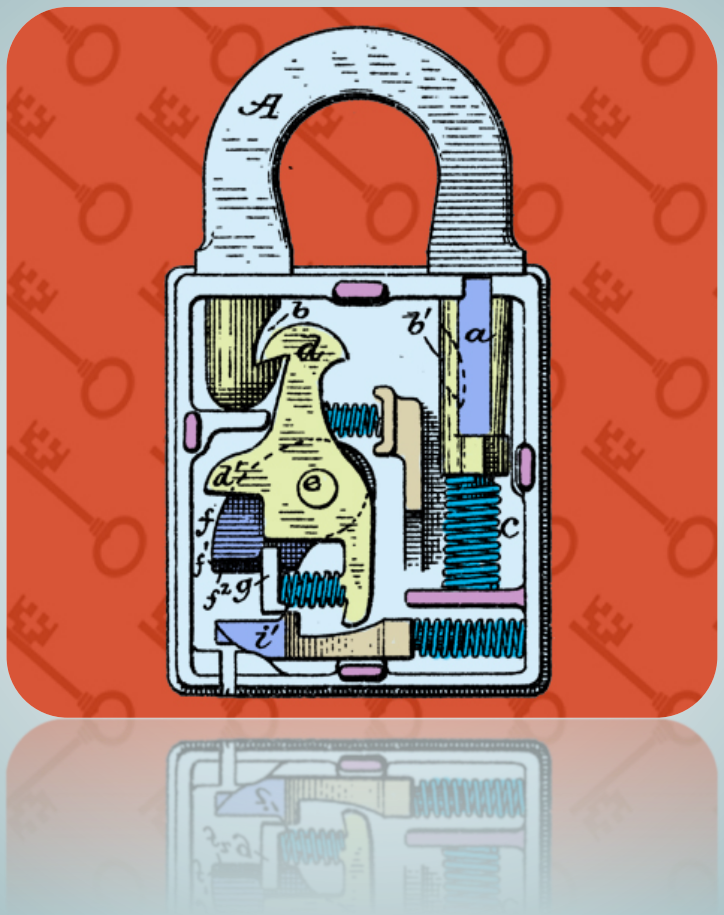
Garantía y Seguridad en Sistemas y Redes

Práctica 2. Gestión de una Autoridad Certificadora Privada



Esteban Stafford

Departamento de Ingeniería
Informática y Electrónica

Práctica 2

Gestión de una autoridad certificadora privada

1. Introducción

En internet es necesario tener un conjunto de entidades certificadoras raiz, en las que confía todo el mundo. Estas entidades firman las llaves públicas de sitios WEB que necesitan demostrar su autenticidad a sus clientes. Las entidades certificadoras raiz permiten que los usuarios de una WEB segura tengan la certeza de que estan accediendo a al sitio que piensan y no una falsificación del mismo.

En ocasiones conviene disponer de una entidad certificadora privada, al margen de el arbol de certificados mundial. Esto puede interesar para aplicaciones privadas en donde se tiene el control, tanto del software cliente, como del servidor. Por ejemplo si una empresa quiere gestionar el acceso a edificios de sus empleados mediante tarjetas inteligentes, no necesita que estas llaves estén firmadas por una entidad externa. O si otra empresa dispone de una plataforma hardware a la que quiere distribuir actualizaciones de software por la red, tanto los clientes como el servidor están totalmente bajo su control. En ambos casos estas empresas podrían recurrir a crear su propia entidad certificadora.

2. Objetivos

Los objetivos fundamentales que persigue esta práctica son los siguientes:

- Aprender la dinámica de trabajo de una autoridad certificadora
- Aprender a crear y revocar certificados
- Implementar un sistema de autentificación basado en llaves asimétricas

3. Desarrollo

3.1. Creación de una autoridad certificadora(CA) y certificación de una llave firmada

En la práctica anterior se vieron los mecanismos de creación y firma de certificados. Lo cual ya permite hacer uso de ellos en una aplicación real, como el servidor WEB. Con estos conocimientos, la creación y gestion de un CA es sencillo, ya que no deja de ser una base de datos de certificados con una llave privada y el correspondiente certificado autofirmado.

3.1.1. Despliegue del arbol de directorios y configuración del CA

Todos los archivos referentes a nuestro CA se aljoarán en un arbol de directorios. Crearlo es trivial:

```
mkdir -p  CA/certs CA/private CA/newcerts
```

Inicialmente el CA necesita tres ficheros. La configuración, que copiaremos de `/etc/ssl/openssl.cnf`, un fichero con un número, que servirá como contador para numerar los certificados que se han firmado, y finalmente un fichero vacío que será la base de datos del CA.

```
cp /etc/ssl/openssl.cnf CA/ca.conf
echo 1000 > CA/serial
touch CA/index.txt
```

En el fichero de configuración hay que escribir la ruta completa del directorio `CA` en la directiva `dir` (línea 42). Observa las política de firma en la sección `[policy_match]`. Esta política va a determinar los requisitos de las peticiones que se pueden firmar con este CA. Finalmente hay que proteger nuestro directorio `private` de la vista de otros usuarios.

```
chmod 700 CA/private
```

Para tener el CA operativo sólo resta crear un certificado autofirmado, que será el que se use para firmar todos las peticiones de certificado que lleguen al CA.

```
openssl req -new -x509 -days 3650 -extensions v3_ca \
   -keyout CA/private/cakey.pem -out CA/cacert.pem \
   -config CA/ca.conf
```

Analiza la información del certificado que acabamos de crear. ¿Cómo se puede ver que es autofirmado?

```
openssl x509 -in CA/cacert.pem -noout -text
```

Observa los cambios que se han producido en el directorio `CA`. ¿Aparte de los dos ficheros que acabamos de crear, se ha modificado algo?

