PGP (Pretty Good Privacy)

Dr. Luis Gerardo de la Fraga

Departamento de Computación
Cinvestav
Correo-e: fraga@cs.cinvestav.mx

Lunes 23 de junio de 2014

PGP (Pretty Good Privacy)

- PGP es un programa de computadora para encriptar y desencriptar datos
- Provee los servicios de:
  - confidencialidad, y
  - autenticación

PGP se usa para firmar, encriptar y desencriptar:

- textos
- correo electrónico
- archivos
- directorios
- particiones de disco

Fue creado por Phil Zimmermann en 1991 cuando trabajado en PKWARE, Inc.

PGP

- Provee autenticación a través del uso de **firmas digitales**
- Provee confidencialidad a través de **encriptación simétrica de bloques**
- Provee compresión de datos a través del **algoritmo ZIP**
- Tiene compatibilidad con el correo electrónico **usando el esquema de codificación radix-64**

- Radix-64 es igual que la codificación “base 64” usado por MIME con un CRC opcional de 24 bits.
- También prove herramientas para manejar llaves para la criptografía de llave pública usando certificados digitales
- MIME (Multipurpose Internet Mail Extensions)

Características de PGP

1. PGP contiene los mejores algoritmos criptográficos disponibles
2. Integra todos estos algoritmos en una herramienta de uso general independiente del sistema operativo y procesador.
3. Está formado por un conjunto pequeño de comandos
4. Los algoritmos que usa son:
   - RSA, DSS, y Diffie-Hellman para criptografía de llave pública
   - CAST-128, IDEA, y 3DES para criptografía de llave simétrica
   - SHA-1 para codificación en picadillo (hash)
5. No es controlado por ninguna organización gubernamental o de estándares

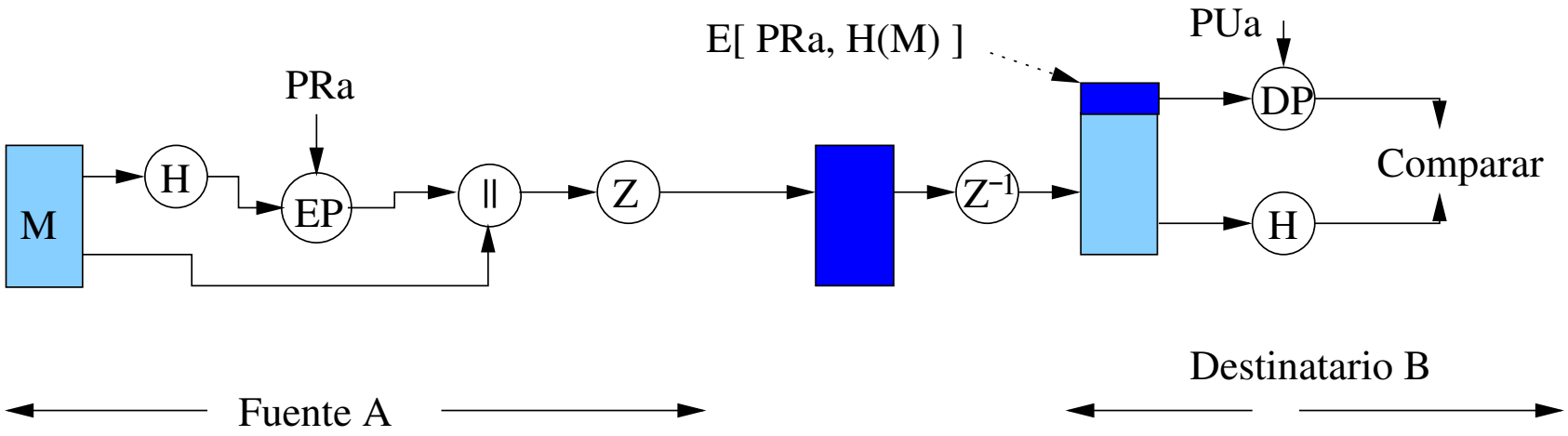
- En la documentación de PGP se usa el término
- “llave secreta”
- para referir la llave emparejada a la llave pública en el esquema de criptografía de llave pública
- Esto es confuso ya que el término “llave secreta” se usa para criptografía simétrica

Notación:

