Redes y Comunicaciones

Ulises J. Cornejo Fandos

Marzo 2017

Practica 2

1.1 Introducción

2. **¿Cuál es la función de la capa de aplicación?**

   La capa de aplicación del modelo TCP/IP maneja protocolos de alto nivel, aspectos de representación, codificación y control de diálogo. El modelo TCP/IP combina todos los aspectos relacionados con las aplicaciones en una sola capa y asegura que estos datos estén correctamente empaquetados antes de que pasen a la capa siguiente. TCP/IP incluye no sólo las especificaciones de Internet y de la capa de transporte, tales como IP y TCP, sino también las especificaciones para aplicaciones comunes. TCP/IP tiene protocolos que soportan la transferencia de archivos, e-mail, y conexión remota, además de los siguientes:

   - **FTP(Protocolo de transferencia de archivos):**
     Es un servicio confiable orientado a conexión que utiliza TCP para transferir archivos entre sistemas que admiten la transferencia FTP.
   - **TFTP (Protocolo trivial de transferencia de archivos:**
     Es un servicio no orientado a conexión que utiliza el Protocolo de datagramaos usuario(UDP).
   - **NFS (Sistema de archivos de red):**
     Es un conjunto de protocolos para un sistema de archivos distribuidos, desarrollado por Sun Microsystems.
   - **SMTP (Protocolo simple de transferencia de correo):**
     Administra la transmisión de correo electrónico a través de las redes informáticas.
   - **SNMP (Protocolo simple de administración de red):**
     Es un protocolo que provee una manera de monitorear y controlar los dispositivos de red.
   - **DNS (Sistema de denominación de dominio):**
     Es un sistema que se utiliza en Internet para convertir los nombres de los dominios y de sus nodos de red publicados abiertamente en direcciones IP.

3. Si dos procesos deben comunicarse: **¿Cómo podrían hacerlo si están en diferentes máquinas?**

   Para comunicarse con un proceso que se encuentra en otra máquina es necesario conocer la direccion IP de la otra máquina, y el puerto en el que se encuentra "escuchando" el proceso.

4. **Explique brevemente cómo es el modelo Cliente/Servidor.**

   El modelo Cliente/Servidor consta de tener una computadora donde se va a realizar la mayor parte de procesamiento, la cual seria el servidor, y una computadora la cual accede al servicio provisto por el servidor, y solo se va a encargar (normalmente) de realizar la parte de visualización de datos. Un ejemplo de sistema Cliente/Servidor en la vida cotidiana puede ser un cliente de mail como es el caso de gmail. Existe otros tipos de modelos, como es el caso del peer-to-peer, modelos híbridos, etc.

5. **Describa la funcionalidad de la entidad genérica "Agente de usuario" o "User agent".**

   Su función, es ser una interfaz entre el usuario y la aplicación de red. Implementa los protocolos necesarios para que funcione la capa de aplicación. Un agente de usuario, por ejemplo un navegador, es un proceso que envía/recibe mensajes por medio de un socket.

1.2 HTTP

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

   HTML es un lenguaje de marcado, en cambio, HTTP es un protocolo definido en la capa de aplicación que utiliza el lenguaje HTML para realizar las respuestas.

7. **Utilizando la VM, abra una terminal. Investigue sobre el comando curl y analice para qué sirven los siguientes parámetros (-I, -H, -X, -s).**

   curl is a tool to transfer data from or to a server, using one of the supported protocols (DICT, FILE, FTP, FTPS, GOPHER, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMB, SMBS, SMTP, SMTPS, TELNET and TFTP). The command is designed to work without user interaction.

   Parámetros:

   - -I: es utilizado para requerir solamente el head de la consulta HTTP.
   - -H: es utilizado para agregar un extra header a la consulta HTTP, con el fin de mejorar la eficiencia de la consulta.
   - -X: siver para especicar el metodo que se utilizapara para la comunicacion HTTP.
   - -s: se utiliza para activar el silent mode. En este modo, todo el procedimiento realizado por el comando y la salida estandar de error es silcenciado.

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.

   Realizó un solo requerimiento GET, y recibió una respuesta HTML.

   (b) ¿Cómo funcionan los atributos href de los tags link e img en html?

   Funcionan redirigiendo a una nueva página, que puede ser local al servidor (la cual va a requerir un nuevo GET a la aplicación), o externos (requiriendo utilizar el metodo GET, pero esta vez con un servidor distinto al anterior.

