Universidad Nacional Autónoma de México

Facultad de Ciencias

Criptografía y Seguridad

Práctica 2: Correos Cifrados

Fecha de entrega: 5/09/2022

Equipo:

Criptonianos

Acosta Arzate Rubén - 317205776

Bernal Marquez Erick - 317042522

Deloya Andrade Ana Valeria - 317277582

Gutiérrez Medina Sebastián Alejandro - 318287021

Rivera Silva Marco Antonio - 318183583

1. Introducción

En esta práctica mostraremos un ejemplo de una forma de cómo aumentar la seguridad al momento de establecer comunicación en un canal inseguro como lo lo pueden ser correos electrónicos. La comunicación es realizada mediante correos cuyo contenido son mensajes cifrados.

2. Desarrollo

2.1. Generación de llaves

Vamos a utilizar *gpg* para generar nuestras llaves con el siguiente comando:

```
erickmain@erickmain:~$ gpg --full-generate-key
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 4096
Requested keysize is 4096 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) 0
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.
```

Figura 1: Creación de la llave

Nos pedirá ciertos datos como el tipo de llave, seleccionamos *RSA* (default); nos pedirá también la longitud de bits, escogemos la de 4096; también nos da la opción de elegir cuando va a expirar, por el momento diremos que no va a expirar escribiendo un 0 en la terminal.

```
GnuPG needs to construct a user ID to identify your key.

Real name: Erick Bernal
Email address: erick07@ciencias.unam.mx
Comment: practica2
You selected this USER-ID:
    "Erick Bernal (practica2) <erick07@ciencias.unam.mx>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
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.
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.
gpg: /home/erickmain/.gnupg/trustdb.gpg: trustdb created
gpg: key 6ACC9EBC3675C2E4 marked as ultimately trusted
gpg: directory '/home/erickmain/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/erickmain/.gnupg/openpgp-revocs.d/A92F6B45F3A3E6219BF0AB676ACC9EBC3
675C2E4.rev'
public and secret key created and signed.

pub   rsa4096 2023-09-02 [SC]
      A92F6B45F3A3E6219BF0AB676ACC9EBC3675C2E4
uid                      Erick Bernal (practica2) <erick07@ciencias.unam.mx>
sub   rsa4096 2023-09-02 [E]
```

Figura 2: Llave pública y secreta creadas

Posteriormente escribimos nuestros datos como nombre, correo y un comentario, al confirmar nuestros datos el proceso tardará un poco ya que está generando la llave pública y secreta.

Para que la persona a la cual le vamos enviar el correo pueda descifrarlo debemos exportar nuestra llave pública y hacersela llegar por algún medio, en este caso Google Drive. No pondremos la carpeta por cuestiones de seguridad (aún mas), pero si el comando necesario para exportar una llave pública, es el siguiente.

```
:~$ gpg --export --armor --output [direccion de llave] [correo a utilizar]
```

Luego es cuando hacemos llegar la llave pública por Google Drive.

2.2. Envío del correo

Para la generación del correo electrónico cifrado utilizamos la extensión *FlowCrypt* y creamos una Pass Phrase para proteger nuestra llave privada.

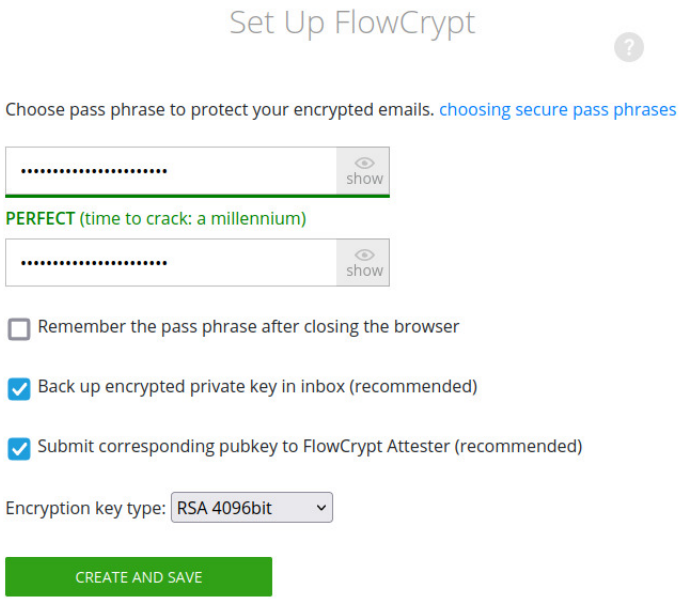

Figura 3: Creación de la llave

Mandamos un correo cifrado.

