Práctica 5: Creación de una Autoridad Certificadora (CA) con OpenSSL

Resumen

El objetivo de está práctica es generar una CA usando OpenSSL. Los pasos que se describen se refieren al desarrollo sobre Linux. Se debe generar un informe asociado a la práctica que será subido al la tarea correspondiente en el campus virtual.

1. Descripción

1.1. Generación de la Autoridad Certificadora

1. Instalar el OpenSSL para ello se escribe en la consola:

   ```
   Nombreequipo:~$ sudo apt-get install openssl
   ```

2. Vamos a crear una CA qué expedirá todos los certificados a los clientes que deseen conectarse con un determinado servidor web. Vamos a ser nosotros mismos nuestra CA, para ellos generamos un directorio con derechos de administrador:

   ```
   cd Directorio_de_trabajo
   Nombreequipo:~$ sudo mkdir certificados
   Nombreequipo:~$ cd certificados
   ```

3. Creación de la Autoridad certificadora (CA)

   Para poder crear un certificado primero tenemos que tener una CA (Autoridad Certificadora). Ésta será la que valide y confirme que nuestro certificado es válido.

   ```
   Nombreequipo:~$  openssl req -x509 -newkey rsa:2048 -days 1095 -keyout
   CAkey.pem -out CAcert.pem

   You are about to be asked to enter information that will be incorporated
   into your certificate request. What you are about to enter is what is
   called a Distinguished Name or a DN. There are quite a few fields but
   you can leave some  blank For some fields there will be a default value,
   Ifyou enter ?., the field will be lef in blank.

   Country Name (2 letter code) [GB]:ES

   State or Province Name (hill name) [Berkshire] : Tenerife
   ```

```
Locality Name (eg, city) [Newbury] :

Organization Name (eg, company) [My Company Ltd]:ULL

Organizational Unit Narne (eg, section) [] Dpto

Common Name (eg, your name or your servers hostnarne) [] :ull.es

Email Address [1: admin@eull.es
```

Descripción de los parámetros:

- `-keyout` determina el nombre del archivo donde guardaremos la clave privada de la CA, en este caso "CAkey.pem",
- `-out` determina el nombre del archivo donde guardaremos la clave pública, al que hemos llamado "CAcert.pem".
- `-days` indicamos el tiempo de validez expresado en días para el certificado de la CA. Este tendrá una validez de 3 años.

1.2. Generación del certificado del servidor

1. Ahora vamos a crear un certificado digital para el servidor, es decir con nuestra CA generamos y firmamos el certificado correspondiente:

   a) Primero generamos la clave privada del que sera nuestro certificado digital:

   ```
   openssl genrsa -des3 -out serv-priv.pem -passout pass:clave 2048
   ```

   Descripción de los parámetros:

   - `genrsa`: Genera una clave privada para el algoritmo de cifrado/firma RSA.
   - `-des3`: Genera una clave asociada al algoritmo de cifrado Triple DES para proteger la clave privada mencionada en el parámetro anterior.
   - `2048`: Longitud de la clave privada RSA, recomendable que sea 2048 bits
   - `-out`:Permite especificar el nombre del fichero donde se almacena la clave privada
   - `passout pass` Indica la clave con la que se cifra la clave privada para protegerla ante posibles robos.

   b) Antes de emitir un certificado X.509, debemos realizar una petición en la que se define el propietario del mismo. Para realizar esta tarea usaremos el parámetro `-req`.

   ```
   openssl req -new -subj "/DC=root.com/OU=com/CN=root" -key
   %  serv-priv.pem -passin pass:clave -out petic-certificado-serv.pem
   ```

   Descripción de parámetros:

   - `-subj`: Define quién es el propietario del certificado (CN es el nombre de la página que usará el certificado, OU es la organización y DC es el nombre del dominio privado).
   - `-key`: Se carga la clave privada generada anteriormente.
   - `-passin pass` ponemos el mismo password que utilizamos para proteger la clave privada, así nos permite acceder a ella.

