PKI- SEGUNDA PARTE

SRDSI

Jose Ángel Gumiel

Índice

Introducción

En este laboratorio se va a continuar trabajando con la autoridad de certificación (CA) que fue creada para la primera parte del certificado. Recordar que la CA es una entidad auto-firmada que tiene la capacidad de gestionar certificados. Puede crear y firmar nuevos certificados y revocar aquellos que hayan sido comprometidos, caducados….

La práctica está orientada a usar trabajar con certificados para páginas web, y se usará también OCSP (Online Certificate Status Protocol). OCSP sirve para determinar el estado de vigencia de un certificado X.509 sin tener que usar CRLs (Certificate Revocation List).

Entre algunas de las ventajas de un OSCP frente a las CRLs destacan:

- OCSP puede proporcionar una información más adecuada y reciente del estado de revocación de un certificado.
- OCSP elimina la necesidad de que los clientes tengan que obtener y procesar las CRL, ahorrando de este modo tráfico de red y procesado por parte del cliente.
- OCSP soporta el encadenamiento de confianza de las peticiones OCSP entre los "responders". Esto permite que los clientes se comuniquen con un "responder" de confianza para lanzar una petición a una autoridad de certificación alternativa dentro de la misma PKI.

Entorno de trabajo

Se va a trabajar con la misma máquina virtual del laboratorio anterior. Se trata en una distribución de Debian 8 que se ejecuta sobre VMWare WorkStation 11.

Desarrollo

La finalidad de la práctica es utilizar los certificados sobre páginas web seguras, es decir, que se usará el protocolo HTTPS. Será necesaria la creación de sitios virtuales, por lo que se usará Apache 2.5.

1. En la primera fase se dejó en funcionamiento un servicio de OCSP. Se comprueba que éste sigue funcionando. Se ha usado el puerto 50000, que no está reservado.

```
root@srdsi:~/pki# openssl ocsp -port 50000 -index db/index.txt -CA certs/ca-root
.crt -rsigner certs/certificado-ca-ocsp.crt -rkey private/clave-privada-ca-ocsp
-text -out log.txt
Waiting for OCSP client connections...
```

2. Se crea un sitio virtual que trabaje bajo el protocolo HTTPS, por eso el puerto de será el 443. Hay que activar SSL y añadir las rutas a los certificados y llaves.

```
  GNU nano 2.2.6  Fichero: /etc/apache2/sites-available/srdsi.conf

<VirtualHost *:443>root
        ServerName              www.srdsi.lab
        DocumentRoot            /var/www/srdsi
        SSLEngine               On
        SSLCertificateFile      /root/pki/certs/serverCert.crt
        SSLCertificateKeyFile   /root/pki/private/serverCert.key
        SSLCertificateChainFile /root/pki/certs/ca-root.key
</VirtualHost>
```

Nota: Es necesario activar el módulo de SSL para que funcione, además del sitio.

```
a2enmod ssl
a2ensite srdsi
```

