Actividad Forwarding

Consideraciones:

- Se considera una lectura de la tabla de rutas en orden descendente para poder implementar correctamente round robin.
- La implementación de round robin se realizó mediante una simple lista con las rutas para hacer forwarding, en caso de utilizar una de estas se procede a quitarla de la lista e insertarla inmediatamente al final de esta para así dar prioridad a las posibles rutas alternativas que existan hacia un router y generar la alternación cíclica.

1. `router.py`

Esta módulo contiene la clase `Router`, la cual crea un router con las funcionalidades para manejar y parsear los paquetes IP, asimismo maneja el forwarding de estos paquetes chequeando las rutas de la tabla de rutas otorgada.

2. `router_test.py`

Este código es utilizado para testear las pruebas realizadas en la sección **Pruebas Mini-Internet sin TTL**, crea un router con una dirección y puerto asociados, entregándole adicionalmente la tabla de rutas, la forma de uso es mediante el esquema de comando mostrado en la actividad:

```
$ python3 router_test.py router_IP router_puerto router_rutas.txt
```

3. Pruebas Mini-Internet sin TTL

A continuación, se presentan los resultados a los test indicados en la sección **Pruebas Mini-Internet sin TTL** de la actividad:

- *Usando las rutas del ejemplo 2 de la sección anterior, pruebe qué ocurre si alguien configura mal una de las tablas de rutas y coméntelo brevemente en su informe. Para ello, cambie la configuración de la tabla de rutas del archivo* ***`rutas_R2_v2.txt`*** *por:*

  Al configurar de manera incorrecta la ruta para el router 2 obtenemos el diagrama mostrado en la Figura 1, donde se observa que no es posible establecer una comunicación con el router R3 en la red, esto se demuestra al tratar de enviar un mensaje desde cualquiera de los routers R1 o R2, en tal caso en la ejecución se realiza un loop sin detención para el envío del paquete con ruta hacia R3, este paquete se envía desde R1 hacia R2 (o viceversa) y luego este envía hacia R1 (R2) nuevamente, ocurriendo este proceso indefinidamente.

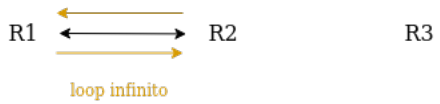

Figura 1: Caso sin ruta en red para R3, loop infinito para envío hacia este.

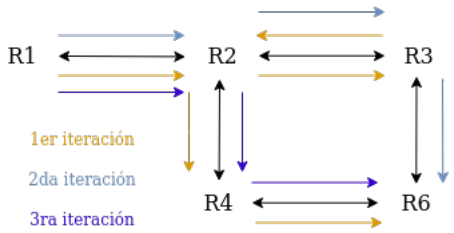

Figura 2: Forwarding con round robin, alternación en rutas para el envío de un paquete desde R1 a R3.

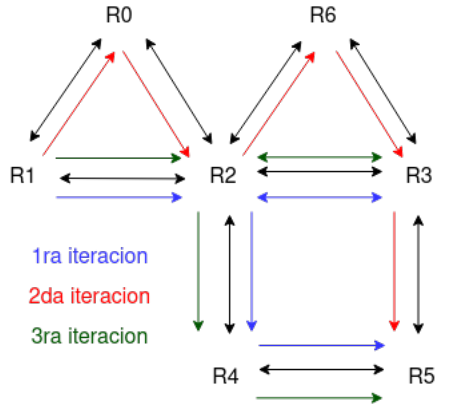

Figura 3: Forwarding paquete desde R1 a R5 en tres iteraciones.

- *Usando las rutas de la estructura del paso 6, pruebe su código enviando paquetes a R1 con destino R5 y vea que su función de rutas efectivamente usa round-robin, es decir que va alternando entre rutas. Haga varias pruebas ¿cuántos saltos dan los paquetes? ¿siempre dan la misma cantidad de saltos?¿cómo se compara la cantidad de saltos que dan los paquetes versus la cantidad de saltos mínima? Anote sus observaciones en el informe.*

  La implementación de round robin efectivamente alterna las rutas de forwarding de paquetes, observándose un patrón repetitivo en la cantidad de saltos que da el paquete para llegar al router de destino R5, en la primera ejecución del comando `netcat` el paquete llega al destino luego de dar 5 saltos, al ejecutar nuevamente el comando el paquete llega luego de 3 saltos, seguidamente llega luego de 3 saltos, posteriormente esta secuencia de cantidad de saltos se repite de manera cíclica. Notemos que la mínima cantidad de saltos que puede dar el paquete antes de llegar al router R5 es 3, como se observa en el diagrama de la Figura 2, lo cual se comprueba pues dado el algoritmo de round robin, el paquete alterna entre las rutas posibles obteniendo en determinados casos una cantidad de 3 saltos hasta llegar al destino.

