**Parte I. - Jugando a ser un cliente HTTP**

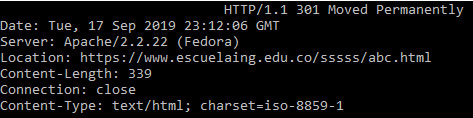
1. Abra una terminal Linux o consola de comandos Windows.
2. Realice una conexión síncrona TCP/IP a través de Telnet al siguiente servidor:
   * Host: www.escuelaing.edu.co
   * Puerto: 80

Teniendo en cuenta los parámetros del comando telnet:

telnet HOST PORT

1. Antes de que el servidor cierre la conexión por falta de comunicación:
   * Revise la página 36 del [RFC del protocolo HTTP](https://tools.ietf.org/html/rfc2616), sobre cómo realizar una petición GET. Con esto, solicite al servidor el recurso ‘sssss/abc.html’, usando la versión 1.0 de HTTP.
   * Asegúrese de presionar ENTER dos veces después de ingresar el comando.
   * Revise el resultado obtenido. ¿Qué codigo de error sale?, revise el significado del mismo en [la lista de códigos de estado HTTP](https://en.wikipedia.org/wiki/List_of_HTTP_status_codes).



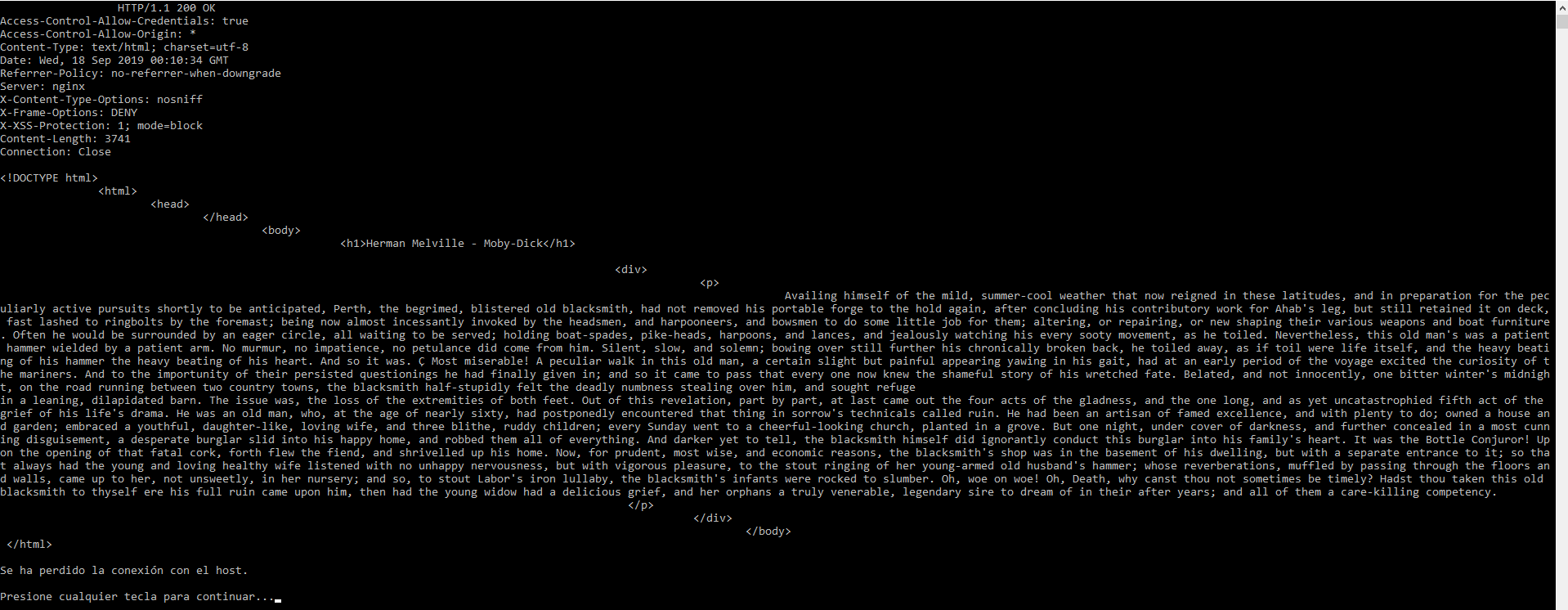


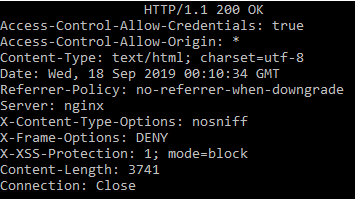
Indica que el host ha sido capaz de comunicarse con el servidor pero que el recurso solicitado ha sido movido a otra dirección permanentemente.

* + ¿Qué otros códigos de error existen?, ¿En qué caso se manejarán?

1. Realice una nueva conexión con telnet, esta vez a:
   * Host: www.httpbin.org
   * Puerto: 80
   * Versión HTTP: 1.1

Ahora, solicite (GET) el recurso */html*. ¿Qué se obtiene como resultado?





¡Muy bien!, ¡Acaba de usar del protocolo HTTP sin un navegador Web!. Cada vez que se usa un navegador, éste se conecta a un servidor HTTP, envía peticiones (del protocolo HTTP), espera el resultado de las mismas, y -si se trata de contenido HTML- lo interpreta y dibuja.

1. Seleccione el contenido HTML de la respuesta y copielo al cortapapeles CTRL-SHIFT-C. Ejecute el comando wc (*word count*) para contar palabras con la opción -c para contar el número de caracteres:

wc -c

Pegue el contenido del portapapeles con CTRL-SHIFT-V y presione CTRL-D (fin de archivo de Linux). Si no termina el comando wc presione CTRL-D de nuevo. No presione mas de dos veces CTRL-D indica que se termino la entrada y puede cerrarle la terminal. Debe salir el resultado de la cantidad de caracteres que tiene el contenido HTML que respondió el servidor.

Claro está, las peticiones GET son insuficientes en muchos casos. Investigue: ¿Cuál es la diferencia entre los verbos GET y POST?

Al momento de enviar información de un formulario el método GET la envía directamente en la URL en los parámetros que van después del “?” mientras que la transmisión de datos con POST quedan “ocultos” y para acceder a ellos se debe utilizar una herramienta adicional (php).

¿Qué otros tipos de peticiones existen?

"OPTIONS"

Es utilizado para describir las opciones de comunicación para el recurso de destino.

"HEAD"

Pide una respuesta idéntica al GET solo que no pide el cuerpo de la respuesta.

"PUT"

Reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.

"DELETE"

Borra un recurso en específico.

"TRACE"

Realiza una prueba de bucle de retorno de mensaje a lo largo de la ruta al recurso de destino.

"CONNECT"

Establece un túnel hacia el servidor identificado por el recurso.

1. En la practica no se utiliza telnet para hacer peticiones a sitios web sino el comando curl con ayuda de la linea de comandos:

curl www.httpbin.org

Utilice ahora el parámetro -v y con el parámetro -i:

curl -v www.httpbin.org

curl -i www.httpbin.org

¿Cuáles son las diferencias con los diferentes parámetros?

-i, - - include

Incluye los encabezados de respuesta HTTP en la salida.

-v, - - verbose

Útil para depurar y ver lo que sucede "debajo del capó". Una línea que comienza con '>' significa "datos de encabezado" enviados por curl, '<' significa "datos de encabezado" recibidos por curl que están ocultos en casos normales, y una línea que comienza con '\*' significa información adicional proporcionada por curl.