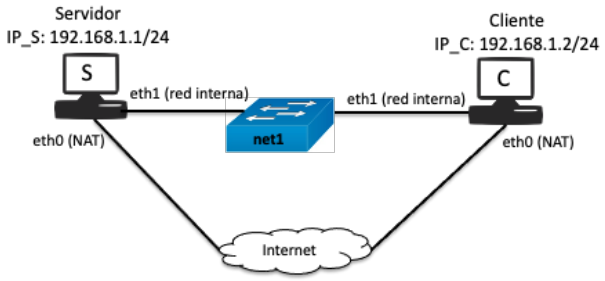
Seguridad en Redes

Práctica 3.1. Servidor web seguro

Preparación del entorno

En esta práctica usaremos dos MVs (`cliente` y `servidor`), tal y como se muestra en la figura.



Para ello, creamos una primera MV importando el servicio virtualizado `SER.ova` y luego creamos una clonación enlazada de dicha máquina, siguiendo las instrucciones del documento "Introducción al entorno de Laboratorio". Puedes cambiar el nombre de las máquinas (`cliente` y `servidor`, respectivamente), desde el menú "Configuración" → "General".

Cada MV tendrá dos interfaces de red:

- El adaptador 1 (`eth0`) configurado en modo NAT para instalar los paquetes necesarios (una vez instalados los paquetes necesarios, se deshabilitará la interfaz `eth0` para cualquier tipo de conflicto.)
- El adaptador 2 (`eth1`) configurado en modo red interna, que se usará comunicarnos entre las distintas máquinas virtuales. Las tres interfaces `eth1` deben estar conectados a la misma red interna (por ejemplo, la llamamos `net1`)

Configura el `servidor` (instala SSH, Apache y PHP):

```
$ sudo apt-get update
$ sudo apt-get install ssh
$ sudo apt-get install apache2 php5 libapache2-mod-php5
$ sudo ifdown eth0
$ sudo ip link set dev eth1 up
$ sudo ip addr add 192.168.1.1/24 broadcast + dev eth1
```

Edita* el fichero `/etc/hosts` y añade la siguiente línea:

```
192.168.1.1      www.lab.ser
```

Edita* el fichero `/etc/apache2/apache2.conf` y añade el siguiente parámetro:

```
ServerName www.lab.ser
```

---

* NOTA IMPORTANTE: Para editar los ficheros de configuración del sistema es necesario ser superusuario, por tanto, se debe llamar al editor (`nano` o `vi`) usando `sudo`. Por ejemplo:

```
$ sudo nano /etc/hosts
```

Reinicia el servidor Apache:

```
$ sudo service apache2 restart
```

Configura el `cliente` (instala SSH):

```
$ sudo apt-get update
$ sudo apt-get install ssh
$ sudo ifdown eth0
$ sudo ip link set dev eth1 up
$ sudo ip addr add 192.168.1.2/24 broadcast + dev eth1
```

Edita el fichero `/etc/hosts` y añade la siguiente línea:

```
192.168.1.1      www.lab.ser
```

Autoridad de certificación (CA)

Siguiendo las instrucciones de la práctica 2.3 (ejercicio A.1), crea una autoridad de certificación (`demoCA`) en el `servidor`. La carpeta `demoCA` deberá colgar del directorio HOME de `usuario` (es decir, `/home/usuario/demoCA`)

Cuando crees el certificado raíz autofirmado, utiliza la opción por defecto para todos los campos salvo para *Common Name* (CN), donde debes poner "`CA`".

Entrega #1: Entrega el archivo con el certificado de la CA y el archivo con la clave privada de la CA creados en el ejercicio anterior

Certificado de servidor web

Siguiendo las instrucciones de la práctica 2.3 (ejercicios A.2 y A.3), crea una solicitud de certificado (CSR) para el servidor web y fírmala con la autoridad de certificación creada anteriormente.

