CREACIÓN Y ADMINISTRACIÓN DE CERTIFICADOS DE SEGURIDAD CON OPENSSL

INTRODUCCIÓN A LA CREACIÓN Y GESTIÓN DE CERTIFICADOS DIGITALES CON OPENSSL

Certificado digital

Un **certificado digital** es un archivo mediante el cual una **autoridad** de **certificación** garantiza la vinculación entre la **llave pública** y la identidad de una persona u organización

Autoridad certificadora

Una **autoridad** de **certificación**, **CA**, es una entidad de confianza responsable de emitir y revocar **certificados digitales** mediante criptografía de **llave pública**

Firma digital

La **firma digital** es un método criptográfico que asocia a una persona o equipo informático con un mensaje o documento

SSL/TLS

SSL, **S**ecure **S**ockets **L**ayer, y su sucesor **TLS**, **T**ransport **L**ayer **S**ecurity son protocolos criptográficos que proporcionan comunicaciones seguras dentro de una red, normalmente internet

Certificados x509

Estándar para infraestructuras de **llave pública**, **PKI**, que especifica el formato para los certificados de **llave pública**

OpenSSL

OpenSSL es una implementación libre, de código abierto, de los protocolos **SSL**, **S**ecure **S**ockets **L**ayer, y **TLS**, **T**ransport **L**ayer **S**ecurity mediante el cual es posible asegurar la capa de transporte para diferentes servicios HTTP, FTP, redes privadas virtuales, etc… y firmar digitalmente documentos

INSTALACIÓN OPENSSL EN UBUNTU

Por norma general, **openSSL** viene instalado por defecto en la distribución Ubuntu, en caso contrario, para instalar **openSSL** desde los repositorios oficiales ejecutamos sudo *apt-get install openssl*. Su configuración se realiza a través del archivo *openssl.cnf* ubicado en el directorio */etc/ssl*

Si se quiere utilizar interfaz, una de las más usadas es TinyCA que puede instalarse desde los repositorios con el comando
sudo apt-get install tinyca

CREACIÓN DE UN PAR LLAVE PÚBLICA/PRIVADA

Para crear y utilizar un método de comunicación seguro es necesario crear un par de **llave**s **pública** y **privada**

Para crear una **llave privada** ejecutaremos *openssl genrsa -des3 1024 > llave*

OpenSSL nos pedirá una frase de paso o contraseña. Cada vez que el servicio que utilice esta llave sea inicializado, se requerirá la introducción de la contraseña para habilitar el servicio

En determinados servicios que se inician de forma frecuente, o por comodidad, podemos crear una llave privada que no necesite contraseña mediante el comando *openssl genrsa > llave*

Para generar una **llave pública** a partir de la **llave privada** ejecutaremos *openssl rsa -in llave_privada -pubout -out llave_pública*

> NOTA El archivo que almacena la **llave privada** debe ser propiedad del usuario administrador y ser inaccesible para el resto de usuarios del sistema. Además, es necesario hacer una copia de seguridad de dicho archivo y guardarla en un lugar seguro, pues si dicho archivo fuera destruido cualquier certificado generado por él no podría utilizarse nunca mas

CREACIÓN DE UN CERTIFICADO

Una vez creada la llave privada, tenemos que crear un certificado, tanto para asegurar nuestra identidad ante el resto de usuarios como para informar a los mismos de nuestra **llave pública**

Existen dos tipos de certificados, los emitidos por un servidor de certificación o autoridad de certificación, CA, y los certificados auto firmados

CREACIÓN Y ADMINISTRACIÓN DE CERTIFICADOS DE SEGURIDAD CON OPENSSL

Creación de un certificado auto firmado

Para crear un certificado auto firmado listo para su uso en el servidor deseado ejecutamos

openssl req -new -key llave -x509 -days 365 -out archivo_crt

Creación de un certificado para su posterior firma por un servidor de certificación

Para crear un certificado que podrá ser usado en nuestros servidores cuando se firme por una CA ejecutamos

openssl req -new -key llave -out archivo_csr

CREACIÓN DE UN SERVIDOR DE CERTIFICACIÓN

A pesar de poder instalar de forma permanente certificados auto firmados, si poseemos una gran cantidad de servicios seguros, puede ser molesto para un usuario tener que instalar todos ellos, o bien, tener que aceptarlos cada vez