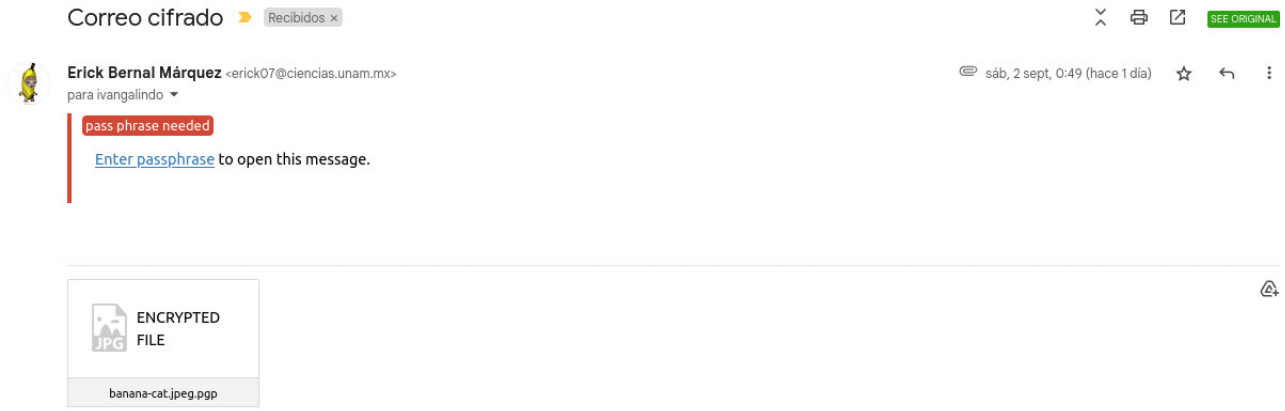

Figura 4: Correo Cifrado

Figura 5: Correo Sin Cifrar

2.3. Recepción de un correo cifrado de vuelta

En principio el correo está cifrado pero como la llave pública la hicimos llegar por otro medio entonces la persona que recibe el mensaje es capaz de descifrar el mensaje y enviarnos un par de preguntas.

El correo que recibimos está cifrado, pero gracias a FlowCrypt podemos descifrar el mensaje. Las preguntas son

- ¿Qué pasa si comprometen mi llave privada? ¿Qué opciones tengo?

  Lo que puede pasar si se compromete una llave privada (utilizada para un cifrado) es **robo de la información** lo cual puede derivar en muchos otros problemas tales como **acceso autorizado** con credenciales robadas, **suplantación de identidad**, **pérdida de información**, etc.

  Las principales opciones que se tienen son:

  ⋆ Cambiar lo antes posible las llaves, tanto la comprometida como las de otros sitios, pues se corre peligro de que también hayan sido expuestas. (Lo más recomendado)

- ⋆ Determinar la magnitud de los daños de la filtración de la información que cifraba la clave e intentar mitigar el impacto.

- ⋆ Dar aviso a todos los implicados con aquella llaves ya sean empresas, compañeros de trabajo, amigos o familiares para mantenerles alerta.

- ⋆ Analizar si es posible un cambio de medio de comunicación, pues si el actual fue expuesto una vez, podrá ser expuesto más veces.

Las medidas se tomarán dependiendo de la información comprometida.

- ¿Bajo qué escenario el sistema PGP puede ser vulnerable a un ataque de MitM y cómo mitigarlo?

  El sistema PGP puede ser vulnerable a un ataque MitM (Man in the Middle) en el caso de que se intercambien claves por medio de canales no seguros y estas sean interceptadas por el atacante. Otro posible caso sea que el atacante falsifique algunas credenciales de alguna de las partes y pueda obtener una o varias claves.

