Práctica 2

Capa de Aplicación - HTTP

Requerimientos

1. Para realizar esta práctica deberá descargar la máquina virtual provista. Puede ver la URL de descarga en el sitio de la cátedra en https://catedras.info.unlp.edu.ar/.

   Una vez descargado el archivo, haciendo doble click en el mismo debería abrirse un cuadro de diálogo que permita configurar algunos parámetros del sistema. Se pueden aceptar los valores por defecto haciendo simplemente click en Importar.

   Se recomienda hacer un snapshot de la VM antes de empezar a usarla, para poder volver atrás en caso de que algo deje de funcionar.

   Los datos de acceso a la máquina virtual son:

   - Usuario: alumno
   - Contraseña: redes
   - Para acceso con permisos de administrador, usar el comando sudo

   Aclaracion: El dominio redes.unlp.edu.ar solo existe dentro de la VM provista por la cátedra, no es válido en Internet, lo que implica que todos los ejercicios de esta práctica y las siguientes en las que se utilice dicho domino solo podrán ser resueltos dentro dicha VM.

Introducción

2. ¿Cuál es la función de la capa de aplicación?
3. Si dos procesos deben comunicarse:
   a. ¿Cómo podrían hacerlo si están en diferentes máquinas?
   b. Y si están en la misma máquina, ¿qué alternativas existen?
4. Explique brevemente cómo es el modelo Cliente/Servidor. De un ejemplo de un sistema Cliente/Servidor en la “vida cotidiana” y un ejemplo de un sistema informático que siga el modelo Cliente/Servidor. ¿Conoce algún otro modelo de comunicación?
5. Describa la funcionalidad de la entidad genérica “Agente de usuario” o “User agent”.

HTTP

6. Observe el indice de la RFC2616, busque el apartado donde se describe el requerimiento y la respuesta. ¿Qué son y en qué se diferencian HTML y HTTP? ¿En que entidad ubicaría a HTML?

7. Utilizando la VM, abra una terminal e investigue sobre el comando curl. Analice para qué sirven los siguientes parámetros (-I, -H, -X, -s).

8. Ejecute el comando curl sin ningún parámetro adicional y acceda a www.redes.unlp.edu.ar. Luego responda:

   a. ¿Cuántos requerimientos realizó y qué recibió? Pruebe redirigiendo la salida($>$) del comando curl a un archivo con extensión html y abrirlo con un navegador.
   b. ¿Cómo funcionan los atributos **href** de los tags **link** e **img** en html?
   c. Para visualizar la página completa con imágenes como en un navegador, ¿alcanza con realizar un único requerimiento? ¿Cuántos requerimientos serían necesarios para obtener una página que tiene dos CSS, dos Javascript y tres imágenes? Diferencie como funcionaría un navegador respecto al comando curl ejecutado previamente.

9. Ejecute a continuación los siguientes comandos:

```
curl -v -s www.redes.unlp.edu.ar > /dev/null
curl -I -v -s www.redes.unlp.edu.ar
```

   - Observe la salida y luego repita la prueba, pero previamente inicie una nueva captura en wireshark. Utilice la opción Follow Stream. ¿Qué se transmitió en cada caso?
   - ¿A que se debió esta diferencia entre lo que se transmitió y lo que se mostró en pantalla?

10. Investigue cómo define las cabeceras la RFC.

    a. ¿Establece todas las cabeceras posibles?.
    b. ¿Cuántas cabeceras viajaron en el requerimiento y en la respuesta del ejercicio anterior?
    c. ¿La cabecera Date es una de las definidas en la RFC? ¿Qué indica?

11. Utilizando curl, realice un requerimiento con el método HEAD al sitio www.redes.unlp.edu.ar e indique:

    a. ¿Qué información brinda la primer línea de la respuesta?
    b. ¿Cuántos encabezados muestra la respuesta?
    c. ¿Qué servidor web está sirviendo la página?
    d. ¿El acceso a la página solicitada fue exitoso o no?
    e. ¿Cuándo fue la última vez que se modificó la página?

f. Solicite la página nuevamente con curl usando GET, pero esta vez indique que quiere obtenerla sólo si la misma fue modificada en una fecha posterior a la que efectivamente fue modificada. ¿Cómo lo hace? ¿Qué resultado obtuvo? ¿Puede explicar para qué sirve?

12. En HTTP/1.0, ¿cómo sabe el cliente que ya recibió todo el objeto solicitado completamente? ¿Y en HTTP/1.1?

13. Investigue los distintos tipos de códigos de retorno de un servidor web y su significado en la RFC. ¿Qué parte se ve principalmente interesada de esta información, cliente o servidor? ¿Es útil que esté detallado y clasificado en la RFC?. Dentro de la VM, ejecute los siguientes comandos y evalue el estado que recibe.

