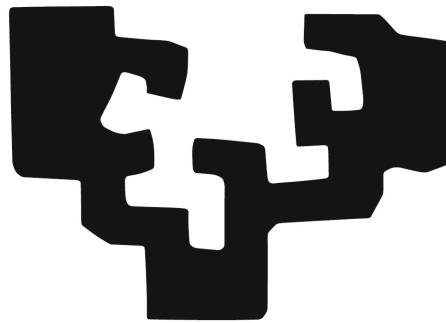


Práctica : PKI (Fase 2)

Grupo 01



UPV EHU

Alex Rivas Machín , Borja Gómez Calvo y Martín López de Ipiña Muñoz

Situación actual.....	3
Tareas.....	3
Lanzar servidor OCSP.....	3
Regular OCSP.....	4
Crear sitio web.....	4
Configurar el navegador.....	6
Ejercicio 1.....	6
OCSP STAPLING.....	8
Modificar sitios web seguros.....	8
Ejercicio 2.....	9

Situación actual

Para minimizar posibles problemas y aprovechar una solución previamente validada en la primera fase, se ha decidido utilizar los archivos y configuraciones proporcionadas. La estructura de directorios inicial es la siguiente:

```
home/user1/
├── PKI2/
│   ├── ca-ocsp.crt
│   ├── ca_openssl.cnf
│   ├── CA-root.crt
│   ├── certs/
│   │   ├── 070E850967D61066.pem
│   │   ├── 070E850967D61067.pem
│   │   ├── 070E850967D61068.pem
│   │   ├── webserverGX.crt
│   │   └── webserverGX.csr
│   ├── db/
│   │   ├── crlnumber
│   │   ├── index.txt
│   │   ├── index.txt.attr
│   │   ├── index.txt.attr.old
│   │   ├── index.txt.old
│   │   ├── serial
│   │   └── serial.old
│   ├── generar_example.sh
│   ├── private/
│   ├── ca-ocsp.key
│   ├── CA-root.key
│   └── webserverGX.key
```

Tareas

Lanzar servidor OCSP

Empezamos lanzando el servidor ocsp, que se queda a la espera.

Para lanzarlo ejecutamos:

```
→ openssl ocsp -port 55555 -index db/index.txt \
  -CA CA-root.crt -rsigner ca-ocsp.crt \
  -rkey private/ca-ocsp.key -text -out log.txt
```

Donde la salida, indica que se queda a la espera de conexión:

```
ACCEPT 0.0.0.0:55555 PID=161085
ocsp: waiting for OCSP client connections..
```

Actualizamos */etc/hosts*, añadiendo la siguiente línea para que el nombre se resuelva correctamente:

```
sudo vi /etc/hosts
>> 127.0.0.1 ocsp.srdsgx.lab
```

De esta forma forma podemos acceder a: *http://ocsp.srdsgx.lab:55555*
visitamos la URL, donde obtenemos un archivo que contiene lo siguiente:

```
HTTP/1.0 200 OK
Content-type: application/ocsp-response
Content-Length: 5

0
```

Esto indica que el servidor OCSP responde correctamente con un mensaje de estado "OK".

Regular OCSP

Crear sitio web

Para completar esta parte, necesitamos crear el sitio web seguro. Para ello vamos a configurar apache.

Dentro de *sites-available*, se ha configurado *grupo01.conf* con el contenido siguiente:

```
<VirtualHost *:443>
  ServerName grupo01
  DocumentRoot /var/www/grupo01

  SSLEngine on

  SSLCertificateFile /home/user1/PKI2/certs/webserverGX.crt
  SSLCertificateKeyFile /home/user1/PKI2/private/webserverGX.key
  SSLCACertificateFile /home/user1/PKI2/CA-root.crt

  SSLStaplingForceURL http://ocsp.srdsiGX.lab:55555
  SSLStaplingResponseMaxAge 90

  ErrorLog ${APACHE_LOG_DIR}/grupo01.log
  CustomLog ${APACHE_LOG_DIR}/grupo01_access.log combined
</VirtualHost>
```

NO coinciden en nombre!