  Las mejores formas de mitigarlo son:

  - Intercambiar las claves (En caso de ser necesario) por medio de canales seguros.

  - Realizar diagnósticos de forma periódica a las redes que se estén usando.

  - Seguir prácticas de seguridad de forma adecuada.

  - De ser posible, utilizar más complementos de seguridad que no comprometan la compatibilidad de la seguridad estándar.

  - Utilizar tablas ARP o DAI, y todas las opciones posibles de la práctica anterior, que justo fue de MitM

- ¿La comunicación por correo electrónico en servicios como Outlook o Gmail está cifrada por defecto?

  Según la información recolectada de Gmail y Outlook, ambos cuentan con cifrado por defecto. En el caso de ambos, el cifrado por defecto es encriptación estándar TLS (sin embargo Gmail ofrece otro tipo de cifrado superior siendo encriptación mejorada S/MIME)

- ¿La comunicación por correo electrónico en servicios como Outlook o Gmail está cifrada de extremo a extremo (E2EE) por defecto?

  Recientemente han sido implementados en ambos, sin embargo, no se contaba con E2EE. Por desgracia sólo se menciona que se tiene en la *mayoría de servicios*.

2.4. Diferencias entre el texto claro y el cifrado, la imagen cifrada y en claro.

A continuación realizaremos un análisis de las diferencias que observamos entre tener texto e imágenes cifradas y no cifradas.

Texto Cifrado y en Claro:

Lo que podemos observar en la imágen que se muestra a continuación (Figura 6) es el texto cifrado del correo enviado al ayudante mencionado anteriormente (Figura 5).

Recordemos que el correo contiene un pequeño texto que dice *"Hola, este es el texto cifrado c:"*, a comparación del texto en su versión cifrada que es mucho más extenso, sin dejar ni rastro de lo que fue el texto originalmente. Se nos muestra un texto conformado tanto por letras mayúsculas como minúsculas, números y entre otros símbolos.

Notemos que aparecen algunos caracteres especiales como +, estos pueden ser espacios, quizá palabras con acentos, alguna separación, etc, pero realmente no hay manera de saberlo a simple

vista. Lo mismo ocurre con el caracter /, muy pocas veces se repite de manera doble //; por último al final del cifrado tenemos un símbolo de igualdad =, el cual es único en todo el texto cifrado.

Figura 6: Texto Cifrado

Imágen Cifrada y en Claro:

Haciendo uso de *hexdump* podemos observar en la Figura 7 la diferencia entre una imágen cifrada y una imágen sin cifrar. Vemos que en terminal ambas imágenes abarcan diez renglones de estructura. Comienzan por un grupo de ocho números cuyos primeros seis números son 0 y los dos números van aumentando 10 en cada renglón. Por ejemplo, si tenemos 00000010, en el renglón posterior es 00000020, esto ocurre en los renglones de ambas imágenes, a excepción del renglón 9 de la imágen sin cifrar en el que sólo se encuentra un asterisco.

Después de este grupo de 8 números en los renglones, hay dos grandes grupos cada uno consiste de ocho pares conformado por números y letras, siendo diferentes en la imágen cifrada y en la imágen sin cifrar.

