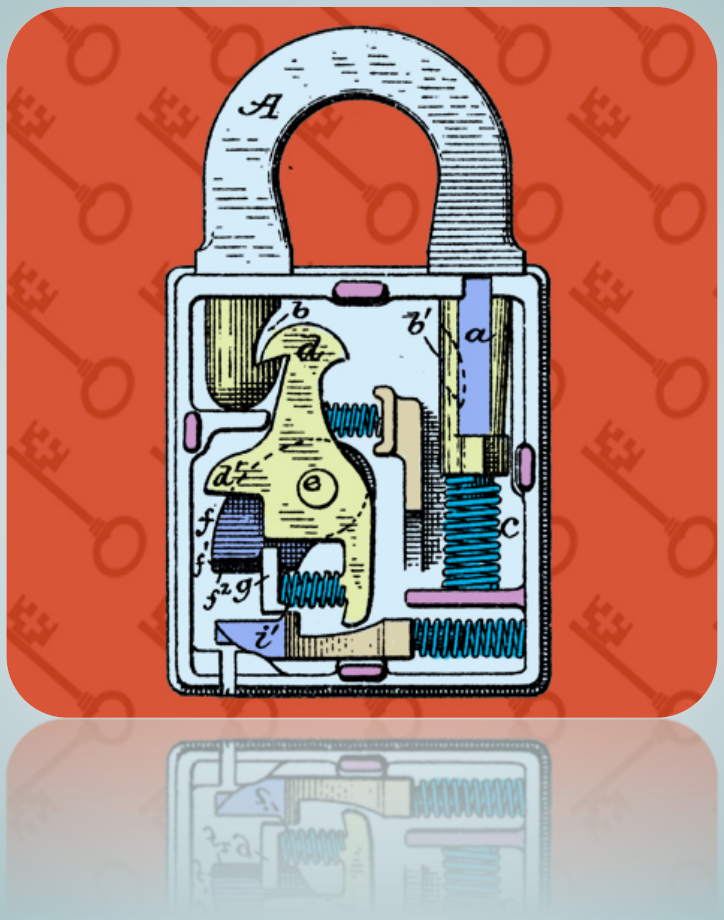
Garantía y Seguridad en Sistemas y Redes

Práctica 1. Gestión y Uso de Certificados



Esteban Stafford

Departamento de Ingeniería
Informática y Electrónica

Práctica 1

Gestión y uso de certificados

1. Introducción

2. Objetivos

Los objetivos fundamentales que persigue esta práctica son los siguientes:

- Conocer los comandos de gestion de claves de OpenSSL
- Instanciar un certificado autofirmado
- Emitir un certificado y ponerlo en uso en un servicio (https)
- Explorar el arbol de certificados de un certificado real

3. Desarrollo

3.1. Creación de un entorno virtual de trabajo

En la carpeta `/extra/gssr` se encuentra una máquina virtual de VirtualBox comprimida en el archivo `gssr.tar.gz`. Para trabajar con ella es necesario descomprimir dicho archivo. Lo haremos dentro de la propia carpeta `/extra/gssr` y le cambiaremos el nombre para identificarla como nuestra. Los siguientes comandos realizan esta tarea.

```
cd /extra/gssr/
tar -xzvf gssr.tar.gz
mv gssr $USER
```

Ahora se puede añadir la máquina a VirtualBox. Antes de arrancarla es necesario configurar una red que conecte la máquina virtual con el anfitrión. Esto se hace pulsando en el botón + en el dialogo de preferencias: File¿Preferences¿Network¿Host-only Networks. Durante el arranque vmplayer mostrará varios avisos que no es necesario tener en cuenta.

La máquina virtual tiene instalado un sistema operatvo ubuntu 14.04.3 LTS, configurado como servidor. Es decir que no tiene los componentes de interfaz gráfica típicos de una instalación para ordenador de escritorio o portátil. Las credenciales de acceso son:

```
Login: gssr
Password: gssr
```

La máquina tiene instalado un servidor LAMP en el puerto 80 y uno `ssh` en el 22. Para poder probar estos servicios tenemos que primero averiguar la direccion IP de la máquina virtual. La manera más sencilla es entrar por la consola virtual y consultarla con

la utilidad `ifconfig`. Hay que tener en cuenta que la máquina virtual tiene dos interfaces de red: el primero para conexiones exteriores y el otro para conectar con el anfitrión. Es la dirección de este segundo interfaz la que nos interesa en este caso.

Como la consola virtual no es excesivamente cómoda, conociendo la dirección IP podemos conectarnos a la máquina virtual a través de `ssh`. Además tambien podemos probar el servidor LAMP con el navegador. Este último debe mostrar una página web con el título *It works*.

```
ssh gssr@172.31.188.30
Password: <gssr>
```

3.2. Creación de una llave RSA

Vamos a crear dos llaves para los dos personajes habituales de la criptograf´ıa, Alice y Bob. Las crearemos de diversas maneras, pero su longitud será siempre 1024 bits. El manejo de llaves se realiza de manera muy cómoda con los comandos incluidos en la librería OpenSSL (http://www.openssl.org/).