Para crear la solicitud de certificado usa la siguiente orden:

```
$ openssl req -new -keyout serverkey.pem -out servercsr.pem
  -nodes
```

Selecciona la opción por defecto para todos los campos salvo para CN, donde debes poner "`www.lab.ser`". Con la opción `-nodes`, la clave privada se almacena sin cifrar para poder ser usada por un servicio sin proporcionar una contraseña.

A continuación firma la solicitud con el certificado de la CA (con la configuración por defecto de OpenSSL, lo buscará en `./demoCA`), usando la siguiente orden:

```
$ openssl ca -in servercsr.pem -out servercert.pem
```

Configuración de HTTPS en Apache

Para configurar HTTPS en Apache es necesario realizar dos operaciones:

1) Habilitar SSL/TLS en Apache (módulo `mod_ssl`)

La configuración del módulo `mod_ssl` está definida en los dos siguientes archivos:

`/etc/apache2/mods-available/ssl.conf` y
`/etc/apache2/mods-available/ssl.load`

Para habilitar dicho módulo usar la siguiente orden:

```
$ sudo a2enmod ssl
```

Esta orden crea enlaces simbólicos a los ficheros anteriores en el directorio `/etc/apache2/mods-enabled`. Revisa la configuración del módulo.

Una vez habilitado el módulo `mod_ssl` es necesario reiniciar el servicio Apache:

```
$ sudo service apache2 restart
```

Con esta operación se abre el puerto TCP 443, que es que utiliza el servidor HTTPS. Comprueba, mediante `netstat -ant` que el puerto 443 está abierto.

Entrega #2: Copia y entrega la salida de la orden `netstat -ant` en el servidor donde se vea que el puerto 443 está abierto

2) Configurar SSL en el sitio (*site*) web

La configuración por defecto del sitio web con SSL está definida en siguiente archivo:

```
/etc/apache2/sites-available/default-ssl
```

Para habilitar esta configuración es necesario ejecutar la siguiente orden:

```
$ sudo a2ensite default-ssl
```

Esta orden crea un enlace simbólico al fichero anterior en siguiente directorio:

```
/etc/apache2/sites-enabled.
```

Revisa la configuración del sitio y modifica el valor de los siguientes parámetros en el fichero `/etc/apache2/sites-enabled/default-ssl`:

```
SSLCertificateFile      /etc/ssl/certs/servercert.pem
SSLCertificateKeyFile   /etc/ssl/private/serverkey.pem
```

Copia el certificado y la clave del servidor creados en el apartado anterior a los directorios especificados.

A continuación, recarga la configuración de Apache:

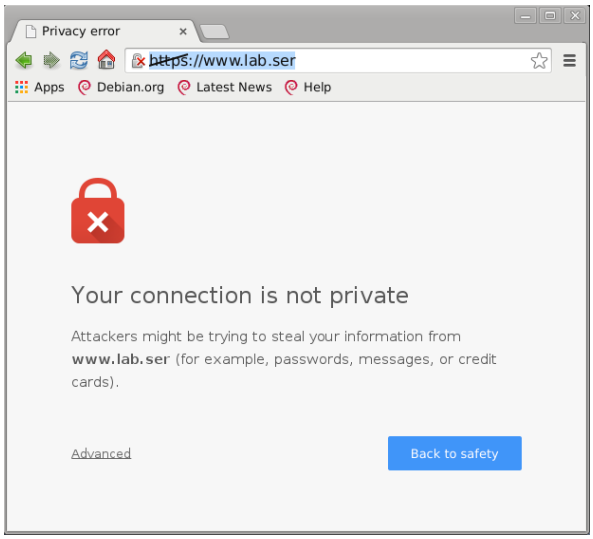
```
$ sudo service apache2 reload
```

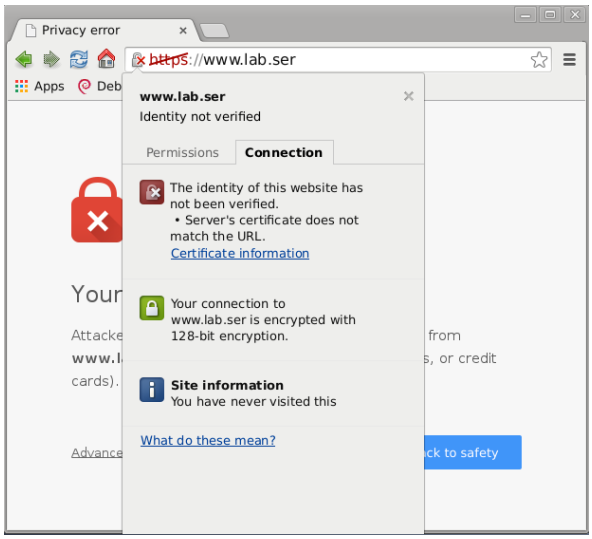
Si falla, revisa los ficheros de *log* en `/var/log/apache2`.

Configuración de la aplicación cliente (navegador)

Arranca el navegador web en `cliente` e intenta conectarte al servidor HTTPS, introduciendo la siguiente URL: `https://www.lab.ser`

Aparecerá un mensaje en el navegador con el siguiente mensaje “Your connection is not private”. Si haces click en el candado junto a la URL, aparecerá una ventana que indica que la identidad de `www.lab.ser` no está verificada.





Esto se debe a que el navegador del cliente no tiene almacenado el certificado raíz de la CA que firmó el certificado del servidor y por tanto reconoce a éste como un certificado válido.

Para solucionar este problema, es necesario importa el certificado raíz de la CA en el navegador del cliente. En primer lugar, mediante SSH, copia el certificado raíz de la CA en el `cliente`, ejecutando la siguiente orden en el `servidor`:

```
$ scp /home/usuario/demoCA/cacert.pem usuario@192.168.1.2:.
```

A continuación, en el navegador del cliente, ve a “Settings” y pulsa en “Show advanced settings…”. En “HTTPS/SSL”, pulsa “Manage certificates…”. Selecciona la pestaña “Authorities” y pulsa “Import”. Tras seleccionar el fichero `cacert.pem` con el certificado de la CA, en “Edit trust settings” selecciona “Trust this certificate for identifying websites”.

Introduce de nuevo la URL `https://www.lab.ser` en el navegador del cliente. Comprueba las propiedades de la conexión pulsando sobre el candado que aparece delante de la URL y luego en “Connection”.

Entrega #3: En las propiedades de la conexión, describe qué versión de TLS se está usando, así como los mecanismos criptográficos utilizados para cifrado, autenticación de mensajes e intercambio de clave.

Utiliza wireshark para ver los mensajes intercambiados entre el cliente y el servidor HTTPS. Analiza el contenido de los mensajes del protocolo de *handshake* (`ClientHello`, `ServerHello`, `Certificate`, `ServerKeyExgange`, `ServerHelloDone`, `ClientKeyExchange`, `ChangeCipherSuite`)

Entrega #4: A partir de los mensajes del protocolo de *handshake* capturados con wireshark, responde a las siguientes preguntas:
1) ¿Qué versión de TLS soporta el cliente (mensaje `ClientHello`)?
2) ¿Cuántas *cipher suites* diferentes soporta el cliente (mensaje `ClientHello`)?
3) ¿Qué versión de TLS soporta el servidor (mensaje `ServerHello`)?
4) ¿Cuál es la *cipher suite* seleccionada por el servidor (mensaje `ServerHello`)?
5) ¿Cuál es el algoritmo de firma (*signature*) usado en el certificado del servidor (mensaje `Certificate`)?
6) ¿Cuál es la longitud de la clave pública tipo EC Diffie-Hellman utilizada en el intercambio de clave (mensajes `ServerKeyExchange` y `ClientKeyExchange`)?

Crea una página PHP en `servidor` con el siguiente contenido en `/var/www/index.php`:

```
<html>
<?php
echo "Web server "  . $_SERVER['SERVER_NAME'] . "<br><br>";
if ($_SERVER['HTTPS'] == "on" ) {
    echo "Secure connection from ". $_SERVER['REMOTE_ADDR'] . "<br><br>";
    echo "Server Id: " . $_SERVER['SSL_SERVER_S_DN'] . "<br>";
    echo "Signing CA: " . $_SERVER['SSL_SERVER_I_DN'] . "<br><br>";
    }
?>
</html>
```

Accede a https://www.lab.ser/index.php con el navegador del cliente.

