UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Seguridad Web

Curso 2021/22

Práctica 1.
El protocolo HTTP

Contenido

1. Objetivos 2
2. Introducción 2
   - 2.1. Navegadores web 2
   - 2.2. Herramientas de manejo de conexiones mediante línea de órdenes 2
   - 2.3. *Burp Suite Community Edition* 3
   - 2.4. Material de la práctica 3
3. El protocolo HTTP 4
   - 3.1. Analizando las cabeceras de HTTP 4
   - 3.2. Modificando las cabeceras de los mensajes de petición HTTP 6
   - 3.3. Modificando las cabeceras de los mensajes de respuesta HTTP 7
   - 3.4. Herramientas del desarrollador del navegador web *Firefox* 9
   - 3.5. Utilización del proxy de interceptación de la *Burp Suite* 10
4. Entrega de resultados 11

1 Objetivos

Los objetivos y actividades que se va a abordar en está práctica son:

- Conocer las bases del protocolo HTTP.
- Conocer y comprender aquellos aspectos que se relacionan con la seguridad.
- Inspeccionar las cabeceras de los mensajes entre el cliente y el servidor.
- Utilizar servidores *proxy* que permitan inspeccionar y modificar algunos de los campos de los mensajes HTTP.

2 Introducción

El protocolo HTTP es el principal mecanismo de comunicación entre un navegador web ("el cliente") y un servidor web ("el servidor"). En esta práctica se llevará a cabo un introducción sobre su estructura y algunos de sus campos más relevantes en relación a la seguridad web.

Para llevar a cabo el estudio de los mensajes HTTP entre el cliente y el servidor, y en general para realizar muchas de las prácticas de la asignatura, se van a utilizar diversas herramientas que se describen a continuación.

2.1 Navegadores web

Durante las sesiones de prácticas utilizaremos dos navegadores: *Mozilla Firefox* y *Chromium*. Ambos poseen herramientas de depuración que permiten inspeccionar y editar el código de una página web, los mensajes enviados entre el navegador y el servidor web, los datos almacenados en el cliente, los estilos utilizados, etc.

En general se utilizará Mozilla Firefox para trabajar los temas relacionados con el desarrollo de la práctica, dejando el Chromium para otro tipo de búsquedas y para cuando se necesite utilizar dos sesiones/usuarios distintos con el mismo servidor.

Cualquier referencia a un menú o una opción del navegador, se referirán en general al navegador Firefox. Cuando nos refiramos al *Menú* se entenderá que es el menú desplegable que se encuentra normalmente en la parte superior derecha. Cualquier opción del menú que requiera navegación y no tenga una combinación de teclas asociada o ésta no sea estándar, se indicará de la siguiente forma: *Menú* >> *Ayuda* >> *Acerca de Firefox*. Por otro lado, para indicar una combinación de teclas se utilizará el formato mostrado en el siguiente ejemplo: `Ctrl + Alt + Meta + F4`, siendo `Meta` la tecla típicamente con el logo de Windows.

Entre otras herramientas importantes, la que más utilizaremos será las herramientas del desarrollador que llevan integradas. Éstas se pueden activar o desactivar en *Menú* >> *Desarrollador Web* >> *Alternar herramientas* o pulsando `Ctrl + Shift + I`.

2.2 Herramientas de manejo de conexiones mediante línea de órdenes

Se van a utilizar un conjunto de programas lanzados desde la línea de órdenes. Entre ellos destacan:

- `nc` o *netcat*, que permite realizar peticiones TCP o UDP y recibir conexiones. Con ella se puede implementar un proxy TCP básico.

- `wget`, es una herramienta muy completa para interacción con servidores web. Permite hacer peticiones HTTP, bajarse ficheros, webs completas, etc.
- `curl` permite hacer múltiples tipos de transferencias hacia o desde una URL. Soporta múltiples protocolos: FILE, FTP, HTTP, IMAP, LDAP, POP3, RTMP, SMB, SMTP, TELNET, TFTP, etc. y sus versiones seguras. Permite además realizar secuencias de peticiones a una o múltiples webs.
- `php` es el interprete de línea de órdenes del conocido lenguaje de programación de servidores web. Además de ejecutar scripts de cualquier índole (php se puede utilizar para casi cualquier tipo de aplicación), el interprete incluye un servidor web integrado que utilizaremos para ejecutar aplicaciones web sencillas.