No necesario en Regular OCSP,

Actualizamos *etc/hosts* para añadir a grupo01:

```
127.0.0.1 grupo01
```

Tras esto, hemos añadido a la configuración de apache en */etc/apache2/apache2.conf* fuera de *<VirtualHosts>* para definir la caché:

No se corresponde con el modo Regular OCSP

```
→ ~ sudo vi /etc/apache2/apache2.conf

>> SSLStaplingCache "shmcb:logs/ssl_stapling(12000)"
```

Habilitamos la configuración:

```
→ ~ sudo a2ensite grupo01.conf
Site grupo01 already enabled
```

Como da problemas al verificar la configuración necesitamos crear la carpeta para el grupo:

```
→ ~ sudo mkdir -p /var/www/grupo01
```

Finalmente recargamos apache:

```
→ ~ sudo systemctl reload apache2
```

Prueba de conexión

En el navegador ingresamos la url: <https://grupo01> y comprobamos que todo funciona correctamente tras aplicar y establecer la configuración. ¿Cómo? No hay capturas

Una vez dentro con el servidor OCSP lanzado queremos comprobar que este funcione correctamente, donde en el servidor podemos leer la siguiente líneas, tras la conexión:

```
→ openssl ocsp -port 55555 -index db/index.txt \
  -CA CA-root.crt -rsigner ca-ocsp.crt \
  -rkey private/ca-ocsp.key -text -out log.txt
```

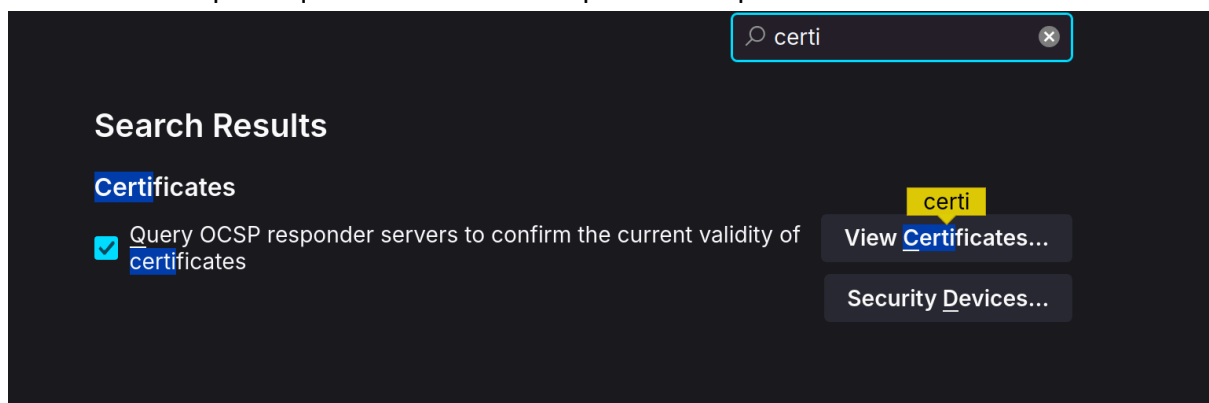
```
ACCEPT 0.0.0.0:55555 PID=47402
ocsp: waiting for OCSP client connections...
ocsp: Received request, 1st line: POST / HTTP/1.0
```

cierra el servidor OCSP

De forma que hemos comprobado que funciona la conexión.

Configurar el navegador

Habilitamos la opción que los servidores ocsp deben responder:



```
➔ PKI2 openssl ocsp -port 55555 -index db/index.txt \
    -CA CA-root.crt -rsigner ca-ocsp.crt \
    -rkey private/ca-ocsp.key -text -out log.txt

ACCEPT 0.0.0.0:55555 PID=47402
ocsp: waiting for OCSP client connections...
ocsp: Received request, 1st line: POST / HTTP/1.0
ocsp: Received request, 1st line: POST / HTTP/1.0
ocsp: Received request, 1st line: POST / HTTP/1.0
ocsp: Received request, 1st line: POST / HTTP/1.0
^C
```

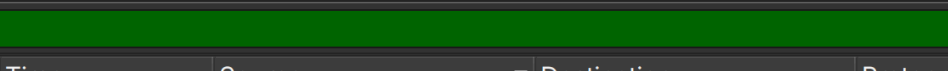
+	4687	469.510190496	127.0.0.1	127.0.0.1	OCSP	271 Request
+	4689	469.512937057	127.0.0.1	127.0.0.1	OCSP	1596 Response

```

  ✓ Online Certificate Status Protocol
    responseStatus: successful (0)
  ✓ responseBytes
    ResponseType Id: 1.3.6.1.5.5.7.48.1.1 (id-pkix-ocsp-basic)
  ✓ BasicOCSPResponse
    tbsResponseData
      responderID: byName (1)
      producedAt: Apr 6, 2025 10:17:07.000000000 CEST
      responses: 1 item
        SingleResponse
          certID
            certStatus: good (0)
            good

```

Qué ocurre si no está en marcha el servidor OCSP? Comprobar con Wireshark cuáles son los mensajes intercambiados



The screenshot shows the Wireshark interface with a packet capture of an OCSP response. The top bar indicates the capture is on the 'eth0' interface. The packet list on the left shows a single packet of type 'OCSP' with a length of 104 bytes. The packet details pane on the right shows the 'OCSP' protocol structure, including the 'OCSP Response' and 'OCSP Response Body' fields. The packet bytes pane at the bottom shows the raw data of the packet.

