Práctica 4 - Capa de Transporte

Revisión 2.0

1) ¿Cuáles son las funciones de la capa de transporte en el modelo OSI? ¿Cuáles implementa TCP y cuáles UDP? ¿Existen otros protocolos de transporte en el stack TCP/IP? ¿Cuáles son los más utilizados?
2) Indique que asumen TCP y UDP con respecto a los servicios provistos por la capa de Red/Internetworking sobre la cual se implementan. ¿Cuál es el campo del datagrama IP y los valores utilizados para diferenciarlos en la multiplexación (Hint: buscar en `/etc/protocols`)?
3) La PDU de la capa de transporte es nombrada de forma genérica segmento. Indique para los protocolos indicados en los puntos anteriores cómo se llaman específicamente las unidades de datos y realice un diagrama de su estructura.
4) Responder:
   a) ¿Qué sucede si llega un datagrama IPv6 a un host y éste no tiene implementado IPv6?
   b) ¿Qué sucede si llega un segmento TCP a un host que no tiene soporte de este protocolo de transporte?
   c) ¿Qué sucede si llega un segmento TCP a un host y éste no tiene un proceso esperando en el puerto destino indicado? ¿Qué sucede si el mensaje es UDP?

5) ¿En qué se diferencian los checksum de UDP, TCP, IPv4 e IPv6?

6) Un proceso desde un host “X” inicia una conexión TCP, por ejemplo mediante el comando telnet, hacia un servidor “Y”. Al mismo tiempo otro proceso inicia otra conexión TCP hacia el mismo servicio. En ninguna de las conexiones existen transferencia de datos de usuario y el proceso servidor está activo.
   a) Indicar números de puerto origen y destino de la primera conexión.
   b) Indicar números de puerto origen y destino para la segunda conexión suponiendo que proviene de un host diferente que el primero.
   c) Indicar números de puerto origen y destino para la segunda conexión suponiendo que proviene del mismo host “X”.
   d) Indicar la cantidad de segmentos TCP transmitidos.
   e) ¿Qué sucede con la segunda conexión, debe esperar hasta que termine la primera para ser atendida?
   f) Indicar los “flags” que se verán en los segmentos TCP transmitidos.

7) Dada la siguiente salida del comando (`netstat o ss`), responder:

```
# netstat -atun
Active Internet connections (servers and established)
Proto R-Q S-Q Local Address      Foreign Addres     State
tcp     0   0   127.0.0.1:43695    0.0.0.0:*          LISTEN
tcp     0   0   127.0.0.1:7634     0.0.0.0:*          LISTEN
tcp     0   0   0.0.0.0:22         0.0.0.0:*          LISTEN
tcp     0   0   127.0.0.1:631      0.0.0.0:*          LISTEN
tcp     0   0   0.0.0.0:17500      0.0.0.0:*          LISTEN
tcp     0   0   0.0.0.0:8000       0.0.0.0:*          LISTEN
tcp     0   0   127.0.0.1:40963    0.0.0.0:*          LISTEN
tcp     0   0   127.0.0.1:2628     0.0.0.0:*          LISTEN
tcp     0   0   10.168.1.163:51222 173.194.42.54:443 ESTABLISHED
tcp     0   0   127.0.0.1:43695    127.0.0.1:45547    ESTABLISHED
tcp     1   0   10.168.1.163:50120 191.189.39.14:80   CLOSE_WAIT
tcp     0   0   127.0.0.1:8000     127.0.0.1:35250    ESTABLISHED
tcp     0   0   127.0.0.1:35250    127.0.0.1:8000     ESTABLISHED
tcp     0   1   10.168.1.163:45123 1.1.1.1:9000       SYN_SENT
tcp     0   0   10.168.1.163:40123 99.59.148.17:443   TIME_WAIT
tcp     1   1   10.168.1.163:58432 104.154.94.81:443  LAST_ACK
tcp     0   0   10.168.1.163:36121 138.160.162.116:7  ESTABLISHED
tcp     1   0   10.168.1.163:58433 205.154.94.81:443  CLOSE_WAIT
tcp     0   0   10.168.1.163:60123 91.189.89.76:443   ESTABLISHED
tcp     0   0   10.168.1.163:42133 173.194.42.33:443  ESTABLISHED
tcp     0   0   10.168.1.163:50121 199.16.156.83:443  ESTABLISHED
tcp     0   0   127.0.0.1:45547    127.0.0.1:43695    ESTABLISHED
tcp6    0   0   :::80              :::*               LISTEN
tcp6    0   0   :::22              :::*               LISTEN
tcp6    0   0   ::1:631            :::*               LISTEN
udp     0   0   0.0.0.0:68         0.0.0.0:*
udp     0   0   0.0.0.0:69         0.0.0.0:*
udp     0   0   0.0.0.0:59744      0.0.0.0:*
udp     0   0   0.0.0.0:17500      0.0.0.0:*
udp     0   0   0.0.0.0:5353       0.0.0.0:*
udp6    0   0   :::59695           :::*
udp6    0   0   :::5353            :::*
```