- Ks = llave de sesión usando en el esquema de encriptación simétrica
- PRa = llave privada del usuario A, usando en el esquema de encriptación de llave pública
- PUa = llave pública del usuario A, usando en el esquema de encriptación de llave pública
- EP = Encriptación de llave pública
- DP = Desencriptación de llave pública
- EC = Encriptación simétrica
- DC = Desencriptación simétrica
- H = función picadillo
- $||$ = concatenación
- Z = compresión usando el algoritmo ZIP
- R64 = conversión a formato ASCII raíz 64

Servicio de autenticación:

1. El usuario crea un mensaje
2. Se usa SHA-1 para generar un picadillo de 160 bits del mensaje
3. Se encripta el picadillo usando RSA con la llave privada del que envía el mensaje. El resultado se agrega al mensaje
4. El que recibe usa RSA con la llave pública del que envía el mensaje para desencriptar y recuparar el picadillo
5. El que recibe genera un nuevo picadillo con el mensaje y lo compara con el picadillo desencriptado, si son iguales se considera que el mensaje es auténtico

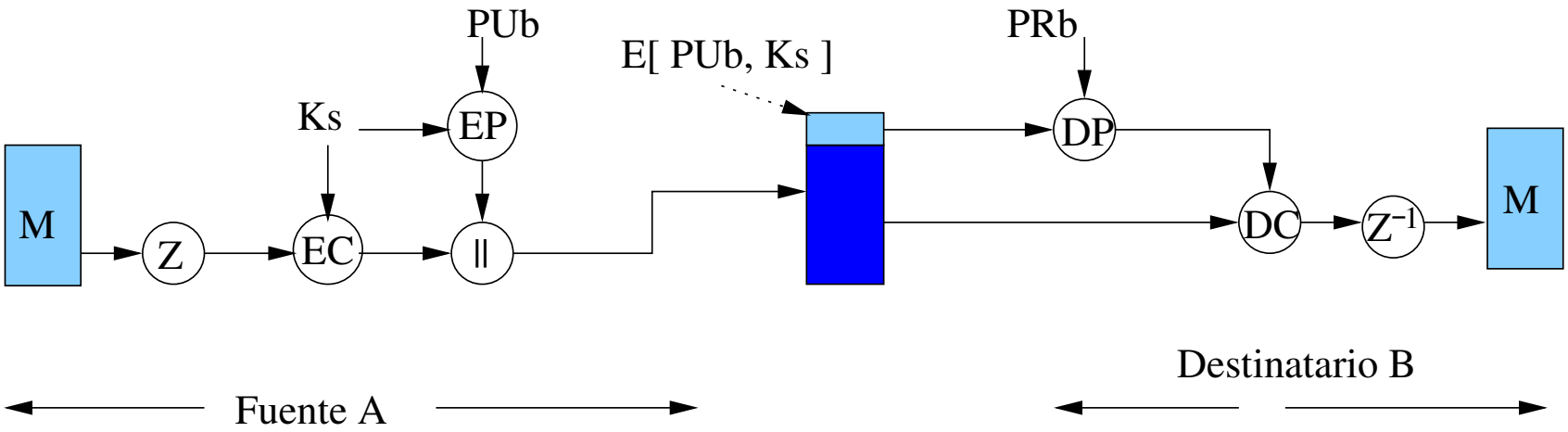


Función PGP de solo autenticación

Servicio de confidencialidad:

1. El que envia genera su mensaje y un número aleatorio de 128 bits. Este número será usado como llave de sesión solamente para este mensaje.
2. El mensaje se encripta usando CAST-128 (o IDEA o 3DES) con la llave de sesión
3. La llave de sesión se encripta con RSA usando la llave pública del recipiente y se agrega al mensaje
4. El que recibe usa RSA con su llave privada para desencriptar y recuperar la llave de sesión.
5. La llave de sesión recuperada se usa para desencriptar el mensaje