No.	Time	Source	Destination	Protocol	Length
1	0.000000	192.168.1.1	192.168.1.100	OCSP	104

Poner en marcha el servidor OCSP y comprobar con Wireshark cuáles son los mensajes intercambiados ¿Quién contacta con el servidor OCSP? ¿En qué momento contacta?

Como hemos comprobado en la puesta en marcha inicial del servidor hay un intercambio de request response: **No está completa, falta el contexto en el que se realizan**

9213	373.269709793	127.0.0.1	127.0.0.1	OCSP	271 Request
9215	373.273065449	127.0.0.1	127.0.0.1	OCSP	1596 Response

Donde la request solicita el certificado al servidor OCSP y este respondería.

¿Quién contacta con el servidor OCSP? **captura wireshark???**

Esto ocurre al entrar a la web de apache, en este como ya estaba iniciada al recargar la web sin chache con ctrl shift + r obligamos a que haga la petición de nuevo.

Si revocamos el certificado del servidor web y volvemos a repetir el acceso seguro. ¿Qué ocurre?

Se revoca el certificado del servidor utilizando OpenSSL:

```
openssl ca -config ca_openssl.cnf -revoke certs/webserverGX.crt
```

```
→ PKI2 openssl ca -config ca_openssl.cnf -revoke certs/webserverGX.crt

Using configuration from ca_openssl.cnf
Revoking Certificate 070E850967D61068.
Database updated
→ PKI2
```

Tras esto reiniciamos el servidor OCSP (lanzarlo de nuevo) para reflejar los cambios. Tras entrar de nuevo a la página no encontramos problemas adicionales, pero tras analizar el intercambio en Wireshark podemos ver que el certificado esta revocado:

```
File Data: 1403 bytes
  Online Certificate Status Protocol
    responseStatus: successful (0)
    responseBytes
      responseType Id: 1.3.6.1.5.5.7.48.1.1 (id-pkix-ocsp-basic)
      BasicOCSPResponse
        tbsResponseData
          responderID: byName (1)
            producedAt: Apr  5, 2025 10:49:19.000000000 CEST
          responses: 1 item
            SingleResponse
              certID
                certStatus: revoked (1)
                  revoked
                    thisUpdate: Apr  6, 2025 10:49:19.000000000 CEST
              signatureAlgorithm (sha256WithRSAEncryption)
                padding: 0
                signature [truncated]: 8264278bf77ce364e79b7c4ec14a13b169a998cc336c634b8e88f94f860427580c7b440249d0073ca88c76cecf9cf2cb9aac6d9fb518f71630e8bbdf9d03812c376d514b810b2769
```

certStatus: revoked (1)

No sirven de mucho estás capturas si no se muestra su contexto.

OCSP STAPLING

Modificar sitios web seguros

Lo primero a realizar en este apartado es comprobar los módulos de apache, para ello en bash:

→ **~ sudo apache2ctl -M**

AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message

Loaded Modules:

core_module (static)
so_module (static)
watchdog_module (static)
http_module (static)
log_config_module (static)
logio_module (static)
version_module (static)
unixd_module (static)
access_compat_module (shared)
alias_module (shared)
auth_basic_module (shared)
authn_core_module (shared)
authn_file_module (shared)
authz_core_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
deflate_module (shared)
dir_module (shared)
env_module (shared)
filter_module (shared)
mime_module (shared)
mpm_event_module (shared)
negotiation_module (shared)
reqtimeout_module (shared)
setenvif_module (shared)
socache_shmcb_module (shared)
ssl_module (shared)
status_module (shared)