   (c) Para visualizar la página completa con imágenes como en un navegador, ¿alcanza con realizar un único requerimiento? ¿Cuántos requerimientos serın 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.

   No alcanza con realizar un único requerimiento, para visualizar dicha página serian necesarios 8 requerimientos, uno para cada archivo (la página inicial, los dos CSS, los dos JS y las tres imágenes).

9. **Ejecute los siguientes comandos:**

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

   (a) ¿Qué diferencia nota entre cada uno?

   La diferencia es que en el primer comando todo el cuerpo de la respuesta es redireccionado a /dev/null, en cambio en el segundo solo se omite el cuerpo de la respuesta.

(b) ¿Qué ocurre si en el primer comando quita la redireción a /dev/null? ¿Por qué no es necesario en el segundo comando?

Si se quita la redirección a /dev/null, se muestra la consulta HTTP completa (con el código HTML recibido), junto con las especificaciones de la respuesta (proveniente del -v).

En cambio, en el segundo comando no es necesaria la redirección, ya que el comando -I solo visualiza la cabecera de la request (proveniente de -v) y el header de la respuesta (proveniente del comando -I).

(c) ¿Cuántas cabeceras viajaron en el requerimiento? ¿Y en la respuesta?
Solo viaja un header en el requerimiento y un header en la respuesta.

10. **Ejecute el comando curl www.redes.unlp.edu.ar y responda:**

(a) ¿Es posible determinar que servidor web se utiliza para servir la página?
Es posible mediante el campo "server" del header.

(b) ¿Cuál es el código de respuesta que devolvió el servidor? ¿Qué otros códigos existen y que significan? Investigue genericamente los tipos de error 2XX, 3XX, 4XX y 5XX.
Devolvió el código 200 que significa que la consulta se resolvió correctamente.

Existen codigos para especificar que no se encontró el archivo buscado, que se movió el directorio temporalmente, que se movió permanentemente, ...

- 1XX - Respuestas Informativas
  - 100: Continue
    El navegador puede continuar realizando su petición (Se utiliza para indicar que la primera parte de la petición del navegador ha recibido correctamente).
  - 101: Switching Protocols
    El servidor acepta el cambio de protocolo propuesto por el navegador (Puede ser por ejemplo un cambio de HTTP 1.0 a HTTP 1.1).
  - 102: Processing (WeDAV - RFC 2518)
    El servidor está procesando la peticion del navegador pero todavía no ha terminado (esto evita que el navegador piense que la petición se ha perdido cuando no recibe ninguna respuesta).
  - 103: Checkpoint
    Se va a reanudar una peticón POST o PUT que fue abortada previamente.
- 2XX - Peticiones Correctas
  - 200: OK
    Respuesta estandar para peticiones correctas.
  - 201: Created
    La petición ha sido completada y ha resultado en la creación de un nuevo recurso.
  - 202: Accepted
    La peticion ha sido aceptada para procesamiento, pero este no ha sido completado. La peticion eventualmente pudiere no se satisfecha, ya que podria ser no permitida o prohibida cuando el procesamiento tenga lugar.
  - 203: Non-Authoritative Information (desde HTTP/1.1)
    La petición se ha completado con exito pero su contenido no se ha obtenido de la fuente originalmente solicitada sino de otro servidor.
  - 204: No Content
    La petición se ha completado con éxito pero su respuesta no tiene ningun contenido (la respuesta sí que puede incluir informacion en sus cabeceras HTTP).
  - 205: Reset Content
    La petición se ha completado con éxito, pero su respuesta no tiene contenidos y además, el navegador tiene que inicializar la página desde la que se realizó la petición (este código es util por ejemplo para páginas con formularios cuyo contenido debe borrarse despues de que el user la envíe).