Se puede obtener información sobre el uso de estas herramientas utilizando la orden `man`.

2.3 *Burp Suite Community Edition*

Es un conjunto de herramientas integradas enfocadas a la depuración, monitorización y ataque de servidores web. Entre sus características y herramientas más destacables dentro de la edición abierta están:

- Un proxy de interceptación que retiene todas los mensajes entre el cliente y el servidor web para su análisis, modificación, descarte o reenvío. También se puede seleccionar qué mensajes se interceptan y qué mensajes se dejan pasar.
- Un histórico de todos los mensajes enviados o recibidos entre ambos para su posterior análisis y/o etiquetado.
- Permite la modificación automática de algunos mensajes, tanto del cuerpo como de las cabeceras.
- Construye un mapa incremental de la web que se está visitando así como de todos los enlaces que en ella se mencionan. Para los nodos que se han visitado se mantienen tanto los mensajes de petición como de respuesta.
- Un *repetidor* que permite editar y reenviar cualquiera de los mensajes que ya se han solicitado.
- Un *decodificador* para convertir entre las codificaciones y formatos más habituales en la web.

2.4 Material de la práctica

Todo el material de apoyo para realizar esta sesión de prácticas se puede encontrar en *PoliformaT >> Recursos >> Prácticas >> Práctica 1 HTTP >> Material*, en el fichero `prac-1-http.tgz`. Para descomprimir dicho fichero una vez descargado se deberá utilizar la siguiente orden:

```
$> cd SEW
$> tar xvzf ~/Descargas/prac-1-http.tgz
prac-1-http/
...
$> cd prac-1-http
```

Siendo el directorio `~/Descargas/` el directorio donde se ha bajado dicho fichero desde PoliformaT y `SEW` el directorio que cada alumno utiliza para almacenar los trabajos realizados en esta asignatura.

Al descomprimir el fichero se creará un directorio `prac-1-http/` que se deberá utilizar como directorio de trabajo durante toda la sesión. Cualquier ruta relativa utilizada en esta memoria supondrá dicho directorio como directorio actual.

3 El protocolo HTTP

Utilizando las herramientas mencionadas se va a llevar a cabo en esta sesión un estudio práctico del funcionamiento del protocolo HTTP y de cómo se utilizar para detectar algunas de las vulnerabilidades de las aplicaciones web.

3.1 Analizando las cabeceras de HTTP

Como primer ejercicio se va a llevar cabo el análisis sencillo de la estructura de los mensajes HTTP utilizando las órdenes `nc` y `wget`. Para ello se va a construir un sencillo proxy TCP utilizando la orden `nc` y el uso de colas FIFO. A continuación se muestra un ejemplo sencillo de como utilizar estas órdenes. Recordad que el símbolo `$>` representa el *prompt* del terminal; **no** se teclea.

```
$> mkfifo /tmp/fifo
$> nc -l -p 8080 </tmp/fifo | nc www.upv.es 80 >/tmp/fifo
$> rm /tmp/fifo
```

Estas órdenes escuchan cualquier petición en el puerto `8080` de la máquina local y la envían al puerto `80` del servidor `www.upv.es`. Si ahora utilizamos la orden `wget` que se muestra a continuación la petición le llegará al servidor web de la universidad.

```
$> wget localhost:8080
```

Deberíamos obtener una respuesta similar a la que se muestra a continuación. Además el fichero de respuesta se debería haber guardado como un fichero con el nombre `index.html`.