→ **~ sudo a2enmod socache_shmcb**

Module socache_shmcb already enabled

Una vez comprobados los módulos, se ha adaptado la configuración de servidor Apache como se indica:

```
SSLStaplingCache "shmcb:logs/ssl_stapling(12000)"
<VirtualHost *:443>
    ServerName grupo01
    DocumentRoot /var/www/grupo01

    SSLEngine on

    SSLCertificateFile /home/user1/PKI2/certs/webserverGX.crt
    SSLCertificateKeyFile /home/user1/PKI2/private/webserverGX.key
    SSLCACertificateFile /home/user1/PKI2/CA-root.crt

    SSLUseStapling On
    SSLStaplingForceURL http://ocsp.srdsiGX.lab:55555
    SSLStaplingResponseMaxAge 90

    ErrorLog ${APACHE_LOG_DIR}/grupo01.log
    CustomLog ${APACHE_LOG_DIR}/grupo01_access.log combined
</VirtualHost>
```

Ejercicio 2

Comprobar con Wireshark cuáles son los mensajes intercambiados ¿Quién contacta con el servidor OCSP? ¿En qué momento contacta?

Iniciamos el servidor ocsp de nuevo, y recargamos:

De primeras en wireshark ocurre lo mismo que en el apartado anterior y que funciona correctamente. donde podemos ver el intercambio habitual de mensajes OCSP y TLS.

104728	4846.660023238	127.0.0.1	127.0.0.1	TCP	76 47214 → 443 [SYN, Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1014028864 TSecr=0 WS=128
104729	4846.660074902	127.0.0.1	127.0.0.1	TCP	76 443 → 47214 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM TSval=1014028864 TS...
104730	4846.660110836	127.0.0.1	127.0.0.1	TCP	68 47214 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1014028864 TSecr=1014028864
104731	4846.661461924	127.0.0.1	127.0.0.1	TLSv1.3	1960 Client Hello [SNI=grupo01]
104732	4846.661522235	127.0.0.1	127.0.0.1	TCP	68 443 → 47214 [ACK] Seq=1 Ack=1993 Win=86784 Len=0 TSval=1014028865 TSecr=1014028865
104733	4846.663146468	127.0.0.1	127.0.0.1	TCP	76 47214 → 55555 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1014028867 TSecr=0 WS...
104734	4846.663189522	127.0.0.1	127.0.0.1	TCP	76 55555 → 47214 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM TSval=1014028867 ...
104735	4846.663185101	127.0.0.1	127.0.0.1	TCP	68 47214 → 55555 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1014028867 TSecr=1014028867
104736	4846.663235791	127.0.0.1	127.0.0.1	OCSP	271 Request
104737	4846.663251138	127.0.0.1	127.0.0.1	TCP	68 55555 → 47214 [ACK] Seq=1 Ack=204 Win=65280 Len=0 TSval=1014028867 TSecr=1014028867
104738	4846.665194137	127.0.0.1	127.0.0.1	OCSP	1613 Response
104739	4846.665216540	127.0.0.1	127.0.0.1	TCP	68 47214 → 55555 [ACK] Seq=204 Ack=1546 Win=81408 Len=0 TSval=1014028869 TSecr=1014028869
104740	4846.666240247	127.0.0.1	127.0.0.1	TCP	68 55555 → 47214 [FIN, ACK] Seq=1546 Ack=204 Win=65536 Len=0 TSval=1014028870 TSecr=1014028869
104741	4846.666390763	127.0.0.1	127.0.0.1	TCP	68 47214 → 55555 [FIN, ACK] Seq=204 Ack=1547 Win=81408 Len=0 TSval=1014028870 TSecr=1014028870
104742	4846.666334402	127.0.0.1	127.0.0.1	TCP	68 55555 → 47214 [ACK] Seq=1547 Ack=205 Win=65536 Len=0 TSval=1014028870 TSecr=1014028870
104743	4846.670942935	127.0.0.1	127.0.0.1	TLSv1.3	4129 Server Hello, Change Cipher Spec, Application Data, Application Data, Application Data, Ap...
104744	4846.670065280	127.0.0.1	127.0.0.1	TCP	68 47214 → 443 [ACK] Seq=1893 Ack=4062 Win=103040 Len=0 TSval=1014028874 TSecr=1014028874
104745	4846.673413970	127.0.0.1	127.0.0.1	TLSv1.3	132 Change Cipher Spec, Application Data
104746	4846.673028798	127.0.0.1	127.0.0.1	TLSv1.3	572 Application Data
104747	4846.673923454	127.0.0.1	127.0.0.1	TLSv1.3	147 Application Data
104748	4846.674094964	127.0.0.1	127.0.0.1	TLSv1.3	147 Application Data
104749	4846.674326990	127.0.0.1	127.0.0.1	TCP	68 47214 → 443 [ACK] Seq=2461 Ack=4220 Win=103040 Len=0 TSval=1014028878 TSecr=1014028878
104750	4846.675478265	127.0.0.1	127.0.0.1	TLSv1.3	676 Application Data

mejor, mensajes OCSP dentro de su contexto!!