- `-out`: Escribe la petición en el archivo petic-certificado-serv.pem.

c) Generamos un fichero de configuración denominado `config1.txt`, que contenga lo siguiente:

```
basicConstraints = critical,CA:FALSE

extendedKeyUsage = serverAuth
```

d) Finalmente se emite el certificado del servidor:

```
 openssl x509 -CA CAcert.pem -CAkey CAkey.pem -req
-in petic-certificado-serv.pem -days 15 -extfile config1.txt -sha1
-CAcreateserial -out servidor-cert.pem
```

Indicamos que el certificado es para un servidor usando el fichero de configuración generado en el paso anterior (-extfile config1.txt) y se utilizaremos la función hash SHA (-sha1).

Con esta instrucción creamos un certificado de servidor del tipo x509 usando la función hash `sha1`, con el certificado de la CA almacenado en `CAcert.pem`. También se usó la clave privada `CAkey.pem` y con los parámetros `-req` e `-in`, para indicar que el fichero `petic-certificado-serv.pem` es el que contiene las especificaciones de la petición. Además se ha estipulado un periodo de 15 días de validez con el parámetro `-days`. Con el parámetro `-CAcreateserial` se numera el certificado y con el parámetro `-out` se especifica el archivo de salida.

Ahora ya tenemos los archivos para utilizar en nuestro sitio web que son los siguientes:

`CAcert.pem`: Clave pública CA, `CAkey.pem`: Clave privada CA, `serv-priv.pem`: Clave privada servidor, `servidor-cert.pem` Certificado Servidor.

Ahora sólo nos queda configurar nuestro servidor web para que utilice OpenSSL.

1.3. Generación de los certificados de los clientes

Hasta ahora los ficheros que se han generado son:

- Asociados a la CA: `CAcert-pem, CAcert.srl, CAkey.pem`. [1]
- Asociados al certificado de servidor: `servidor-cert.pem, serv-priv.pem` (certificado y clave privada)

Ahora generaremos los certificados de los clientes que le tendremos que dar a los mismos para que puedan entrar en el sitio web.

1. Primero generamos la clave privada del cliente:

   ```
   openssl genrsa -des3 -passout pass:clave -out client-priv.pem 2048
   ```

   Se ha usado el algoritmo de cifrado triple des (-des3) generando una clave privada de 2048 bits que se almacenara en el fichero (-out) `client-priv.pem` y con el parámetro `-passout pass:` indicamos la passphrase para la calve privada de la CA que será clave.

2. Ahora generamos la petición del certificado:

[1] el fichero de extensión srl (Shared Recipient List) contendrá la lista de certificados emitidos.

```
 openssl req -new -key client-priv.pem -passin pass:clave -subj
"/DC=localhost/OU=com/CN=Fsv" -out petic-cert-client.pem
```

3. Antes de emitir el certificado, se debe editar un archivo específico de configuración openssl. En este caso se denomina `config2.txt`. Las líneas que debe contener son:

```
basicConstraints = critical,CA:FALSE

extendedKeyUsage = clientAuth
```

4. Finalmente ya se puede emitir el certificado:

```
 openssl x509 -CA CAcert.pem -CAkey CAkey.pem -req -in
petic-cert-client.pem -set_serial 3 -days 15 -extfile config2.txt
-out client-cert.pem
```

   Se ha indicado que será un certificado del tipo x509 cuya CA (-CA) está definida en el fichero CA-cert.pem y que usa como clave privada (-CAkey) el fichero CAkey.pem y que el certificado a generar tendrá las especificaciones definidas en el apartado anterior(-req -in), las cuales están en el fichero de petición petic-cert-client.pem.

   El certificado tendrá una validez de quince dias (-days 15),le indicamos que el certificado es para un cliente, y como esto lo tenemos en nuestro fichero de configuración se lo indicamos poniendo -extfile y nuestro fichero config2.txt y usamos la función hash SHA (-sha1).