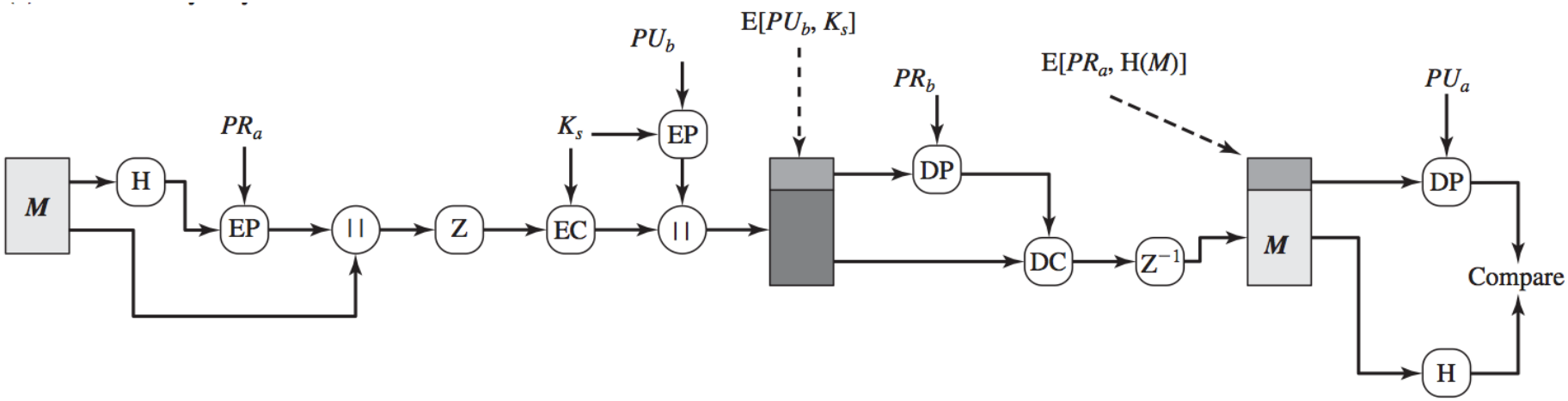
Una alternativa para RSA es el esquema de ElGamal
No se necesita un protocolo de intercambo de llave de sesión porque no se inicia una sesión de salida.



Función PGP de solo confidencialidad

Confidencialidad y autenticación

1. Se crea una firma y se agrega al mensaje
2. Se encripta todo usando CAST-128 (o IDEA o 3DES) con la llave de sesión
3. Se encripta la llave de sesión usando RSA (o ElGamal) con la llave pública del recipiente