  - 206: Partial Content
    La petición servirá parcialmente el contenido solicitado.Esta característica es utilizada por herramientas de descarga como wget para continuar la transferencia de descargas anteriormente interrumpidas, o para dividir una descarga y procesar las partes simultaneamente.
  - 207: Multi-Status (Multi-Status, WebDAV)
    El cuerpo del mensaje que sigue es un mensaje XML y puede contener algun numero de códigos de respuesta separados, dependiendo de cuantas subpeticiones sean hechas.
  - 208: Already Reported (WebDAV)
    El listado de elementos DAV ya se notificó previamente, por lo que no se van a volver a listar.
- 3XX - Redirecciones
  - 300: Multiple Choices
    Indica opciones multiples para el URL que el cliente podria seguir. Esto podria ser utilizado, por ejemplo, para presentar distintas opciones de formato para video, listar archivos con distintas extensiones o word sense disambiguation.
  - 301: Moved Permanently
    Esta y todas las peticiones futuras deberían ser difigidas a la URL dada.
  - 302: Found
    Este es el código de redirección más popular, pero también un ejemplo de las prácticas de la industria contradiciendo el estándar. La especificación HTTP/1.0 (RFC 1945) requería que el cliente realizara una redirección temporal (la frase descriptiva original fue "Moved Temporarily"), pero los navegadores populares lo implementaron como 303 See Other. Por tanto, HTTP/1.1 añadió códigos de estado 303 y 307 para eliminar la ambigüedad entre ambos comportamientos. Sin embargo, la mayoría de aplicaciones web y bibliotecas de desarrollo aún utilizan el código de respuesta 302 como si fuera el 303.
  - 303: See Other (desde HTTP/1.1)
    La respuesta a la petición puede ser encontrada bajo otra URI utilizando el método GET.
  - 304: Not Modified
    Indica que la petición a la URL no ha sido modificada desde que fue requerida por última vez. Típicamente, el cliente HTTP provee un encabezado como If-Modified-Since para indicar una fecha y hora contra la cual el servidor pueda comparar. El uso de este encabezado ahorra ancho de banda y reprocesamiento tanto del servidor como del cliente.
  - 305: Use Proxy (desde HTTP/1.1)
    Muchos clientes HTTP (como Mozilla3 e Internet Explorer) no se apegan al estándar al procesar respuestas con este código, principalmente por motivos de seguridad.
  - 306: Switch Proxy
    Este código se utilizaba en las versiones antiguas de HTTP pero ya no se usa (aunque está reservado para usos futuros).
  - 307: Temporary Redirect (desde HTTP/1.1)
    Se trata de una redirección que debería haber sido hecha con otra URI, sin embargo aún puede ser procesada con la URI proporcionada. En contraste con el código 303, el método de la petición no debería ser cambiado cuando el cliente repita la solicitud. Por ejemplo, una solicitud POST tiene que ser repetida utilizando otra petición POST.
  - 308: Permanent Redirect
    El recurso solicitado por el navegador se encuentra en otro lugar y este cambio es permanente. A diferencia del código 301, no se permite cambiar el

método HTTP para la nueva petición (así por ejemplo, si envías un formulario a un recurso que ha cambiado de lugar, todo seguirá funcionando bien).

- 4XX - Errores del cliente
  - 400 - Bad Request
    La solicitud contiene sintaxis errónea y no debería repetirse.
  - 401 - Unauthorized
    Similar al 403 Forbidden, pero específicamente para su uso cuando la autentificación es posible pero ha fallado o aún no ha sido provista. Vea autenticación HTTP básica y Digest access authentication.
  - 402 - Payment Required
    La intención original era que este código pudiese ser usado como parte de alguna forma o esquema de Dinero electrónico o micropagos, pero eso no sucedió, y este código nunca se utilizó.
  - 403 - Forbidden
    La solicitud fue legal, pero el servidor rehúsa responderla dado que el cliente no tiene los privilegios para hacerla. En contraste a una respuesta 401 No autorizado, la autenticación no haría la diferencia.
  - 404 - Not Found
    Recurso no encontrado. Se utiliza cuando el servidor web no encuentra la página o recurso solicitado.
  - 405 - Method Not Allowed
    Una petición fue hecha a una URI utilizando un método de solicitud no soportado por dicha URI; por ejemplo, cuando se utiliza GET en un formulario que requiere que los datos sean presentados vía POST, o utilizando PUT en un recurso de solo lectura.
  - 406 - Not Acceptable
    El servidor no es capaz de devolver los datos en ninguno de los formatos aceptados por el cliente, indicados por éste en la cabecera "Accept" de la petición.
  - 407 - Proxy Authentication Required
  - 408 - Request Timeout
    El cliente falló al continuar la petición - excepto durante la ejecución de videos Adobe Flash cuando solo significa que el usuario cerró la ventana de video o se movió a otro.
  - 409 - Conflict
    Indica que la solicitud no pudo ser procesada debido a un conflicto con el estado actual del recurso que esta identifica.
  - 410 - Gone
    Indica que el recurso solicitado ya no está disponible y no lo estará de nuevo. Debería ser utilizado cuando un recurso ha sido quitado de forma permanente. Si un cliente recibe este código no debería volver a solicitar el recurso en el futuro. Por ejemplo un buscador lo eliminará de sus índices y lo hará más rápidamente que utilizando un código 404.
  - 411 - Length Required
    El servidor rechaza la petición del navegador porque no incluye la cabecera Content-Length adecuada.
  - 412 - Precondition Failed
    El servidor no es capaz de cumplir con algunas de las condiciones impuestas por el navegador en su petición.
  - 413 - Request Entity Too Large
    La petición del navegador es demasiado grande y por ese motivo el servidor no la procesa.
  - 414 - Request-URI Too Long

