Confidencialidad con clave secreta en OpenSSL

Seguridad de Sistemas Informáticos

Primera sesión OpenSSL

Resumen

OpenSSL (`https://www.openssl.org/`)[1] es una implementación abierta de los protocolos SSL y TLS para comunicaciones seguras, proporcionando una infinidad de funciones utilizadas en la gestión de seguridad. Además puede usarse como librería criptográfica de propósito general.

1. OpenSSL: una herramienta de código abierto

Algunas de las funcionalidades disponibles en OpenSSL son las siguientes:

- Generación de números aleatorios.
- Utilidades numéricas: comprobación de primalidad, cálculo de inversos, etc.
- Cifrado y descifrado.
- Generación y gestión de claves privadas, públicas y parámetros.
- Evaluación de algoritmos criptográficos.
- Cálculo y verificación de resúmenes (funciones hash criptográficas) para la integridad de ficheros.
- Emisión y gestión de certificados X.509 y CRLs.
- Generación y verificación de marcas (sellado) de tiempo.
- Operaciones criptográficas de clave pública: firma electrónica.
- Testeo de clientes y servidores SSL/TLS.
- Gestión de correo S/MIME firmado o cifrado.

Contiene tres elementos diferenciados:

- la librería SSL `https://www.openssl.org/docs/man1.1.1/man7/ssl.html`,
- la librería Crypto `https://www.openssl.org/docs/man1.1.1/man7/crypto.html`

[1] Este documento se ha elaborado de acuerdo con la versión OpenSSL 1.1.1 (openssl version)

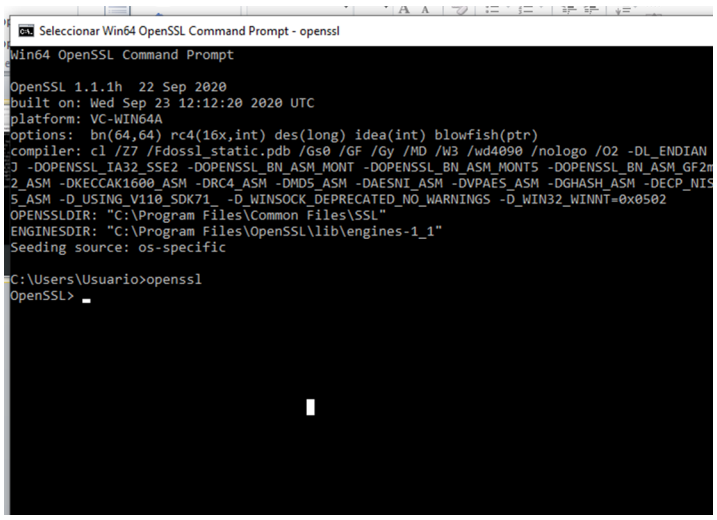

Figura 1: Entorno OpenSSL en Windows

- y una línea de comandos muy potente `https://www.openssl.org/docs/man1.1.1/man1/`. En las sesiones prácticas se hará uso de la línea de comandos.

Está disponible para diferentes sistemas operativos y permite el uso de funciones incluidas en su API en el desarrollo de aplicaciones seguras.

Desde el indicador del sistema operativo Linux puedes ejecutarlo en modo comando el programa tecleando OpenSSL, para Windows puedes encontrar los fichero ejecutables en `https://slproweb.com/products/Win32OpenSSL.html`.

2. Confidencialidad con clave secreta en OpenSSL: comando *"enc"*

La sintaxis del comando `enc` es la siguiente:

```
openssl enc -ciphername [-help] [-list] [-ciphers] [-in filename] [-out filename]
[-pass arg] [-e] [-d] [-a] [-base64] [-A] [-k password] [-kfile filename] [-K key]
[-iv IV] [-S salt] [-salt] [-nosalt] [-z] [-md digest] [-iter count] [-pbkdf2] [-p] [-P]
[-bufsize number] [-nopad] [-debug] [-none] [-rand file...] [-writerand file]  [-engine id]
```

1. *-ciphername* selecciona el cifrado a utilizar. El formato que usa OpenSSL para nombrar a los diferentes cifrados es: *name-keylength-blockoption*, siendo "name" el nombre del algoritmo, "keyleng" [2] la longitud usada por el algoritmo de cifrado y "blockoption" indica cuál es el **modo de cifrado** que usan los algoritmos de cifrado en bloque (ecb, cbc, ofb, cfb). Los dos últimos parámetros dependen del algoritmo de cifrado seleccionado.