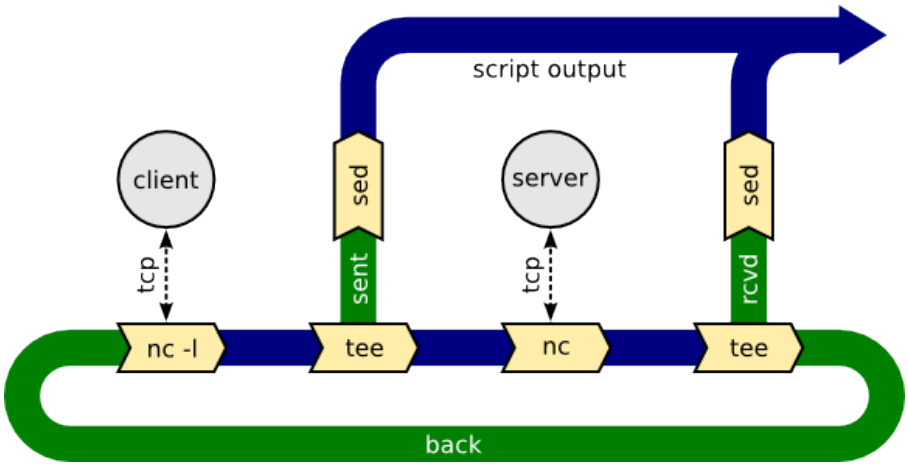
```
--2019-02-07 01:17:01--  http://localhost:8080/
Resolviendo localhost (localhost)... 127.0.0.1
Conectando con localhost (localhost)[127.0.0.1]:8080... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: no especificado [text/html]
Grabando a: “index.html”

index.html        [ <=>                          ]  53,88K  --.-KB/s  in 0,05s

2019-02-07 01:17:01 (1024 KB/s) - “index.html” guardado [55171]
```

Ejercicio 1. Ejecutad las órdenes mencionadas con anterioridad. Necesitaréis lanzar dos terminales, uno para el *proxy* y otro para lanzar la orden **`wget`**. Comprobad que el contenido del fichero **`index.html`** es el adecuado.

Ejercicio 2. Solicitad ahora alguna página del mismo servidor que no exista y comprobad el código de error mostrado por **`wget`**.

```
$> wget localhost:8080/noexiste.html
```

Hemos comprobado que el proxy funciona, pero no podemos ver los mensaje intercambiado entre el cliente y el servidor. Vamos a proceder a ampliar el proxy, almacenándolo en script, para que nos muestre los mensaje enviados y recibidos. El script seguirá el siguiente esquema:



A continuación se encuentra el código del script correspondiente, aunque el fichero `tcp-proxy.sh` que lo contiene se encuentra en *PoliformaT >> Recursos >> Prácticas >> Práctica 1 HTTP >> Material*.

```
if [ $# != 3 ]
then
    echo "usage: $0 <src-port> <dst-host> <dst-port>"
    exit 0
fi

TMP=`mktemp -d`
BACK=$TMP/pipe.back
SENT=$TMP/pipe.sent
RCVD=$TMP/pipe.rcvd
trap 'rm -rf "$TMP"' EXIT

mkfifo -m 0600 "$BACK" "$SENT" "$RCVD"
sed -e 's/^/ => /' | grep HTTP <"$SENT" &
sed -e 's/^/<=  /' | grep HTTP <"$RCVD" &

nc -l -p "$1" < "$BACK" | tee "$SENT" | nc "$2" "$3" | tee "$RCVD" > "$BACK"
```

Al igual que en el caso anterior, este proxy se quedará escuchando en el puerto indicado `<src-port>` de la máquina local (debe ser mayor que el puerto 1024) y reenviará las peticiones al servidor indicado a continuación: `<dst-host> <dst-port>`.

Ejercicio 3. Probad a lanzar el servidor proxy contra el servidor **`cursophp.disca.upv.es`**.

```
$> scripts/tcp-proxy.sh 8080 cursophp.disca.upv.es 80
```

Ejercicio 4. Comprobad los mensajes de envío y recepción que aparecen en la consola del proxy.
- ¿Cómo se identifica el cliente **`wget`** ante el servidor?
- ¿Qué servidor web se está utilizando en dicha dirección?
- ¿Cuál es la fecha de modificación de la página?

Ejercicio 5. Comprobad con el navegador que el contenido de la página **`cursophp.disca.upv.es`** es el correcto. En caso de duda, consultad el código fuente de la página utilizando el `Botón derecho` >> *Ver código fuente de la página*.
- ¿El contenido de la página cargada en el navegador parece coincidir con lo recibido?
- ¿Qué indicaba el campo **`Host`** de la petición que se le ha enviado al servidor web?
- ¿Corresponde con la dirección del servidor web o con la del *proxy*?

