Comprobar el acceso seguro al servidor

A continuación una serie de actuaciones que te servirán para comprobar que el acceso seguro que estableces con el servidor es el esperado:

- Siempre que te conectes mediante SSL a un página web y el certificado no sea admitido, debes ver los campos descriptivos del certificado antes de generar la excepción que te permita visitar la página.
- Debes comprobar en el certificado si la página a la que intentas acceder es la misma que dice el certificado.
- Típicamente en los navegadores, si no está configurado lo contrario, cuando accedes mediante cifrado SSL a una página web puedes ver en algún lugar del mismo un icono: un candado, por lo cual debes verificar su existencia para asegurarte que estás accediendo por https.

Puedes ver en la barra de direcciones indicaciones del tipo de certificado con el que se cifra la comunicación.

- Revisar la lista de certificados admitidos que posee tu navegador. En **Firefox**, versión > 3.x , donde x es el número de revisión de la versión 3, puedes verlas dirigiéndote por las pestañas a:

Editar → Preferencias → Avanzado → Cifrado → Ver certificados

- Revisar la lista de revocaciones que posee tu navegador. En **Firefox**, versión > 3.x , donde x es el número de revisión de la versión 3, puedes verlas dirigiéndote por las pestañas a:

Editar → Preferencias → Avanzado → Cifrado → Listas de revocación

Puedes **Importar/Exportar** certificados en los navegadores, con lo cual los puedes llevar a cualquier máquina. Esto es muy útil cuando necesitas un certificado personal en máquinas distintas.

Autenticación y control de acceso

Puede que interese impedir el acceso a determinadas páginas ofrecidas por el servidor web, pues podría no convenir a una empresa que cualquiera tuviera acceso a determinada información confidencial, o simplemente que interese controlar el acceso hacia un servicio a través de la web, como el correo electrónico. Para este tipo de casos tenemos que pensar en la autenticación y el control de acceso.

Cuando nos autenticamos en una web suele transferirse la información de autenticación a una base de datos, que puede existir en la misma máquina que el servidor web o en otra totalmente diferente. Suelen emplearse bases de datos SQL o LDAP para la autenticación de usuarios, siendo OpenLDAP una de las alternativas más empleadas.

HTTP proporciona un método de autenticación básico de usuarios: **basic**. Este método ante una petición del cliente (navegador web) al servidor, cuando se solicita una URL, mostrará un diálogo pidiendo usuario y contraseña. Una vez autenticado el usuario, el cliente volverá a hacer la petición al servidor pero ahora enviando el usuario y contraseña, en texto claro (sin cifrar) proporcionados en el diálogo. Es recomendable entonces si empleas este método que lo hagas combinado con conexión SSL (HTTPS).

En la autenticación HTTP Basic es muy típico utilizar archivos .htaccess en los directorios que queremos controlar el acceso. Puedes encontrar un ejemplo sobre basic con https en el archivo virtualhost-ssl-basic y un ejemplo sobre .htaccess en el archivo htaccess.

Para usar archivos **.htaccess**, se necesita tener una configuración en el servidor que permita poner directivas de autenticación en estos archivos, mediante la directiva **AllowOverride**, así: **AllowOverride AuthConfig.**

También se puede controlar el acceso mediante IP. Puedes encontrar un ejemplo en el archivo virtualhost-control-por-IP.

Autenticar usuarios en apache mediante LDAP

El servidor web Apache permite la autenticación de usuarios mediante LDAP. Esto es posible mediante los módulos **ldap** y **authnz_ldap**.

Cómo instalar y configurar un servidor OpenLDAP en un GNU/Linux basado en Debian, con el que Apache puede realizar la autenticación.

Instalación y configuración del servidor OpenLDAP

Para una instalación de OpenLDAP en Linux podemos ver la siguiente página:

Instalar y configurar el servidor LDAP de Linux

Más información sobre la autenticación LDAP para el servidor web Apache.

Autenticación LDAP en Apache mediante el módulo **authnz_ldap.**

Para el buen funcionamiento de lo expuesto a continuación se asume que tanto Apache2 como OpenLDAP están instalados y configurados:

1. Habilita el soporte LDAP para Apache2:

```
a2enmod authnz_ldap
/etc/init.d/apache2 restart
```

2. Configura el virtualhost **autenticacion-ldap-apache** como sigue:

```
<VirtualHost *:80>
        DocumentRoot "/var/www/autenticacion-ldap"
ServerName www.empresa-proyecto.panel-de-control.com
ServerAlias www.autenticacion-ldap.empresa-proyecto.com
<Directory "/var/www/autenticacion-ldap">
        AllowOverride All
        </Directory>
ErrorLog /var/log/apache2/error-autenticacion-ldap.log
        LogLevel warn
        CustomLog /var/log/apache2/access-autenticacion-ldap.log combined
</VirtualHost>
```

   La directiva **AllowOverride All** es necesaria para habilitar ficheros .htaccess

3. Crea el fichero **/var/www/autenticacion-ldap/.htaccess** que permite configurar la autenticación ldap para el virtualhost anterior:

```
AuthName "Autenticacion por LDAP"
AuthType Basic
AuthBasicProvider ldap
AuthzLDAPAuthoritative on
AuthLDAPUrl ldap://127.0.0.1/ou=usuarios,dc=proyecto,dc=com?uid
Require ldap-user user1LDAP
```

   La directiva **Require ldap-user admin** permite la autenticación al usuario **user1LDAP**, todos los demás usuarios tienen el acceso denegado.