2. *-in file.input*: especifica el nombre del fichero a cifrar. Por defecto se usa la entrada estándar.

[2] Si se requieren cadenas hexadecimales en algún argumento y se introduce una cadena de longitud inferior a la requerida, ésta se completa con ceros, si la longitud es superior, la cadena es truncada.

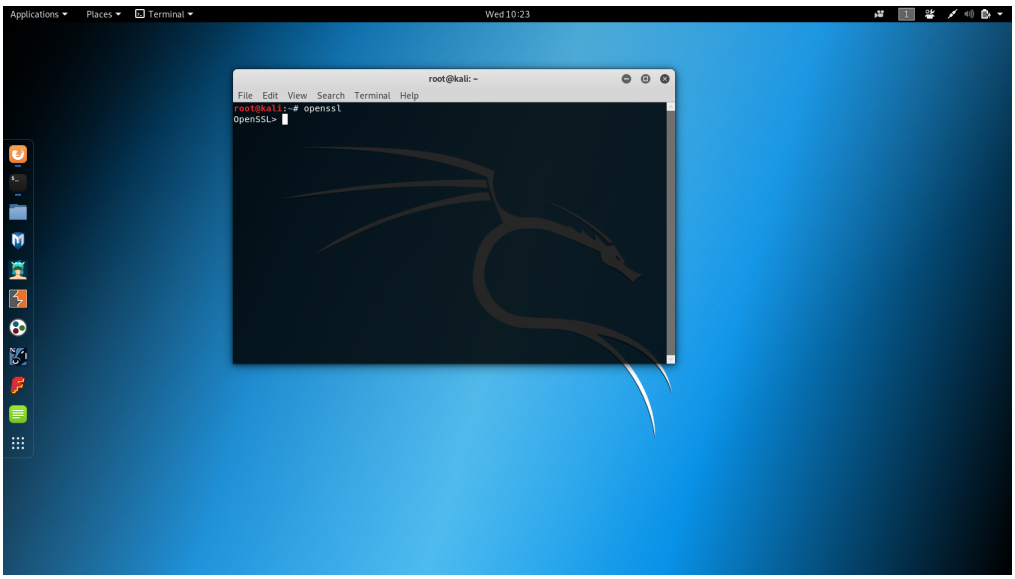

Figura 2: Entorno OpenSSL en Linux

3. *-out file.enc*: especifica el nombre del fichero resultante del cifrado. Por defecto se usa la salida estándar.

4. -e y -d: especifica si la operación a desarrollar es de cifrado (-e) o descifrado (-d). La operación por defecto es cifrado.

5. *Modos de especificar/derivar la clave*: todo esquema simétrico de cifrado requiere una clave para cifrar/descifrar (puesto que estamos en cifrados de clave secreta, la clave es la misma para cifrar que para descifrar) la información correspondiente. En OpenSSL este valor puede generarse a partir de una contraseña (solicitada al usuario) o especificada directamente a través de la línea de comandos. La función que se encarga de generar la clave a partir de la contraseña y la cadena usada para incrementar la entropía (salting) es EVP_BytesToKey() y está incluida en openssl/evp.c (`http://www.openssl.org/docs/crypto/EVP_BytesToKey.html`)[3]

   a) Opciones básicas:

   - *-k password*: se salta la solicitud de la clave y usa directamente desde la contraseña para la generación de la clave. Esta opción no se considera segura.
   - *-kfile password.file*: lee la contraseña del la primera línea del fichero especificado (password.file). Opción considerada más segura que la anterior.
   - *-K key*: selecciona la clave real directamente sin usar el proceso de generación. Debe ser una cadena expresada en hexadecimal. Cuando se utiliza esta opción, también se tiene que especificar el vector de inicialización, salvo que se se proporcione una contraseña para su generación. Este mecanismo de derivación de clave está especificado en PKCS5#5 (RFC-8018, RFC-8018).

   b) Opción de palabra de paso: Las opciones -k y -kfile continúan considerándose por temas de compatibilidad entre las diferentes versiones de OpenSSL y han sido sustituidas por el argumento *-pass*

[3] Usa la generación de funciones hash encadenadas.