3.2.1. Llave de Alice (Sin contraseña)

El comando para generar la llave sin contraseña es sencillo:

```
openssl genrsa -out alice.pem 1024
```

Si no se especifica la longitud de la llave se toma 512 bits. El fichero `alice.pem` contiene tanto la llave privada como la pública. Este fichero tiene que mantenerse en secreto y no difundirlo en absoluto. Para inspeccionar el contenido del fichero se usa el comando:

```
openssl rsa -in key.pem -noout -text
```

Observa los diferentes componentes del fichero de llave. Sobre esto se puede encontrar más información en la web (http://www.ietf.org/rfc/rfc3447.txt)

3.2.2. Llave de Bob (encriptada)

Para ayudar en la custodia de la llave privada, el fichero `.pem` se puede encriptar en el momento de su creación con el comando:

```
openssl genrsa -des3 -out bob_enc.pem 1024
```

Estudia el contenido del fichero `bob_enc.pem`. ¿Que tipo de encriptación se usa para proteger la clave privada? ¿Es simétrtica o asimétrica? ¿Es práctico utilizar encriptación asimétrica?

3.2.3. Encriptación de una llave existente

Una vez creadas las llaves se pueden encriptar con el comando:

```
openssl rsa -des3 -in alice.pem -out alice_enc.pem
```

Del mismo modo se puede eliminar la clave de una llave encriptada:

```
openssl rsa -in alice_enc.pem -out alice_2.pem
```

Compara los ficheros `alice.pem` y `alice_2.pem`. ¿Son iguales?
Tambien es útil poder cambiar la clave de una llave encriptada:

```
openssl rsa -des3 -in bob_enc.pem -out bob_enc_new.pem
```

3.2.4. Extracción de llave pública

En el los ficheros de llave de Alice y Bob se encuentra tanto la llave privada como la pública. Para obtener un certificado de una de estas llaves es necesario enviar la llave pública a la autoridad certificadora (La privada no se puede enviar a ningún sitio) en un fichero independiente.

```
openssl rsa -in alice.pem -pubout -out alice_pub.pem
openssl rsa -in bob_enc.pem -pubout -out bob_pub.pem
```

3.2.5. Creación de una petición de certificado

Como se ha visto en clase el estándar de distribución de llave pública x509 estipula que un certificado se compone de la llave pública, de los datos de identificación del propietario y de una firma de una autoridad certificadora. Entonces la creación de un certificado consta de dos fases. Primero se crea una petición de certificación, en la que el propietario de la llave aporta sus datos de identificación, y segundo, la autoridad certificadora procesa la solicitud, firmandola y devolviendo el certificado completo.

Aunque la petición no contiene la llave privada, es requisito para la generación de la petición, ya que ésta va firmada.

```
openssl req -new -key alice.pem -out alice_req.pem
openssl req -new -key bob_enc.pem -out bob_req.pem
```

Y para ver el contenido del certificado:

```
openssl req -in alice_req.pem -noout -text
```

Observa el contenido del certificado y búsca qué información de la llave original aparece ahí.

El certificado de Alice se puede generar, tanto de la versión encriptada como de la no encriptada. ¿Se genera el mismo certificado?

Para comprobar el la corrección de un certificado se puede hacer:

```
openssl req -verify -in alice_req.pem -noout -text
```

Pero tambien se puede comprobar la firma del certificado. Para ello hay que usar el fichero de llave.

```
openssl req -verify -in alice_req.pem -key alice.pem -noout -text
```

Comprueba qué sucede si tratas de validar el certificado de Alice con la llave de Bob. En este proceso de verificación, ¿se usa la llave privada?

3.2.6. Firma de un certificado

El paso final para la obtención de un certificado es la firma por parte de la autoridad certificadora. Habitualmente se envía a ésta la petición, como las que hemos hecho en el aparado anterior, y la autoridad nos devuelve el certificado firmado. Como no estamos en contacto con ninguna autoridad certificadora, vamos a firmar los certificados nosotros mismos. Como si fuésemos una autoridad raiz.

```
openssl req -x509 -key alice.pem -in alice_req.pem -out \
>   alice_cert.pem -days 365
```

¿Podemos firmar la petición de Bob con la llave de Alice?
Como apunte para el futuro, la creación de un certificado autofirmado se puede hacer con un sólo comando:

```
openssl req -x509 -new -key key.pem -out selfcert.pem -days 365
```

Tambien es posible crear la llave privada y el certificado autofirmado de una vez. Tanto no encriptado como encriptado:

```
openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout \
   key.pem -out selfcert.pem
openssl req -x509 -days 365 -newkey rsa:1024 -keyout \
   enc\_key.pem -out selfcert.pem
```

Además las herramientas de OpenSSL permiten convertir llaves y certificados entre diferentes formatos.