Monitorización del acceso: Archivos de registro (logs).

Tan importante es configurar un servidor web como mantener y comprobar su correcto funcionamiento, y para ello debes ayudarte de los logs o archivos de registro que te permiten revisar y estudiar su funcionamiento

Apache permite mediante diversas directivas crear archivos de registro que guardarán la información correspondiente a las conexiones con el servidor. Esta información es guardada en formato CLF **(Common Logon Format)** por defecto. Ésta es una especificación utilizada por los servidores web para hacer que el análisis de registro entre servidores sea mucho más sencillo, de tal forma que independientemente del servidor web utilizado podamos emplear el mismo método de análisis de registro, ya sea mediante lectura, mediante programas ejecutables (scripts) o mediante programas propios de análisis de registro.

En un archivo de registro en formato CLF cada línea identifica una solicitud al servidor web. Esta línea contiene varios campos separados con espacios**.** Cada campo sin valor es identificado con un guión (-). Los campos empleados en una configuración por defecto de Apache2 son los definidos en la siguiente tabla:

Ejemplo log Apache en formato CLF		
192.168.200.100 - - [05/May/2020:17:19:18 +0200] "GET /index.html HTTP/1.1" 200 20		
Campos (especificadores)	**Definición**	**Ejemplo**
host (%h)	Identifica el equipo cliente que solicita la información en el navegador.	192.168.200.100
ident (%l)	Información del cliente cuando la máquina de éste ejecuta identd y la directiva IdentityCheck está activada.	
authuser (%u)	Nombre de usuario en caso que la URL solicitada requiera autenticación HTTP.	
date (%t)	Fecha y hora en el que se produce la solicitud al servidor. Va encerrado entre corchetes. Este campo tiene su propio formato:	[05/May/2020:17:19:18 +0200]

	[dia/mes/año:hora:minuto:segundo zona]	
request (%r)	Petición del cliente, esto es, la página web que está solicitando. En el ejemplo: /index.html, esto es, dentro de la raíz del dominio que se visite la página	/index.html
status (%s ó %>s)	Identifica el código de estado HTTP de tres dígitos que se devuelve al cliente.	200
Bytes (%b)	Sin tener en cuenta las cabeceras HTTP el número de bytes devueltos al cliente.	20

Cada campo tiene su especificador, el cual se emplea en las directivas de Apache para indicar que campo queremos registrar.

Directivas para archivos de registro.

El contexto de aplicación de todas las directivas que se indican a continuación en la siguiente tabla puede ser el de la configuración principal del servidor así como el de la configuración de los host virtuales

Directivas para archivos de registro.	
Directivas	**Definicion**
TransferLog	Directiva que define el nombre del archivo de registro o al programa al que se envía la información de registro. Emplea los especificadores asignados por la directiva **LogFormat.**
LogFormat	Directiva que define el formato del archivo de registro asignado con la directiva **TransferLog**
ErrorLog	Directiva que permite registrar todos los errores que encuentre Apache. Permite guardar la información en un archivo de registro o bien en **syslog**
CustomLog	Directiva similar a la directiva **TransferLog**, pero con la particularidad que permite personalizar el formato de registro empleando los especificadores anteriormente vistos.
CookieLog	Directiva que define el nombre del archivo de registro donde

Directivas para archivos de registro.	
Directivas	**Definicion**
	registrar información sobre cookies

La tabla siguiente muestra la sintaxis y el uso de las anteriores directivas

Sintaxis y uso de directivas para archivos de registro	
Directiva TransferLog	
Sintaxis	**TransferLog nombre_fichero_archivo_registro | tubería_para_enviar_al_programa_la_información de_registro**
Uso	**TransferLog logs/acceso_a_empresa1.log**
Directiva LogFormat	
Sintaxis	**LogFormat nombre_fichero_archivo_registro [opcional_alias] [opcional_alias]** permite definir un logformat con un nombre de tal forma que cuando hacemos referencia al nombre lo hacemos al logformat vinculado.
Uso	**LogFormat logs/acceso_a_empresa1.log**
Directiva ErrorLog	
Sintaxis	**ErrorLog nombre_fichero_archivo_registro**
Uso	**ErrorLog logs/acceso_a_empresa1.log**
Directiva CustomLog	
Sintaxis	**CustomLog nombre_fichero_archivo_registro| tubería_para_enviar_al_programa_la_información de_registro [variable_de_entorno_opcional]**
Uso	**CustomLog logs/acceso_a_empresa1.log**
Directiva CookieLog	
Sintaxis	**CookieLog nombre_fichero_archivo_registro**
Uso	**CookieLog logs/acceso_a_empresa1.log**