   Una cuestión importante es el numerar los certificados emitidos. En el caso anterior, el certificado tendrá el numero 3; éste número debe ser diferente para cada certificado emitido puesto que a la hora de revocarlos openssl lo hace por el número de certificado.

1.4. Exportando los certificados de los clientes

Los certificados de clientes deben ser entregados a los usuarios para que puedan ser utilizados en su navegador. Por tanto, deben exportarse en un formato estándar que el navegador entienda: pkcs12.

```
 openssl pkcs12 -export -in client-cert.pem -inkey client-priv.pem
-certfile CAcert.pem -out cert-pck12.p12
```

Esto nos genera el archivo de salida `cert-pck12.p12` a partir de nuestro certificado cliente `client-cer.pem` que tiene como clave privada `client-priv.pem` y cuya entidad certificadora esta definida en `CAcert.pem`. El nombre `cert-pck12.p12` podemos cambiarlo a nuestro gusto para crear certificados personalizados para cada cliente. Este archivo .p12 es el que cada cliente debe añadir a su navegador para poder ver la página web vía https.

1.5. Definiendo la lista de revocación

Otra tarea que debe desarrollar la CA es revocar los certificados que han sido comprometidos, de manera que la perdida de un certificado no suponga entradas no autorizadas a clientes que no les hemos dado el certificado correspondiente. Para ellos se debe crear una lista de revocación de certificados (CRL).

1. Para poder definirla correctamente, debemos tener en el directorio `/etc/ssl` un archivo de texto llamado `index.txt`, el cual tendrá todas las entradas de certificados revocados (base de datos de certificados revocados).
2. Además se debe configurar correctamente el archivo `openssl.cnf` ubicado en`/etc/ssl`. A continuación se muestran las entradas que debe contener dicho fichero:

```
[ CA_default ]

dir= c:/openssl/bin # Directorio donde esta OpenSSl

certs= $dir/certs # El directorio de certificados (da lo mismo)

#crl_dir= $dir/crl #La dejamos comentada. CRL en raiz

database= $dir/index.txt #Donde tengamos nuestro fichero index.txt

#unique_subject= no #Lo dejamos en ?no? para poder crear certificados con el mismo subject

new_certs_dir = $dir/newcerts # Directorio de nuevos certificados(tampoco es importante)

certificate= $dir/CAcert.pem #Ruta del certificado de nuestra CA

serial= $dir/serial #El actual numero de serie para los certificados

#crlnumber= $dir/crlnumber #El numero actua de la lista de revocacion. Debe estar comentado

#crl= $dir/crl.pem #La actual lista de revocacion

private_key= $dir/CAkey.pem # Localizaci\'on de la clave privada

RANDFILE= $dir/.rand #Archivo para la generaci\'on aleatoria de numeros.

x509_extensions = usr_cert #Extension para a\~nadir a los certificados
```

3. Una vez tenemos este archivo bien configurado pasamos a crear nuestra lista de revocación o CRL de la siguiente manera.

```
openssl ca -gencrl -out listarev.crl
```

4. También deberemos crear el mismo archivo pero con extensión .pem

```
openssl ca -gencrl -out listarev.pem
```

5. Ahora ya tenemos la lista de revocación creada y ya podemos revocar los certificados cliente que queramos, para ello utilizaremos la siguiente instrucción para revocar un certificado.

```
openssl ca -revoke client-cert.pem
```

Con esto le estamos diciendo que revoque el certificado `client-cert.pem` que habíamos creado anteriormente. OpenSSL añadirá dicha entrada a la base de datos hasta que actualicemos nuestra lista de revocación que deberemos hacerla con el mismo comando utilizado para crearla.

```
-openssl ca -gencrl -out listarev.crl
```

2. Resultados

En el informe debes incluir las dificultades que has tenido así como los ficheros que se han generado en cada paso.

3. Material de apoyo

- Página de documentación de OPenSSL: `http://www.openssl.org/docs/`
- Tutorial web