Resumen Tema 4

Seguridad en redes

Autor: @BlackTyson

Índice

1. Introducción **2**
1.1. Aspectos de seguridad (CACNrID) . . . 2
1.2. Ataques de seguridad . . . 2
1.3. Mecanismos de seguridad . . . 2

2. Cifrado **3**
2.1. Características . . . 3
2.2. Cifrado simétrico . . . 3
2.3. Cifrado asimétrico . . . 5

3. Autenticación con clave secreta **6**

4. Intercambio de Diffie-Hellman **7**
4.1. Definición . . . 7
4.2. Procedimiento . . . 7
4.3. Vulnerabilidad Man-in-the-Middle (MitM) . . . 7

5. Funciones Hash (HMAC) **8**
5.1. Características Función HASH . . . 8
5.2. MD-5 y SHA-1: Resumen de funciones hash . . . 8

6. Firma digital **10**
6.1. Objetivos . . . 10
6.2. Firma digital con clave asimétrica. Doble cifrado . . . 11

7. Certificados digitales **11**

8. Protocolos seguros **12**
8.1. Seguridad . . . 12

1. Introducción

Una red es segura cuando se garantizan **todos los aspectos**. No hay protocolos 100 % seguros. La seguridad se debe **situar en todas las capas**.

1.1. Aspectos de seguridad (CACNrID)

- **Confidencialidad/privacidad**: El contenido es **comprensible sólo** para las **entidades autorizadas**.
- **Autenticación**: Las **entidades son** quien dicen ser.
- **Control de accesos**: Los **servicios son accesibles** solo para **entidades autorizadas**.
- **No repudio**: El sistema **impide la renuncia de autoría** de una acción.
- **Integridad**: el sistema **detecta** todas las **alteraciones** de la información.
- **Disponibilidad**: El sistema m**antiene las prestaciones** con **independencia** de la demanda.

1.2. Ataques de seguridad

Acción intencionada o no que vulnera cualquier aspecto de seguridad. Tipos

- **Sniffin** = vulnera la **confidencialidad** (husmeas)
- **Spoofin**(phishing) = **suplanta identidad** de entidades
- **Man_in_the_middle** = **interceptación**
- **Distributed Denial_of_Service (DDOS)** = **denegación de servicio** distribuido.
- **Malware**= troyanos, gusanos, spyware, ransomware, backdoors, rootkits, keyloggers...

1.3. Mecanismos de seguridad

- **Prevención:**
  - Autenticación
  - Control de acceso
  - Separación
  - Seguridad en las comunicaciones
- **Detección**:
  - IDS (intruder dectection system)
- **Recuperación**
  - Copias de seguridad
  - Mecanismos de análisis forense.

2. Cifrado

2.1. Características

- Basado en **criptografía**
- Garantiza la **confidencialidad**
- **Texto llano** se transforma en **texto cifrado**
- Se basa la existencia un **algoritmo de cifrado/descifrado** conocido como $E_K()$ y $D_{K'}()$.

2.2. Cifrado simétrico

- Existe **una sola clave para cifrar y descifrar** (k=k')
- **DES**: equema de sustitución **monoalfabético** (debilidad).
- **Doble DES**
  1. Se **cifra** el **bloque de datos** con la **primera clave** (K_1)
  2. **Se cifra** el **resultado** del paso anterior con una **segunda clave** (K_2)

  No mejora la seguridad frente a ataques de **man-in-the-middle**.
- **3DES**
  1. **Cifra** el bloque con la **primera clave** K_1.
  2. **Descifra** el **resultado** con la **segunda clave** K_2
  3. **Cifra** nuevamente el **resultado** con la **primera clave** K_1. También se puede emplear una **tercera clave** para mejorar la seguridad.

  Es mucho **más seguro** pero también mucho **más lento.**
- **IDEA** (International Data Encryption Algorithm)
  1. **Divide los datos** en bloques de 64 bits
  2. Se aplican **subclaves derivadas** de la clave principal
  3. **Sustituyen y permutan** constantemente

  Ventajas:

  - Excelente **seguridad**
  - **Menos recursos**

  Desventajas:

  - Sigue siendo un poco **lento**
  - Se utiliza en **sistemas antiguos**

- **AES** (Advance Encryption Standard)

  1. Se **dividen los datos** en bloques de 128 bits.
  2. **Cada bloque** pasa por **10, 12 o 14 rondas de transformación**
  3. Las rondas incluyen operaciones como sustitución de bytes, mezcla de columnas...

  Ventajas:

  - **Muy seguro**
  - **Rápido y eficiente**
  - Estándar más utilizando en la actualidad.