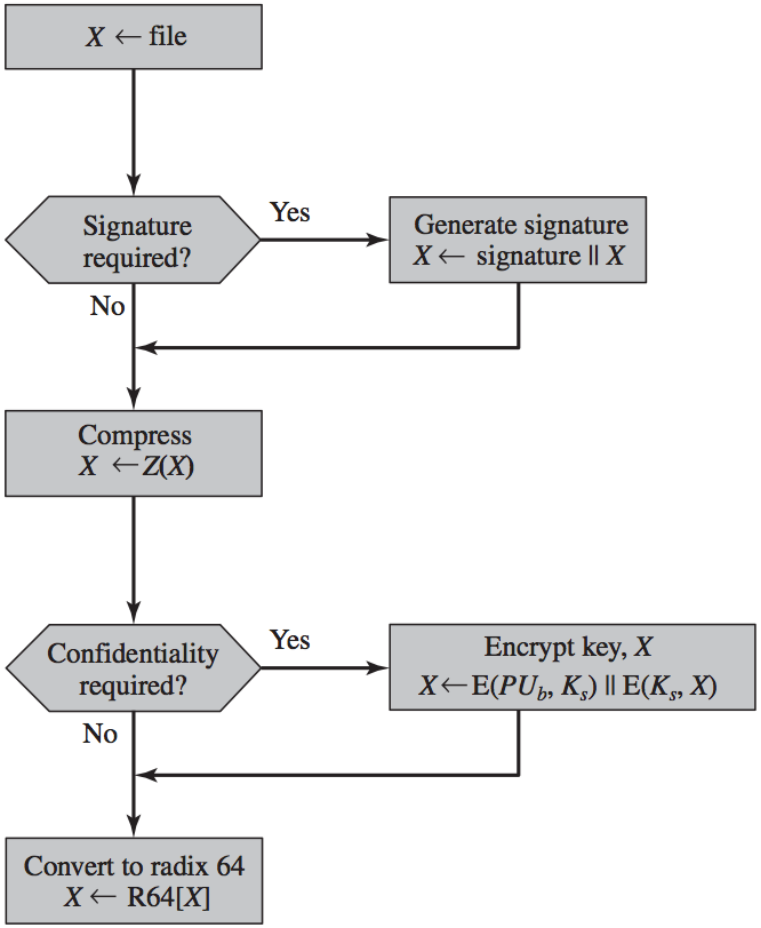
(c) Confidentiality and authentication

Compresión

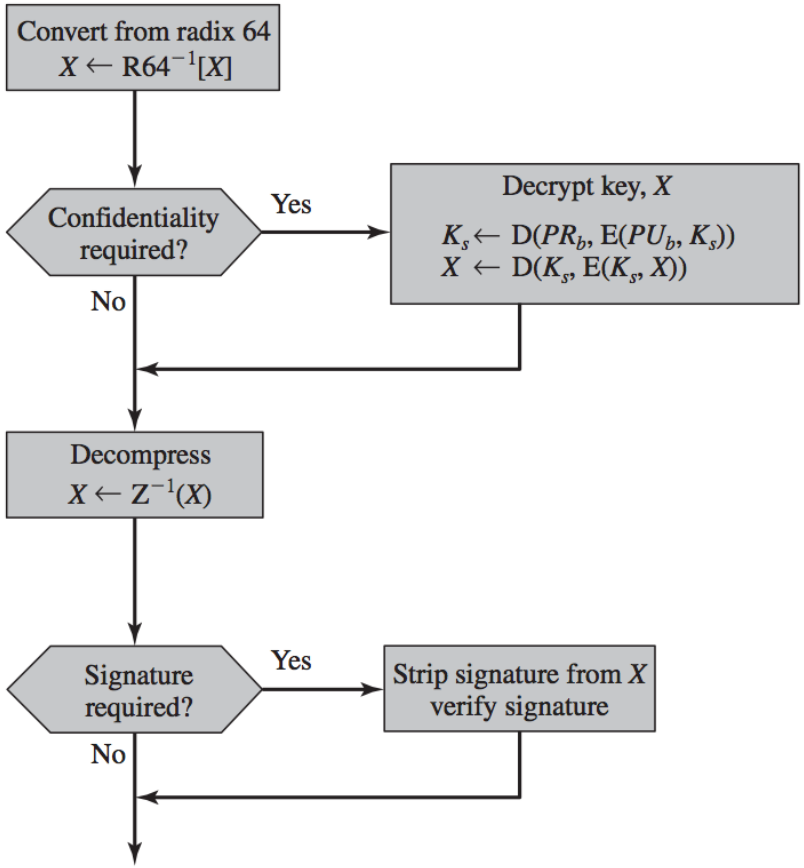
- Por defecto, PGP comprime el mensaje antes de calcular la firma y antes de encriptar
- Esto tiene el beneficio de ocupar menos espacio tanto para la transmisión de correo-e como para almacenamiento de un archivo.

Compatibilidad con correo-e

- Si solo se usa el servicio de firma (autenticación), entonces el digesto del mensaje se encripta (con la llave privada del que envía)
- Se se usa el servidio de confidencialidad, se encripta el mensaje más la firma (con una llave simétrica de un solo uso)



(a) Generic transmission diagram (from A)



(b) Generic reception diagram (to B)

Figure 18.2 Transmission and Reception of PGP Messages

PGP usa cuatro tipos de llaves:

- Llaves simétricas de sesión de un solo uso
- Llaves públicas
- Llaves privadas
- Llaves simétricas basadas en una frase de paso

Tres requerimientos pueden identificarse con respecto a estas llaves: (1/2)

1. Se necesita un medio para generar llaves de sesión impredecibles
2. Se quisiese permitir a un usuario contar con múltiples pares de llave-pública/llave-privada.
   - Una razón es que el usuario desearía cambiar su par de llaves
   - Cuando esto sucede, cualquier mensaje en espera será construido con las llaves obsoletas.
   - Los recipientes sabrán de la nuevo llave hasta que se pongan al día.
   - Además el usuario quisiese multiples llaves para interactuar con grupos distintos.
   - o mejorar la seguridad usando de forma limitada una llave para una cierta cantidad de material
   - Entonces, no existe una relación uno a uno entre usuarios y sus llaves públicas

Tres requerimientos pueden identificarse con respecto a estas llaves: (2/2)