3.1.2. Firma de una petición de certificado

Búsca en la práctica anterior como crear una llave y la petición de firma correspondiente. Para evitar confusiones y acentuar la diferencia entre los roles del solicitante de certificado y el CA, crea un directorio en la máquina anfitrión llamado `llaves`. Utiliza los comandos necesarios de la práctica anterior dentro de este directorio para crear una llave para Carol. Ten en cuenta la poítica del firma del CA, que has visto antes.

```
mkdir llaves;  cd llaves
openssl ...
   ... -out carol-req.pem
```

Para obtener el certificado lo tenemos que enviar al CA, que se encuentra en la máquina virtual.

```
scp carol_req.pem gssr@192...:
```

Supongamos somos de nuevo el CA y que nos han enviado la petición de Carol. Para firmarla:

```
openssl ca -config CA/ca.conf -out carol-cert.pem \
   -infiles carol-req.pem
```

Ahora deberíamos enviar el `carol-cert.pem` de vuelta para que el usuario lo pueda usar.

```
scp gssr@192...:carol-cert.pem .
```

Observa los pasos que sigue `openssl` para llevar a cabo la fima. Revisa los cambios que se han producido en el arbol de directorios del CA. ¿En qué lugar queda almacenado el certificado de Carol? ¿Y su llave privada? Explora el nuevo certificado. ¿En donde se puede ver la identidad del CA?

Finalmente debemos importar la llave privada y el certificado de Carol en el navegador. En firefox solo se puede hacer con llaves de tipo PKCS12, para convertir la llave y certificado recien creados a este formato, se puede usar el siguiente comando:

```
openssl pkcs12 -export -in carol-cert.pem \
   -inkey carol-key.pem -out carol.p12
```

3.2. Autentificación de clientes en apache con SSL

Una vez creado el certificado de Carol, vamos a configurar apache para que sólo se pueda accedr por `https` presentando un certificado de nuestro CA. Esto es sencillo, cambiando en `001-default-ssl` la siguiente línea:

```
SSLVerifyClient require
```

Tras reiniciar apache, ¿se puede acceder a `https`? ¿Qué errores se producen? ¿A qué son debidos?

Lee los comentarios del fichero de configuración y modifícalo para poder establecer una conexión `https` correctamente.

3.3. Revocación de certificados emitidos por el CA

Los certificados que hemos generado hasta el momento incluyen fechas de validez, de manera que un usuario es avisado cuando accede a un sitio con un certificado caducado. En cualquier caso, los certificados pueden invalidarse de manera unilateral por parte del CA, basta con incluirlos en una lista de certificados revocados. En inglés *Certificate Revocation List (CRL)*.

3.3.1. Creación de una CRL inicial

```
mkdir CA/crl
echo 1000 > CA/crlnumber
openssl ca -config CA/ca.conf  -gencrl  -out CA/crl.pem
```

Observa cómo se ha modificado el directorio `CA`. ¿Por qué ha sido necesario introducir la clave de la llave privada? Se puede ver la lista de revocación `crl.pem` con el comando:

```
openssl crl -in CA/crl.pem -noout -text
```

3.3.2. Revocación de un certificado

Ahora cada vez que sospechemos de la seguridad de un certificado, lo podemos revocar con el siguiente comando:

```
openssl ca -config CA/ca.conf -revoke CA/newcerts/1000.pem
```

Observa lo que ha sucedido en `CA`. ¿Qué ha cambiado en el fichero `index.txt`? ¿Se ha actualizado el fichero `crl.pem`?

3.3.3. Publicación de un CRL actualizado

Para que los cambios de validez de los certificados se reflejen en el `crl.pem`, hay que rehacer la lista de revocación.

```
openssl ca -config CA/ca.conf  -gencrl  -out CA/crl.pem
```

Observa ahora el `crl.pem`. ¿Qué información se usa para hacer referencia a los certificados revocados?

3.4. Configuración de CRL en apache

Se ha revocado el certificado de Carol, pero se sigue pudiendo hacer la autentificación por `https`. Esto es debido a que apache no está comprobando si el certificado de Carol esta en la CRL. De nuevo revisa el fichero de configuración de apache y modifícalo para que lea el CRL y no se pueda establecer la conexión con el certificado de Carol.

3.4.1. Opcional: Acceso a contenido del certificado desde PHP

Hasta ahora, apache se ocupa de realizar la autentificación de los clientes a través de su certificado. Una manera de implementar un control de acceso, al margen de apache, sería a traves de PHP. Para ello tenemos que conseguir que apache pase a los scripts de PHP la información del certificado del cliente. Esto se hace a través de variables de entorno y se activa añadiendo `+ExportCertData` a las opciones SSL en el fichero de configuración de apache.

```
<FilesMatch "\.(cgi|shtml|phtml|php)$">
   SSLOptions +StdEnvVars +ExportCertData
</FilesMatch>
```

Despues un script de PHP como el siguiente es todo lo que se necesita para hacer una autorización rudimentaria.

```
<?php
   session_start();
   $authorized_users = array("esteban","carol");
   $cert_data = openssl_x509_parse($_SERVER['SSL_CLIENT_CERT']);
   $commonname = (  is_array($cert_data['subject']['CN'])
                                  ? $cert_data['subject']['CN'][0]
                                  : $cert_data['subject']['CN']);
   if (in_array($commonname, $authorized_users)) {
?>
   Hola <?=$commonname?>
<?php
   } else {
?>
   No me simpatizas!!
<?php
   }
?>
<pre>
   <?php print_r($cert_data) ?>
<\pre>
```