Entrega #5: Copia y entrega la salida mostrada en el navegador del cliente

Autenticación del cliente mediante certificado en Apache

En una conexión HTTPS el servidor debe disponer obligatoriamente de un certificado digital para permitir su autenticación, no obstante el uso de un certificado digital para la autenticación de la parte cliente es opcional. Esta sección tiene dos partes: 1) crearemos e instalaremos un certificado digital para la parte cliente; y 2) configuraremos Apache para que solicite la autenticación de la parte cliente.

1. Creación e instalación del certificado digital para la parte cliente

El certificado digital del cliente lo crearemos y lo firmaremos en el servidor (que tiene instalada la CA), y luego lo transferiremos mediante SSH al cliente, para finalmente importarlo en su navegador.

En `servidor`, crea una solicitud de certificado para el cliente:

```
$ openssl req -new -keyout clientkey.pem -out clientcsr.pem
```

Selecciona la opción por defecto para todos los campos salvo para CN, donde debes poner "usuario".

Firma la solicitud con el certificado de la CA:

```
$ openssl ca -in clientcsr.pem -out clientcert.pem
```

Convierte el certificado a formato PKCS12:

```
$ openssl pkcs12 -export -in clientcert.pem
  -inkey clientkey.pem -out client.p12
```

Transfiere el certificado de `servidor` a `cliente` mediante SSH:

```
$ scp /home/usuario/client.p12 usuario@192.168.1.2:.
```

Importa el certificado `client.p12` en el navegador de `cliente`: ve a "Settings" y pulsa en "Show advanced settings…". En "HTTPS/SSL", pulsa "Manage certificates…". Selecciona la pestaña "Your certificates", pulsa "Import" y selecciona el certificado del cliente `client.p12`.

2. Configuración del servidor Apache para autenticación de la parte cliente

En `servidor`, modifica el valor de los siguientes parámetros en el fichero `/etc/apache2/sites-enabled/default-ssl`:

```
SSLCACertificateFile /etc/ssl/certs/cacert.pem
SSLVerifyClient require
```

Copia el certificado de la CA al directorio especificado. Así, se habilita la autenticación del cliente para todo el sitio. También se puede habilitar por directorios con la directiva `Location`.

Recarga la configuración de Apache:

```
$ sudo service apache2 reload
```

Introduce la URL `https://www.lab.ser` en el navegador del cliente y comprueba que el navegador solicita el certificado con el que debe identificarse el cliente.

Utiliza wireshark para ver los mensajes intercambiados entre el cliente y el servidor HTTPS. Identifica el mensaje TLS de tipo `Certificate` donde el `cliente` envía su certificado al `servidor`.

Crea una página PHP en `servidor` con el siguiente contenido en `/var/www/index.php`:

```
<html>
<?php
echo "Web server "  . $_SERVER['SERVER_NAME'] . "<br><br>";
if ($_SERVER['HTTPS'] == "on" ) {
    echo "Secure connection from ". $_SERVER['REMOTE_ADDR'] . "<br><br>";
    echo "Server Id: " . $_SERVER['SSL_SERVER_S_DN'] . "<br>";
    echo "Signing CA: " . $_SERVER['SSL_SERVER_I_DN'] . "<br><br>";
    }
if ($_SERVER['SSL_CLIENT_VERIFY'] == "SUCCESS") {
    echo "Client verified successfully.<br><br>";
    echo "Client Id: " . $_SERVER['SSL_CLIENT_S_DN'] . "<br>";
    echo "Signing CA: " . $_SERVER['SSL_CLIENT_I_DN'] . "<br>";
    }
?>
</html>
```

Accede a https://www.lab.ser/index.php con el navegador del cliente.

Entrega #6: Copia y entrega la salida mostrada en el navegador del cliente