Captura 1. Enviar por el chat privado de Teams.
Realizar una o varias capturas de pantalla del terminal con la ejecución del proxy **`tcp-proxy.sh`** en las que se muestre la parte inicial del acceso a la web **`cursophp.disca.upv.es`**.

El problema que aparece al utilizar un proxy *tcp* básico es que los mensaje se envían sin modificar. Sin embargo, dado que el servidor web `cursophp.disca.upv.es` está alojado en el la misma máquina que el servidor `muses.upv.es`, estos servidores *virtuales* sólo se pueden diferencias por el nombre utilizado. Si el campo `Host` no está establecido correctamente el servidor web devuelve la página del servidor virtual por omisión (en este caso `muses.upv.es`).

Ejercicio 6. Comprobad con el navegador que el contenido de la dirección **`muses.upv.es`** se corresponde efectivamente con lo que se había obtenido utilizando el *proxy*.

Para solventar el problema vamos a llevar a cabo una ampliación del *proxy* que permita modificar las peticiones antes de enviárselas al servidor web.

3.2 Modificando las cabeceras de los mensajes de petición HTTP

En este apartado se va a ampliar el proxy utilizado con anterioridad para permitir la modificación al vuelo de cabeceras HTTP. Utilizaremos como base el script `tcp-filtering-proxy.sh` que se encuentra en el fichero descargado desde *PoliformaT*.

La diferencia fundamental se encuentra en la cadena de invocación a la orden `nc`. En esta ocasión hemos colocado un filtro entre el servidor y el cliente que invoca un script implementado con `awk` (se usará realmente la implementación `mawk`). El fragmento se puede observar a continuación:

```
DIR=$(dirname $0)

FILTER_CLT_SRV="mawk -W interactive -f ${DIR}/filter-clt-srv.awk"

nc -l -p "$1" < "$BACK" | tee "$SENT" | \
    ${FILTER_CLT_SRV} -v HOST="$2" -v PORT="$3" | tee "$FILTERIN"  | \
    nc "$2" "$3" | tee "$RCVD" \
    > "$BACK"
```

El filtro que vamos a aplicar es sencillo. Dado que `awk` es una herramienta de procesamiento de líneas, lo que hace el fragmento que se muestra a continuación:

1. Si la línea comienza por la cadena `Host:` el filtro escribe una nueva cabecera *Host* utilizando las variables recibidas desde el script principal. Es de vital importancia imprimir al final de la cabe-

cera los caracteres `CR LF`.

2. Si la línea *no* comienza por la cadena `Host:` el filtro imprime la línea original (`$0`).

```
/^Host: / {
    printf("Host: %s:%d\r\n", HOST, PORT);
}

! /^Host: / {
    print $0;
}
```

Ejercicio 7. Probad a acceder a la página **`cursophp.disca.upv.es`** con **`wget`** utilizando el nuevo proxy **`tcp-filtering-proxy.sh`**. Los scripts se lanzan como en el apartado anterior.

Ejercicio 8. Analizar cuál será la respuesta del servidor web si quitamos el carácter **`\r`** de la orden **`printf`**. Editar el fichero **`filter-clt-srv.awk`** y probadlo.

Fijaos que en el texto que aparece por la consola el aspecto de la cabecera *parece* correcto, dado que el carácter `\r` no se visualiza y no es necesario en los sistema Unix.

Tras reponer la secuencia correcta `\r\n`, vamos a añadir algún filtro adicional en alguna cabecera inofensiva por el momento.

Ejercicio 9. Modificar el fichero **`filter-clt-srv.awk`** para que cambien la cadena de identificación del cliente **`User-Agent`** por una identificación personalizada ... al gusto.

Para que se impriman las líneas que no cumplen ninguno de los patrones se puede utilizar la combinación `!/Patron 1/ && !/Patron 2/`.

Captura 2. Enviar por el chat privado de Teams.
Realizar una o varias capturas de pantalla del terminal con la ejecución del proxy **`tcp-filtering-proxy.sh`** en las que se muestre la parte inicial del acceso a la web **`cursophp.disca.upv.es`**.

