Redes y Comunicaciones 2017

Práctica 2

Autor: Fermín Minetto (http://www.github.com/ferminmine)

2) ¿Cuál es la función de la capa de aplicación?

La capa de aplicación es la capa que proporciona la interfaz entre las aplicaciones que utilizaremos para comunicarnos y la red subyacente en la cual se transmiten los mensajes. Los protocolos de la capa de aplicación se utilizan para intercambiar los datos entre los programas que se ejecutan en los hosts de origen y destino.

3) ¿Cómo podrían comunicarse dos procesos si están en diferentes máquinas?

Cuando un programador desarrolla una aplicación de red, este requerirá que haya una comunicación entre los sistemas terminales que ejecutan el programa, ya sea para compartir datos o archivos que servirán para cumplir el propósito de la aplicación. Para esto, un proceso envía mensajes a la red y los recibe de ellas a través de una interfaz de software llamada socket. Cuando un proceso desea enviar un mensaje a otro proceso que se está ejecutando en otro host, envía el mensaje a través de la puerta.

Este proceso emisor supone que existe una infraestructura de transporte al otro lado de la puerta que llevará el mensaje hasta la puerta del proceso destino. Una vez que el mensaje llega al host de destino, este pasa a través de la puerta del proceso receptor, actuando entonces el proceso receptor sobre el mensaje.

Especificar el orden y contenidos de los mensajes enviados dependerá del protocolo utilizado por las aplicaciones.

4) ¿Cómo es el modelo cliente-servidor? ¿Cuál otro existe?

En una arquitectura cliente-servidor siempre existe un host activo, denominado servidor, que da servicio a las solicitudes de muchos otros hosts, que son los clientes. Un ejemplo clásico es la web, en la que un servidor siempre activo sirve las solicitudes de los navegadores que se ejecutan en los hosts clientes.

Otro paradigma arquitectónico predominante en Internet es el P2P. En una arquitectura P2P existe una mínima (o ninguna) dependencia de una infraestructura de servidores siempre activos. Los pares no son propiedad del proveedor del servicio. Una de las características más convincentes de las arquitecturas P2P es su auto-escalabilidad. Cada peer genera una carga de trabajo solicitando archivos, también añade capacidad de servicio al sistema (por ejemplo distribuyendo archivos a otros pares). Por lo general, no requieren una infraestructura de servidores.
Ejemplos: Utorrent, Skype.

5) ¿Cuál es la funcionalidad e la entidad genérica “agente de usuario”?

Un agente de usuario es una aplicación informática que funciona como cliente en un protocolo de red; el nombre se aplica generalmente para referirse a aquellas aplicaciones que acceden a la WWW.

6) ¿Qué son y en qué se diferencian HTML y HTTP?

HTML es un lenguaje de maquetado que utiliza etiquetas. Suele utilizarse para la vista en aplicaciones web estáticas o dinámicas junto con CSS. En cambio, HTTP es un protocolo de la capa de aplicación de Internet, que utiliza la web para recuperar documentos. Este protocolo define el formato y la secuencia de los mensajes que pasan entre el navegador web y el servidor web.

7, 8 y 9) CURL

CURL es una librería de funciones para conectar con servidores y para trabajar con ellos. El trabajo se realiza con formato URL. Es decir, sirve para realizar acciones sobre archivos que hay en URLs de Internet, soportando los protocolos más comunes, como HTTP, HTTPS, FTP, etc.

Parámetros:

-I : Recupera solo los headers de la respuesta.

-H : Pasa un header customizado al servidor.

-s : Modo silencioso. No muestra progreso o mensajes de error.

-v : Verborrágico. Útil para debuggear. Brinda más información cuando puede.

-X : Especifica un método HTTP para utilizar. Ej: CURL -X HEAD (ver man).

-F : Especifica información enviada en el cuerpo del mensaje. Útil para datos de un post.

Cuando se ejecuta CURL sobre una “página” (URL) se hace un requerimiento que trae el archivo HTML base mediante un mensaje GET HTTP. Etiquetas como IMG que muestran un recurso no van a poder encontrarlo porque el CURL solo trae el archivo base. En otros sitios ni siquiera va a encontrar el archivo CSS. Para traer una “página web” completa va a haber que hacer un requerimiento de cada uno de esosr ecursos para traerlos. Un cliente web como Firefox haría esos requerimientos de manera automática y formaría la estructura de la página para poder indexar los recursos al archivo base HTML de forma automática. Entonces para una página que tiene 2 CSS, 3 JS y 3 imágenes serían necesarios 8 requerimientos (se suma el archivo HTML base).