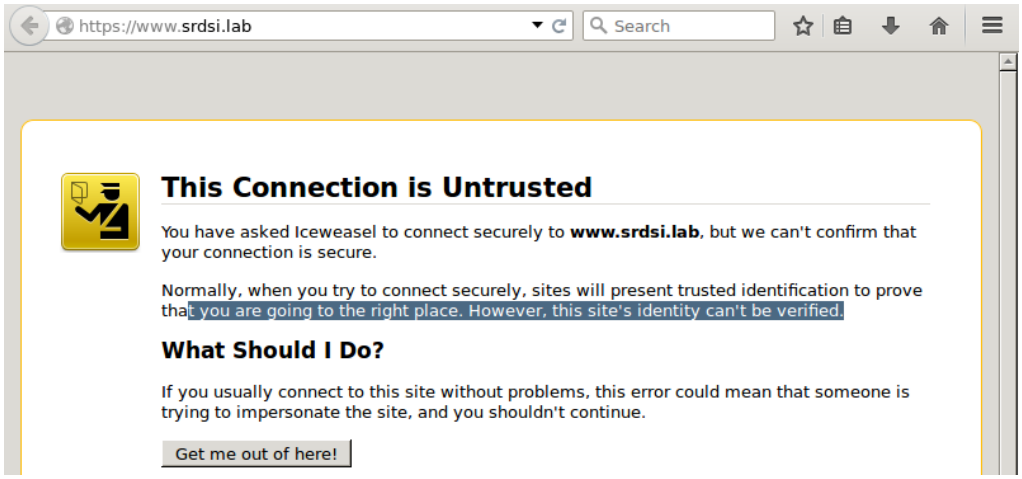
3. Se crea otro sitio virtual para el ocsp. En ésta práctica se ha llamado “ca-root.conf”. No tiene nada de especial, utiliza HTTP (puerto 80) y sólo tiene las líneas de ServerName y DocumentRoot.

4. A continuación, se genera un certificado para el servidor firmado por la CA del laboratorio. Es necesario para que haya una cadena de certificación.

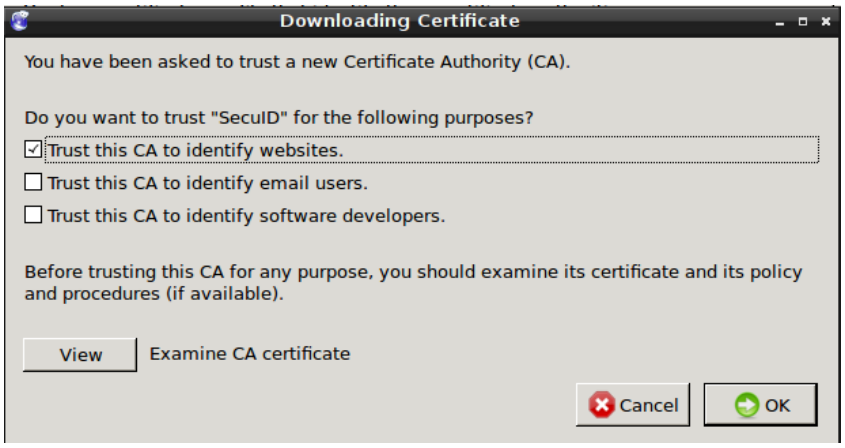
```
root@srdsi:~/pki# openssl req -new -nodes -keyout /root/pki/private/serverCert.
key -out /root/pki/certs/serverCert.crt -config /root/pki/ca_openssl.cnf
Generating a 2048 bit RSA private key
..................+++
.....................................+++
writing new private key to '/root/pki/private/serverCert.key'
-----
root@srdsi:~/pki# ls certs
1-ca-root.crt          certificado-ca-ocsp.crt     crl-ca-root.pem
68DD234C238A1158.pem   certificado-exportado.pem   serverCert.crt
68DD234C238A1159.pem   certificado-usuario2.crt    solicitud-ca-ocsp
68DD234C238A115A.pem   certificado-usuario3.crt    solicitud-usuario2.crt
68DD234C238A115B.pem   certificado-usuario.crt     solicitud-usuario3.crt
ca-root.crt            clave-exportada.p12         solicitud-usuario.crt
```

5. Se modifica el archivo “`/etc/hosts`” para asignar los nuevos nombres a localhost.

6. Por último, se prueba la conexión en el navegador y se agregan los certificados pertinentes.



El navegador avisa de que la conexión no es de fiar. Hay que introducir el certificado:

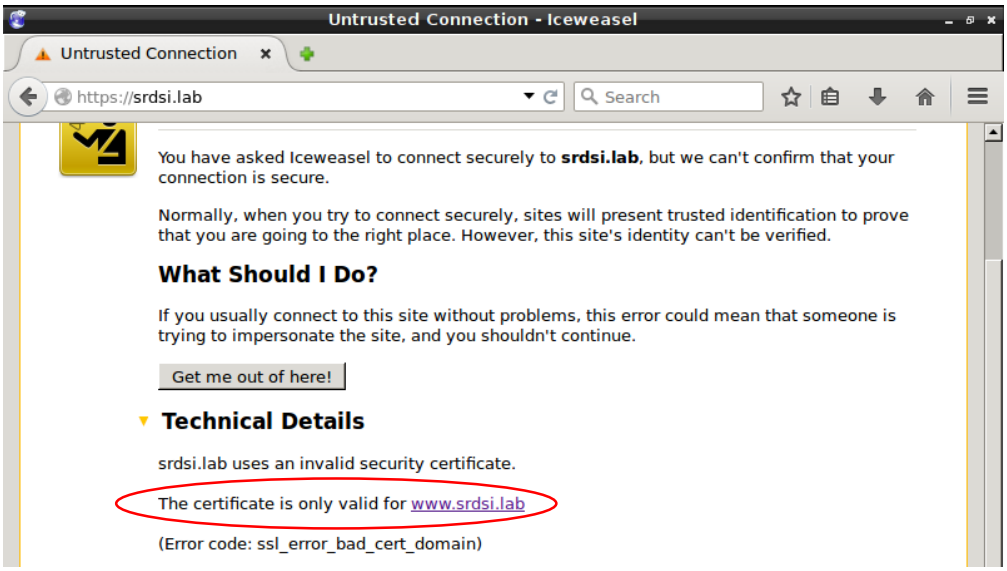


Tras introducir el certificado, el resultado obtenido es el siguiente:

Como en la imagen se puede apreciar, en la pestaña "*Details*", asociada al certificado, se ve como la "*root-ca*" vinculada al certificado del servidor web ha sido incluida. Todo ello mediante el archivo de configuración de Apache, empleando el parámetro "*SSLCertificateChainFile*". Por lo tanto, la validación del certificado se basará en verificar que la firma digital del certificado del servidor al ser descifrada coincide con el algoritmo de resumen del propio certificado, que es un SHA-256.

Se observa en el lado izquierdo de la imagen que se ha podido acceder a la página web, y que en la barra de navegación aparece el candado de HTTPS.

¿Qué sucede si en vez de entrar por "`www.srdsi.lab`" se entra por "`srdsi.lab`"?



El certificado sólo es válido para "`www.srdsi.lab`". Se podría crear otro certificado para usar ambas direcciones URL.

¿Qué ocurre si no está en marcha el servidor OCSP? ¿Cómo lleva a cabo esta validación?

Se ha tomado con WireShark la siguiente captura del protocolo.

No.	Time	Source	Destination	Protocol	Length	Info
5	0.434811000	127.0.0.1	127.0.0.1	TLSv1.2	305	Client Hello
7	0.437831000	127.0.0.1	127.0.0.1	TLSv1.2	2197	Server Hello, Certificate, Server Key Exch
9	0.445113000	127.0.0.1	127.0.0.1	TLSv1.2	194	Client Key Exchange, Change Cipher Spec, H
10	0.445707000	127.0.0.1	127.0.0.1	TLSv1.2	342	New Session Ticket, Change Cipher Spec, Er
11	0.455714000	127.0.0.1	127.0.0.1	TLSv1.2	482	Application Data
12	0.456088000	127.0.0.1	127.0.0.1	TLSv1.2	277	Application Data
20	5.368274000	127.0.0.1	127.0.0.1	TLSv1.2	99	Encrypted Alert

Se inicia el TLS Handshake de la siguiente forma:

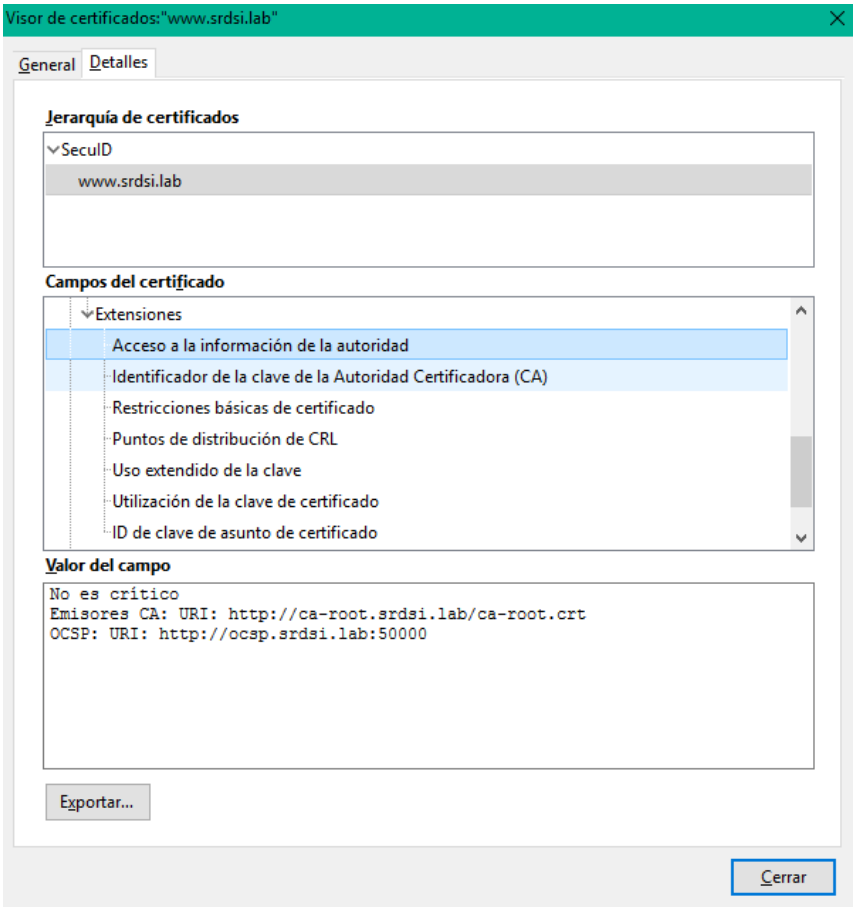
El cliente envía el mensaje "*Client Hello*" con una lista de algoritmos de los cifrados soportados por el navegador, además de dos parámetros aleatorios utilizados para la derivación de la Master Key.

El servidor responde con "*Server Hello, Certificate, ServerKeyExchange*" y "*Server Hello done*". "*Server Hello*" contiene la suite de cifrado que se va a emplear en la negociación de claves. En "*Certificate*" está el certificado entregado por el servidor. El navegador lo validará mediante la cadena de certificación y, si es válido, cogerá el "*ServerKeyExchange*" (parámetros de ECDHE) y comprobará su integridad para emplear la matemática y decidir sus parámetros de Cliente, que intercambiará en "*ClientKeyExchange*". A partir de aquí ambos extremos conocerán la clave privada y se intercambiarán los mensajes "*ChangeCipherSpec*" y "*Finished*".

Configurando OCSP

Para trabajar con OCSP hay que modificar el archivo de configuración del sitio virtual. Queda del siguiente modo:

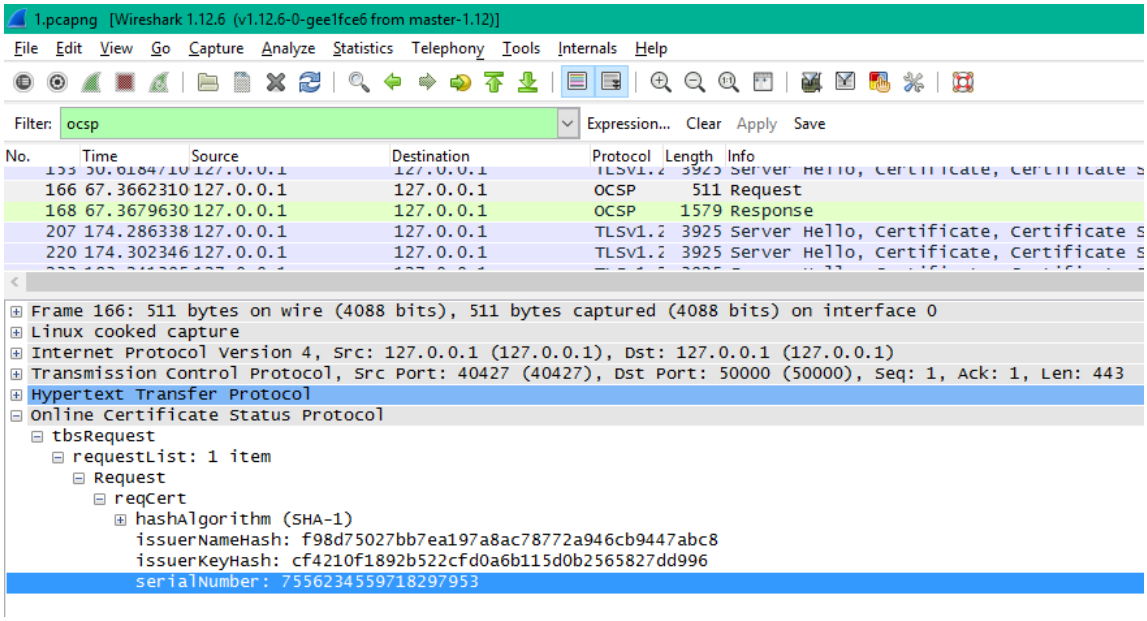
```
SSLStaplingCache shmcb:logs/ssl_stapling(12000)
<VirtualHost *:443>
        ServerName                      www.srdsi.lab
        DocumentRoot                    /var/www/srdsi
        SSLUseStapling                  On
        SSLStaplingResponseMaxAge       90
        SSLEngine                       On
        SSLOCSPEnable                   On
        SSLStaplingForceURL             http://ocsp.srdsi.lab:50000
        SSLCertificateFile              /root/pki/certs/serverCert.crt
        SSLCertificateKeyFile           /root/pki/private/serverCert.key
        SSLCertificateChainFile         /root/pki/certs/ca-root.crt
        ErrorLog                        "/var/log/ApacheError.log"
</VirtualHost>
```



Se ha puesto en marcha el servidor openssl para responder a las peticiones OCSP.