Captura 3. Enviar por el chat privado de Teams.
Realizar también una captura con el código de **`filter-clt-srv.awk`** en un editor.

Entregable.
Se deberá entregar el fichero **`filter-clt-srv.awk`**.

3.3 Modificando las cabeceras de los mensajes de respuesta HTTP

A continuación vamos a añadir un filtro adicional que cambie ahora una cabecera del mensaje de respuesta del servidor. Necesitaréis crear otra cola FIFO similar a la creada para filtrar la entrada (`FILTE-RIN`).

Para utilizar un formato de fecha similar al utilizado por los servidores web se puede utilizar la orden `date` desde la terminal con la opción `-d` para especificarle una fecha (formato `YYYY-MM-DD`) y la opción `-R` para que utilice el formato de salida rfc-2822 (ver `man date`). A continuación se muestra un ejemplo:

```
$> date -d "1970-10-27" -R
Tue, 27 Oct 1970 00:00:00 +0100
```

Ejercicio 10. Crear un nuevo filtro **`filter-srv-clt.awk`**, similar al script awk **`filter-clt-srv.awk`**, en el que sólo se cambie la cabecera **`Date`** y ponga la fecha de vuestro cumpleaños en el formato apropiado.
Incorporad el nuevo filtro al script **`tcp-filtering-proxy.sh`** con el nombre **`FILTER_SRV_CLT`** e insertadlo detrás del **`tee "$RCVD"`**.
Nota: La cadena con la fecha, que habréis copiado de la consola, estará en una variable FECHA del script y se la pasaréis al filtro awk a través de un parámetro DATE. Podéis ver un ejemplo al principio de la sección anterior.

Ejercicio 11. Se deberá incorporar otra cola FIFO **`FILTEROUT`** y otro filtro para ver la versión de la respuesta después de que se haya filtrado lo que envía el servidor. Éste se deberá insertar en el script después de la ejecución de **`filter-srv-clt.awk`**. El nuevo filtro de salida será similar al que se muestra a continuación.
Nota: Acordaos de añadir **`FILTEROUT`** en la orden de creación de colas FIFO **`mkfifo`**.

```
sed -e 's/^/<#  /' -n -e '1,20p' <"$FILTEROUT" &
```

Al añadir las opciones `-n -e '1,20p'` la orden `sed` sólo procesará y mostrará las 20 primeras líneas.

Captura 4. Enviar por el chat privado de Teams.
Realizar una o varias capturas de pantalla del terminal con la ejecución del nuevo proxy **`tcp-filtering-proxy.sh`** en las que se muestre la parte inicial del acceso a la web **`cursophp.disca.upv.es`**.

Captura 5. Enviar por el chat privado de Teams.
Realizar también unas capturas con el código de los ficheros **`tcp-filtering-proxy.sh`** y **`filter-srv-clt.awk`** en un editor.

Entregable.
Se deberán entregar los ficheros **`tcp-filtering-proxy.sh`** y **`filter-srv-clt.awk`**.

Ejercicio 12. Probad el funcionamiento del proxy accediendo a la página web del departamento DISCA **`www.disca.upv.es`**. ¿Funciona correctamente? ¿Qué ha mostrado el proxy por la consola? ¿Qué ha mostrado la orden **`wget`**?

En este último caso hemos visto que algunos códigos de error causan acciones adicionales en cliente.

Vamos a continuación a analizar la interacción entre el navegador y el servidor utilizando las propias herramientas de desarrollo del navegador web.

3.4 Herramientas del desarrollador del navegador web *Firefox*

En está sección se va a utilizar el navegador *Firefox* para observar el contenido de los mensajes intercambiados entre un cliente y un servidor web. Para ello utilizaremos las herramientas de desarrollador que podemos obtener al presionar las teclas `Ctrl + Shift + I`.

En los casos anteriores, cuando utilizábamos la herramienta `wget`, ésta sólo realizaba una petición al servidor web: la URL solicitada. A continuación vamos a ver como un navegador web realiza múltiples peticiones adicionales, además de la relacionada con la URL introducida en el navegador, cuando comienza a analizar su contenido para llevar a cabo la visualización.