En el primer comando se necesita la redirección a /dev/null para que no aparezca el HTML del cuerpo de la respuesta en la terminal, así se pueden ver los encabezados de respuesta que aparecen por el -v (curl -v -s www.redes.unlp.edu.ar >> dev/null).
En el segundo comando no es necesaria la redirección porque el parámetro -I solamente va a mostrar los encabezados de la respuesta omitiendo el cuerpo.

10) Ejecutando CURL -I www.redes.unlp.edu.ar y observando los encabezados…

Es posible determinar que servidor web respondió la solicitud. En caso de redes.unlp.edu.ar fue APACHE. *En el encabezado de respuesta server se puede ver esta información*.
El código de respuesta *fue el número 200*, que significa que no hubo inconvenientes y se entrega el documento con éxito. Otros códigos de resultado HTTP:

-301 Moved Permanently: El objeto solicitado ha sido movido de forma permanente; el software cliente recuperará automáticamente el nuevo URL.

-**400 Bad Request**: Se trata de un código de error genérico que indica que la solicitud no ha sido comprendida por el servidor.

-**404 Not Found:** El documento solicitado no existe en el servidor.

-**505 HTTP Version Not Supported**: La versión del protocolo HTTP solicitada no es soportada por el servidor.

Se puede ver cuándo fue la última vez que el documento se modificó en el encabezado “Last-Modified: …”.
Para que nos envíe una página solo si fue modificada hay que enviar un header en la solicitud llamado “If-Modified-Since: X“ y enviando una fecha con el mismo formato que los encabezados DATE en cualquier otra solicitud. Comando:
CURL -I -H “If-Modified-Since: Wed, 16 Mar 2016 20:43:33 GMT” www.redes.unlp.edu.ar
En caso de existir una versión más reciente del archivo va a llegar con un mensaje de respuesta OK(200). si no existe una versión más nueva del archivo entonces llegará una respuesta sin contenido y con un encabezado con código 304 Not Modified.

El ETAG es un campo en el encabezado de solicitud y respuesta de un mensaje HTTP que es opcional. Básicamente sirve para indicarle al servidor web que versión de un documento tenemos almacenada localmente, para que éste determine si ha cambiado y nos mande nuevamente la página.

La primera vez que recibimos un recurso, nos llega el encabezado ETAG: xxyy. Luego nuestro agente de usuario manda el encabezado "If-None-Match: xxyy", y en caso de que exista un documento con ese ETAG nos mandará una respuesta 304 Not Modified.

curl -I -H 'If-None-Match: "138-a..." ' www.redes.unlp.edu.ar

12) Conexiones persistentes / no persistentes

Con el comando telnet se está estableciendo una conexión TCP con el host ficticio www.redes.unlp.edu.ar y se le manda una solicitud HTTP para recuperar un documento.
Con HTTP1.0 la conexión se realiza de manera no persistente, es decir que el servidor cierra la conexión inmediatamente después de entregar el documento.
Con HTTP1.1 la conexión queda abierta y no es cerrada por el servidor. Una conexión persistente solo es cerrada cuando transcurre un tiempo determinado sin utilizarse.
Para solicitar una página completa con HTTP1.0 habría que abrir una conexión por cada objeto de la misma. Con HTTP1.1 se pueden recuperar los objetos empleando la misma conexión TCP.

13) Recuperar páginas en otro idioma

Para poder recuperar páginas en inglés hay que enviar el encabezado de solicitud "Accept-Language: en" para poder recibir la página en el idioma inglés, o cualquier otro idioma deseado. Si mandamos una solicitud con un idioma no soportado entonces se nos enviará la versión del documento por defecto, en el caso del ejercicio en español.

14) Formularios GET y POST

El método HTTP utilizado para enviar los datos varía según la página. Esto se especifica como parámetro method dentro de la etiqueta HTML form.
La diferencia entre estos métodos es que si se utiliza el POST los datos que el usuario ingresa en el formulario viajarán en el cuerpo de la entidad. Si no se utiliza el post y se utiliza el GET entonces los datos viajarán en la URL solicitada, lo cual es riesgoso en caso que se envíe información privada.

curl -X POST -F "materia: redes" -F "tema:curl" http://localhost