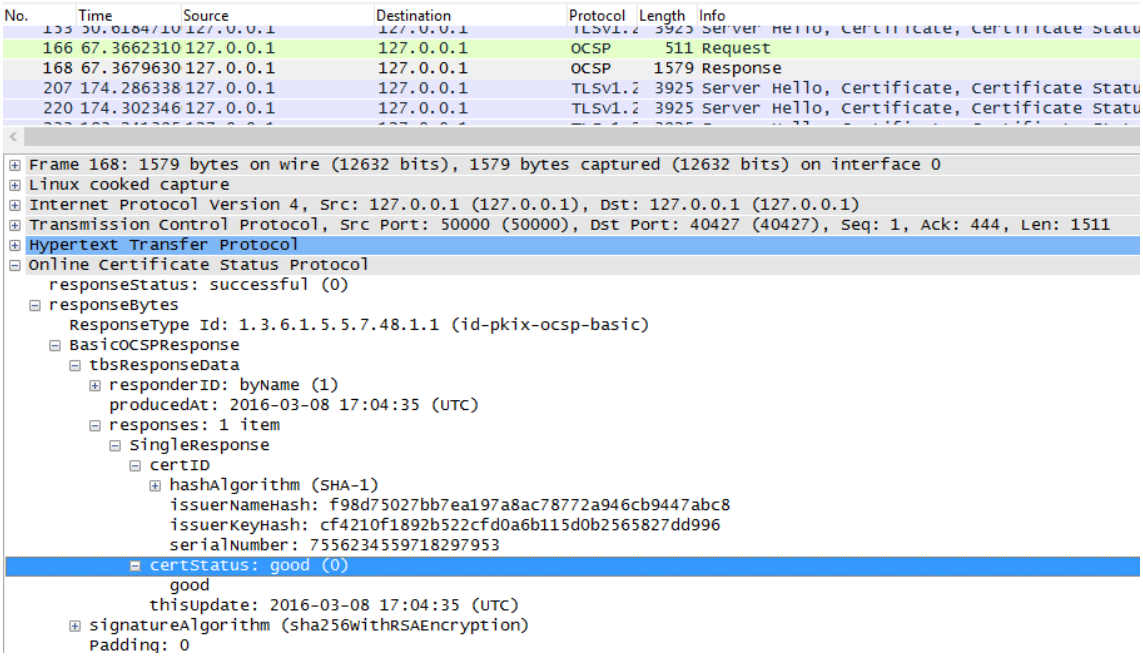
Dentro del certificado se puede apreciar la OCSP URL, hay que crear aparte un nuevo sitio virtual con la ruta que apunte al servidor, es decir, puerto 50000. Además el directorio tiene que contener el certificado.

COMPROBAR CON WIRESHARK EL PROTOCOLO OCSP.



Se le envía el "*serial number*" del certificado del servidor y dos hashes en sha1 de los campos "*keyissuer*" e "*issuername*".

Si abrimos el response se observa lo siguiente:



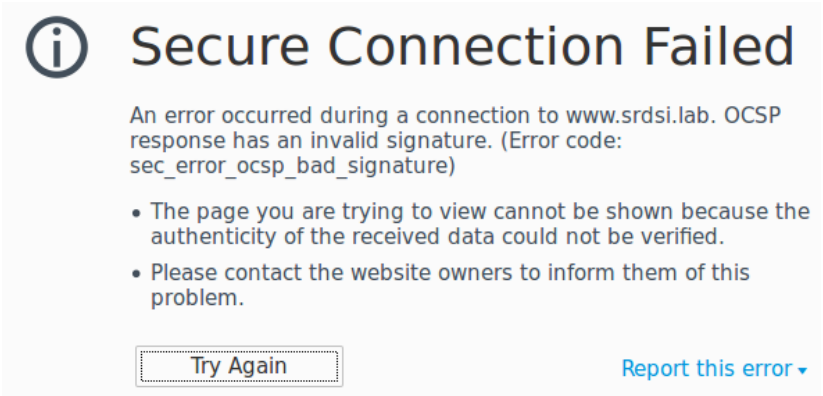
Los hashes que ha enviado el usuario en la "*Request*" coinciden con los que envía el servidor a través de la "*Response*". Además, se ha incluido un nuevo campo, "*certStatus*", que indica en este caso que la validación ha sido la correcta (good).

¿QUÉ OCURRE SI SE REVOCA EL CERTIFICADO?

Con el siguiente comando se revoca el certificado:

```
root@srdsi:~/pki# openssl ca -revoke certs/serverCert.crt -crl_reason keyComprom
ise -config ca_openssl.cnf
Using configuration from ca_openssl.cnf
Revoking Certificate 68DD234C238A1161.
Data Base Updated
```

Se comprueba el resultado usando el navegador:



El OCSP ha verificado que el certificado del servidor está revocado, devolviendo un error de bad_signature. En un primer lugar se pensó que esta respuesta podría ser un error, pero posteriormente, después de hacer pruebas, se consultó en Internet y parece ser correcto.

Dedicación

La dedicación para ésta actividad ha sido la siguiente:

	Desarrollo	Documentación	Total
Tiempo	6h.	2h.	8h.