



ACTIVIDAD PRÁCTICA Y DE DESARROLLO
APD-1: Comunicación web. Protocolos y depuración

Objetivos

- Refrescar conceptos de protocolos y estándares relacionados con las aplicaciones Web (HTTP, HTTPS, CGI, MIME, Cookies)
- Conocer y usar herramientas para el trazado y la depuración de dichos protocolos.

Introducción

Protocolo HTTP

El *hypertext transfer protocol* [http-1][http-2][http-3] es un protocolo de transferencia de contenidos web. Se basa en un modelo cliente/servidor. El cliente establece una conexión TCP con el servidor (normalmente al puerto TCP/80) y sobre ella se transfieren las peticiones del primero y las respuestas del segundo.

Peticiones HTTP: están formadas por una línea de petición, varias líneas de cabeceras, una línea en blanco y, opcionalmente, el cuerpo del mensaje.

La línea de petición tiene el formato

<acción> <espacio> <ruta> <espacio> <versión>

Las dos acciones más comunes son GET (obtener contenido almacenado en la ruta indicada) y POST (enviar contenido al sitio indicado por la ruta, para que sea procesado). La versión del protocolo a emplear suele ser HTTP/1.0 o HTTP/1.1

Las cabeceras (con formato <nombre>: <valor>) dan indicaciones adicionales sobre la petición. Las cabeceras más importantes en peticiones HTTP son: Host, User-Agent, Referrer, Accept-*...

Tras las cabeceras debe enviarse obligatoriamente una línea en blanco.

Tras esa línea de fin de cabeceras, en determinados casos (principalmente, con la acción POST) se puede incluir un cuerpo del mensaje.

Respuestas HTTP: formadas por una línea de estado, varias cabeceras, una línea en blanco y el cuerpo de la respuesta.

La línea de estado tiene la forma

<versión> <espacio> <código de estado> <espacio> <texto explicativo>

La versión es como en la línea de petición (HTTP/1.x). El código de estado indica el resultado de la petición. Los códigos más importantes [codigos-estado] son:

- 200: todo correcto
- 302: redirección a otra URL
- 401: se requiere autenticación
- 404: recurso no encontrado
- 500: error interno del servidor



Parámetros HTTP

Una petición HTTP, en caso de que referencie a un objeto dinámico (un script PHP o ASP, un servlet Java, una página JSP...) puede contener parámetros. Éstos están representados en forma de parejas <nombre>=<valor> separadas por '&'. Pueden enviarse de dos formas:

- En el segmento de contenido de una petición POST (es decir, tras la línea en blanco que va tras las cabeceras.

```
POST /procesa.php HTTP/1.1
Host: www.pagoelectronico.com
User-Agent: Mozilla/4.0
Content-Length: 64
Content-Type: application/x-www-form-urlencoded
<línea en blanco>
usuario=cliente&clave=312eoq&tarjeta=192838281823&importe=192.20
```

- En la propia URL de una petición GET, en este caso precedidos por '?'

```
GET /procesa.php?usuario=cliente&clave=312eoq
&tarjeta=192838281823&importe=192.20
Host: www.pagoelectronico.com
User-Agent: Mozilla/4.0
<línea en blanco>
```

Cookies

HTTP es un protocolo sin estado: el servidor no tiene constancia ("memoria") de las peticiones anteriores, ni posibilidad de relacionarlas con nuevas peticiones del mismo cliente.

Para mantener información persistente relacionada con un visitante o usuario, se puede almacenar en el cliente HTTP de dicho visitante un pequeño fragmento de información llamado cookie [cookies] que consta, entre otros, de un nombre, un valor y una caducidad.

El servidor solicita al cliente que almacene esta cookie y, en caso de que se acepte, el cliente se compromete a reenviar la cookie al servidor en cada petición subsiguiente, hasta que esta caduque.

Bibliografía

[http-1] <http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node13.html>

[http-2] <http://es.kioskea.net/contents/internet/http.php3>

[http-3] <http://www.jmarshall.com/easy/http>

[codigos-estado] <http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>

[cookies] <http://es.wikipedia.org/wiki/Cookie>

[live] <http://livehttpheaders.mozdev.org>

[webdev] <http://chrispederick.com/work/web-developer>

[aec] <http://addneditcookies.mozdev.org>

[wget] <http://es.wikipedia.org/wiki/Wget>

[wget-win] <http://www.christopherlewis.com/WGet/WGetFiles.htm>



Experimentos

Ex1. (5 min) Instalar en el navegador Firefox la extensión "Live HTTP headers" [live]. Esta extensión añade una opción al menú de herramientas que permite ver "en vivo" las peticiones y respuestas HTTP que gestiona Firefox. Practicar con la extensión accediendo a varias páginas y viendo los resultados.

Ex2. (10 min) Usaremos otra extensión de Firefox: Web Developer [webdev]. Añade un menú contextual (botón derecho) con multitud de opciones para recabar información sobre la página actual. Instalarla, acceder al área de autenticación del portal de la EPS

<https://www.upo.es/eps/portal/autentificaUsuarios.php>

y probar las opciones

- "Web Developer -> Formularios -> Ver información de los formularios"
- "Web Developer -> Formularios -> Mostrar detalles del formulario"
- "Web Developer -> Cookies -> Ver información de las cookies"

Ex3. (5 min) Instalar el programa Wget [wget], ya sea en Linux o en Windows [wget-win] y practicar obteniendo la URL de la EPS vista en el experimento 1.

