

Taller Práctico de Criptografía: Integridad, Simetría y Asimetría

Objetivo: Aplicar los conceptos fundamentales de funciones Hash, cifrado simétrico y cifrado asimétrico mediante comandos reales de la industria para garantizar confidencialidad e integridad.

Parte 1: Integridad y Funciones Hash (Windows)

Las funciones Hash generan una "huella digital" única e irreversible de cualquier archivo. Si cambia un solo carácter del archivo, el hash cambia por completo.

1. Creación del documento confidencial:

Abra **PowerShell** en Windows y cree un archivo de texto con un mensaje corporativo:

PowerShell

```
echo "Reporte confidencial de ventas: 100,000 dolares" > reporte.txt
```

2. Generación del Hash inicial (SHA-256):

Calcule la huella digital del archivo utilizando la herramienta nativa de Windows:

PowerShell

```
Get-FileHash -Path .\reporte.txt -Algorithm SHA256
```

(Anoten el hash generado en un bloc de notas).

3. Prueba de alteración (Integridad):

Modifiquen ligeramente el archivo alterando una cifra:

PowerShell

```
echo "Reporte confidencial de ventas: 900,000 dolares" > reporte.txt
```

4. Validación del Hash modificado:

Vuelvan a calcular el hash:

PowerShell

```
Get-FileHash -Path .\reporte.txt -Algorithm SHA256
```

- **Pregunta:** ¿Coincide el hash nuevo con el que anotaron al inicio? ¿Por qué este mecanismo **no** se considera un cifrado?

Parte 2: Cifrado Simétrico con AES-256 (Ubuntu WSL)

El cifrado simétrico utiliza una **única clave secreta (contraseña)** tanto para cifrar como para descifrar. Es muy rápido, ideal para proteger volúmenes grandes de información.

1. Instalación y verificación de OpenSSL:

Abra su terminal de **Ubuntu WSL** y asegúrese de tener OpenSSL disponible:

Bash

```
openssl version
```

2. Creación del archivo secreto en Linux:

Bash

```
echo "Contraseña maestra del servidor de bases de datos: P@ssw0rdSuperSegura"  
db_pass.txt
```

3. Cifrado del archivo con AES-256:

Ejecute el comando de OpenSSL (el sistema les pedirá una contraseña secreta que ambos deben acordar):

Bash

```
openssl enc -aes-256-cbc -salt -in db_pass.txt -out db_pass.enc
```

4. Verificación de ilegibilidad:

Intente leer el contenido del archivo cifrado directamente:

Bash

```
cat db_pass.enc
```

(Verán caracteres extraños, demostrando que la información está protegida ante ataques de interceptación o Sniffing).

5. Eliminación del texto plano y Descifrado:

Bash

```
rm db_pass.txt
```

```
openssl enc -aes-256-cbc -d -in db_pass.enc -out db_recuperado.txt
```

```
cat db_recuperado.txt
```

Parte 3: Cifrado Asimétrico y Generación de Llaves

La criptografía asimétrica utiliza un par de llaves matemáticamente vinculadas: una **Clave Pública** (para cifrar o verificar) y una **Clave Privada** (para descifrar o firmar), asegurando que la llave privada **nunca** salga del equipo origen.

1. **Generación del par de llaves en Windows (Estudiante A):** Asegúrese de tener OpenSSL instalado (winget install openssl si aún no lo tiene). En PowerShell, genere su Clave Privada RSA de 2048 bits:

PowerShell

```
openssl genpkey -algorithm RSA -out mi_privada.pem -pkeyopt rsa_keygen_bits:2048
```

2. **Extracción de la Clave Pública:** A partir de la privada, extraiga la clave pública (el "candado" que compartirá con su compañero):

PowerShell

```
openssl rsa -pubout -in mi_privada.pem -out mi_publica.pem
```

Parte 4: Intercambio y Cifrado Asimétrico

Simulemos que el **WSL** quiere enviarle un mensaje secreto al **Windows** utilizando exclusivamente la Clave Pública de A.

1. Transferencia de la Clave Pública:

Como WSL comparte el sistema de archivos de Windows, el Estudiante B puede copiar la clave pública generada por el Estudiante A hacia su entorno de Linux (reemplace TuUsuario por su usuario real de Windows):

Bash

```
cp /mnt/c/Users/TuUsuario/mi_publica.pem ~/mi_publica.pem
```

2. Creación del mensaje secreto en WSL (Estudiante B):

Bash

```
echo "Plan de despliegue a producción aprobado para medianoche" >  
mensaje_confidencial.txt
```

3. Cifrado usando la Clave Pública del Estudiante A:

Bash

```
openssl pkeyutl -encrypt -pubin -inkey ~/mi_publica.pem -in mensaje_confidencial.txt  
-out mensaje_seguro.enc
```

(El Estudiante B puede ahora enviar mensaje_seguro.enc por cualquier red insegura sabiendo que solo el Estudiante A podrá leerlo).

4. Descifrado usando la Clave Privada en Windows: El Estudiante A toma el archivo cifrado y utiliza su llave maestra privada (que nunca abandonó su PC) para revelar el mensaje:

PowerShell

```
openssl pkeyutl -decrypt -inkey mi_privada.pem -in mensaje_seguro.enc -out  
mensaje_final.txt
```

```
Get-Content .\mensaje_final.txt
```

Preguntas de Análisis (Debate del Taller)

Al finalizar la ejecución del taller, discutan y respondan en su reporte de pareja:

1. Si un atacante intercepta mediante un ataque de *Sniffing* el archivo db_pass.enc (Parte 2), ¿puede leer su contenido sin conocer la contraseña simétrica? Justifiquen su respuesta.
2. En la Parte 4, si un atacante roba el archivo mi_publica.pem mientras es transportado, ¿podría utilizarlo para descifrar el mensaje secreto? ¿Para qué sirve realmente esa clave pública?
3. ¿Por qué en la práctica de cifrado asimétrico utilizamos un mensaje corto en lugar de cifrar un archivo de video de 2 Gigabytes directamente con RSA?