a) ¿Cuántas conexiones TCP hay establecidas?
b) ¿Cuántas conexiones TCP hay establecidas solo como cliente y cuántas como servidor?
c) ¿Cuántas conexiones TCP están aún pendientes por establecerse?
d) ¿A qué servicios bien conocidos (puertos 0 a 1023) tiene el host conexiones TCP establecidas?
e) ¿Qué servicios bien conocidos (TCP y UDP) tiene corriendo el host (esperando por recibir información)?
f) ¿Cuántas conexiones están en proceso de cierre?
g) ¿Cuáles conexiones TCP tuvieron el cierre iniciado por el host local y cuáles por el remoto?
h) ¿En qué puertos podría recibir datos sobre IPv6 desde otros hosts, tanto TCP6 como UDP6?

8) Analizar la salida de comando del anexo I de la práctica y responder:

   a) ¿Cuántas conexiones TCP hay establecidas sobre IPv4?
   b) ¿Cuántas conexiones TCP hay establecidas sobre IPv6?
   c) ¿Cuántas conexiones TCP sobre IPv4 hay establecidas como cliente con hosts remotos?
   d) ¿Cuántas conexiones TCP sobre IPv6 hay establecidas como cliente con hosts remotos?
   e) ¿Cuántas conexiones TCP6 están aún pendientes por establecerse?
   f) Investigue qué protocolo UDP sobre IPv6 el cliente tiene mensajes enviados que aparecen como conexiones “establecidas”. ¿A quién pertenece la IPv6: `2800:3f0:4003:c01`?
   g) ¿Cuántas conexiones TCP sobre IPv4 y cuántas sobre IPv6 están en proceso de cierre?
   h) ¿Cuáles conexiones TCP (sobre IPv4 e IPv6) tuvieron el cierre iniciado por el host local y cuáles por el remoto?

9) UDP es un protocolo no orientado a conexión, ¿Qué significa la siguiente salida? (Puede saltear y directamente realizar el ejercicio UDP entregable, punto 24)

```
# netstat -atun | grep 8000
udp        0      0 127.0.0.1:5000  127.0.0.1:8000  ESTABLISHED
udp        0      0 0.0.0.0:8000    0.0.0.0:*
```

10) Armando la topología de la figura 1 con el simulador, realizar los siguientes puntos (Puede saltearlo y directamente realizar el ejercicio TCP entregable, punto 25):

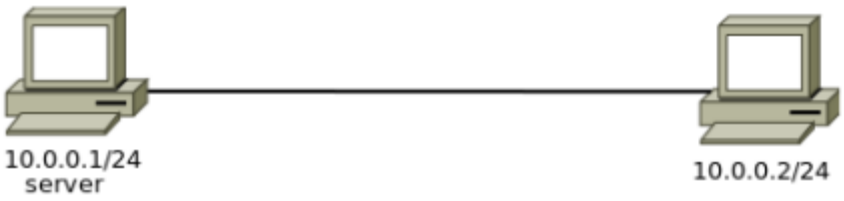

Figura 1: TCP Cliente/Servidor

   a) Utilizando la herramienta nc(netcat) levantar un servidor TCP escuchando en el port 8001.
   b) Conectarse desde otro host al port 8001 utilizando la herramienta telnet o el mismo nc en modo cliente.
   c) Inspeccionar el estado de las conexiones con el comando netstat en ambos equipos
   d) Cerrar la conexión terminando el proceso cliente y ver cuáles son los estados que quedan las conexiones.

e) Intentar nuevamente la conexión utilizando el mismo port origen con nc usando la opción -p. ¿Cuál es el resultado?

f) Volver a correr el servidor y lograr una nueva conexión desde el cliente. Generar datos y capturar tráfico con la herramienta wireshark/tcpdump. Inspeccionar los números de secuencia con los cuales se generan los segmentos.

