Redes de Computadoras

Obligatorio 2 – 2012

Protocolo Confiable de Transporte (*PCT*) con Ventana Go-Back-N

Facultad de Ingeniería
Instituto de Computación
Departamento de Arquitectura de Sistemas

En este obligatorio se implementará el servicio de transferencia de datos confiable (PCT), con política Go Back N de Automatic Repeat reQuest (ARQ). El mismo debe implementarse sobre el protocolo IP, que provee un servicio de datagramas no confiable (best effort), implementado por medio de raw sockets.

Nota previa - IMPORTANTE

Se debe cumplir íntegramente el "Reglamento del Instituto de Computación ante Instancias de No Individualidad en los Laboratorios", disponible en `http://www.fing.edu.uy/inco/pm/uploads/Ense%f1anza/NoIndividualidad.pdf`

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.).

Forma de entrega

La entrega se realizará a través de un recurso habilitado para tal fin en la plataforma EVA, que será oportunamente avisado por dicho medio. 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 entregar correctamente. Se considerará como válida la última entrega dentro del plazo asignado.

Se debe entregar un solo archivo **ob2.tar.gz** que contenga los ítems descritos en el apartado **Entregable**. Dicho archivo 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.

Fecha de entrega

Los trabajos deberán ser entregados siguiendo el procedimiento descrito anteriormente antes del domingo 14 de octubre a las 23:30 horas, sin excepciones. No se aceptará ningún trabajo pasada la citada fecha. En particular, no se aceptarán trabajos enviados por mail a los docentes del curso, ni entregados en cualquier otro medio.

Observaciones

El laboratorio se realizará en la máquina virtual (VM) **BackTrack3** disponible en las computadoras de las salas linux, en el directorio /home/redes. Puede trabajar con esta VM en cualquier entorno, pero recuerde que la defensa de la tarea se se realizará en las salas de FING, por lo que debe probarse su funcionamiento en ese ambiente previo a la entrega.

Descripción del problema a resolver

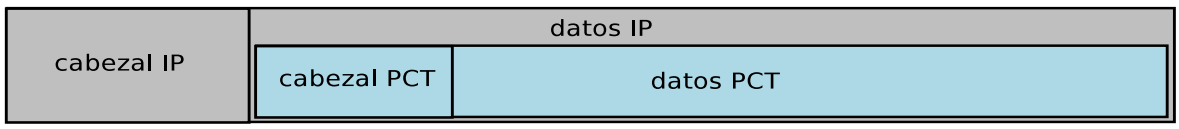
En el presente obligatorio, se implementará un servicio de capa transporte confiable y orientado a conexión el cual denominaremos PCT *(Protocolo Confiable de Transporte*), sobre el protocolo de capa red *Internet Protocol (IP)*. El término confiable indica que los datos enviados por el extremo origen, serán entregados a la aplicación destino sin pérdidas, y en el mismo orden en el que fueron enviados, independientemente de los problemas que pueda presentar la transferencia de datos por la red (pérdida o duplicación).

Las características generales de una conexión *PCT* son las siguientes:

- Debe establecerse entre un extremo activo que inicia una conexión utilizando el método `conectarPCT` y un extremo pasivo que espera conexiones utilizando el método `aceptarPCT`.
- Se enviarán datos en forma unidireccional desde el extremo que inició la conexión hacia el otro extremo.
- Deben proveerse dos métodos `escribirPCT` y `leerPCT`, que implementarán la solicitud de envío y recepción respectivamente, entre las aplicaciones conectadas en los extremos.
- Debe permitir el cierre de conexiones, que podrá ser solicitado por cualquiera de los extremos.
- Debe permitir la transmisión de bytes de un extremo al otro (tanto texto como de datos binarios).
- Debe respetar el orden de entrega. Los datos se deben entregar a la capa de aplicación como un flujo continuo de bytes recibidos, independientemente de los mecanismos utilizados para la transmisión.
- Se implementarán buffers de recepción y envío auxiliares, que serán utilizados por las aplicaciones para escribir/leer los bytes enviados/recibidos.
- Se implementará un mecanismo de ARQ *Go Back N*. Se numerarán los **paquetes** transmitidos en la red (**y no bytes como en TCP**), debiendo resolver la recuperación de información en caso de pérdida. La ventana N debe ser modificable, y se tomará de una constante predefinida (`GBN_WINDOW`). Será responsabilidad del protocolo garantizar la entrega de los datos que se hayan perdido por cualquier motivo (típicamente, fallas en la red). En caso de saturación de los buffers de envío y/o recepción del *PCT*, debe rechazarse la solicitud de envío de la aplicación, o de recepción desde el extremo opuesto. Asimismo se deben descartar los datos recibidos en forma duplicada.
- Solo existirá una sesión *PCT* por aplicación. Las sesiones *PCT* se identificarán con los siguientes 4 elementos,{*IP origen*, *puerto PCT origen*, *IP destino, puerto PCT destino*}. El *puerto PCT origen* debe tener un valor diferente al *puerto PCT destino*.

