Redes de Computadoras

Capas de Aplicación y Transporte

Obligatorio 2 (v.2) – 2015

Facultad de Ingeniería - UdelaR
Instituto de Computación
Departamento de Arquitectura de Sistemas

Nota previa - IMPORTANTE
Se debe cumplir íntegramente el "*Reglamento del Instituto de Computación ante Instancias de No Individualidad en los Laboratorios"* . En particular, está prohibido utilizar documentación de otros grupos, de otros años, de cualquier índole, o hacer público código a través de cualquier medio (foros, correo, papeles sobre la mesa, etc.).

Fecha de entrega
Los trabajos deberán ser entregados antes del ***11 de octubre a las 23:30 horas***. No se aceptará ningún trabajo pasada la citada fecha y hora. En particular, no se aceptarán trabajos enviados por e-mail a los docentes del curso, ni entregados en medios físicos en el instituto.
El sistema de entregas soporta múltiples entregas por grupo, llevando un histórico de las mismas. Se recomienda realizar una entrega vacía con tiempo, a los efectos de verificar que su sistema le permite enviar su trabajo correctamente.
En cada monitoreo se deberán realizar las entregas parciales al docente.

Forma de entrega
Una clara, concisa y descriptiva documentación es clave para la evaluación de los trabajos entregados. La entrega del laboratorio consiste en un único archivo **obligatorio2-redesXX.tar.gz** que deberá a su vez contener los siguientes archivos:

- **obligatorioXX.pdf**, donde se documenta el trabajo realizado en el presente obligatorio.
- Los *fuentes* y scripts de compilación que implementan la solución.

El archivo **obligatorio2-redesXX.tar.gz** deberá ser generado utilizando la herramienta *GNU tar* y compresión *gzip*. Otros formatos (bz2, rar, zip, cab, jar, etc.) no son válidos y serán rechazados, con la consecuente pérdida del curso para todos los integrantes del grupo. La entrega se realizará a través de un recurso habilitado para tal fin en la plataforma EVA, que será oportunamente avisado por dicho medio.

La documentación del obligatorio debe incluirse dentro del archivo de la entrega. La misma debe entregarse como un único archivo tipo PDF. El mismo deberá respetar la numeración de secciones acorde a los requerimientos.
Es necesario que en el informe figuren el nombre y la cédula de identidad de cada integrante del grupo. En caso que esto no se cumpla el obligatorio no será corregido, con la consecuente pérdida del curso de sus autores. NO se aceptarán otros formatos de informe. (ver http://www.universidad.edu.uy/odfpdf/)

Objetivo general del Laboratorio

Aplicar los conceptos teóricos de capas de aplicación y transporte, el desarrollo de un servicio simple de comunicación para una LAN.

Descripción general del problema

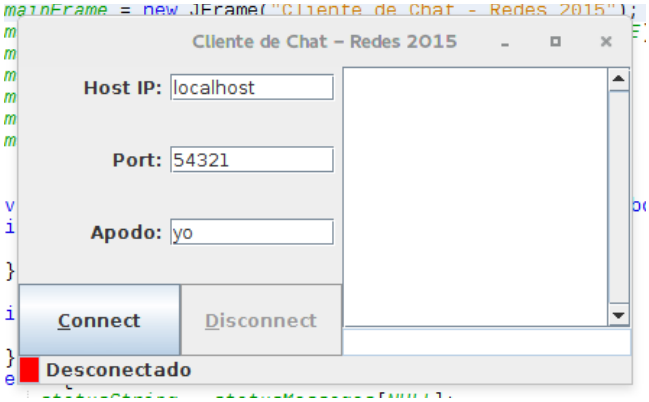
Se desea implementar una aplicación simple de comunicación en LAN que permita el intercambio de información (mensajería) entre múltiples usuarios.

Descripción del problema

Se desea implementar una aplicación (cliente y servidor) de mensajería en línea para grupos en una LAN. La misma utilizará el protocolo UDP y buscará ofrecer un servicio de mensajería confiable sobre éste, permitiendo el intercambio de mensajes entre todos los clientes conectados simultáneamente. El nivel de confiabilidad deseado es equivalente al RDT 3.0 visto en el curso sin considerar mensajes corruptos.

