Práctica de laboratorio: Cifrar y Descifrar Datos con OpenSSL Objetivos

Parte 1: Cifrar mensajes con OpenSSL

Parte 2: Descifrar mensajes con OpenSSL

Aspectos Básicos / Escenario

OpenSSL es un proyecto de código abierto que proporciona un kit de herramientas robusto, comercial y completo para los protocolos de Seguridad de capa de transporte (Transport Layer Security, TLS) y Capa de sockets segura (Secure Sockets Layer, SSL). También es una biblioteca de criptografía de uso general. En esta práctica de laboratorio utilizarán OpenSSL para cifrar y descifrar mensajes de texto.

Nota: Si bien actualmente OpenSSL es la biblioteca de criptografía obligada, el uso que se presenta en esta práctica de laboratorio NO se recomienda para lograr una protección sólida. A continuación, se presentan dos problemas de seguridad con esta práctica:

1) El método que se describe en esta práctica utiliza una función de derivación de claves débiles. La ÚNICA medida de seguridad es una contraseña muy fuerte.
2) El método que se describe en esta práctica no garantiza la integridad del archivo de texto.

Esta práctica de laboratorio debe utilizarse solo con fines instructivos. Los métodos aquí presentados NO se deben emplear para asegurar datos realmente sensibles.

Recursos necesarios

- Máquina virtual "CyberOps Workstation"

Instrucciones

Parte 1: Cifrar mensajes con OpenSSL

OpenSSL se puede utilizar como una herramienta independiente para el cifrado. Aunque pueden utilizarse muchos algoritmos de cifrado, esta práctica de laboratorio se enfoca en AES. Si quieren utilizar AES para cifrar un archivo de texto directamente desde la línea de comando utilizando OpenSSL, sigan los pasos que se indican a continuación:

Paso 1: Cifrar un archivo de texto

a. Iniciar sesión en la Máquina Virtual "CyberOPS Workstation."

b. Abrir una ventana del terminal.

c. El archivo de texto que se debe cifrar se encuentra en el directorio /home/analyst/lab.support.files/, cambie a ese directorio:

```
[analyst@secOps ~]$ cd ./lab.support.files/
[analyst@secOps lab.support.files]$
```

d. Escribir el siguiente comando a continuación para generar una lista del contenido del archivo de texto encriptado **letter_to_grandma.txt** en la pantalla:

```
[analyst@secOps lab.support.files]$ cat letter_to_grandma.txt
Hi Grandma,
```

```
I am writing this letter to thank you for the chocolate chip cookies you sent
me. I got them this morning and I have already eaten half of the box! They
are absolutely delicious!


I wish you all the best. Love,
Your cookie-eater grandchild.
[analyst@secOps lab.support.files]$
```

e. En la misma ventana del terminal, emitir el siguiente comando para cifrar el archivo de texto. El comando usará AES-256 para cifrar el archivo de texto y guardar la versión cifrada como **message.enc**. OpenSSL nos pedirá una contraseña y pedirá que la confirmemos. Proporcionar la contraseña tal como se les solicita y recordarla.

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc -in
letter_to_grandma.txt -out message.enc
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
[analyst@secOps lab.support.files]$
```

Documentar la contraseña.

f. Cuando hayamos terminado el proceso, hay que volver a utilizar el comando **cat** para mostrar el contenido del archivo **message.enc**.

```
[analyst@secOps lab.support.files]$ cat message.enc
```

¿Se mostró correctamente el contenido del archivo **message.enc**? ¿Qué aspecto tiene? Explique.

g. Para que el archivo sea legible, volvamos a ejecutar el comando OpenSSL pero esta vez agreguemos la opción **-a**. La opción **-a** le indica a OpenSSL que debe cifrar el mensaje cifrado con un método de codificación diferente a Base64 antes de guardar el resultado en un archivo.

Nota: Base64 es un grupo de esquemas similares de codificación binario a texto que se utiliza para representar datos binarios en un formato de ASCII string

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc -a -in
letter_to_grandma.txt -out message.enc
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
```

h. Nuevamente, utilicen el comando **cat** para mostrar el contenido del archivo **message.enc**, que se acaba de generar otra vez:

Nota: El contenido de **message.enc** variará.

```
[analyst@secOps lab.support.files]$ cat message.enc
U2FsdGVkX19ApWyrn8RD5zNp0RPCuMGZ98wDc26u/vmj1zyDXobGQhm/dDRZasG7
rfnth5Q8NHValEw8vipKGM66dNFyyr9/hJUzCoqhFpRHgNn+Xs5+TOtz/QCPN1bi
08LGTSzOpfkg76XDCk8uPy1hl/+Ng92sM5rgMzLXfEXtaYe5UgwOD42U/U6q73pj
a1ksQrTWsv5mtN7y6mh02Wobo3A1ooHrM7niOwK1a3YKrSp+ZhYzVTrtksWDl6Ci
XMufkv+FOGn+SoEEuh7l4fk0LIPEfGsExVFB4TGdTiZQApRw74rTAZaE/dopaJn0
sJmR3+3C+dmgzZIKEHWsJ2pgLvj2Sme79J/XxwQVNpw=
```

```
[analyst@secOps lab.support.files]$
```

¿Ahora se muestra correctamente el archivo **message.enc**? Explique.

¿Podemos pensar en algún beneficio de tener **message.enc** codificado como Base64?

Parte 2: Descifrar mensajes con OpenSSL

Con un comando OpenSSL similar se puede descifrar **message.enc**.

a. Utilizar el siguiente comando para descifrar message.enc:

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc –a -d -in message.enc
-out decrypted_letter.txt
```

b. OpenSSL les pedirá la contraseña que se utilizó para cifrar el archivo. Vuelvan a introducir la misma contraseña.

c. Cuando OpenSSL termine de descifrar el archivo **message.enc**, lo guardará en un archivo de texto de nombre **decrypted_letter.txt**. Utilicen el comando **cat** para mostrar el contenido de **decrypted_letter.txt**:

```
[analyst@secOps lab.support.files]$ cat decrypted_letter.txt
```

¿La carta se descifró correctamente?

El comando que se utilizó para descifrar también contiene una opción -a. ¿Podemos explicarlo?