Al final de los renglones hay un último grupo que es diferente para cada imágen. Este grupo está conformado en su mayoría por puntos y lineas verticales. En la imágen sin cifrar también hay letras, mientras que en la imágen cifrada hay letras, números y entre otros caracteres.

```
valeria@valeria-VirtualBox:~/Descargas$ hexdump -C photo_2023-09-03_16-55-10.jpg | head
00000000  ff d8 ff e0 00 10 4a 46  49 46 00 01 01 01 00 48  |......JFIF.....H|
00000010  00 48 00 00 ff db 00 43  00 04 03 03 04 03 03 04  |.H.....C........|
00000020  04 03 04 05 04 04 05 06  0a 07 06 06 06 06 0d 09  |................|
00000030  0a 08 0a 0f 0d 10 10 0f  0d 0f 0e 11 13 18 14 11  |................|
00000040  12 17 12 0e 0f 15 1c 15  17 19 19 1b 1b 1b 10 14  |................|
00000050  1d 1f 1d 1a 1f 18 1a 1b  1a ff db 00 43 01 04 05  |............C...|
00000060  05 06 05 06 0c 07 07 0c  1a 11 0f 11 1a 1a 1a 1a  |................|
00000070  1a 1a 1a 1a 1a 1a 1a 1a  1a 1a 1a 1a 1a 1a 1a 1a  |................|
*
00000090  1a 1a 1a 1a 1a 1a 1a 1a  1a 1a 1a 1a 1a 1a ff c2  |................|
valeria@valeria-VirtualBox:~/Descargas$ hexdump -C PGP.jpg.pgp | head
00000000  c1 c1 4c 03 07 b6 e0 75  50 bb 08 11 01 0f fe 29  |..L....uP......)|
00000010  30 40 87 f8 17 c0 69 89  bb be 4a 4b 93 58 1c e8  |0@....i...JK.X..|
00000020  85 f3 52 d0 8b 53 4a 70  98 c4 f8 45 39 f7 90 bf  |..R..SJp...E9...|
00000030  0d d9 62 5a b3 ef 5d b1  9e ee 16 ac 13 33 b0 14  |..bZ..]......3..|
00000040  81 24 42 c8 ba 4d ca d4  00 1e 8d 78 f3 7b 12 9e  |.$B..M.....x.{..|
00000050  bd 62 35 7f 4d 6b 82 ce  41 18 2e 97 cb b7 e5 5d  |.b5.Mk..A......]|
00000060  a1 8c 74 83 63 27 7f 4c  2a bd aa a5 c1 d7 e5 00  |..t.c'.L*.......|
00000070  9d 6a d5 6c 81 e3 89 67  ef 9d 8c f1 d6 c2 c6 8a  |.j.l...g........|
00000080  ad 90 56 3b 64 38 29 b6  14 f3 36 e3 0e a4 72 10  |..V;d8)...6...r.|
00000090  40 64 e0 3f a9 d6 69 d6  07 f1 b6 b9 b7 fb d7 96  |@d.?..i.........|
```

Figura 7: Imágen sin cifrar y cifrada

2.5. Análisis de llave pública con *pgpdump*

```
Old: Public Key Packet(tag 6)(525 bytes)
        Ver 4 - new
        Public key creation time - Sat Sep  2 00:12:27 CST 2023
        Pub alg - RSA Encrypt or Sign(pub 1)
        RSA n(4096 bits) - ...
        RSA e(17 bits) - ...
```

Figura 8: pgpdump de la llave pública

```
Old: User ID Packet(tag 13)(51 bytes)
        User ID - Erick Bernal (practica2) <erick07@ciencias.unam.mx>
Old: Signature Packet(tag 2)(590 bytes)
        Ver 4 - new
        Sig type - Positive certification of a User ID and Public Key packet(0x13).
        Pub alg - RSA Encrypt or Sign(pub 1)
        Hash alg - SHA512(hash 10)
        Hashed Sub: issuer fingerprint(sub 33)(21 bytes)
         v4 -   Fingerprint - a9 2f 6b 45 f3 a3 e6 21 9b f0 ab 67 6a cc 9e bc 36 75 c2 e4
        Hashed Sub: signature creation time(sub 2)(4 bytes)
                Time - Sat Sep  2 00:12:27 CST 2023
        Hashed Sub: key flags(sub 27)(1 bytes)
                Flag - This key may be used to certify other keys
                Flag - This key may be used to sign data
        Hashed Sub: preferred symmetric algorithms(sub 11)(4 bytes)
                Sym alg - AES with 256-bit key(sym 9)
                Sym alg - AES with 192-bit key(sym 8)
                Sym alg - AES with 128-bit key(sym 7)
                Sym alg - Triple-DES(sym 2)
        Hashed Sub: preferred hash algorithms(sub 21)(5 bytes)
                Hash alg - SHA512(hash 10)
                Hash alg - SHA384(hash 9)
                Hash alg - SHA256(hash 8)
                Hash alg - SHA224(hash 11)
                Hash alg - SHA1(hash 2)
        Hashed Sub: preferred compression algorithms(sub 22)(3 bytes)
                Comp alg - ZLIB <RFC1950>(comp 2)
                Comp alg - BZip2(comp 3)
                Comp alg - ZIP <RFC1951>(comp 1)
        Hashed Sub: features(sub 30)(1 bytes)
                Flag - Modification detection (packets 18 and 19)
        Hashed Sub: key server preferences(sub 23)(1 bytes)
                Flag - No-modify
        Sub: issuer key ID(sub 16)(8 bytes)
                Key ID - 0x6ACC9EBC3675C2E4
        Hash left 2 bytes - 49 d5
        RSA m^d mod n(4096 bits) - ...
                -> PKCS-1
```

