ESTO ES UN BORRADOR (OJO CON LO AMARILLO)

TP D: Desarrollo de una aplicación distribuida con BSD sockets

Fecha y hora límite de entrega: 8/6 23:59:59
Cáracter: grupal (grupos de 2)

Objetivo

Que los alumnos se familiaricen con la biblioteca BSD Sockets y se interioricen con las particularidades de TCP y UDP. Que logren intercomunicar varios sistemas y sean capaces de adaptarse a protocolos ya existentes.

Los alumnos deberán desarrollar un medidor de throughput y latencia. Los resultados de cada medición deberán ser enviados por UDP a un servidor Logstash con la información en formato JSON.

Descripción de la aplicación

El sistema para medir throughput y latencia constará de un cliente y un servidor.

Requisitos funcionales

- el cliente es quien siempre inicia las pruebas (y conexiones) tanto de download como de upload
- el cliente es quien luego de finalizar la prueba envía el resultado de la misma a un host remoto a través de UDP con la información relevante codificada en JSON
- el servidor deberá estar esperando las pruebas de download como de upload
- los clientes y servidores implementados deberán interoperar contra los de los demás grupos y contra la implementación de los docentes. Es decir se deberán respetar rigurosamente las definiciones protocolares que más adelante se detallen

ESTO ES UN BORRADOR (OJO CON LO AMARILLO)

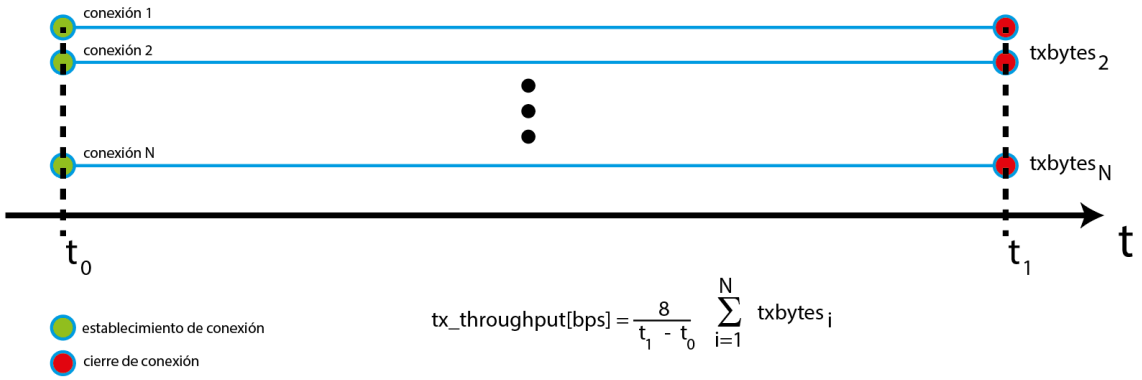
Requisitos no funcionales

- La prueba de download como de upload se realizará con múltiples conexiones TCP concurrentes
- Las pruebas de latencia se harán usando datagramas UDP. Se harán en las tres fases: fase previa, fase de medición de download y fase de medición de upload.
- Las fase de medición tanto download como upload deben durar T=20 segundos cualquiera sea la condición de la red o de los equipos extremos. Este valor T debería poder variarse fácilmente tocándolo en un sólo lugar del código.
- El puerto TCP del server para la fase de download es 20251
- El puerto TCP del server para la fase de upload es 20252
- El puerto UDP del server para medir la latencia y consultar resultados es 20251

Medición de throughput

La **medición de throughput de download** se realiza acorde al siguiente procedimiento.

- El cliente establece N conexiones en simultáneo contra el server al port 20251.
- El servidor debe proveer un flujo de información (bytes) a la máxima velocidad que sus recursos lo permitan por un tiempo de al menos T=20 segundos en cada una de estas N conexiones. Se deberán tomar los recaudos que pasados T+ 3 segundos se cierren las conexiones si es que el cliente no inició el cierre de conexión.
- El cliente debe durante un período de tiempo T=20 segundos ir contabilizando los bytes recibidos en cada una de esas N conexiones.
- La medición de throughput de download será la sumatoria de bytes recibidos en las N conexiones divido el tiempo transcurrido que se desea que sea T=20 segundos, pero se debe usar el verdadero tiempo que tardó transcurrió en abrirse y cerrarse las N conexiones. El tiempo es medido por el cliente.



La **medición de throughput de upload** se realiza acorde al siguiente procedimiento.

- El cliente debe generar **aleatoriamente** un número de 4 bytes que será usado como identificador de prueba. El primer byte de este identificador no podrá ser 0xff.
- El cliente establece N conexiones en simultáneo contra el server al port 20252
- En cada conexión el cliente debe enviar en los primeros bytes un identificador de prueba compuesto por 4 bytes y un identificador de conexión compuesto por 2 bytes.