Para la implementación, se utilizarán sockets del tipo *Raw Sockets* de la capa de red [2], que permiten el envío y recepción de datagramas, sin utilizar los mecanismos usuales aplicados por el sistema operativo. De ésta forma se tendrá acceso a **todos los datagramas** intercambiados en la interfaz de *loopback (lo)*, siendo responsabilidad de la API diseñada la selección de los datagramas que tengan como destino la aplicación conectada por ella.

Se utilizará el protocolo IP para el intercambio de datos y control. *PCT* intercambiará los datagramas identificados con *0xFF* (255) en el campo *Protocol* de IP.



Es responsabilidad de la API implementada el armado de los datagramas IP a enviar, para lo que se deberá colocar los datos exigidos por el protocolo IP en los campos de control y datos.

Los datagramas IP intercambiados, contarán en su carga útil con datos en formato *PCT*, con los campos de control especificados en la letra.

Esquema general del protocolo *PCT*

Para el establecimiento de la conexión se utilizará un mecanismo *three-way handshake*, como el visto en el teórico para TCP.

Cada un tiempo de `TIMEOUT` segundos, el protocolo deberá enviar un segmento *PCT* que permita verificar que el otro extremo se encuentra activo. La no recepción de ningún segmento por un tiempo de (2 * `TIMEOUT`), será tomado como la pérdida de la conexión y en este caso cualquier método devolverá un error (-1).

Para el cierre de la conexión se enviará un segmento `FIN`, y se debe devolver `FIN`,`ACK`. En caso de pérdida del segmento `FIN` se saldrá por `TIMEOUT`.

Se deben implementar los siguientes métodos:

- `int crearPCT(struct in_addr localIPaddr)` Crea las estructuras de datos, e inicia los procedimientos necesarios para el control del *PCT*. Devuelve un valor mayor o igual a `0`. En caso de error devuelve -1.
- `int aceptarPCT(unsigned char localPCTport)` Queda esperando en forma pasiva la conexión de otro equipo a través del puerto PCT especificado (`localPCTport`). Esta función bloquea la aplicación hasta recibir una conexión. En caso de realizar en forma satisfactoria el procedimiento de conexión devuelve `0`, en caso contrario devuelve `-1`.
- `int conectarPCT(unsigned char localPCTport,unsigned char peerPCTport, struct in_addr peerIPaddr)` Inicia en forma activa la conexión con otro equipo (`peerIPaddr`) a través del puerto PCT especificado (`peerPCTport`). Esta función bloquea la aplicación hasta establecer la conexión. En caso de realizar en forma satisfactoria el procedimiento de conexión devuelve `0`, en caso contrario devuelve `-1`.
- `int escribirPCT(const void *buf, size_t len)` Solicita el envío al otro extremo de la conexión, de los datos pasados en `buf` de largo `len`. Esta función, dependiendo del estado y capacidad de los buffers disponibles, aceptará para envío hasta `len` bytes pasados en la variable `buf`, los que se almacenarán hasta tener la certeza de su recepción por el extremo opuesto. La función devuelve la cantidad de bytes aceptados para su transmisión; en caso de no ser posible por falta de buffers devuelve `0` y en otros casos el procedimiento devuelve `-1`. En caso de no haberse establecido conexión previamente devuelve `-1`.
- `int leerPCT(void *buf, size_t len)` Entrega como máximo `len` bytes recibidos por la conexión. En caso de no contar con datos devolverá `0`. Los datos se devuelven a través del parámetro `buf`. El procedimiento devuelve cantidad de bytes devueltos en `buf`, y en caso de error devuelve `-1`. En caso de no haberse establecido conexión devuelve `-1`.
- `int cerrarPCT()` Cierra la conexión en forma ordenada y destruye las estructuras de datos, y los procedimientos necesarios para el protocolo *PCT*. Previo al cierre deben enviarse los datos que se encuentran en los buffers de *PCT*. En caso de error devuelve `-1`.