Ejercicios

Lo que realices para resolver estos ejercicios debes escribirlo en un documento que convertirás a PDF y entregarás en la tarea WebCT correspondiente a esta práctica. Se espera algo más que un simple listado de acciones realizadas: comenta, ilustra con capturas de pantalla de lo que se te pide, destaca aspectos interesantes, saca conclusiones (aunque no sean correctas) y, en general, dale un poco de vida al "cuaderno de prácticas".

Ej1. (10 min) Acceder a cinco sitios web diferentes. Hacer una tabla indicando las cabeceras que envían los servidores en su respuesta. Incluir, además de las cabeceras más conocidas, alguna que llame la atención, siempre y cuando se haya podido encontrar una explicación de su significado.

Ej2. (5 min) Investigar sobre las cabeceras content-length y content-type. Encontrar los tipos de contenido más comunes que se sirven.

Ej3. (10 min) Describir las diferencias entre el paso de parámetros GET y POST. Encontrar dos páginas de ejemplo de cada tipo y capturar el tráfico al enviar los parámetros.

Ej4. (10 min) Acceder a la página de la Sociedad de Estudios Medievales y Renacentistas (<http://www.la-semyr.es>) y, tras activar la ventana de Live HTTP Headers, pulsar en el enlace "Pague su cuota (TPV)". Revisar el tráfico y las redirecciones HTTP (a pgw.teca.es). Analizar los parámetros que se le pasan a pgw.teca.es. Analizar el formulario presentado por esta última página.

Ej5. (5 min) Acceder a <http://www.pittarese.com/cookie demo.php> y usar el formulario "cookie demonstration". Monitorizar las cabeceras con Live HTTP Headers y detectar la cabecera que fija la cookie. Observar con Web Developer el valor de la misma. Recargar la página y detectar la cabecera del cliente que envía la cookie. Borrar la cookie ("To delete your cookie...") y observar qué cabecera se envía para borrarla.



- Ej6. (5 min) Acceder al buzón de sugerencias de la Escuela Politécnica (<http://www.upo.es/eps/portal/enviarSugerencias.php>) y describir los componentes del formulario. ¿Por qué método HTTP se envía el formulario?
- Ej7. (10 min) Acceder a las tres páginas anteriores mediante wget. Indicar las cabeceras HTTP que fijan las cookies de la pregunta anterior. Recargar varias veces y observar si los valores cambian. Encontrar las opciones de esta herramienta relacionadas con las cookies y usarlas para grabar en disco las cookies obtenidas.

Problemas

- Pr1. Probar la extensión "Add & Edit cookies" con la URL de la EPS. ¿Qué cookies se almacenan? ¿Es posible modificarlas?
- Pr2. Sobre el paso de parámetros GET y POST, y desde el punto de vista de la seguridad, ¿hay alguna ventaja si se usa uno de ellos en lugar del otro?
- Pr3. Acceder a tres páginas que sirvan cookies. Analizar el contenido de las mismas y, en su caso, tratar de interpretar su valor.
- Pr4. Descargar con wget varias veces la página del Aula Virtual de la UPO (WebCT). ¿Qué cabeceras se obtienen? ¿Hay variación? ¿Qué cookies se generan?
- Pr5. Investigar el uso de Wget para enviar parámetros POST. Encontrar una página que los admita y probar el envío.



Datos de la práctica

Estimación temporal:

Parte presencial: 90 minutos.

Explicación inicial: 15 minutos.

Experimentos: 20 minutos.

Ejercicios: 55 minutos.

Parte no presencial: 270 minutos.

Lectura y estudio del guión y bibliografía básica: 150 minutos

Problemas: 120 minutos

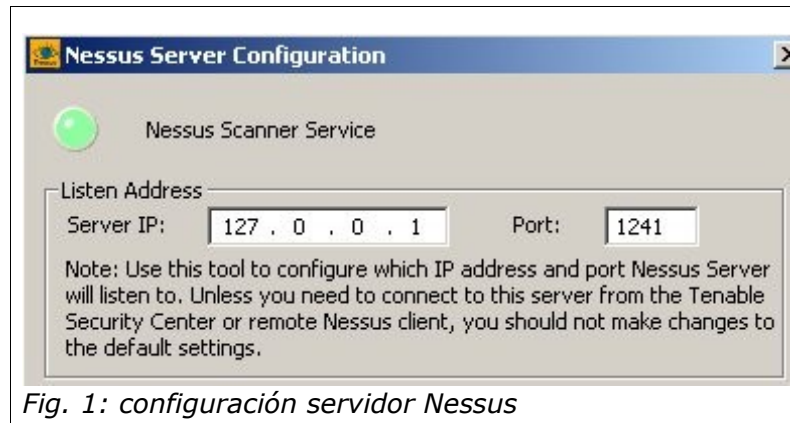


Fig. 1: configuración servidor Nessus

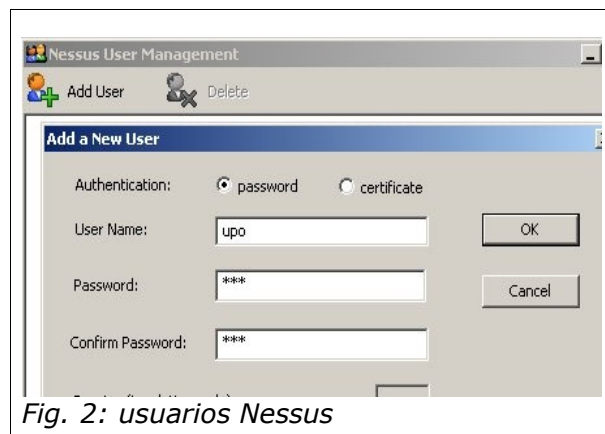


Fig. 2: usuarios Nessus