BROADCASSST

#Conectividad en 200 caracteres

Protocolos de Comunicación TCP/IP

Laboratorio N°5

Interfaz de Socket

Fernando Nahuel Tuquina - fernandotuquina@gmail.com
Santiago José Gianotti - sjgianotti1@gmail.com
Santiago Latina - santilatina99@gmail.com
Santiago Martín Berretta Gali - santiagomartinbg@gmail.com

Índice

Desarrollo

1) ¿Qué tipo de Socket decidió usar para el desarrollo de las aplicaciones? ¿Por qué?

El análisis efectuado en la elección del socket durante desarrollo de las aplicaciones fue el siguiente:

1. Se debe enviar el mensaje a todos los Hosts de la red local
2. El servidor desea enviar un mensaje desde un puerto efímero (50.000 - 50.100)
3. El cliente debe estar escuchando un puerto específico (2500)

Existen dos formas de lograr la comunicación del mensaje:

- Crear una conexión mediante TCP
- Utilizar datagramas UDP

Utilizando TCP:

Para que el servidor sea capaz de enviar información a todos los hosts de la red local mediante TCP, este debe conocer de antemano la dirección IP de cada uno de los hosts.
Luego, debe enviar el mensaje a cada host por separado.
Para ello, el cliente se deberá “suscribir” a estos mensajes, avisando de antemano al servidor que desea recibir mensajes, para que este anote su dirección IP.

Es evidente, que esta alternativa no es la mejor, ya que posee muchas desventajas:

1. El cliente debe conocer el socket del servidor a donde se va a suscribir para este servicio
2. El servidor debe enviar mensajes a cada host por separado, creando paquetes que contienen el mismo mensaje y por lo tanto generando tráfico innecesario en la red.
3. La complejidad de programar tanto el cliente como el servidor es bastante alta debido al punto 1 y 2.

Utilizando UDP:

UDP posee una ventaja inmensa al poder utilizar broadcasts;

1. El servidor solo debe enviar un datagrama en broadcast al puerto 2500
2. El cliente solo debe escuchar en el puerto 2500 y mostrar el mensaje.

Además, al utilizar broadcast se genera un solo paquete, por lo que la red local no será inundada de mensajes.

También, UDP posee una ventaja inherente al ser más rápido, pero para nuestro caso en particular donde se muestra solo un mensaje en pantalla, la velocidad no es crítica.

Por otro lado, UDP no es confiable, por lo que es posible que algunos hosts nunca reciban el mensaje, pero esto no es demasiado relevante al tratarse de un simple mensaje.

En conclusión, se usó socket orientado a datagrama (UDP).

2) ¿Qué tipo de servidor decidió desarrollar? ¿Por qué?

Se decidió utilizar un servidor iterativo ya que el procesamiento de la petición del cliente es simple y "acotada", y la respuesta se realiza en forma rápida y con un intercambio mínimo de información.

Además que la programación del mismo es mucho más sencilla que uno concurrente, ya que no se necesita manejar concurrencia de procesos.

Por otro lado, los servidores iterativos suelen ser asociados con los sockets de datagrama, aunque no necesariamente.

3) ¿Qué rutinas de envío y recepción de mensajes usó? ¿Por qué?

Las rutinas de envío y recepción de mensajes fueron las propias de los protocolos No-orientadas a conexión: tanto en Servidor como en Cliente, luego de creado el socket y haber sido asociado a una dirección IP y a un puerto, inicia el intercambio de datos. Las system call de envío (sendto() para el servidor) y recepción de datos (recvfrom() para el cliente) permiten especificar la dirección IP y puerto de la aplicación a la que se envía o de la cual se recibe la información.

4) Describa las dificultades que tuvo al desarrollar el laboratorio.

La primera dificultad surgió debido a que los integrantes del grupo poseían distintas ideas sobre cuál era la mejor forma de implementar la aplicación. Por lo tanto, se hizo el análisis del punto número 1 antes de comenzar a escribir el código.

Durante este análisis, debido a que los ejemplos en la bibliografía están orientados a conexión, fue necesario buscar manuales y referencias en internet sobre el uso de datagramas UDP con broadcast para ver si es posible realizar la implementación del diseño tentativo de la aplicación.

Una vez que se determinó que es posible implementar, la mayor dificultad fue coordinar la programación debido a que cada integrante del grupo dispone de poco tiempo cuando el resto podía.

La última gran dificultad se encontró a la hora de enviar el broadcast, ya que se desconocían algunas configuraciones que debían ser realizadas para que este funcione. Esto se pudo solucionar buscando referencias en varios foros de programación, como por ejemplo, stackoverflow.

Diagrama de flujo