Para evitar esto, podemos usar **openSSL** para crear nuestra propia CA, **A**utoridad de **C**ertificación, así, los usuarios solo tendrán que descargarse el certificado auto firmado que nos acredita como autoridad de certificación

Crear la estructura del servidor de certificación

La estructura de un servidor de certificación constará de un directorio principal, solo accesible por el administrador, y dentro del cual crearemos los siguientes directorios y archivos

ESTRUCTURA BÁSICA DE UN SERVIDOR DE CERTIFICACIÓN

DIRECTORIO / ARCHIVO	DESCRIPCIÓN
certs	Directorio donde se almacenan los certificados ya firmados y enviados a los clientes
newcerts	Directorio donde se guardan los certificados recién firmados
crl	**C**ertificate **R**evokation **L**ist. Direcotrio donde se almacenan los certificados revocados
csr	**C**ertificate **S**igning **R**equest. Directorio donde se almacenan los certificados pendientes de firmar
private	Directorio donde se almacenarán la clave privada de la autoridad de certificación, así como las demás claves privadas de los diferentes servicios
serial	Archivo con el número de serie que tendrán los certificados firmados. Inicialmente tendrá el valor 01 seguido de un salto de línea
crlnumber	Archivo con el número que tendrá la siguiente lista de certificados revocados. Inicialmente tendrá el valor 01 seguido de un salto de línea
index.txt	"Base de datos" con información sobre los certificados firmados. Inicialmente tiene que estar vacío

NOTA tanto en el archivo *serial* como *crlnumber* pueden crearse con el comando *echo "01" > fichero*

Dentro de esta estructura podemos colocar el archivo de configuración de nuestro servidor de certificaciones. Este archivo se puede copiar directamente del archivo de configuración *openssl.cnf* ubicado en el directorio */etc/ssl*

En el fichero de configuración de nuestro servidor de certificación, normalmente, solo tendremos que modificar las secciones *[CA_default]* y *[req_distinguished_name]*

SECCIÓN [CA_default]

VARIABLE	DESCRIPCIÓN
dir	Directorio raíz del servidor de certificación
certs	Directorio donde se almacenarán los certificados ya firmados
crl_dir	Directorio donde se almacenan los certificados revocados
database	"Base de datos" de los certificados firmados
new_certs_dir	Directorio donde se guardan los certificados recién firmados
certificate	Archivo con la llave pública de nuestro servidor de certificación
serial	Archivo con el número de serie de los certificados
crlnumber	Archivo con el número de serie de revocación
crl	Lista de los certificados revocados
private_key	Llave privada del servidor de certificación

CREACIÓN Y ADMINISTRACIÓN DE CERTIFICADOS DE SEGURIDAD CON OPENSSL

SECCIÓN [REQ_DISTINGUISHED_NAME]

VARIABLE	DESCRIPCIÓN
countryName_default	País de emisión del certificado
stateOrProvinceName_default	Estado o provincia de emisión
localityName_default	Localidad de emisión del certificado
0.organizationName_default	Nombre de la organización
organizationalUnitName_default	Nombre de la sección
commonName_default	Nombre de la autoridad
emailAddress_default	Mail del responsable de la autoridad de certificación

Crear la llave y certificado del servidor de certificación

Para que nuestro servidor de autentificación pueda firmar certificados necesitamos crear una llave privada de nuestro servidor de certificación y un certificado auto firmado. Para crear ambos ejecutaremos

openssl req -new -x509 -config ruta_archivo_configuración -keyout ruta_llave_privada -out ruta_certificado

Si queremos comprobar la llave y el certificado generado ejecutamos respectivamente

openssl rsa -in ruta_llave_privada -text

openssl x509 -in ruta_certificado -text

Crear un certificado CSR y su llave privada para su firma por un servidor de certificación

Si queremos crear un certificado **CSR** para que una entidad de certificación lo firme y pueda usarse como un certificado válido ejecutamos

openssl req -new -nodes -keyout ruta_llave_privada -out ruta_certificado

Esta orden generará tanto la llave privada como el propio certificado **CSR**

NOTA recuerda que al utilizar la opción -*nodes* la llave no tendrá frase de paso

Firmar un certificado CSR a través de nuestro servidor de certificación

Para que un certificado **CSR** pueda ser usado como un certificado válido, primero tendremos examinar dicha solicitud, y si es correcta, firmarla. Para esto ejecutaremos

openssl req -in ruta_archivo_csr -text

openssl ca -config ruta_archivo_configuración -in ruta_archivo_csr -out ruta_certificado

