# Práctica Transporte: TCP y UDP

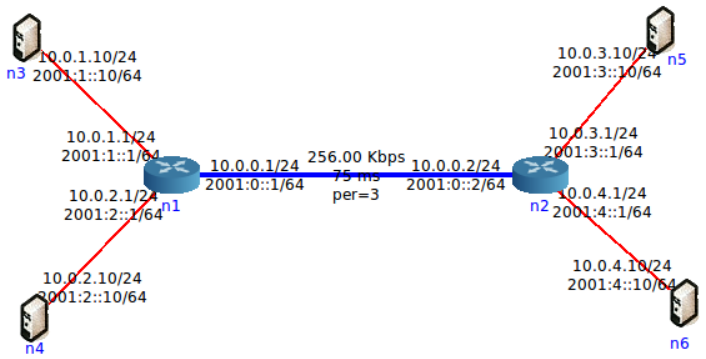


Figura 1: Configuración de la red

1. Configurar una topología de acuerdo a la figura anterior.
2. Ejercicios UDP:
   *a*) Levantar un servicio UDP con la herramienta `nc` en el host **n5** y enviar información desde **n3** usando el mismo comando. Ver el estado de los sockets en ambos extremos. ¿Qué significa el estado `ESTABLISHED` en UDP?
   *b*) Levantar en el super daemon inetd el servicio udp `echo` y probar el cliente `nc` contra este servicio. Inspeccionar el estado.
   *c*) Enviar información desde **n3** a **n5** a un port UDP donde no existe un proceso esperando por recibir datos. ¿Cómo notifica el stack TCP/IP de este hecho? Investigue la herramienta **traceroute** que ports utiliza y como usa estos mensajes.
   *d*) Ver la posibilidad de generar paquetes UDP con el port origen 0 (cero) y de forma broadcast.
   *e*) Para las pruebas anteriores capturar tráfico y ver el formato de los datagramas UDP y como se encapsulan en IP.

3. Ejercicios TCP:

   *a*) En el nodo **n5** levantar con el super daemon inetd algunos servicos extras tcp `echo`, `discard` y otros. Chequear los servicios TCP y UDP activos.

   *b*) Desde el nodo **n3** realizar una conexión TCP utilizando el programa `telnet` o `nc`, al servicio `discard`, y al `echo`. Capturar el tráfico con la herramienta `tcpdump` o `wireshark` y analizar la cantidad de segmentos, los flags utilizados y las opciones extras que llevan los enabezados tcp.

   *c*) Sin cerrar las conexiones chequear los servicios activos y ver los estados.

   *d*) Cerrar las conexiones y ver el estado de los servicios en ambos lados. ¿En que estado queda el que hace el cierre activo?

   *e*) Observando la captura indicar la cantidad de segmentos y los flags utilizados. ¿Con cúantos segmentos se cerró la conexión? ¿Existen otras variantes?

   *f*) Hacer un diagrama de los segmentos intercambiados con los números de secuencia absolutos para una de las dos sesiones tcp.

   *g*) Realizar una conexión mediante `nc` indicando un port específico para el cliente. Luego cerrar la conexión desde el cliente e intentar habrirla nuevamente. ¿En que estado esta el socket? Investigar valor del 2MSL en la plataforma sobre la cual esta haciendo los tests.

   *h*) Realizar varias conexiones simultáneas al servicio tcp `echo`. Investigar los estados. Cerrar la conexiones.

   *i*) Intentar levantar un servicio en **n5** con `nc` en el port usado por el servicio `echo`. ¿Qué sucede?. Levantar un servicio en otro port y ver los estados. Realizar varias conexiones desde **n3** y volver a ver los estados en ambos host (Nota: Para que `nc` puede permitir varias conexiones una trás otra puede ser necesaria la opción `-k`, de acuerdo a la versión). Ver que sucede si se realizan más de 3 conexiones. Ver los estados.

   *j*) Generar un estado `CLOSE_WAIT` en el lado del cliente. Primero generando la conexión, luego enviando a dormir el proceso cliente (por ejemplo con CTRL+Z) y luego matando el proceso del lado del servidor. ¿En que estado queda el extremo del servidor?

¿Cuándo pasará a `TIME_WAIT` y que extremo? ¿Qué significa el estado `CLOSE_WAIT` y `TIME_WAIT`? Llevar al estado `TIME_WAIT` antes del 2MSL.

*k*) Generar un estado `CLOSE_WAIT` en el lado del servidor. (Nota: Primero generando la conexión, luego enviando a dormir el proceso servidor y luego matando el proceso cliente).

*l*) Intentar realizar una conexión a un port donde no exista un servicio activo en **n5**. ¿Qué flags activos, número de secuencia y ACK tiene el segmento de respuesta? Comparar como lo resuleve UDP.

*m*) Ver el funcionamiento del "Timeout of Connection Establishment". Una forma puede ser tratando de alcanzar un host que es inalcanzable. Para que esto funcione los routers no deben generan un ICMP de host inalcanzable o filtrar los mensajes ICMP, con la herramienta `iptables`. Inspeccionar el estado en que queda el cliente. Ver los parámetros del kernel que regulan cuantos SYN trata de enviar y como maneja los timers, bajar el valor a 1 y volver a probar.

*n*) Generar un Half-close y ver el cierre cuantos segmentos son utilizados.

*ñ*) Investigar cual es el valor del MSS que utiliza. Cambiar el MSS que negocia, por ejemplo modificando el MTU.

*o*) Ver el funcionamiento del control de flujo TCP mediante la variable Win. Para probar el funcionamiento consultar el documento de ayuda. Probar con SACK, sin SACK, con escalado de ventana y sin escalado de ventana.