La URI de la petición del navegador es demasiado grande y por ese motivo el servidor no la procesa (esta condición se produce en muy raras ocasiones y casi siempre porque el navegador envía como GET una petición que debería ser POST).

  - 415 - Unsupported Media Type
    La petición del navegador tiene un formato que no entiende el servidor y por eso no se procesa.
  - 416 - Requested Range Not Satisfiable
    El cliente ha preguntado por una parte de un archivo, pero el servidor no puede proporcionar esa parte, por ejemplo, si el cliente preguntó por una parte de un archivo que está más allá de los límites del fin del archivo.
  - 417 - Expectation Failed
    La petición del navegador no se procesa porque el servidor no es capaz de cumplir con los requerimientos de la cabecera Expect de la petición.
  - 418 - I'm a teapot
    Soy una tetera.
  - 422 - Unprocessable Entity (WebDAV - RFC 4918)
    La solicitud está bien formada pero fue imposible seguirla debido a errores semánticos.
  - 423 - Locked (WebDAV - RFC 4918)
    El recurso al que se está teniendo acceso está bloqueado. 424 - Failed Dependency (WebDAV) (RFC 4918) La solicitud falló debido a una falla en la solicitud previa.
  - 425 - Unassigned
    Definido en los drafts de WebDav Advanced Collections, pero no está presente en "Web Distributed Authoring and Versioning (WebDAV) Ordered Collections Protocol" (RFC 3648).
  - 426 - Upgrade Required (RFC 7231)
    El cliente debería cambiarse a TLS/1.0.
  - 428 - Precondition Required
    Ell servidor requiere que la petición del navegador sea condicional (este tipo de peticiones evitan los problemas producidos al modificar con PUT un recurso que ha sido modificado por otra parte).
  - 429 - Too Many Requests
    Hay muchas conexiones desde esta dirección de internet.
  - 431 - Request Header Fileds Too Large)
    El servidor no puede procesar la petición porque una de las cabeceras de la petición es demasiado grande. Este error también se produce cuando la suma del tama˜no de todas las peticiones es demasiado grande.
  - 449
    Una extensión de Microsoft: La petición debería ser reintentada después de hacer la acción apropiada.
  - 451 - Unavailable for Legal Reasons
    El contenido ha sido eliminado como consecuencia de una orden judicial o sentencia emitida por un tribunal.

- 5XX - Errores de servidor
  - 500 - Internal Server Error
    Es un código comúnmente emitido por aplicaciones empotradas en servidores web, mismas que generan contenido dinámicamente, por ejemplo aplicaciones montadas en IIS o Tomcat, cuando se encuentran con situaciones de error ajenas a la naturaleza del servidor web.
  - 501 - Not Implemented
    El servidor no soporta alguna funcionalidad necesaria para responder a la solicitud del navegador (como por ejemplo el método utilizado para la petición).