(frase de paso) mejorando la seguridad. Si no se da ninguna contraseña y es necesaria, se pide al usuario que introduzca una a través del teclado en el instante de ejecutar el comando. El comando -pass tiene sus propias opciones, siendo las siguientes:

- *pass:password*: recibe la contraseña desde la línea de comandos.
- *env:var*: usa una variable de entorno como contraseña para generar la clave.
- *file:path*: lee la contraseña desde un fichero.
- *fd:number* lee la contraseña desde un descriptor de fichero, usado en redes y tuberías.
- *stdin* solicita la contraseña a la entrada estándar.

c) Otras opciones:

- *-iv initialvector* especifica el vector de inicialización usado en cifrados en bloque para evitar ataques de texto claro conocido (usado en algunos de los modos de cifrado). Es una cadena hexadecimal, obligatoria cuando se especifica la opción -K , a menos que se proporcione también una contraseña. En dicho caso, la clave se usa como clave de cifrado, mientras que la contraseña se usa para la generación de vector de inicialización. Si se usa con un algoritmo de cifrado en flujo este parámetro se ignora.
- *-md hash* selecciona la función hash usada en la generación de clave. Tanto MD5 como SHA1 están disponibles.
- *-iter count* Utiliza un número determinado de iteraciones sobre la contraseña para derivar la clave de cifrado. Los valores altos aumentan el tiempo necesario para averiguar por fuerza bruta la clave resultante. Esta opción permite el uso del algoritmo PBKDF2 para derivar la clave.
- *-pbkdf2*(Password-Based Key Derivation Function 2) [4]Usar el algoritmo PBKDF2 con el número de iteraciones por defecto, a menos que se especifique lo contrario.

6. Campo de aderezado: Ésta es una herramienta importante ya que mejora la aleatoriedad del proceso generación de la clave,consiguiendo valores de entropía muy altos. Es crucial incluir siempre este parámetro ya que mejora considerablemente la protección frente a ataques de fuerza bruta y ataques por diccionario. Para evitar inconsistencias en los comandos, cuando se especifican los argumentos -K y -iv, la opción de aderezado se ignora y el aderezado se inicializa con bytes aleatorios. La longitud del aderezado es de 8 bytes que se incluyen en el fichero resultante del cifrado precedido por la cadena “Salted” (otros 8 bytes) haciendo que el fichero resultante contenga 16 bytes más que el fichero sin aderezado.

   Hay tres clases de comandos de aderezado:

   - -salt permite la inserción de aderezado en el proceso de generación de clave;
   - -S salt especifica el aderezado a usar en lugar de permitir que OpenSSL lo calcule a partir de la contraseña. Este argumento debe especificarse como una cadena hexadecimal;

[4]Es una función de derivación clave con un costo computacional variable, utilizadas para reducir las vulnerabilidades a los ataques de fuerza bruta.

- -nosalt se usa sólo para mantener la compatibilidad. Desactiva la generación del aderezado. Se considera una opción muy insegura

7. Opciones menores:

   - -base64, -a, -A codifica el fichero en base 64 después de cifrarlo, esto es útil para enviar datos por correo electrónico.
   - -nopad deshabilita la opción de relleno. En este caso, la entrada de datos debe ser un múltiplo de la longitud del bloque de texto a cifrar dependiendo del cifrado seleccionado, en otro caso no se podrá descifrar el fichero correctamente.
   - -p muestra en pantalla el aderezado, la clave y el vector de inicialización en hexadecimal y realiza la operación de cifrado;
   - -P ejecuta el comando sin generar ningún fichero asociado a la operación de cifrado ni a la de descifrado, sólo se genera el aderezado, la clave y el vector de inicialización se y muestran en la pantalla en modo hexadecimal.
   - -buffer size define el tamaño del buffer para las operaciones de entrada y salida;
   - -debug muestra información de depuración para las operaciones de entrada y salida (por ejemplo, ficheros abiertos).
   - -z Comprimir o descomprimir texto claro usando zlib antes de la codificación o después de la descodificación.
   - -bufsize Establece el tamaño del buffer para E/S.
   - -list y -ciphers proporcionan un listado de los cifrados disponibles.
   - -help
   - -none Utiliza un cifrado NULL (sin cifrado o descifrado de la entrada).
   - -rand file Un archivo o archivos que contienen datos aleatorios utilizados para sembrar el generador de números aleatorios. Se pueden especificar múltiples archivos separados por un carácter dependiente del sistema operativo. El separador es ; para MS-Windows, , para OpenVMS, y : para todos los demás.
   - -writerand file Escribe datos aleatorios en el archivo especificado al salir.
   - -engine id Es la implementación de hardware o software utilizada para realizar operaciones criptográficas. El ID del motor por defecto es openssl y utiliza las funciones incorporadas de OpenSSL.