  Desventajas: **No tiene**

Tabla explicativa

Algoritmo	**Claves**	**Seguridad**	**Problemas**
DES	56 bits	Baja (vulnerable a fuerza bruta)	Clave corta para estándares modernos
Doble DES	112 bits	Mejor que DES, pero no ideal	Ataque del encuentro en el medio
3DES	112/168 bits	Alta (aún usado en sistemas antiguos)	Lento y menos eficiente que AES
IDEA	128 bits	Alta (resistente a ataques diferenciales y lineales)	Más lento que AES y usado en sistemas antiguos
AES	128/192/256 bits	Muy alta (estándar moderno, resistente a análisis diferencial)	Implementación incorrecta puede ser vulnerable a ataques de canal lateral

2.3. Cifrado asimétrico

Características

- Existe **dos claves por usuario** (A): una **pública** K_{PUB_A} y otra **privada** K_{PRI_A}.
- La **pública es conocida** y la privada **imposible de conocer.**
- Claves diferentes para cifrar y descrifar
  - **Cifrar**: $C = E_{K_{PUB_B}}(P)$
  - **Descifrar**: $D_{K_{PRI_B}}(C) = D_{K_{PRI_B}}(D_{K_{PUB_B}}(P)) = P$

RSA

1. **Generación de claves**:
   a) Elige **dos números primos grandes** (p y q)
   b) Se **calcula** $n = p \cdot q$. Este valor n será **parte de ambas claves** y representa el **rango** de valores cifrados.
   c) **Se calcula** $z = (p-1) \cdot (q-1)$. **Z es** conocido como la **función de euler de n** y se utiliza para **generar** el **exponente de cifrado y descifrado.**
   d) Se calcula **e** como $MCD(e, z) = 1$. Se puede verificar con el algoritmo de euclides.
   e) Se calcula **d** como $(ed)\,\%z = 1$ por medio del **algoritmo extendido de euclides**.
   f) **Se generan las claves**:
      - La clave **pública** es **(e,n)**
      - La clave **privada** es **(d,n)**
2. **Proceso de cifrado**
   a) Se transforma el mensaje **p** en un **número menor que n**
   b) Se **cifra** como $C = P^e\,\%n$ **Se usa la clave pública** (e,n)
3. **Proceso de descifrado**
   a) Se **recupera el mensaje original** en el receptor mediante $P = C^d\,\%n$

Ventajas:

- Permite **cifrado y autenticación** con **un único algoritmo**
- **Se puede compartir la clave pública** abiertamente.

Desventajas:

- **Más lento** que AES
- **Requiere generar números primos grandes**, lo que puede ser **computacionalmente costoso**.

3. Autenticación con clave secreta

Autenticar consiste en verificar la identidad de entidades involucradas. Se **recomienda usar nounces** (información de un solo uso)

Esquema reto-respuesta

- **Actores**:
  - **Cliente** (C): Entidad que desea autenticarse
  - **Servidor** (S): Entidad que valida autenticación
  - Ambos **comparten clave secreta** previamente establecida (K)
- **Procedimiento**:
  1. El **servidor envía un reto** (nonce):
     - El servidor genera un **valor aleatorio** (nonce)
     - El valor **se envía al cliente** para que lo **resuelva.**
  2. El cliente responde al reto:
     - El cliente **utiliza** la clave **secreta** para generar la **respuesta**
     - Aplica **función criptográfica** para **combinar** el reto y la clave secreta

       `Respuesta= f(Reto,K)`

     - La funcion f puede ser un algoritmo de cifrado un función hash
  3. **El servidor verifica la respuesta**
     - El **servidor aplica la misma función** al reto para calcular la respuesta
     - **Compara la respuesta** del cliente con su respuesta.
     - Si **ambas coinciden**, el cliente se **autentifica con éxito**.

4. Intercambio de Diffie-Hellman

4.1. Definición

Permite a dos partes **establecer una clave secreta** compartida a través de un **canal inseguro**, sin tener que enviar la clave directamene.

4.2. Procedimiento

1. Ambas partes **acuerdan públicamente dos valores**.
   - **g**: Una base (número primo pequeño conocido)
   - **n**: número primo grande

   Valores enviados por canal público
2. **Generación de claves privadas**. Cada parte genera un número privado secreto.
   - A elige x (clave privada)
   - B elige y (clave privada)
3. **Cálculo de claves públicas**. Usango g y n, cada parte calcula su clave pública y lo comparte con la otra:
   - **A calcula** $g^x \%n$ y lo **envía a B**
   - **B calcula** $g^y \$n$ y lo **envía a A**
4. **Cálculo de clave compartida**
   - Usando clave pública y privada, se calcula:
     - A: $(g^y \%n)^x \%n$
     - B: $(g^x \%n)^y \%n$
   - Usando **exponenciación modular**, ambas partes obtienen el **mismo valor**:

$$K_{AB} = g^{xy} \%n$$

4.3. Vulnerabilidad Man-in-the-Middle (MitM)

El intercambio de claves Diffie-Hellman es vulnerable a un ataque de tipo **hombre en el medio (MitM)** si no se implementa autenticación. En este ataque:

- Un atacante **intercepta las claves públicas** enviadas entre las partes A y B.
- El atacante genera sus **propias claves privadas** (z) y **públicas** (g^z mód n).
- A y el atacante establecen una clave secreta K_{AZ}, mientras que B y el atacante establecen K_{BZ}.
- El atacante puede descifrar, modificar y reenviar los mensajes sin que A y B lo detecten.