Descripción del cliente

El cliente podrá ser programado en Java, partiendo preferentemente basado en la siguiente GUI:



Con el docente de monitoreos se puede discutir una variación en la interfaz y del lenguaje de programación. Se aceptará una interfaz orientada a línea de comandos si el alumno lo prefiere.
Previo a iniciar la sesión el usuario deberá poder seleccionar la dirección del servidor, el puerto en el que atenderá, así como, su apodo a lo largo de la sesión. Luego que el cliente se presente al servidor, podrá mandar mensajes y recibir los mensajes de otros clientes conectados. Los mensajes mostrados llevarán como prefijo el apodo de quién lo envía, que no podrá ser modificable durante la conexión. Será posible enviar mensajes privados a usuarios específicos. En este caso, el servidor solamente enviará el mensaje a un único cliente.
Es responsabilidad del usuario suministrar la dirección IP (o nombre de host) y puerto en el que atiende el servicio. El puerto predeterminado será el 54321.

Descripción del servidor

El servidor deberá ser programado en C/C++, será una aplicación *multithreaded* que repetirá mensajes entre clientes. Aceptará inicios de sesión de los clientes y llevará registro de los mismos. Los mensajes recibidos desde un cliente serán enviados (*relayed*) a todos los clientes registrados en ese momento. No se llevará registro de los mensajes históricos.
Luego de que un cliente inicie sesión recibirá mensajes de todos los clientes y, de igual forma, sus mensajes serán copiados a todos los clientes.
Se deberán aceptar arribos y partidas de los clientes en cualquier momento. Las estructuras compartidas por los *threads* deberán estar correctamente mutuo-excluídas.

El servidor tendrá una interfaz de texto mínima, mostrando información de estado elemental. El servidor debe responder a entradas desde el teclado, mostrando información según la siguiente especificación:

- ***a*** – cantidad de clientes conectados
- ***s*** – cantidad de mensajes enviados
- ***d*** – cantidad de conexiones totales
- ***f*** – tiempo (wall time) de ejecución

Cada vez que un cliente envíe un mensaje, éste será copiado por salida estándar del proceso servidor (además de a los otros clientes) mostrando la dirección IP origen.
Habrá un *timeout* para que el servidor desconecte clientes inactivos, definido por la implementación.

Protocolo de comunicaciones

El protocolo de comunicaciones entre el cliente y el servidor deberá ser confiable, resolviendo pérdidas de paquetes y duplicados. Para transmisiones en la misma LAN se utilizará la dirección de *multicast* 225.5.4.<no_grupo>. El servidor transmitirá a la dirección de *multicast* , donde escucharán

los clientes. Los clientes transmitirán por *unicast* a la dirección del servidor.

Se deberá proponer un protocolo de transporte confiable sobre UDP sobre el cual se implementará el protocolo de mensajería que se describe debajo. El protocolo de la aplicación está orientado a cadenas de caracteres, y cada línea es un comando. El carácter CR es el separador de línea. Los mensajes no excederán el tamaño de un datagrama.

Comandos de cliente al servidor

```
LOGIN <usuario><CR>
LOGOUT<CR>
GET_CONNECTED<CR>
MESSAGE <msg><CR>
PRIVATE_MESSAGE <receptor> <msg><CR>
```

Comandos del servidor al cliente:

```
RELAYED_MESSAGE <emisor> <msg><CR>
PRIVATE_MESSAGE <emisor> <msg><CR>
CONNECTED <usr1>[|<usr2>...]<CR>
GOODBYE<CR>
```

Avance recomendado:

Semana 39 (21-25 de setiembre)

Ejemplo de servidor UDP y de Cliente UDP, en C/C++ y el lenguaje de propuesta para el cliente. Se sugiere entregar una propuesta del protocolo de comunicaciones

Semana 40 (28 de setiembre – 2 de octubre)

Cliente funcional, en su versión beta.

Semana 41 (5 al 9 de octubre)

Servidor funcional, en su versión beta.

Entrega final (11/10 antes de las 23:30hs)

1. Documentación, con énfasis en el protocolo (descripción exhaustiva en idioma Español). Pseudocódigo detallado y comentado de cliente y servidor.
2. Cliente
    1. Código fuente del cliente.
    2. Makefile, ant, maven o gradle para la compilación de éste (siempre que sea más complejo que `javac Cliente.java`).
3. Servidor
    1. Código fuente del servidor (implementando el pseudocódigo).
    2. Makefile o ant para la compilación de éste (siempre que sea más complejo que `gcc ircserver.cc`).

Referencia

Examen de Diciembre de 2009:
https://www.fing.edu.uy/inco/cursos/redescomp/examenes/sirc0912.pdf