Analizar los mensajes intercambiados y mostrar las diferencias que supone utilizar TLSv1.3 frente a TLSv1.2

104732	4846.660329338	127.0.0.1	127.0.0.1	TCP	70 42724 - 443 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1014028864 TSecr=0 WS=128
104731	4846.660374082	127.0.0.1	127.0.0.1	TCP	70 443 - 47214 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1014028864 TSecr=0 WS=128
104730	4846.660110836	127.0.0.1	127.0.0.1	TCP	68 47214 - 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1014028864 TSecr=1014028864
104731	4846.661461924	127.0.0.1	127.0.0.1	TLVsl.3	1960 Client Hello (SHA1-group01)
104732	4846.661522235	127.0.0.1	127.0.0.1	TCP	68 443 - 47214 [ACK] Seq=1 Ack=1893 Win=86784 Len=0 TSval=1014028865 TSecr=1014028865
104733	4846.663146488	127.0.0.1	127.0.0.1	TCP	76 41422 - 55555 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=1014028867 TSecr=0 WS=128
104734	4846.663185222	127.0.0.1	127.0.0.1	TCP	76 55555 - 41422 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM TSval=1014028867 TSecr=0 WS=128
104735	4846.663185101	127.0.0.1	127.0.0.1	TCP	68 41422 - 55555 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1014028867 TSecr=1014028867
104733	4846.663235791	127.0.0.1	127.0.0.1	OCSP	271 Request
104737	4846.663251138	127.0.0.1	127.0.0.1	TCP	68 55555 - 41422 [ACK] Seq=1 Ack=204 Win=65280 Len=0 TSval=1014028867 TSecr=1014028867
104738	4846.665194137	127.0.0.1	127.0.0.1	OCSP	1613 Response
104739	4846.665216549	127.0.0.1	127.0.0.1	TCP	68 41422 - 55555 [ACK] Seq=204 Ack=1546 Win=81408 Len=0 TSval=1014028869 TSecr=1014028869
104740	4846.666249247	127.0.0.1	127.0.0.1	TCP	68 55555 - 41422 [FIN, ACK] Seq=1546 Ack=204 Win=65536 Len=0 TSval=1014028870 TSecr=1014028869
104741	4846.66638763	127.0.0.1	127.0.0.1	TCP	68 41422 - 55555 [FIN, ACK] Seq=204 Ack=1547 Win=81408 Len=0 TSval=1014028870 TSecr=1014028870
104742	4846.666334492	127.0.0.1	127.0.0.1	TCP	68 55555 - 41422 [ACK] Seq=1547 Ack=204 Win=65536 Len=0 TSval=1014028870 TSecr=1014028870
104743	4846.678942835	127.0.0.1	127.0.0.1	TLVsl.3	4129 Server Hello, Change Cipher Spec, Application Data, Application Data, Application Data, Application Data, Ap...
104744	4846.678965280	127.0.0.1	127.0.0.1	TCP	68 47214 - 443 [ACK] Seq=1893 Ack=4062 Win=103040 Len=0 TSval=1014028874 TSecr=1014028874
104745	4846.6793413970	127.0.0.1	127.0.0.1	TLVsl.3	132 Change Cipher Spec, Application Data
104746	4846.679382879	127.0.0.1	127.0.0.1	TLVsl.3	572 Application Data
104747	4846.679323454	127.0.0.1	127.0.0.1	TLVsl.3	147 Application Data
104748	4846.674994964	127.0.0.1	127.0.0.1	TLVsl.3	147 Application Data
104749	4846.674326990	127.0.0.1	127.0.0.1	TCP	68 47214 - 443 [ACK] Seq=2461 Ack=4220 Win=103040 Len=0 TSval=1014028878 TSecr=1014028878
104750	4846.675482865	127.0.0.1	127.0.0.1	TLVsl.3	670 Application Data

Analizando el handshake podemos encontrar más información:

Y hemos visto como se hace!

```

Version: TLS 1.2 (0x0303)
Length: 1
Change Cipher Spec Message
- TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
  Opaque Type: Application Data (23)
  Version: TLS 1.2 (0x0303)
  Length: 42
  Encrypted Application Data: bcb2b126d4e784cd621cd59779b2d871efe12e4f527bd7ab047ae2aac0a3bbd4994720f6c31a37136a2
    [Application Data Protocol: Hypertext Transfer Protocol]
- TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
  Opaque Type: Application Data (23)
  Version: TLS 1.2 (0x0303)
  Length: 3532
  Encrypted Application Data [truncated]: 7d9f67f3a1dc94119514bb51c13a52b1c3894a773d53fcb47054dbec964e1d9453d6ff7fa19ad7635584a42f073d8fd2e5890a85b016d775922e715df0794967
    [Application Data Protocol: Hypertext Transfer Protocol]
- TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
  Opaque Type: Application Data (23)
  Version: TLS 1.2 (0x0303)
  Length: 281
  Encrypted Application Data [truncated]: 57fcd8d1bf5f3bfc979d35974c73f8b3ceb54e22f7855bee6ce0d9544ac0786fefff29a8248cf9beb5f230aa3aa8d61c2df681747502b267f43d4bfa48da41
    [Application Data Protocol: Hypertext Transfer Protocol]

```

```
SSLStaplingCache "shmcb:logs/ssl_stapling(12000)"
<VirtualHost *:443>
    ServerName grupo01
    DocumentRoot /var/www/grupo01

    SSLEngine on

    SSLCertificateFile /home/user1/PKI2/certs/webserverGX.crt
    SSLCertificateKeyFile /home/user1/PKI2/private/webserverGX.key
    SSLCACertificateFile /home/user1/PKI2/CA-root.crt

    SSLProtocol -all +TLSv1.2
    SSLUseStapling On
    SSLStaplingForceURL http://ocsp.srdsgx.lab:55555
    SSLStaplingResponseMaxAge 90

    ErrorLog ${APACHE_LOG_DIR}/grupo01.log
    CustomLog ${APACHE_LOG_DIR}/grupo01_access.log combined
</VirtualHost>
```

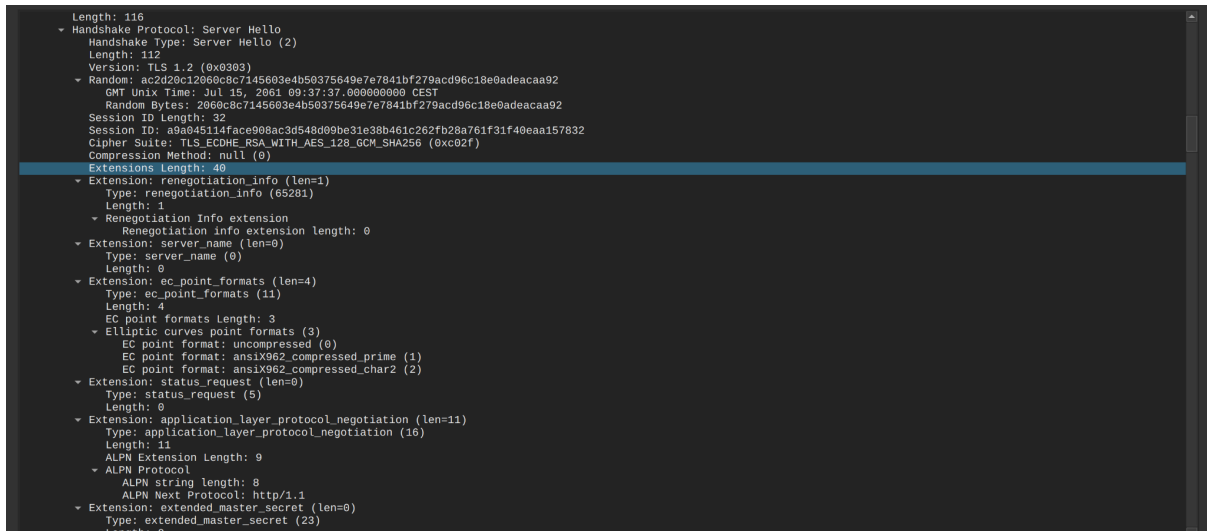
Finalmente recargamos apache utilizando: `sudo systemctl reload apache2`.

Una vez se ha establecido la conexión, analizamos el proceso en wireshark.

EL proceso es el mismo solo que esta vez el protocolo TLS utiliza la versión 1.2.

