Universidad EAFIT
ST0263: Tópicos Especiales en Telemática, 2022-1
Laboratorio No 1.
Modificado: 01-febrero

Título: Aplicación Distribuida Web/HTTP en Sockets TCP

Conceptos Fundamentales:

Dos o más procesos / aplicaciones en Internet se pueden comunicar de diferentes maneras para implementar un servicio distribuido. Uno de los mecanismos de comunicación más básico es a través de Sockets, lo cual plantea una tubería o enlace de comunicación de intercambio de mensajes entre los procesos.

Una de las aplicaciones estándar más populares es la Web, ampliamente conocido y el cual consiste en su versión más simple (web 1.0) en un cliente, browser o navegador (ej: Chrome, Edge, Firefox, etc) que se conecta a un Servidor Web como Apache Web Server para transferir Recursos (páginas web, pdfs, etc) al Cliente.

La web es su concepto más amplio (https://es.wikipedia.org/wiki/World_Wide_Web) utiliza 3 estándares fundamentales:

1. Un Protocolo estándar de comunicación entre un browser y un servidor web: HTTP
    a. https://es.wikipedia.org/wiki/Protocolo_de_transferencia_de_hipertexto
2. Un formato de datos principal de intercambio: HTML
    a. https://es.wikipedia.org/wiki/HTML
3. Un localizador de recursos web en Internet: URL
    a. https://es.wikipedia.org/wiki/Localizador_de_recursos_uniforme

Como un mecanismo para entender y dominar los estándares de la Web (HTTP, HTML y URLs), realizarán un programa con los siguientes requerimientos:

1. Conectarse a un servidor web estándar mediante el protocolo HTTP, esta conexión la realizará mediante sockets TCP, y deberá construir un paquete HTTP-Request con todos sus componentes para comunicarse con el servidor web. Mostrar en pantalla o consola, el mensaje HTTP enviado decodificando cada una de sus componentes. (ver programas como Wireshark - https://www.wireshark.org/)
2. Recibir del servidor web el mensaje HTTP-Response y Decodificar el protocolo HTTP en MODO CONSOLA (ver programas como Wireshark - https://www.wireshark.org/), el cual permita mostrar en pantalla (consola) el mensaje HTTP-request que su programa enviará y mostrar el mensaje HTTP-response recibido. En ambos casos realizará la decodificación e interpretación de los diferentes campos de los mensajes HTTP.

3. Salvar localmente como archivos los recursos que se transfieran por el protocolo HTTP, por ejemplo: si son páginas HTML, deberán salvarse como páginas HTML (*.html), o si son recursos estáticos como PDFs, Imágenes, etc. deberán salvarse como archivos.
    a. NOTA: si se descarga una página HTML, deberá escanear o *parsear* la página HTML identificando y listando los recursos estáticos a los cuales apunta la página HTML y sobre los cuales – OPCIONALMENTE – podría descargar para tener una visualización completa local (como un mirror de la página).
4. El programa Cliente que llamaremos – 'yacurl' (Yet Another CURL), deberá ser implementado en Sockets TCP y HTTP. Ojo: NO ES REALIZAR EL SERVIDOR, SOLO UN CLIENTE. Se utilizará el protocolo HTTP por el puerto estándar 80, sin embargo, como parte de los parámetros de entrada de 'yacurl' deberá especificar la URL y opcionalmente el puerto u otros parámetros que considere, PERO NO DEBERÁ 'quemar' en código ninguna opción especifica.

A modo de ejemplo, vea programas como CURL para realizar un programa parecido con adiciones.

Otras Referencias:

- Socket de Internet - Wikipedia, la enciclopedia libre
- Network socket - Wikipedia
- Socket Programming in Python (Guide) – Real Python
- Unix Socket Tutorial - Tutorialspoint

Fecha de entrega:

Hasta el 13 de febrero 2022, hora 23:59. Por Interactiva Virtual

Trabajo individual.