3.2.7. Utilización de llaves en `ssh`

El protocolo de acceso remoto `ssh` ejecuta comandos en un equipo remoto a través de la red de manera segura. Es decir, que el tráfico entre ambos equipos es encriptado. Cuando se abre la conexión, se pide al usuario una clave de acceso. Esta es la correspondiente a la cuenta del usuario en el equipo remoto. Sin embargo `ssh` permite realizar la autentificación con clave privada. De manera que si ésta no está protegida por clave, se puede acceder al equipo remoto sin teclear la clave cada vez.

Aunque seguro que se pueden utilizar las llaves generadas anteriormente, `ssh` tiene la herramienta `ssh-keygen` que hace la creación de la llave muy sencillo. Tras ejecutar el

comando aparecen dos ficheros en el directorio `.ssh`: `id_rsa` y `id_rsa.pub`. Para acceder al servidor virtual con esta nueva clave hay que añadir el contenido de `id_rsa.pub` al fichero `.ssh/authorized_keys` en el servidor virtual. A partir de este momento debería ser posible conectarse al servidor virtual sin clave.

¿Cómo podemos hacer para conectar desde la máquina virtual a la física sin teclear contraseña?

3.2.8. Fundamentos de administración de apache

Uno de los usos más típicos de uso de llaves asimétricas es en la creación de servicios web seguros con `https`. Este protocolo se suele servir por el puerto 443. Vamos configurar el servidor apache que se encuentra en la máquina virtual para que use la llave de Alice.

Afortunadamente, la configuración apache viene preparada para que activar `https` séa muy sencilla.

Antes de comenzar conviene saber que apache es un servidor http creado en los años noventa y ha evolucionado mucho desde entonces. Inicialmente su configuración se hacía en un sólo fichero `httpd.conf`. Sin embargo en la versión actual este fichero está vacio. Esto indica que el servidor utilizará una configuración modular. El fichero principal es `apache.conf` en donde se especifican aspectos generales de la configuración. Luego la configuración se divide en dos parejas de directorios: `mods-available`, `mods-enabled`, `sites-available` y `sites-enabled`. El primero contiene configuraciones especificas de los módulos de apache, el segundo contiene enlaces simbólicos a las configuraciones correspondientes a los módulos que se quiere activar. Del mismo modo `sites-available` contiene la configuración de uno o varios servicios, o servidores virtuales, y `sites-enabled` enlaces a las configuraciones de los sitios activos. El administrador del sistema simplemente tiene que crear o eliminar enlaces simbólicos para activar o descativar módulos o servicios. No hay que olvidarse de que cuando se cambie la configuración de apache hay que recargarla en el demonio con el comando.

```
/etc/init.d/apache2 reload
```

Como todo demonio, apache escribe información de su estado, errores o incidencias en un fichero. Concretamente, en la carpeta `/var/log/apache2`. El fichero `access.log` muestra la lista de conexiones realizadas y `error.log` muestra mensajes de error. Conviene revisar este último cuando se modifica la configuración, ya que puede ser que por un error apache no esté funcionando correctamente.

3.2.9. Configuración de `https` en apache

Para que apache arranque con el módulo ssl, hay que crear dos enlaces en `mods-enabled`:

```
cd /etc/apache2/mods-enabled
ln -s ../mods-available/ssl.* .
```

Y para activar el servicio de `https` basta con enlazar el fichero `sites-available/default-ssl` en `sites-enabled`.

```
cd /etc/apache2/sites-enabled
ln -s ../sites-available/default-ssl 001-default-ssl
```

Finalmente hay que declarar la llave y el certificado en el fichero `001-default-ssl`. Para ello crea unas carpetas en el propio directorio de configuración de apache y copialas ahí.

```
mkdir -p /etc/apache2/ssl/certs
mkdir -p /etc/apache2/ssl/private
cp bob_cert.pem alice_cert.pem /etc/apache2/ssl/certs
cp bob_enc.pem alice.pem /etc/apache2/ssl/private
```

Luego modifica `001-default-ssl` donde aparece `SSLCertificateFile` y `SSLCertificateKeyFile`, configurando el certificado de Alice. Finalmente reinicia apache y prueba a navegar.

Cuando hayas conseguido ver la página en el navegador, muestra la información del certificado del servidor y contrastala con lo que has observado en apartados anteriores.

Prueba a configurar el apache con el certificado de Bob.

Ahora usa el navegador para explorar la información de los certificados de otros sitios web.