Formato de TPDUs utilizados por PCT

El intercambio de información entre los dos extremos se debe realizar a través de *TPDUs (Transport Protocol Data Unit)* con formato *PCT*. Las *TPDUs* contienen un cabezal cuya estructura es presentada a continuación y un campo de datos con largo entre `0` y `MAX_PCT_DATA_SIZE` bytes.

```
struct pct_header {
  unsigned char        srcPort;
  unsigned char        destPort;
  unsigned char        flags_pct;    /* Flags de control  */
  unsigned char        nro_SEC_pct;  /* nro de secuencia de pct*/
  unsigned char        nro_ACK_pct;  /* nro de ACK de pct*/
};
```

El campo `flags_pct` se utiliza considerando los diferentes bits del octeto, según el siguiente esquema:

- `DAT`, indica si el segmento *PCT* contiene datos (valor en 1). Los segmentos que solo tengan información de control (`DAT` en 0) no se numerarán para el *ARQ*. Se accede con la máscara `0x80`.
- `SYN`, para establecimiento de conexión (similar a TCP). Se accede con la máscara `0x01`.
- `ACK`, para indicar que el segmento contiene información de ACK (similar a TCP). Se accede con la máscara `0x02`.
- `FIN`, para indicar fin de conexión (similar a TCP). Se accede con la máscara `0x04`.

Go Back N para PCT

Se debe implementar el protocolo según se presenta en el libro de texto del curso. Los números de secuencia se deben incrementar modulo 256, para asegurarse que pueden ser contenidos en el campo `nro_SEC_pct` y `nro_ACK_pct`, y se deben redefinir las comparaciones realizadas en la máquina de estados. Las funciones de envío a la red se presentan como `udt_send`, las recepciones desde la red como `rdt_rcv`, mientras que las solicitudes de envío desde la aplicación se referencia como `rdt_send`.

La máquina de estados del que envía es:

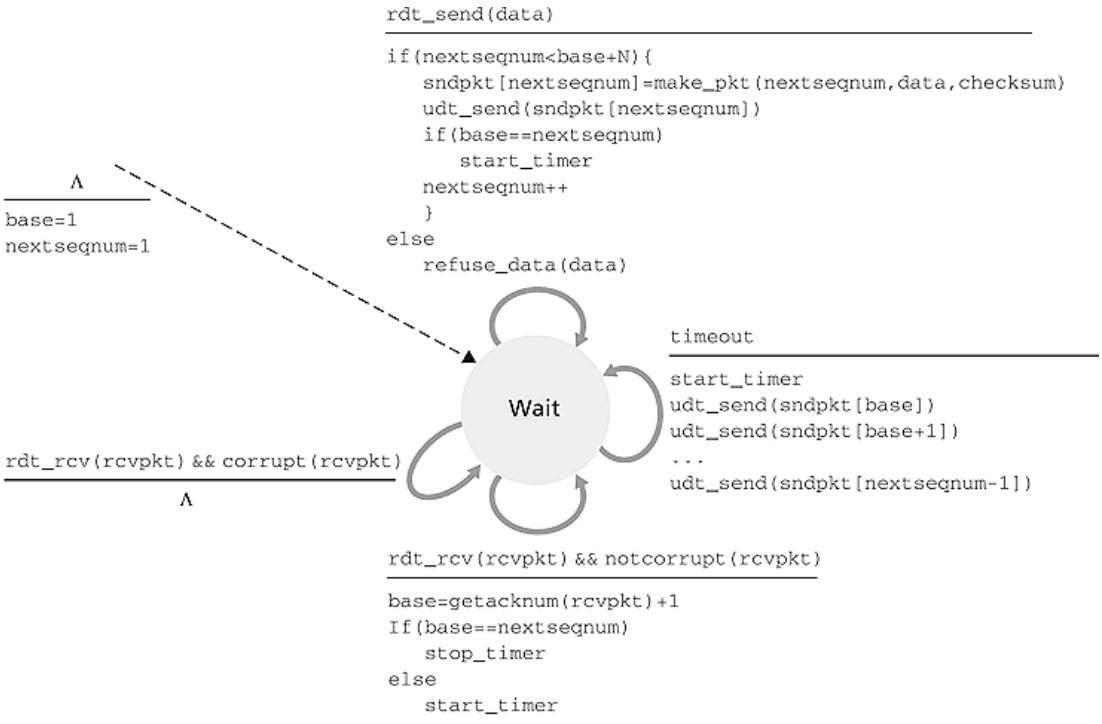


Figure 3.20 ♦ Extended FSM description of GBN sender

La máquina de estados del receptor es:

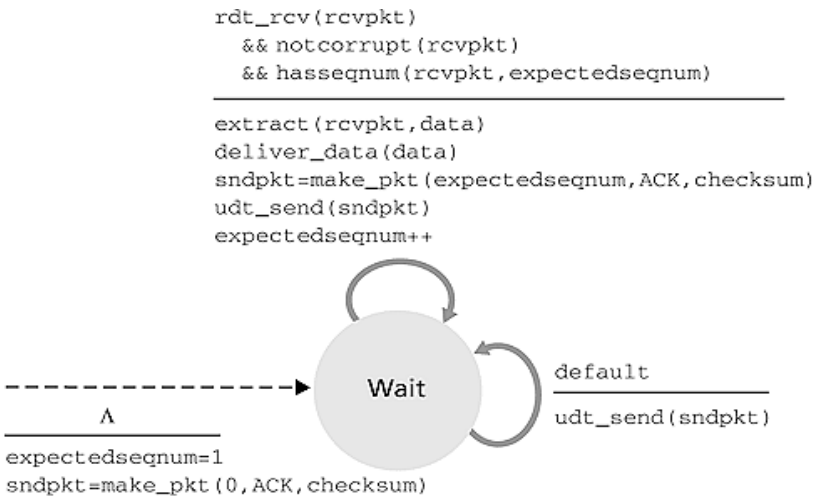


Figure 3.21 ♦ Extended FSM description of GBN receiver

Se pide:

Implementar una API en C/C++, que contenga las funciones definidas anteriormente. Dicha API se linkeditará con programas que utilicen las mencionadas funciones. Deben resolverse los siguientes elementos del protocolo:

- Establecimiento de conexión del *PCT*, generando las estructuras sobre IP requeridas para la conexión real.
- Permitir el intercambio de datos en forma confiable, sin pérdida ni duplicación de datos.
- Desconexión del *PCT*.

Entregable
Se deberá entregar:

1.Implementación:
a)Código fuente.
b)*Makefile* para la compilación del mismo.
c)Instrucciones para su utilización (incluyendo forma de compilación de una aplicación con *PCT*).

2. Análisis del comportamiento del protocolo PCT, frente a la variación del tamaño de ventana `GBN_WINDOW` del ARQ *Go Back N*. En especial considere el caso N=1 (ARQ Stop&Wait) y el comportamiento ante el crecimiento de N.

3.Indique qué cambios debería realizar al protocolo para que fuera capaz de manejar un stream de bytes (como hace TCP) en lugar de manipular paquetes.

4.Documentación. La documentación debe contener:
a)Descripción exhaustiva en idioma Español del protocolo.
b)Propuesta de implementación del mecanismo de transporte confiable *PCT*.
c)Pseudocódigo detallado y comentado del módulo *PCT* entregado.
d)Respuesta a los puntos 2 y 3, de la presente sección.

La documentación del obligatorio constará de un único archivo tipo PDF de nombre

infOb2grupoXX.pdf que debe incluirse en del archivo **ob2.tar.gz**. Se deberá respetar la numeración de secciones acorde a los requerimientos anteriores, y deberá figurar el número de grupo y el nombre y la cédula de identidad de cada integrante. De acuerdo a la normativa de la Universidad de la República para el manejo de documentos de ofimática, NO se aceptarán otros formatos de informe.
(ver http://www.universidad.edu.uy/renderResource/index/resourceId/4755/siteId/1).

El no cumplimiento de esta normativa invalida la entrega, con la consecuente pérdida del curso.

Insumos

Junto con este documento se entregarán los siguientes insumos para el trabajo:

- Archivo `pct.h` conteniendo el cabezal de las funciones a implementar, constantes pre-establecidas, y formato del cabezal de los segmentos PCT.
- Programas ejemplo que utilizan la API especificada para transmitir información utilizando la implementación de PCT realizada durante este obligatorio.
- Programa ejemplo de envío de datagramas IP sobre *raw socket*, y otro de recepción, con el envío de datos por IP.
- Shell script `netEmulator.sh` que permite introducir distorsión en la red, mediante duplicación y pérdida de datagramas. Esto permitirá probar que el protocolo *PCT* efectivamente implementa el transporte confiable.

Documentación

Notas adicionales:

- Se sugiere la utilización del *wireshark*, para visualizar la información generada en la red por la aplicación. El mencionado software no decodifica *PCT*, pero permitirá analizar el envío de flags y valores del cabezal, así como el contenido de los datos.
- Se sugiere la utilización de *threads*, asegurando la ejecución simultánea de código así como la mutua exclusión.
- Para la comprensión del ARQ a utilizar se recomienda ver [1] (caso *Go-Back-N Protocol-Chapter 3*).

Referencias:

[1] Applets (Computer Networking, James F. Kurose / Keith W. Ross) http://wps.aw.com/aw_kurose_network_4/63/16303/4173752.cw/index.html

[2] raw(7): IPv4 raw sockets - Linux man page: http://linux.die.net/man/7/raw