- 502 - Bad Gateway
  El servidor está actuando de proxy o gateway y ha recibido una respuesta inválida del otro servidor, por lo que no puede responder adecuadamente a la petición del navegador.
- 503 - Service Unavailable
  El servidor no puede responder a la petición del navegador porque está congestionado o está realizando tareas de mantenimiento.
- 504 - Gateway Timeout
  El servidor está actuando de proxy o gateway y no ha recibido a tiempo una respuesta del otro servidor, por lo que no puede responder adecuadamente a la petición del navegador.
- 505 - HTTP Version Not Supported
  El servidor no soporta o no quiere soportar la versión del protocolo HTTP utilizada en la petición del navegador.
- 506 - Variant Also Negotiates (RFC 2295)
  El servidor ha detectado una referencia circular al procesar la parte de la negociación del contenido de la petición.
- 507 - Insufficient Storage (WebDAV - RFC 4918)
  El servidor no puede crear o modificar el recurso solicitado porque no hay suficiente espacio de almacenamiento libre.
- 508 - Loop Detected (WebDAV)
  La petición no se puede procesar porque el servidor ha encontrado un bucle infinito al intentar procesarla.
- 509 - Bandwidth Limit Exceeded
  Límite de ancho de banda excedido. Este código de estatus, a pesar de ser utilizado por muchos servidores, no es oficial.
- 510 - Not Extended (RFC 2774)
  La petición del navegador debe a˜nadir más extensiones para que el servidor pueda procesarla.
- 511 - Network Authentication Required
  El navegador debe autenticarse para poder realizar peticiones (se utiliza por ejemplo con los portales cautivos que te obligan a autenticarte antes de empezar a navegar).
- 512 - Not updated
  Este error prácticamente es inexistente en la red, pero indica que el servidor está en una operación de actualizado y no puede tener conexión.

(c) ¿Cuándo fue la última vez que se modífico la página?

La última vez que se modificó la página fue el 16 de marzo de 2016 a las 20:41

(d) Solicite a 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 definitivamente fue modificada. ¿Cómo lo hace? ¿Qué resultado obtuvo? ¿Puede explicar por qué y para queé sive?

Se puede realizar mediante el siguiente comando: curl -I -H 'If-Modified-Since: DATE-FORMAT-HERE' URL

Si la fecha especificada es posterior a la fecha de modificacion se obtiene el codigo de respuesta 304 (Not Modified), de caso contrario retorna el código 200 (OK).

Puede servir para consultar las páginas que fueron modificadas solamente despues de determinada fecha.

(e) ¿Qué significa el encabezado ETag?

Un ETag es un identificador asignado por un servidor web a una versión específica de un recurso que se encuentra en una URL.

(f) Investigue el encabezado If-Modified-Since. ¿Para qué cree que pueden servir los tres encabezados anteriores?

Sirve para realizar una request condicional, que retorna el estado 200 si la página cumple, o 304 si no cumple.

11. **Realice las siguientes pruebas:**

    (a) Ejecute el comando 'cat /home/redes/prueba-http-1.0.txt' y copie la salida completa (incluyendo los dos ente que aparecen debajo del texto).

    (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. ¿Que ocurre luego de hacerlo?

    Me retorna automaticamente la respuesta.

    (c) Repita el proceso anterior, pero copiando el contenido del archivo /home/redes/prueba-http-1.1.txt. Verifique que debería poder pegar varias veces lo mismo en el contenido sin tener que ejecutar telnet nuevamente.

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

    (a) Qué está haciendo al ejecutar el comando telnet?

    Al ejecutar el comando, se establece una conexion con el servidor www.redes.unlp.edu.ar a la espera de una request.

    (b) Qué comando HTTP utilizó? ¿Qué recurso solicitó?

    Se utilizo el comando GET, y solicitó cualquier tipo de respueta.

    (c) ¿Qué diferencias notó entre los dos casos? ¿Puede explicar por qué?

    La diferencia es que utilizando el protocolo HTTP 1.0 el comando telnet realizo la consulta automaticamente, en cambio con el protocolo HTTP 1.1 se podia mandar muchas consultas juntas, para luego esperar las respuestas en el mismo orden que se mandaron. La explicación es que el protocolo 1.1 acepta pipelining, que basicamente acepta encadenar varias consultas.

    (d) ¿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?

    Es más eficiente el protocolo 1.1, ya que en el caso de requerir varios recursos para visualizar una página, gracias al pipelining, es posible requerirlos en una sola consulta al servidor.

13. **La página www.redes.unlp.edu.ar/http/idioma.php tiene soporte para visualizarse en inglés y en español. Manipule los encabezados de HTTP para visualizar la página en los diferentes idiomas.**

    Para visualizar en español se realiza el requerimiento normalmente, en cambio, para visualizar en ingles se puede utilizar el siguiente comando:

    curl -H 'Accept-language: en' www.redes.unlp.edu.ar/http/idiomas.php

    De esta forma se visualiza la página en ingles.