La opción *Network*

Esta opción permite ver todo el trasiego de información entre el cliente y el servidor web, así como llevar a cabo pequeñas ediciones sobre los mensajes enviados. Esta opción se puede lanzar directamente con las teclas `Ctrl + Shift + E`.

Ejercicio 13. Abrir la ventana de las herramientas de desarrollador, seleccionar la opción *Network* o *Red* y acceder a la página principal de la UPV `www.upv.es`.

Se pueden observar todos los recursos solicitados, el método utilizado para obtenerlo, el proveedor, el nombre del archivo, por qué motivo se ha solicitado, de qué tipos es, los datos transferidos o si se ha obtenido de la cache, el tamaño, etc.

Si se selecciona uno de dichos recursos, se puede acceder a las cabeceras utilizadas, a las cookies, los parámetros enviados, las respuesta obtenida, etc. Además, la solicitud utilizada para su petición se puede editar y volver a enviar. Existen además múltiples puntos en la información mostrada en que se puede consultar ayuda en línea sobre su significado.

Ejercicio 14. Seleccionar algunos de los símbolos/iconos de información para saltar a la documentación asociada en la web de *MDN* https://developer.mozilla.org/.

Ejercicio 15. Modificar alguna petición para solicitar, por ejemplo, un recurso no existente y analizar la respuesta obtenida por el servidor.

Captura 6. Enviar por el chat privado de Teams.
Realizar una captura del navegador mostrando la opción *Network* y el acceso a un recurso no existente solicitado en el ejercicio anterior.

La opción *Inspector*

Esta opción, a parte de ayudar en el desarrollo y diseño de páginas web, dada su capacidad de edición interactiva del contenido de una página web y sus estilos, tiene otras utilidades desde el punto de vista

de la seguridad. Esta opción se puede lanzar directamente con las teclas `Ctrl + Shift + C`.

En nuestro caso, se podría utilizar el editor y localizador de elementos integrado en esta opción para la edición de formularios y así eliminar de forma manual, restricciones, comprobaciones en java, limitaciones de tamaño, campos ocultos, etc.

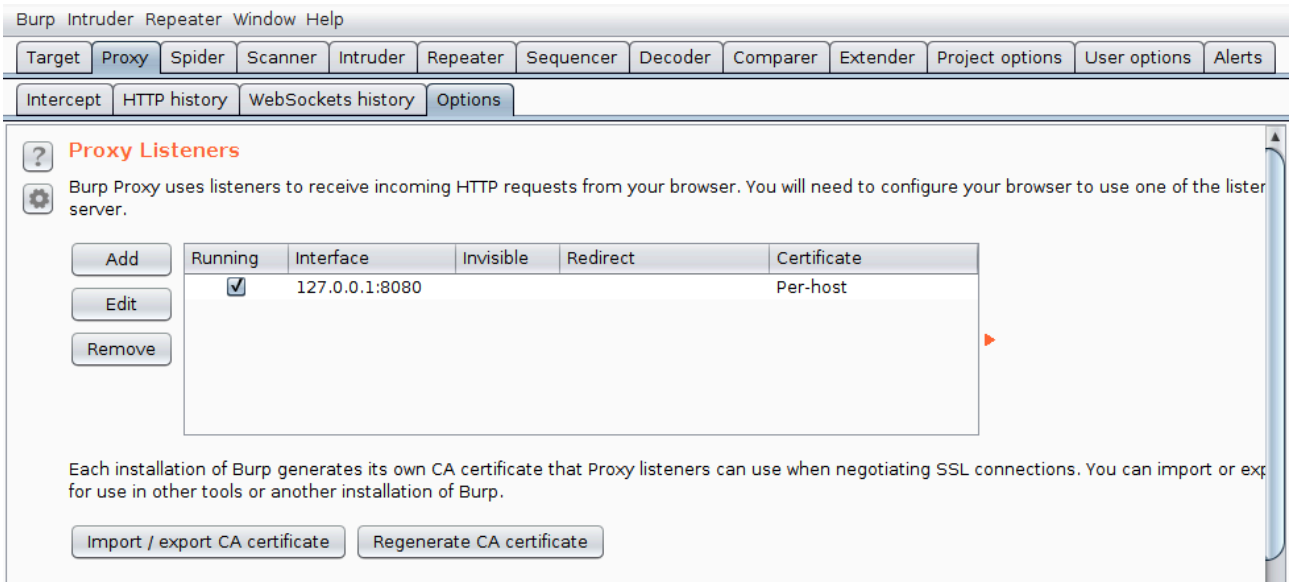
Ejercicio 16. Navegar a una página con un formulario, localizarlo y modificar el campo **`action`** del elemento **`form`** para que envíe la información a un servidor distinto, v.g. a nuestro proxy que nos debería mostrar el mensaje de la petición y de la respuesta.