Medidas contra el ataque MitM

Para prevenir este ataque, se pueden tomar las siguientes medidas:

- **Autenticación de las partes:** Usar certificados digitales o firmas digitales para verificar que las claves públicas provienen de la parte correcta.
- **Integración con cifrado asimétrico:** Combinar Diffie-Hellman con RSA u otros métodos que permitan autenticar el intercambio de claves.
- **Protocolos seguros:** Utilizar protocolos como TLS/SSL, que incluyen autenticación mutua y cifrado.

5. Funciones Hash (HMAC)

5.1. Características Función HASH

- Funciones **unidireccionales** de cálculo sencillo
- Texto de entrada (**M**) de longitud variable
- Dado M se genera el **resumen H(M)**, de longitud física
- Es **imposible obtener M** a partir de H(M)
- Son **invulnerables a ataques de colisión**
- Se utiliza **HMAC** para garantizar **integridad y autenticación**.

5.2. MD-5 y SHA-1: Resumen de funciones hash

1. MD-5 (Message Digest 5)

Características principales:

- Longitud del resumen: **128 bits**.
- Procesa los datos en bloques de **512 bits**.
- Especificado en el **RFC 1321**.

Proceso:

1. **Relleno:** El mensaje se rellena con un bit "1"seguido de ceros hasta alcanzar una longitud congruente a 448 $\equiv$ mód 512. Se añade un campo de 64 bits que contiene la longitud original del mensaje.
2. **División en bloques:** El mensaje rellenado se divide en bloques de **512 bits**.
3. **Procesamiento secuencial:** Cada bloque se procesa utilizando un algoritmo iterativo con funciones f_F, f_G, f_H y f_I que combinan XOR, suma modular y rotaciones.
4. **Salida:** El estado final (128 bits) se concatena en un único valor hash.

Problemas:

- Vulnerable a ataques de colisión, por lo que no es adecuado para aplicaciones críticas como firmas digitales.

2. SHA-1 (Secure Hash Algorithm 1)

Características principales:

- Longitud del resumen: **160 bits**.
- Procesa los datos en bloques de **512 bits**.
- Publicado en 1993 por **NIST**.

Proceso:

1. **Relleno:** Igual que en MD-5: el mensaje se rellena con un bit "1"seguido de ceros hasta alcanzar 448 mód 512. Se añade un campo de 64 bits con la longitud original del mensaje.
2. **División en bloques:** El mensaje rellenado se divide en bloques de **512 bits**.
3. **Procesamiento secuencial:** Cada bloque pasa por un algoritmo iterativo con **4 funciones principales** (f_1, f_2, f_3, f_4), operando en 20 pasos cada una (total: 80 pasos). Se utilizan operaciones como rotaciones, XOR y suma modular.
4. **Salida:** El estado final de **160 bits** se concatena como el valor hash resultante.

Problemas:

- Más seguro que MD-5, pero también vulnerable a ataques de colisión.

3. Diferencias entre MD-5 y SHA-1

Característica	**MD-5**	**SHA-1**
Longitud del resumen	128 bits	160 bits
Procesamiento	4 funciones (64 pasos)	4 funciones (80 pasos)
Seguridad	Vulnerable a colisiones	Más seguro que MD-5, pero vulnerable
Estandarización	RFC 1321	NIST 1993
Aplicaciones	Histórico, no recomendado	Histórico, no recomendado

Cuadro 1: Comparación entre MD-5 y SHA-1

6. Firma digital

6.1. Objetivos

- Receptor pueda autenticar al emisor
- Garantizar **no repudio**
- El emisor tenga garantías de no **falsificación** por el destinatario

Firma digital con clave secreta y Big Brother (BB)

Hipótesis: En este esquema, existe una autoridad de confianza denominada **Big Brother (BB)** que:

- Comparte una clave secreta K_A con A.
- Comparte una clave secreta K_B con B.
- Posee una clave secreta interna K_{BB}, que solo conoce BB.

Proceso:

1. A genera un mensaje firmado utilizando su clave compartida K_A y envía:

$$A, K_A(B, R_A, t, P)$$

   Aquí:

   - B: Identidad del receptor.
   - R_A: Un valor aleatorio generado por A.
   - t: Marca temporal para evitar ataques de repetición.
   - P: El mensaje que se desea autenticar.

