Protocolos de Comunicación
y Sistemas Embebidos

Tarea 2: "Red virtual Anillo"

Objetivos

Implementar comunicación Serial a través de puertos virtuales,a través del uso del protocolo SLIP e IP simplificado.

Grupos de Trabajo

Se debe trabajar en grupos de 2 personas donde ambos deberán trabajar en misma medida.

Programa

Se pide desarrollar **un único programa** llamado "nodo", el cual debe ejecutarse en el entorno de una Raspberry PI. El programa debe ser capaz de enviar o recibir mensajes, dependiendo de los parámetros de entrada que se le ingresen al programa, cada ejecución del programa corresponderá a un nodo. El mensaje por enviar debe ser encapsulado con los protocolos SLIP e IPv4 Modificado (mostrado en tabla 1).

- Al momento de ejecutar el programa en la consola, se debe especificar en la misma línea de comandos la IP del nodo, la ruta del puerto tx y la ruta del puerto de recepción respectivamente (usar nomenclatura *../tmp/p2*) *Ejemplo: ./nodo 192.168.0.1 ../tmp/p1 ../tmp/p2*. Para facilitar la ejecución incluir estos comandos en el archivo makefile.

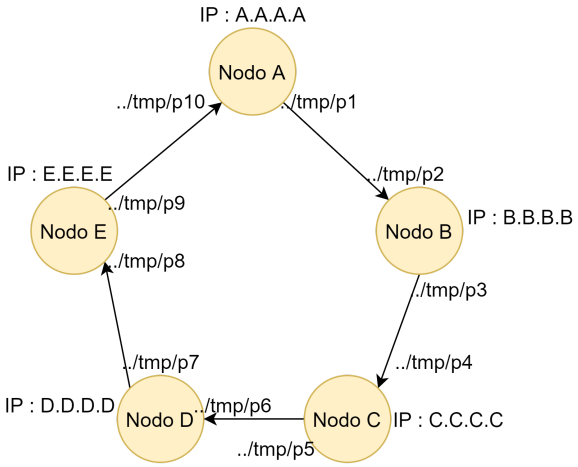

Figura 1: Red anillo a formar.

- Cada nodo utilizará 2 puertos, uno como receptor conectado al nodo anterior y otro como emisor conectado al nodo siguiente de modo que se forme la red anillo donde la información tiene un solo sentido de flujo. Con el fin de generar las siguientes interconexiones virtuales

mostradas en la figura 1 en donde se observa la IP de cada nodo y sus respectivos puertos de comunicación.

- Todos los nodos tienen que poder mandar 2 tipos de mensajes:
  - Broadcast: Este tipo de mensaje tiene una dirección de destino con todos sus bits en alto, sin incluir el identificador de red, aunque para este caso se utilizará la dirección de destino F.F.F.F. Al recibir un mensaje con esta dirección, el programa debe informar que se recibió un broadcast, mostrar el mensaje en la consola, indicando que nodo envió el mensaje, reducir en 1 su TTL y reenviar el mensaje por su puerto emisor. Si un nodo recibe un mensaje tipo broadcast con su dirección, el mensaje debe ser descartado.
  - Unicast: Este tipo de mensajes tiene una dirección de destino de un nodo en particular. Al enviar este mensaje los nodos que no tengan la IP de destino deben simplemente reducir en 1 el TTL del paquete y enviarlo por su puerto emisor. Al momento del nodo destino recibir este mensaje debe anunciar por pantalla que recibió un mensaje tipo unicast y mostrar en la consola el contenido del mensaje.
- Todos los mensajes enviados deben ser encapsulados en el protocolo IPv4 modificado (Capa 3) y luego empaquetados en SLIP (Capa 2) para su posterior envío.
- Todos los menús por consola deben ser auto explicativos.

Información Adicional:

En la tabla 1 se muestra el Protocolo IPv4 modificado en donde se observan diferentes campos, con sus respectivos pesos en bits, para este caso el Flag de fragmento, el offset de Fragmento se consideran para uso en la sección de puntos extras, en donde el primer campo menciona indica con un 0 que el paquete no es fragmentado, con un 1 que es un fragmento y con un 2 que es último fragmento (el campo “Identificación” es el mismo para los fragmentos del dato fragmentado), los demás campos deben ser utilizados para la tarea de la siguiente manera, la longitud de datos permite una capacidad de 2^{16} - 1 bytes originalmente pero para este caso solo se limita a 1500 Bytes, el TTL será el campo en donde se guarde el Tiempo de vida del paquete, la identificación es única por dato y es ingresada como un contador por el emisor (para paquetes fragmentados este identificador es el mismo), la Suma de verificación es solo de la cabecera, la IP de origen y destino indican el direccionamiento de los nodos y los datos son el mensaje a enviar.

Tabla 1: Protocolo IPv4 Modificado.

<table>
<tr><td>0</td><td>7</td><td>8</td><td>15</td><td>16</td><td>31</td></tr>
<tr><td>Flag de fragmento</td><td colspan="3">Offset de Fragmento</td><td colspan="2">Longitud de los datos</td></tr>
<tr><td colspan="2">TTL</td><td colspan="2">Identificación</td><td colspan="2">Suma de verificación</td></tr>
<tr><td colspan="6">IP origen</td></tr>
<tr><td colspan="6">IP destino</td></tr>
<tr><td colspan="6">Datos</td></tr>
</table>

Protocolos de Comunicación y Sistemas Embebidos

...

Para la fácil comprensión de su código es necesario comentar por secciones, describiendo qué es lo que se está llevando a cabo. En caso de usar funciones, describir qué hace, sus salidas y entradas.
La tarea debe incluir un archivo Makefile para compilar ambos programas. En caso de que los programas no compilen, la tarea no será revisada.

Informe

Se debe incluir un informe con una breve investigación y descripción de lo realizado que incluya:

- Describir los protocolos IPv4 (original) y describiendo la utilidad de los campos que posee y SLIP, y explicar cómo se encapsulan éstos, es decir, el orden del encapsulamiento de los 2 protocolos.
- Fundamentar la elección del valor de TTL.
- Además, debe incluir el "manual del usuario" del programa donde se debe especificar claramente cómo se debe compilar, montar, correr la aplicación y cómo se utiliza. Incluya capturas de pantalla de todo el proceso. Éste debe servir para replicar el funcionamiento, por lo que debe considerar todos los pasos que ustedes siguieron para implementarlo.

Puntos extra

Incluir la opción de mandar un mensaje tipo unicast fragmentado en un número de partes a elección, utilizando los campos Flag de fragmento y Offset de fragmento. Este mensaje debe ser mostrado igual que el unicast, pero además informando que es un paquete fragmentado e indicar cuantos fragmentos tenía. La cantidad de fragmentos del mensaje no debe ser superior a 20.

Entrega

Debe ser subida a Campus Virtual la fecha indicada con el texto. Recuerde que en el archivo comprimido deben estar todos los archivos necesarios para el correcto funcionamiento de la tarea, este debe tener los apellidos de los integrantes en formato "Apellido1_Apellido2.zip". Además, se debe incluir una carpeta "doc" con el informe solicitado en latex y pdf.

Fechas

Entrega: 20 de Junio de 2024.

Observación

Se descontará **1 punto** por día de atraso. **La copia será sancionada con un 1.0.** Recuerde documentar adecuadamente el código fuente, de lo contrario tendrá descuentos. **Se descontará puntaje si la tarea no cumple con los formatos solicitados.**

Referencias útiles

https://www.geeksforgeeks.org/command-line-arguments-in-c-cpp/