3.5 Utilización del proxy de interceptación de la *Burp Suite*

Por último vamos a llevar a cabo una visita exploratoria de la *Burp Suite Community Edition* y más concretamente su uso como proxy de interceptación para visualizar y modificar los mensaje HTTP entre el cliente y el servidor.

Para ello antes hay que configurar el navegador *Firefox* para que utilice la herramienta *Burp Suite Community* (en adelante *BSC*) como proxy de red. Lo primero será lanzar dicha herramienta y averiguar en qué puerto está escuchando el proxy de interceptación.

Una vez lanzada la herramienta *BSC*, con la orden `BurpSuiteCommunity` desde un terminal, y aceptadas las licencias oportunas y entendidas las limitaciones de la versión *CE* se debería visualizar la pantalla principal. Se organiza como dos niveles de pestañas. El primer nivel, las herramientas (*Target*, *Proxy*, *Spider*, ...) y en el segundo nivel las vistas u opciones de la herramienta seleccionada. En la figura a continuación se muestra las *opciones* de la herramienta *Proxy*, que referenciaremos como *Proxy >> Options*.



En el se puede observar la dirección y el puerto utilizado por *BSC* como dirección de escucha del proxy.

Ejercicio 17. Buscad la opción de establecer el proxy en el navegador *Firefox* y establecer el ofrecido por *BSC*.

Ejercicio 18. Visitar la página web de la UPV, navegando por varias de sus opciones y observad como se va construyendo un mapa de la web y los enlaces alcanzables desde sus páginas en la sección *Target >> Site map*.

Ahora activaremos el proxy de interceptación y probaremos algunas de sus opciones. Se sugieren algunos ejercicios a continuación, pero se puede llevar a cabo cualquier *experimento* que os llame la atención o despierte vuestra curiosidad.

Ejercicio 19. Acceder al servidor web de la UPV y ver como se puede interceptar las peticiones antes de que lleguen al servidor, editarlas y enviarlas al servidor de forma definitiva.

Ejercicio 20. Activar la interceptación de los mensaje de respuesta del servidor en *Proxy >> Options >> Intercept Server Responses* y editar el contenido de la página web antes de que se muestre en el navegador.

Ejercicio 21. Modificar la cabera de alguna respuesta y añadir una nueva *Cookie* personalizada. Comprobad que el navegador tiene ahora almacenada dicha cookie.

Ejercicio 22. Intentar utilizar algunas de las reglas de remplazamiento automático que se encuentran en *Proxy >> Options >> Match and Replace*.

Para más información sobre el uso de *Burp Suite* se puede visitar el *Burp Suite Support Center* en https://support.portswigger.net/

4 Entrega de resultados

Los ejercicios marcados como **entregables** podrán ser parte del proceso de evaluación presencial de la asignatura en el que se valorará su contenido, corrección y autenticidad.

Entregable.
Subir los ficheros indicados en esta prácticas a *PoliformaT >> Espacio Compartido >> Nombre Alumno >> Práctica 1*. Si la carpeta *Práctica 1* no existe la deberá crear el alumno. En caso de haber más de un alumno autor de los entregables, se deberá indicar en un fichero de texto **`autores.txt`** el nombre completo de los participantes.
Es importante respetar los nombres de los ficheros solicitados para facilitar el proceso de evaluación de la práctica. Para entregar los ficheros solicitados en un único archivo se puede utilizar la orden **`tar`** o la orden **`zip`**.