- *Repita las pruebas del punto 2 utilizando la estructura que se le pidió crear en el Test 2 del paso 7 y escriba sus observaciones en su informe. Añada en su informe los contenidos de los distintos archivos de rutas que debió crear.*

  La implementación de round robin funciona correctamente, no se entrará en detalle al respecto de las rutas que tomaron los paquetes en las iteraciones producidas pues se pueden observar en las Figuras 3 y 4, ademas se encuentran los archivos con las tablas de ruta en la sección **Anexo**, sin embargo para este ultimo caso en que se envía el paquete desde R5 a R1 es necesario mencionar que en la tercera iteración, al comienzo, el paquete salta a R4 para luego volver a R5, esto se produce debido a que R4 ya había enviado un paquete anteriormente a R2 por tanto la única ruta alternada que le queda para enviar el paquete hacia R1 es R5, este mismo proceso ocurre para R3 que obtiene el paquete desde R5 y lo devuelve a R5, para este ultimo enviarlo a R4 y seguir con su envío de manera normal.

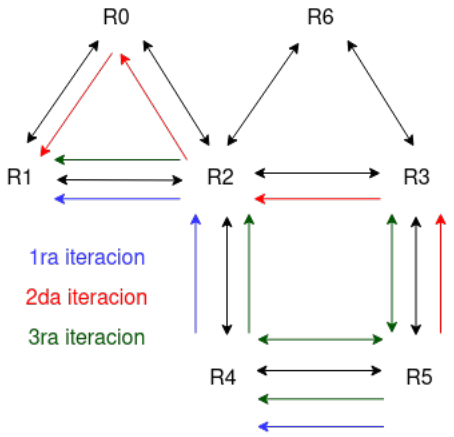

Figura 4: Forwarding paquete desde R5 a R1 en tres iteraciones.

4. Pruebas Mini Internet con TTL

Como consideración para esta parte de la actividad se debe tener en cuenta que el archivo `prueba_router_ttl.py`, creado específicamente para el ítem 4, considera que los parámetros que se entregan son un archivo con los paquetes IP (`ip_packets.txt`), el cual se lee linea por linea, enviando cada uno de estos a la dirección de destino `(routerp_ip, router_port)`.

```
$ python3 prueba_router.py ip_packets.txt router_ip router_port router_route_table.txt
```

Dada esta consideración y que para ejecutar este archivo se debe tener en ejecución previamente los routers apropiados usando el archivo `router_test.py`, se procede a responder las pruebas de la correspondiente sección:

- *Repita la primera prueba de la parte 1 (donde indujo error en la tabla de rutas) y vea qué ocurre. Use un TTL inicial TTL = 10 ¿Qué diferencias observa? Anote sus observaciones brevemente en su informe.*

  Al añadir al header un TTL = 10, los routers comienzan a enviar de manera no satisfactoria el paquete entre el router R1 y R2, como lo observado en la Figura 1, cada uno de estos lo envía 5 veces de manera intercalada hasta que vuelve a recibirlo el router R1, en este punto el TTL a decrementado a 0 por tanto simplemente informa en la consola que se recibió un paquete con TTL = 0 y lo descarta.

- *Considere la configuración de 5 routers vista en la parte 1. Use el código implementado en el paso 4 con un archivo grande (de varias líneas) ¿Qué ocurre con el orden de los paquetes? Anote sus observaciones en el informe.*

  El envío de los paquetes al ejecutar `prueba_router.py` es por orden descendente, sin embargo como estos paquetes luego son forwardeados por el router al que se enviaron y puede caber la posibilidad de que estos paquetes sean devueltos por otro router, se van intercalando la recepción de los paquetes por los routers, aun así estos llegan a los destinos correspondientes.

5. Anexo

A continuación se presentan los códigos para el ultimo ítem de la actividad **Pruebas Mini Internet sin TTL**, cabe mencionar que se modificaron los archivos entregados para el ejemplo de una red con 6 routers, por tanto los archivos tienen los mismo nombres entregados para ese ejemplo.

```
# rutas_R0_v3.txt
127.0.0.1 8881 8881 127.0.0.1 8881
127.0.0.1 8882 8885 127.0.0.1 8882
```

```
# rutas_R1_v3.txt
127.0.0.1 8882 8885 127.0.0.1 8882
127.0.0.1 8880 8885 127.0.0.1 8880

# rutas_R2_v3.txt
127.0.0.1 8881 8881 127.0.0.1 8881
127.0.0.1 8883 8885 127.0.0.1 8883
127.0.0.1 8883 8885 127.0.0.1 8884
127.0.0.1 8880 8881 127.0.0.1 8880
127.0.0.1 8883 8886 127.0.0.1 8886

# rutas_R3_v3.txt
127.0.0.1 8881 8882 127.0.0.1 8882
127.0.0.1 8881 8882 127.0.0.1 8885
127.0.0.1 8884 8885 127.0.0.1 8882
127.0.0.1 8884 8885 127.0.0.1 8885
127.0.0.1 8881 8882 127.0.0.1 8886

# rutas_R4_v3.txt
127.0.0.1 8881 8883 127.0.0.1 8882
127.0.0.1 8881 8883 127.0.0.1 8885
127.0.0.1 8885 8885 127.0.0.1 8885

# rutas_R5_v3.txt
127.0.0.1 8881 8884 127.0.0.1 8884
127.0.0.1 8881 8884 127.0.0.1 8883

# rutas_R6_v3.txt
127.0.0.1 8880 8882 127.0.0.1 8882
127.0.0.1 8883 8885 127.0.0.1 8883
```