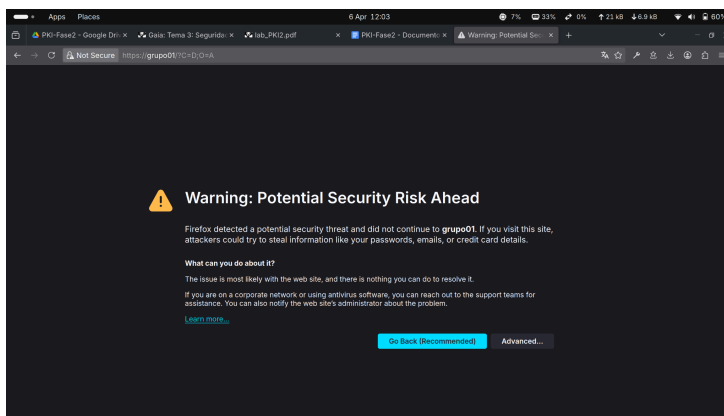
No, en el tratamiento que da a OSCP

Entrando a analizar el Handshake **la diferencia clave es la visibilidad**, donde apreciamos que los mensajes están en texto claro, lo que puede dar problemas si capturan la trama.



Si revocamos el certificado del servidor web y reintentar el acceso seguro ¿Qué ocurre?

Cuando revocamos el certificado del servidor web y volvemos a acceder mediante HTTPS, se observa que la conexión se puede llegar a establecer, solo que el navegador nos da una advertencia visual.



La advertencia nada tiene que ver con la revocación del certificado

Ejercicio 3

Añadimos la directiva indicada en el archivo de configuración de la PKI:

```
sudo vi PKI2/ca_openssl.cnf
```

```
[ server-ext ]
basicConstraints = CA:FALSE
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
tlsfeature = status_request
```

Generamos un certificado nuevo para la PKI:

```
openssl req -new -key private/webserverGX.key -out certs/webserverGX.csr -config
ca_openssl.cnf
openssl ca -config ca_openssl.cnf -in certs/webserverGX.csr -out certs/webserverGX.crt
-extensions server-ext
```

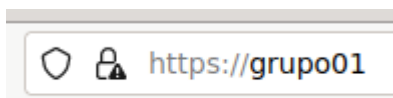
Datos del certificado??

Reiniciamos apache:

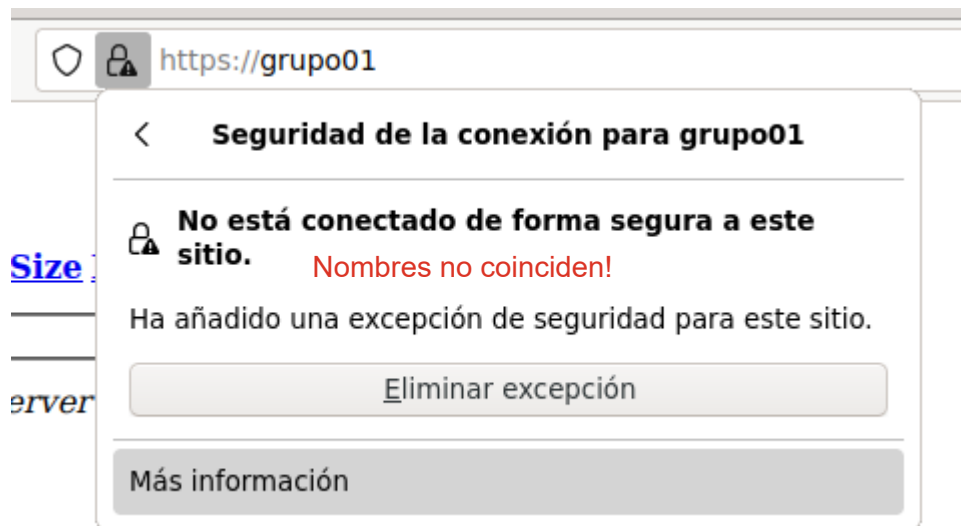
```
sudo systemctl restart apache2
```

Comprobar que aparece la nueva extensión en el certificado del sitio web seguro.

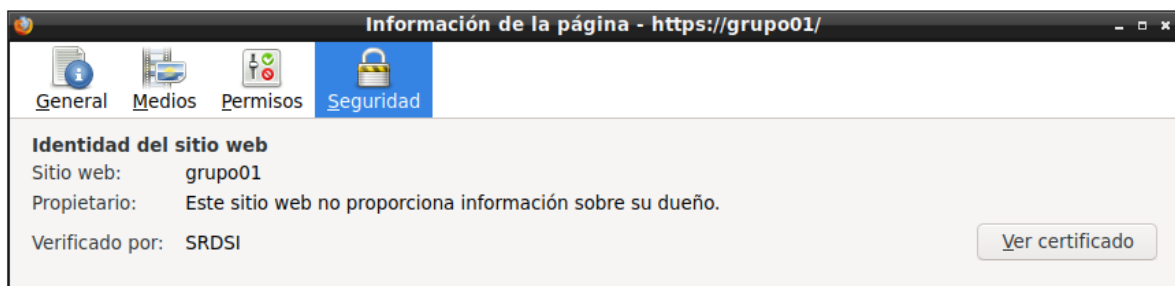
Accedemos al sitio web:



Hacemos click sobre el candado y "Más información":



Seleccionamos “Ver certificado” en la pantalla emergente:



Podemos comprobar que marca OCSP como requerido:

Sello de tiempo OCSP

Sin contexto no tiene validez!

Requerido **Sí**

Con el OCSP Responder detenido, comprobar que se deniega el acceso al sitio web seguro:

Buscando en firefox about:config, modificamos la opción `security.OCSP.require` a true.



Poner en marcha el OCSP Responder, y comprobar que ahora sí accede

EJERCICIO Extra

En los apartados anteriores, nos hemos centrado en cómo el navegador valida el certificado del servidor web. Sin embargo, en escenarios donde se requiere autenticación mutua (el servidor también valida al cliente), es crucial que el servidor pueda verificar si el certificado presentado por el usuario está revocado. Este ejercicio configura el servidor Apache para que utilice OCSP para validar el certificado del cliente.

Configuración del servidor Apache

Primero, debemos modificar la configuración (grupo01.conf) para que solicite un certificado al cliente y active la comprobación OCSP para dicho certificado.

Editamos el archivo /etc/apache2/sites-available/grupo01.conf

Descomentamos/añadimos las siguientes directivas dentro del bloque <VirtualHost *:443>:

```
# (Opcional) Para la validación del certificado del usuario
SSLCSPEnable on
SSLVerifyClient require
SSLVerifyDepth 10
```

Guardamos los cambios y recargamos Apache:

```
sudo systemctl restart apache2
```

Si aún no has importado el certificado de la CA, vete a la configuración de los certificados de tu navegador y añade el certificado CA-root.crt **en el servidor**

Crear un Certificado de Usuario

Generar la clave privada para el usuario.

```
openssl genpkey -algorithm RSA -out private/usuario01.key
```

Generar la CSR para el usuario.

```
openssl req -new -key private/usuario01.key -out certs/usuario01.csr -config
ca_openssl.cnf
```

Firmar la CSR del usuario con la CA.

```
openssl ca -config ca_openssl.cnf -in certs/usuario01.csr -out certs/usuario01.crt
-extensions client_cert
```

Crear un archivo PKCS 12

```
openssl pkcs12 -export -out certs/usuario01.p12 -inkey private/usuario01.key -in  
certs/usuario01.crt -certfile CA-root.crt
```

Prueba de Acceso Inicial (Cliente Válido)

Con el servidor OCSP en funcionamiento y la configuración de Apache recargada:

- Accedemos a <https://grupo01> capturas resultados??
- El navegador ahora debería solicitar que seleccionemos un certificado personal para autenticarnos. Elegimos el certificado de usuario que acabamos de importar
- Si todo es correcto (certificado de cliente válido y firmado por la CA, Apache lo verifica y comprueba con OCSP que no está revocado), deberíamos acceder correctamente al sitio web seguro

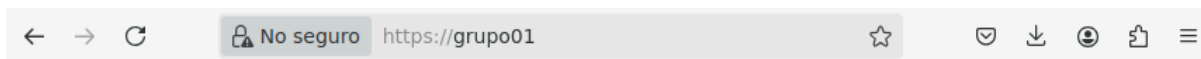
Revocación del Certificado de Cliente

Ahora revocaremos el certificado del *cliente* que hemos usado

```
openssl ca -config ca_openssl.cnf -revoke certs/usuario01.pem
```

Al acceder de nuevo a la página web nos encontraremos con el siguiente fallo, que indica que no se ha podido verificar la identidad del usuario.

no hay resultados



Conexión segura fallida

Ha ocurrido un error al conectar con grupo01. El otro extremo de la conexión SSL no ha podido negociar un conjunto aceptable de parámetros de seguridad.

Código de error: SSL_ERROR_HANDSHAKE_FAILURE_ALERT

- La página que está intentando ver no se puede mostrar porque la autenticidad de los datos recibidos no ha podido ser verificada.
- Contacte con los propietarios del sitio web para informarles de este problema.

[Más información...](#)

Reintentar