ESTO ES UN BORRADOR (OJO CON LO AMARILLO)

Por ejemplo, supongamos que los primeros 6 bytes son: 0x42 0x3a 0xaf 0xc9 0x00 0x01 donde 0x42 0x3a 0xaf 0xc9 es el identificador de medición y debe ser común a las N conexiones, y 0x00 0x01 es el identificador de conexión en el contexto de la prueba. Las conexiones hay que numerarlas secuencialmente empezando desde el número 1.

- El servidor deberá contabilizar durante T=20 segundos (tiempo contabilizado desde que el system call accept()) los bytes recibidos del cliente. El servidor utilizará internamente también la IP origen del cliente y el identificador de prueba para que la probabilidad de colisión disminuya en un ambiente múlti-cliente)
- El cliente cuando le finalicen la totalidad de las conexiones deberá solicitar los contadores de la prueba al server, identificando los datos **sólo** con el identificador de prueba (4 bytes).
- La estructura de datos devuelta por el servidor será consultada por UDP al port 20251. El cliente enviará un datagrama UDP cuyo payload sean únicamente los 4 bytes del identificador de prueba. Por ejemplo: 0x42 0x3a 0xaf 0xc9 . La estructura de datos del servidor será la siguiente:
    - TO BE DEFINED
    - identificador de prueba<LINESEP>
      identificador de conexion <COLSEP>bytes<COLSEP>tiempo net<LINESEP>

Medición de latencia

La medición de latencia se realiza acorde al siguiente procedimiento

- El cliente envía un datagrama UDP con un payload de 4 Bytes **aleatorio** cuyo **primer byte es 0xff**. Justo después de enviar el datagrama (system call send() o la que corresponda), tomará un timestamp.
- El servidor deberá contestar dicho datagrama copiando los 4 Bytes recibidos. Debe asegurarse de que sean sólo 4 Bytes, ni más ni menos. Ante una diferencia a lo establecido, no deberá contestar nada. Y logueará en pantalla y/o en un archivo dicho error.
- El cliente al recibir la respuesta, tomará un timestamp y deberá validar que es la respuesta a la consulta que envió. Deberá validar los 4 Bytes enviados y la IP de dónde lo recibió. El RTT será la diferencia entre los timestamps tomados.
- Estos pasos se llamarán medición de RTT

Se harán 3 mediciones de RTT antes de las pruebas de download y upload. Las mediciones deberán estar separadas por un segundo entre la recepción de la respuesta y el envío. Si no llega ninguna respuesta en un tiempo prudencial (10 segundos), el cliente debe abortar las otras mediciones e informar claramente en la salida del programa.

Exportación de resultados

TBD

ESTO ES UN BORRADOR (OJO CON LO AMARILLO)

Funcionamiento general

El funcionamiento general del programa podrá ser visto en la siguientes etapas del diagrama

Recomendaciones generales

- Se sugiere hacer uso de multi-process para la paralelización de las pruebas de download y upload. Si bien podría hacerse con un solo proceso esto podría tener limitaciones de performance que incurran en gastar mucho tiempo en optimización.
- Se sugiere controlar la disponibilidad del recurso socket para operaciones de lectura y escritura usando la system call `select()` o `poll()` en su defecto.
- Se sugiere loguear en pantalla y/o en un archivo las conexiones que el server establece para realizar las mediciones.
- Usar el comando `nc` para ir probando pequeños códigos (ejemplo: `nc -k -l 0.0.0.0 20251`)
- Se recomienda probar usando WANem acotando el bandwidth en un entorno de máquinas virtuales. También se sugiere comparar resultados contra el utilitario `iperf`. (Ver diapositivas para configurar rutas de hosts).

Entregables

Se deberá entregar el código junto con sus instrucciones de uso y compilación.

El programa será probado desde Internet idealmente en ambos modos. Cliente y servidor.
(el lado servidor tiene el desafío de como hacer público el server en Internet)
El programa será probado con la maqueta de routers en vivo y debe dar un valor acorde al traffic shaper configurado

Se deberá hacer una breve presentación Powerpoint grabada en video (Youtube o similar) que será expuesta el 17/06/2025 y el 18/06/2025. Se debe mostrar la arquitectura diseñada y un diagrama de la misma. A su vez se debe hacer énfasis en la experiencia personal de programar usando BSD Sockets (desafíos, dificultades, vivencias, expectativas, etc). La presentación deberá durar aproximadamente 8 minutos.

ESTO ES UN BORRADOR (OJO CON LO AMARILLO)

Cabe la posibilidad de que se haga alguna pregunta a cualquiera de los integrantes del grupo al respecto de lo expuesto o relacionado al código o a la estrategia que se utilizó para satisfacer los requerimientos de la aplicación.