    - curl -I http://unlp.edu.ar
    - curl -I www.redes.unlp.edu.ar/restringido/index.php
    - curl -I www.redes.unlp.edu.ar/noexiste

14. Utilizando curl, acceda al sitio www.redes.unlp.edu.ar/restringido/index.php y siga las instrucciones y las pistas que vaya recibiendo hasta obtener la respuesta final. Será de utilidad para resolver este ejercicio poder analizar tanto el contenido de cada página como los encabezados.

15. Utilizando la VM, realice las siguientes pruebas:

    a. Ejecute el comando 'curl www.redes.unlp.edu.ar/extras/prueba-http-1-0.txt' y copie la salida completa (incluyendo los dos saltos de linea del final).
    b. Desde la consola ejecute el comando telnet www.redes.unlp.edu.ar 80 y luego pegue el contenido que tiene almacenado en el portapapeles. ¿Qué ocurre luego de hacerlo?
    c. Repita el proceso anterior, pero copiando la salida del recurso /extras/prueba-http-1-1.txt. Verifique que debería poder pegar varias veces el mismo contenido sin tener que ejecutar telnet nuevamente.

16. En base a lo obtenido en el ejercicio anterior, responda:

    - ¿Qué está haciendo al ejecutar el comando telnet? ¿Qué lo diferencia con curl?
    - Observe la definición de método y recurso en la RFC. Luego responda, ¿Qué método HTTP utilizó? ¿Qué recurso solicitó?
    - ¿Qué diferencias notó entre los dos casos? ¿Puede explicar por qué?
    - ¿Cuál de los dos casos le parece más eficiente? Piense en el ejercicio donde analizó la cantidad de requerimientos necesarios para obtener una página con estilos, javascripts e imágenes. El caso elegido, ¿puede traer asociado algún problema?

17. En el siguiente ejercicio veremos la diferencia entre los métodos POST y GET. Para ello, será necesario utilizar la VM y la herramienta **Wireshark**. Antes de iniciar considere:

- Capture los paquetes utilizando la interfaz con IP 172.28.0.1. (Menú "**Capture** ->**Options**". Luego seleccione la interfaz correspondiente y presione **Start**).
- Para que el analizador de red sólo nos muestre los mensajes del protocolo http introduciremos la cadena 'http' (sin las comillas) en la ventana de especificación de filtros de visualización (display-filter). Si no hiciéramos esto veríamos todo el tráfico que es capaz de capturar nuestra placa de red. De los paquetes que son capturados, aquel que esté seleccionado será mostrado en forma detallada en la sección que está justo debajo. Como sólo estamos interesados en http ocultaremos toda la información que no es relevante para esta práctica (Información de trama, Ethernet, IP y TCP). Desplegar la información correspondiente al protocolo HTTP bajo la leyenda "Hypertext Transfer Protocol".
- Para borrar la cache del navegador, deberá ir al menu "Herramientas->Borrar historial reciente". Alternativamente puede utilizar Ctrl+F5 en el navegador para forzar la petición HTTP evitando el uso de caché del navegador.
- En caso de querer ver de forma simplificada el contenido de una comunicación http, utilice el botón derecho sobre un paquete HTTP perteneciente al flujo capturado y seleccione la opción **Follow TCP Stream**.

a. Abra un navegador e ingrese a la URL: www.redes.unlp.edu.ar e ingrese al link en la sección "Capa de Aplicación" llamado "Métodos HTTP". En la página mostrada se visualizan dos nuevos links llamados: Método GET y Método POST. Ambos muestran un formulario como el siguiente:

b. Analice el código HTML.
c. Utilizando el analizador de paquetes Wireshark capture los paquetes enviados y recibidos al presionar el botón Enviar.
d. ¿Qué diferencias detectó en los mensajes enviados por el cliente?
e. ¿Observó alguna diferencia en el browser si se utiliza un mensaje u otro?

18. HTTP es un protocolo stateless, para sortear esta carencia muchos servicios se apoyan en el uso de Cookies. ¿En qué RFC se definió dicha funcionalidad? Investigue cuál es el principal uso que se le da a Set-Cookie y Cookie en HTTP. ¿Qué atributo de la RFC original fue en parte aprovechado para la implementación?

19. ¿Cuál es la diferencia entre un protocolo binario y uno basado en texto? ¿de que tipo de protocolo se trata HTTP/1.0, HTTP/1.1 y HTTP/2?

20. Analice de que se tratan las siguientes características de HTTP/2: stream, frame, server-push

21. Responder las siguientes preguntas:

    a. ¿Qué función cumple la cabecera Host en HTTP 1.1? ¿Existía en HTTP 1.0? ¿Qué sucede en HTTP/2? (Ayuda: https://undertow.io/blog/2015/04/27/An-in-depth-overview-of-HTTP2.html para HTTP/2)

    b. ¿Cómo quedaría en HTTP/2 el siguiente pedido realizado en HTTP/1.1 si se está usando https?

```
GET /index.php HTTP/1.1
Host: www.info.unlp.edu.ar
```