3. Cada entidad PGP debe mantener un archivo con sus propias pares de llaves públicas/privadas así como también un archivo de llaves públicas correspondientes

Generación de llaves de sesión

- Se usa la sincronización de teclado y las mismas teclas oprimidas por el usuario como texto de entrada
- También se usa la llave anterior generada
- Se usa el modo retroalimentado del cifrador CAST-128 que genera dos bloques de 64 bits
- Se concatenan los dos bloques y forman la llave de 128 bits
- De esta forma se garantiza que la llave sea única

Identificador de llave

- La llave de sesión se encripta con la llave pública del recipiente
- Si se usara un solo para de llaves pública/privada es fácil desencriptarla: se usa única llave privada del recipiente
- ¿Pero se se tienen varios pares de llaves?
- Una solución sería usar un identificador para cada llave
- El identificador de la llave son sus últimos 64 bits

El mensaje consiste entonces de tres componentes:

1. El mensaje (incluye su fecha y nombre de archivo)
2. La firma (opcional)
   - La fecha en que fue hecha
   - El digesto del mensaje de 160 bits producido por SHA-1. Este va encriptado con la llave privada del que envía.
   - Dos octetos para el digesto del mensaje
3. Y un componente de la llave de sesión (opcional)
   Incluya la llave de sessión (encriptada) y el identificador de la llave pública del recipiente

Anillo de llaves

Se contruye con:

- Fecha de creación
- Identificador de la llave
- Llave pública
- Llave privada

Manejo de las llaves públicas

La tarea de proteger las llaves públicas para que no sean manipuladas es el problema más dificil en las aplicaciones prácticas con llaves públicas. Es el "talón de Aquiles" de la criptografía de llave pública y mucha de la complejidad en el software está atado a resolver ente problema.

Supongamos que A desea obtener una llave pública de B, los siguientes escenarios con posibles:

1. Obtener físicamente la llave de B.
   - B podría guardar su llave pública en una memoria USB y dársela a A.
   - A podría cargar la llave en su sistema de la memoria USB.
   - Este es un método muy seguro pero tiene limitaciones práctivas obvias.

2. Verificar una llave vía telefónica. Si A reconoce a B en el teléfono, A podría llamar a B preguntarle que dicte la llave, en formato radix-64. Una alternativa más práctica: B podría trasmitir su llave en un mensaje de correo-e a A. A podría tener PGP y generar un digesto SHA-1 de 160-bits de la llave y desplegarla en formato exadecimal; y este sería la huella dactilar de la llave. A podría entonces llamar a B y preguntarle que dicte la huella dactilar por teléfono. Si las dos huellas dactilares concuerdan, entonces la llave está verificada.

3. Obtener la llave pública de un invididuo D en que ambos confíen.
   - Para esto D crea un certificado firmado.
   - El certifivado incluye: fecha de creación de la llave, periódo de validez de la llave,
   - D genera un digesto de 160 bits de su certificado, lo encripta con su llave privada,
   - y pega la firma a su certificado.
   - Como solo D puede crear la firma, nadie más puede crear un llave pública falsa y pretender que está firmada por D.
   - El certificado firmado podría enviarse directamente a A por B o D, o podría poderse dispobible en algún publicación.

4. Obtener la llave pública de un autoridad certificadora confiable.
   - De nuevo, se crea un certificado de llave pública y es firmado por la autoridad.
   - A tendría acceso a la autoridad, proveyendo un numbre de usuario y recibiendo el certificado firmado.

¿Cúales son los comandos de pgp?

- `https://www.gnupg.org`
- GnuPG viene en dos sabores: 1.4.17 es la versión bien conocida y portable
  2.0.23 es la versión mejorada y más dura de construir
- **gpg2** – OpenPGP encryption and signing tool
- El anillo de llaves se guarda por defecto en `~/.gnupg`
- Generar una clave: `gpg2 --gen-key`
- `gpg2 --list-keys`
- `gpg2 --list-secret-keys`

Una prueba para firmar algún documento

- Primero hay que crear las llaves
- Para firmar: `gpg2 -s [archivo]`
- Para verificar la firma: `gpg2 --verify [archivo]`

Cryptography and Network Security
Principles and Practice
Fifth edition
W. Stallings
Prentice Hall