphrase.

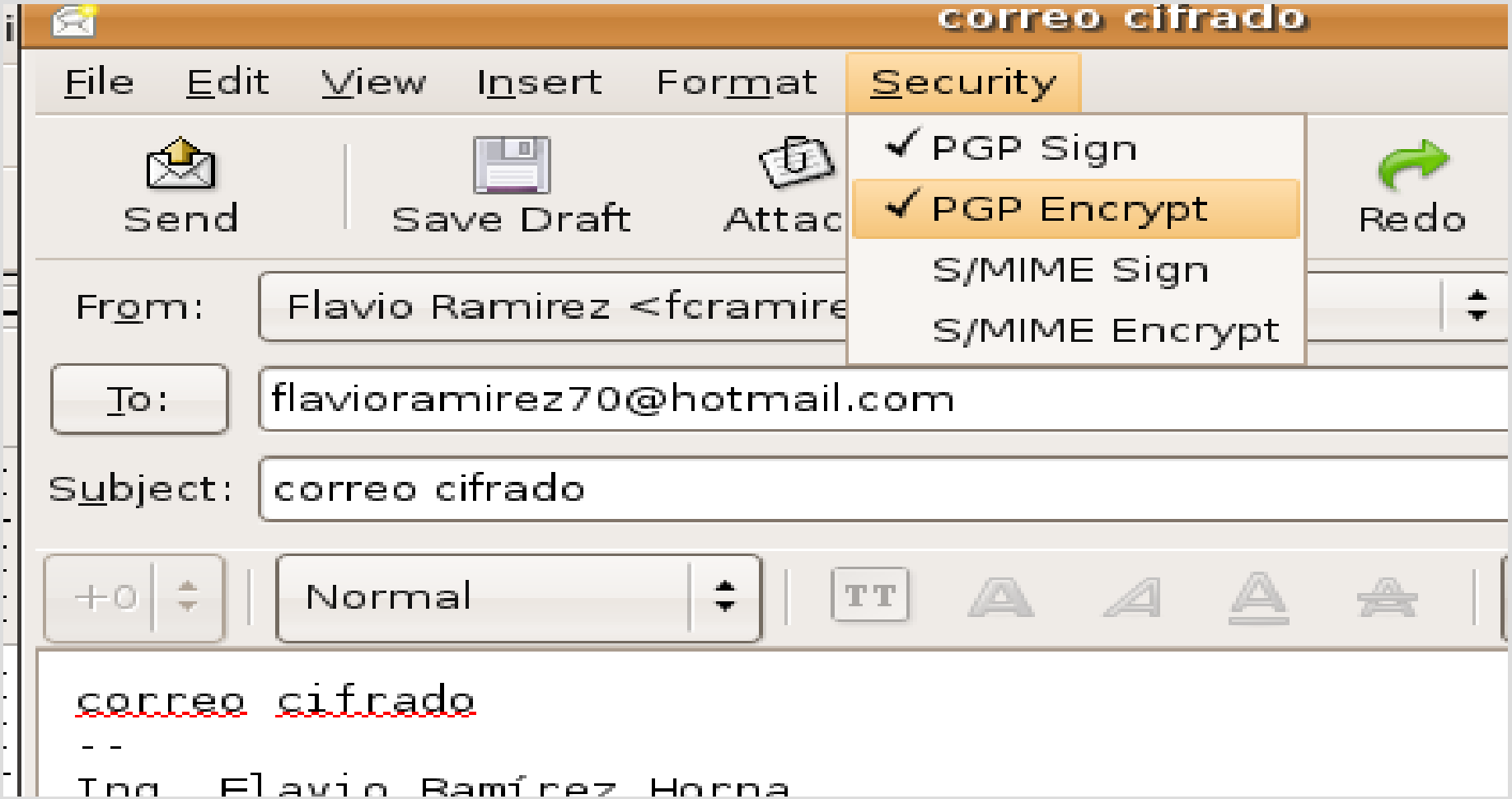
Para descifrar un mensaje se usa la opción --decrypt. Para ello es necesario poseer la clave privada para la que el mensaje ha sido cifrado. De igual modo que en el proceso de cifrado, el documento a descifrar es la entrada, y el resultado descifrado la salida.