2. BB, al recibir el mensaje, verifica la validez del mismo usando K_A. Luego, genera un nuevo mensaje para B utilizando K_B y añade su propia firma interna $K_{BB}(A, t, P)$:

$$K_B(A, R_A, t, P, K_{BB}(A, t, P))$$

3. B, al recibir el mensaje de BB, verifica:

   - La autenticidad del mensaje mediante K_B.
   - La firma de BB para confirmar que el mensaje es válido.

Ventajas:

- BB actúa como una autoridad de confianza, facilitando la autenticación entre A y B.
- Las claves secretas K_A y K_B aseguran la confidencialidad en la comunicación.

Este esquema permite que A y B intercambien mensajes autenticados a través de BB, sin necesidad de compartir claves directamente.

6.2. Firma digital con clave asimétrica. Doble cifrado

- Procedimiento

  1. Se usa cifrado para proporcionar privacidad con K_{PUB_B}
  2. Se cifra de manera previa, para autenticación del firmante, con K_{PRI_A}
  3. La acción de firmar es el envío de $K_{PUB_B}(K_{PRI_A}(T))$
  4. El receptor recupera el texto mediante $K_{PUB_A}(K_{PRI_B}(K_{PUB_B}(K_{PRI_A}(T)))) = T$

- Este sistema garantiza

  - **Autenticidad del mensaje**. Solo puede provenir de A
  - **Integridad del mensaje**: Si el mensaje es alterado, el receptor no podrá verificarlo correctamente.
  - **Confidencialidad dle mensaje**: Al cifrar posteriormente el mensaje, se asegura que B puede descifrar el contenido
  - **No repudio**, garantizado por la firma de A. El receptor puede emostrar que A firmó el mensaje. Para poder hacer esto se necesita garantizar mediante **certificado digital** la asociación de A con su clave pública.

7. Certificados digitales

- **Identifica fehacientemente** una **identidad** con su **clave pública**
- Es necesario la intervención de una **Autoridad de Certificación(AC)**. Procedimiento

  1. El usuario **A obtiene sus claves** pública y privada.
  2. Envía una solicitud **firmada**, cifrada con clave privada a la AC, indicando su **identidad y su clave pública.**
  3. **AC comprueba** la firma y emite el certificado. Contiene la identidad de AC, la identidad de A, la clave pública, validez del certificado.
  4. Este proceso está **firmado digitalmente por AC**, para evitar falsedad

8. Protocolos seguros

8.1. Seguridad

Perimetral

Por medio de **firewalls** y sistemas de detección de intrusiones **(IDS)** y respuesta **(IRS)**

Protocolos

- Capa de **aplicación** (ejemplos):
  - Pretty Good Privacy**(PGP)**: e-mail seguro
  - Secure Shell **(SSH)**: acceso remoto seguro
- Capa de **transporte**
  - **Transport Layer Security**(TSL)
    - HTTPS
    - IMAPS
    - VPN
    - SSL-POP
  - TLS = Handshake+Protocolo
  - Garantiza confidencialidad, autenticación e integridad
- Capa de **red**: IPSec(VPN)
- Capas **inferiores**

TRANSPORT LAYER SECURITY

- **SSL Record protocol**: **encapsula los protocolos** y ofrece un **canal seguro**.
- **SSL Handshake Protocol**:
  - Negocia el **algoritmo de cifrado**.
  - Negocia la **función hash**
  - **Autentifica** al servidor
  - El cliente genera claves de sesión aleatorias o con Diffie-Hellman
- **SSL Assert protocol**: informa sobre **errores** en la sesión
- **Change Cipher Espec Protocol**: notifica **cambios de estado**.

IPSec

- Objetivo: Garantizar **autenticación**, **integridad** y (a poder ser) privacidad a nivel ip
- **Procedimiento 1**:
  - **Establecimiento de Asociación** de seguridad mediante el protocolo **IKE**
  - Establece una **clave secreta** (Diffie-Hellman)
  - Incluye **autenticación mutua** con **certificados**.
  - Es **simplex** (un único sentido)
  - Se identifica con IP origen + SPI
  - Vulnera el carácter no orientado a conexión de IP
- **Procedimiento 2**:
  - Garantiza **autenticación e integriad**
  - Usa el protocolo de **Çabeceras de autenticación"**
  - Usa **HMAC** con la Ks obtenida en IKE
- Procedimiento 3 (opcional):
  - Garantiza autenticación e integridad + privacidad
  - Protocolo .[En]capsulado de seguridad de la carga"
- **2 modos de operación**:
  - **Túnel**: asociación entre dos **router intermediarios**
  - **Transporte**: **extremo a extremo** entre host origen y destino.