Figura 9: pgpdump de la llave pública

Figura 10: pgpdump de la llave pública

Al usar el comando **pgpdump** con nuestra clave pública podemos ver distintos datos como:

- Fecha y hora en la que fue creada.
- Algoritmo utilizado (En nuestro caso RSA de 4096 bits)
- Id del usuario quien creó la llave (En nuestro caso Erick Bernal)
- El correo del usuario que creó la llave
- El algoritmo SHA que se usó (SHA512)

Entre otros detalles sobre el proceso del cifrado de la llave pública.

3. Conclusiones

El haber realizado esta práctica nos hizo conocer la importancia del proceso y los motivos del cifrado de mensajes y archivos (PGP principalmente), además de su relevancia histórica. Aunque las empresas digan que sus correos están cifrados pueden ocurrir miles de cosas que permitan interceptar y descifrar el mensaje. Así que no está demás añadir una capa extra.

Otro punto que pudimos observar son las posibles vulnerabilidades que se tienen al usar cifrados de llave pública, tales como un MitM y el robo de estas, comprometiendo nuestra información

De igual forma pudimos comparar archivos sin cifrar con los archivos cifrados por medio de **hexdump** y **pgpdump**, comprendiendo el cambio que le ocurre a la información bit a bit para que no pueda ser robada por agentes externos.

4. Referencias

- *Are public-key encryption systems vulnerable to man-in-the-middle (MITM) attacks? How can they be defended against such attacks?* (s. f.). Quora. https://www.quora.com/Are-public-key-encryption-systems-vulnerable-to-man-in-the-middle-MITM-attacks-How-can-they-be-defended-against-such-attacks

- KCCross. (2023, 12 agosto). *Cifrado de correo electrónico en Microsoft 365.* Microsoft Learn. https://learn.microsoft.com/es-es/purview/email-encryption

- *Encriptación del correo electrónico durante el envío - Ayuda de Gmail.* (s. f.). https://support.google.com/mail/answer/6330403?hl=es-419#zippy=%2Ctls-encriptaci%C3%B3n-est%C3%A1ndar

- *Email Encryption FAQs - Transparency Report Help Center.* (s. f.). https://support.google.com/transparencyreport/answer/7381230?hl=en#zippy=%2Cif-my-email-

is-encrypted-in-transit-does-it-mean-that-no-one-can-ever-snoop-on-my-email%2Cis-tls-the-be-all-end-all-solution-for-protecting-my-email-while-its-in-transit%2Cis-email-from-google-users-to-other-google-users-encrypted-in-transit%2Chow-does-encryption-in-transit-relate-to-https-access-to-gmail%2Chow-does-encryption-in-transit-relate-to-other-forms-of-email-encryption-like-pgp

- Stanojevic, M. (2023, 4 enero). End-to-end encryption is now available for Outlook.com users. *Windows Report - Error-free Tech Life.* https://windowsreport.com/outlook-com-end-to-end-encryption/

- *Generating a new GPG key - GitHub Docs.* (s.f.). GitHub Docs. https://docs.github.com/en/authentication/managing-commit-signature-verification/generating-a-new-gpg-key

- Nine, A. (2022). Google to introduce End-to-End Gmail Web Encryption. *ExtremeTech.* https://www.extremetech.com/internet/341671-google-to-introduce-end-to-end-gmail-web-encryption