En **GNU/Linux** puedes **comprobar en tiempo real** desde un terminal en el equipo que guarda los logs -que puede ser el propio equipo servidor web- que es lo que ocurre cuando accedes a una página web observando el contenido de los archivos de registro mediante el comando:

tail -f nombre_archivo_de_registro.log

Rotación de los archivos de registro

Como los archivos de registro a medida que pasa el tiempo van incrementando su tamaño, debe existir una política de mantenimiento de registros para que éstos no consuman demasiados recursos en el servidor, así es conveniente rotar los archivos de registro, esto es, hay que depurarlos, comprimirlos y guardarlos. Básicamente tienes dos opciones para rotar tus registros: **rotatelogs** un programa proporcionado por Apache, o **logrotate**, una utilidad presente en la mayoría de los sistemas GNU/Linux.

No debes olvidar que la información recopilada en los **ficheros log** se debe conservar al menos durante 1 año por eventuales necesidades legales, de este modo, además de rotarlos se opta habitualmente por **comprimir logs**.

Uso de rotatelogs
CustomLog "|ruta_rotatelogs ruta_log_a_rotar numero_segundos|tamaño_máximoMB" alias_logformat
Ejemplos
Rotar el archivo de registro access.log cada 24horas **CustomLog "|/usr/sbin/rotatelogs /var/log/apache2/access.log 86400" common**
Rotar el archivo de registro access.log cada vez que alcanza un tamaño de 5 megabytes **CustomLog "|/usr/sbin/rotatelogs /var/logs/apache2/access.log 5M" common**
Rotar el archivo de registro error.log cada vez que alcanza un tamaño de 5 megabytes y el archivo se guardará con el sufijo de formato : YYYY-mm-dd-HH_MM_SS (Año-Mes-Día-Hora_Minutos_Segundos) **ErrorLog "|/usr/sbin/rotatelogs /var/logs/errorlog.%Y-%m-%d-%H_%M_%S 5M" common**

Los ficheros rotados por intervalo de tiempo, lo harán siempre y cuando en el intervalo de tiempo definido existan nuevos datos.

Por defecto, si no se define formato mediante ningún modificador % para guardar los archivos de registro, el sufijo nnnnnnnnn (10 cifras) se agrega automáticamente y es el tiempo en segundos traspasados desde las 24 horas (medianoche).

El **alias logformat** es muy interesante, porque permite definir un grupo de modificadores en una palabra, de tal forma que incorporando esa palabra en la directiva log correspondiente estás activando todo un grupo de modificadores. En Apache existen predefinidos en el archivo **/etc/apache2/apache2.conf** los alias **logformat: vhost_combined, combined, common, referer y agent**, que puedes ver a continuación:

Alias logformat predefinidos en /etc/apache2/apache2.conf
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" vhost_combined
LogFormat "%h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %O" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

El programa **logrotate** rota, comprime y envía archivos de registro a diario, semanalmente, mensualmente o según el tamaño del archivo. Suele emplearse en una tarea diaria del cron.

En **Debian** puedes encontrar los siguientes archivos de configuración para **logrotate**:

- **/etc/logrotate.conf**: Define los parámetros globales, esto es, los parámetros por defecto de logrotate. Puedes encontrar un archivo tipo en el siguiente enlace: logrotate.conf
- **/etc/logrotate.d/apache2**: Define para apache2 el rotado de logs, todos aquellos parámetros que no se encuentren aquí recogen su valor del fichero **/etc/logrotate.conf**. Puedes encontrar un archivo tipo en el siguiente enlace: logrotate.d/apache2

Uso de logrotate
Comprobar la correcta configuración de la rotación de un log
/usr/sbin/logrotate -d /etc/logrotate.d/apache2
Forzar la ejecución de logrotate
/usr/sbin/logrotate-f /etc/logrotate.conf
/etc/cron.daily/logrotate: Fichero tipo para ejecutar logrotate diariamente en el

cron

```
#!/bin/sh

test -x /usr/sbin/logrotate || exit 0

/usr/sbin/logrotate /etc/logrotate.conf
```

Ejemplo para añadir al archivo crontab del sistema (crontab -e)

```
               # Rotar logs de apache con logrotate a las 3 am

0 03 * * * root /usr/sbin/logrotate /etc/logrotate.conf > /dev/null 2>&1
```