[root@mineas ~]#gpg --output resultado.txt --decrypt pruebax.txt

Leemos el archivo

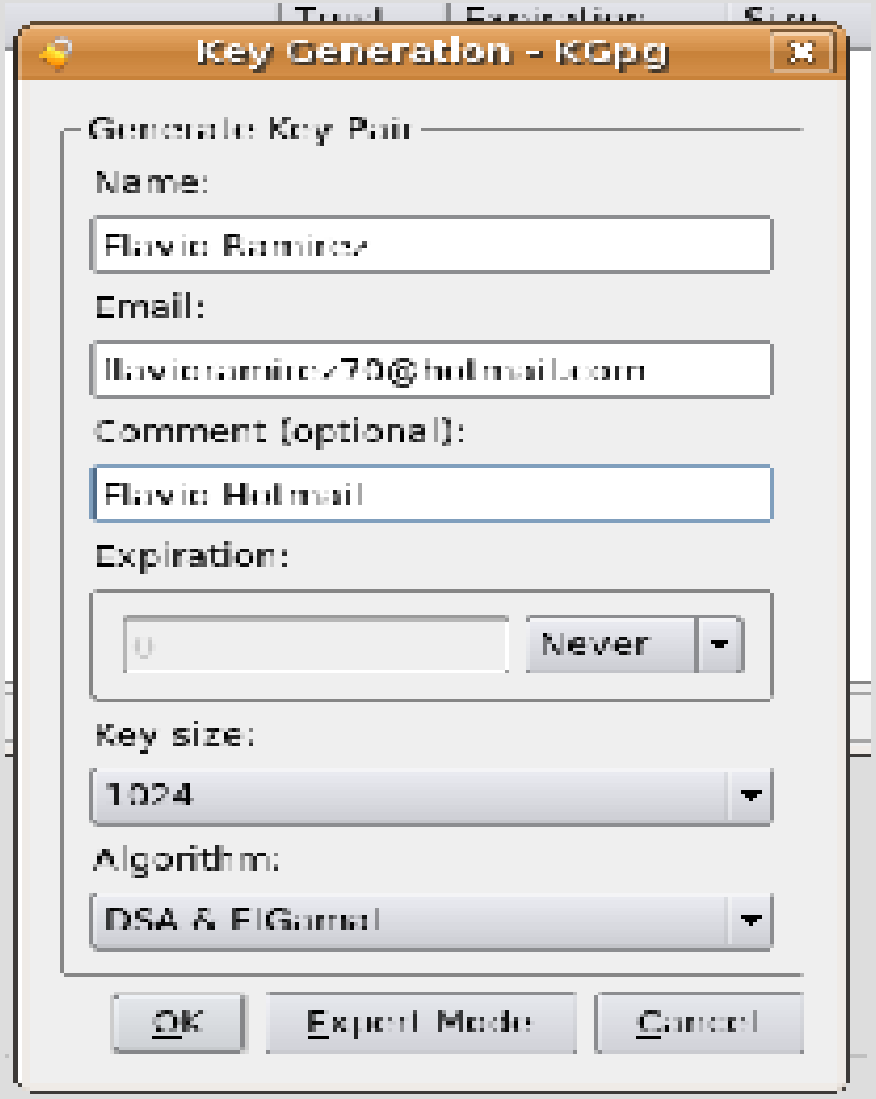
[root@mineas ~]#cat resultado.txt

Utilitario Gráficos-Evolution Mail Client

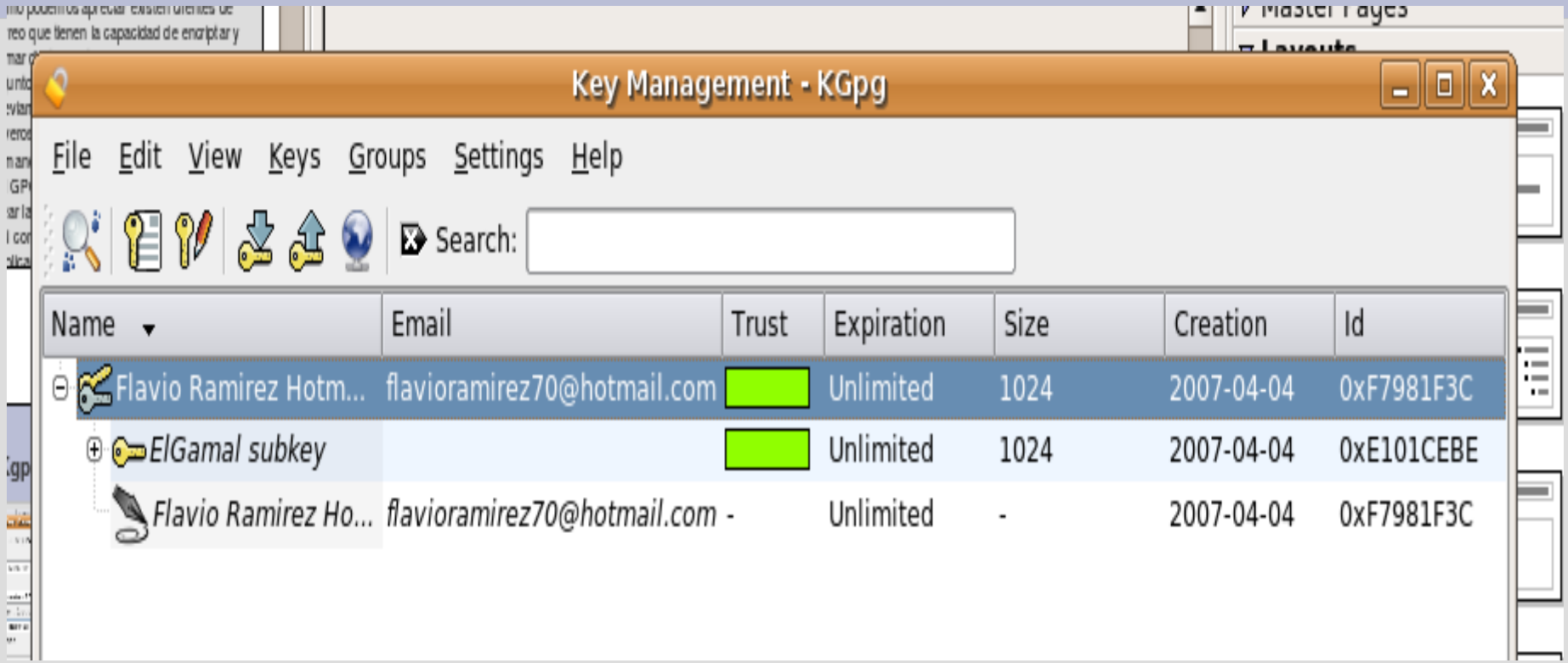


- Como podemos apreciar existen clientes de correo que tienen la capacidad de encriptar y firmar digitalmente los correos y archivos adjuntos como se ve en la figura anterior.
- Previamente se debio haber creado los llaveros públicos y privados con la linea de comando PGP o con algún utilitario front-end del GPG como el Kgpg que se usa para crear las llaves y llaveros públicos y privados ,asi como exportar o importar las llaves públicas.

Kgpg-Generación de Claves



Podemos apreciar que necesitamos indiarle algunos detalles como el correo,el tamaño de la llave y el algoritmo de encriptación.



Como podemos apreciar usando este front-end es muy fácil crear las llaves y llaveros públicos y privados ,también tenemos la funcionalidad de exportar la llave publica a un servidor de llaves

Exportando la Llaves Públicas

Tenemos varias opciones para exportar en importar tales como :

- Por email
- Hacia un servidor de llaves públicas.
- A un archivo ASCCII

Enviando el Correo

- El cliente de correo sabe que tiene que utilizar la llave pública del destinatario de correo para encriptar y la buscara en el llavero público local, en base a la dirección del correo del destinatario.
- Si ademas queremos firmar los correos es necesaria usar la llave privada del emisor del correo
    - Muchas Gracias