NOTA de este modo solo podemos firmar certificados creados por nosotros mismos, de ma misma organización, para firmar certificados de cualquier organización tenemos que ejecutar

openssl ca -config ruta_archivo_configuración -in ruta_archivo_csr -out ruta_certificado -policy policy_anything

Revocar certificados

Todo servidor de certificación tiene la posibilidad de revocar algún certificado emitido. El motivo de revocar un certificado puede ser tan sencillo como la caducidad del mismo, o porque ya no se considera confiable.

Para revocar un certificado y actualizar la lista de certificados revocados ejecutaremos respectivamente

openssl ca -config ruta_archivo_configuración -revoke ruta_certificado

openssl ca -config ruta_archivo_configuracion -gencrl -out ruta_certificado_revocado

NOTA para comprobar que un certificado no ha sido modificado podemos comprobar su huella digital mediante el comando

openssl x509 -noout -fingerprint -in ruta_archivo_crt

Ahora tendríamos que comprobar el resultado obtenido con la huella digital del certificado almacenado en el directorio *newcerts* de nuestro servidor de certificación

Si queremos examinar la lista de certificados revocados ejecutaremos

openssl crl -in ruta_archivo_revocado -text

CREACIÓN Y ADMINISTRACIÓN DE CERTIFICADOS DE SEGURIDAD CON OPENSSL

NOTA la lista de certificados revocados debe ser puesta a disposición de los usuarios de la red de forma similar a como ponemos nuestra autoridad de certificación. Esto permite a los usuarios descargarse dicha lista e instalarla en sus ordenadores y así, mantener actualizados los certificados emitidos por nuestro servidor de certificación

OPERACIONES SOBRE CERTIFICADOS DIGITALES

OpenSSL permite realizar muchas operaciones criptográficas sobre certificados y con certificados como manipular certificados, encriptar y desencriptar datos, firmar documentos, etc…

Detalles de los certificados

Todos los comandos para conocer los detalles de los certificados tienen la forma *openssl X509 -noout -in archivo_crt* seguido de las siguientes opciones

CONOCER DETALLES DE LOS CERTIFICADOS

OPCIÓN	DESCRIPCIÓN
-subject	Muestra detalles sobre el dueño del certificado
-issuer	Muestra detalles sobre la autoridad de certificación que ha firmado el certificado
-dates	Detalles sobre la vigencia del certificado
-fingerprint	Muestra la huella digital del certificado
-serial	Muestra el número de serie del certificado
-text	Muestra todos los detalles de forma conjunta

Firma digital de archivos

Mediante una clave privada, openSSL permite firmar digitalmente archivos y posteriormente verificar dicha firma mediante los comandos respectivamente

openssl dgst -md5 -sign ruta_llave_privada -out firma_archivo archivo_a_firmar

openssl dgst -md5 -verify llave_pública -signature firma_archivo archivo_firmado

Cifrar y descifrar archivos

OpenSSL permite cifrar y descifrar archivos con **encriptación simétrica** y **asimétrica** de forma similar a como lo hace **GPG**. Con este método no hace falta generar ningún tipo de **llave pública** y **privada**

Cifrado simétrico

Para cifrar un archivo con **criptografía simétrica**,esto es, mediante una contraseña, utilizando el algoritmo **3DES**, uno de los más utilizados, ejecutamos

openssl enc -des3 -in archivo_original -out archivo_cifrado

Para descifrar dicho archivo ejecutaríamos

openssl enc -d -des3 -in archivo_cifrado -out archivo_descifrado

Cifrado asimétrico

Para cifrar archivos haciendo uso de **criptografía asimétrica** tendremos que utilizar la **llave pública** para cifrar archivos y la **llave privada** para descifrarlos

Creamos nuestro par de **llave**s **pública**/**privada**

openssl genrsa > llave_privada

openssl rsa -in llave_privada -pubout -out llave_pública

Ciframos un archivo con la **llave pública**

openssl rsautl -encrypt -pubin -inkey llave_pública -in archivo_original -out archivo_encriptado

Para descifrar dicho archivo con la **llave privada** ejecutamos

openssl rsautl -decrypt -inkey llave_privada -in archivo_encriptado -out archivo_desencriptado