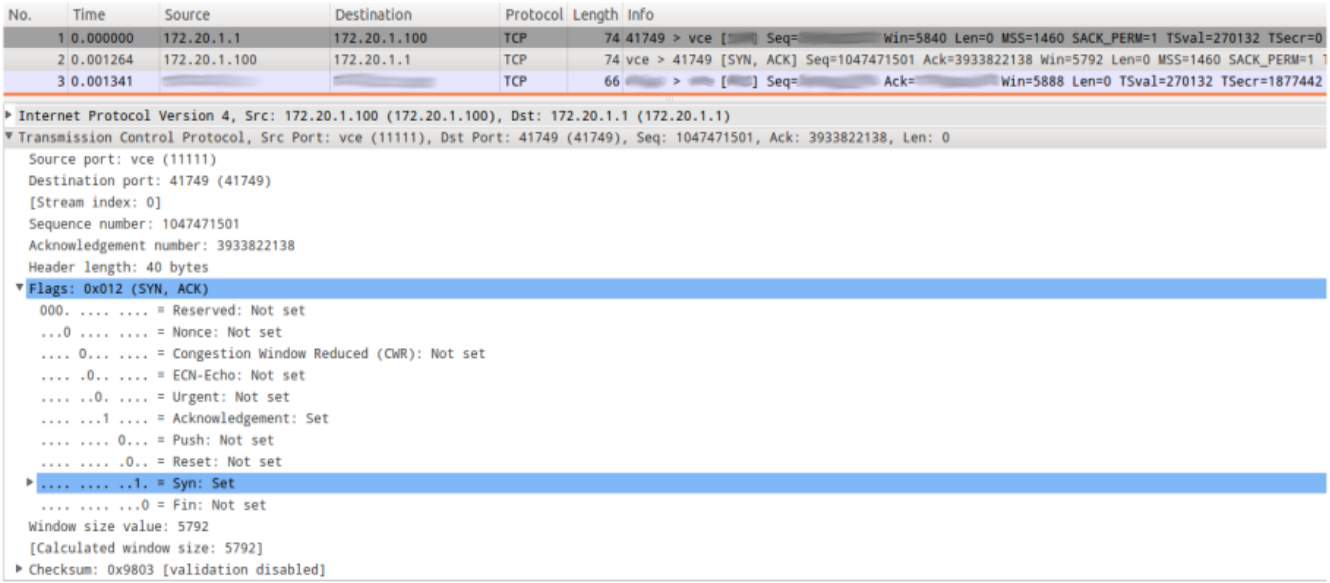

Figura 2: Captura TCP 1

11) De acuerdo a la captura de la figura 2 indicar los valores de los campos que están difuminados (“blur”).

12) De acuerdo a la siguiente salida del comando netstat responder:

```
# netstat -atunp
Active Internet connections (servers and established)
Proto . Local Address   Foreign Addres   State
tcp     0.0.0.0:22      0.0.0.0:*          LISTEN  999/sshd
tcp     127.0.0.1:631   0.0.0.0:*          LISTEN  11351/cupsd
tcp     13.10.0.14:5217 91.189.95.73:443   CLOSE_ 10418/remot
tcp     13.10.0.14:22   11.191.89.18:49357 ESTABL 11696/sshd:
tcp6    :::80           :::*               LISTEN  1847/apache2
tcp6    :::22           :::*               LISTEN  999/sshd
tcp6    ::1:631         :::*               LISTEN  11351/cupsd
```

a) ¿Cuál es el proceso asignado al port de servicios de impresión de red? (Hint: buscar en `/etc/services`).

b) ¿Se podría enviar a imprimir al servidor de impresión de este equipo a través de la red desde otros equipos?
c) ¿Qué servicios pueden recibir conexiones sobre IPv6?

13) Dadas las salidas de los siguientes comandos ejecutados en el cliente y el servidor, responder:

```
srv# netstat -atun | grep 4500
tcp    0      0  0.0.0.0:4500      0.0.0.0:*           LISTEN
tcp    0      0  157.0.0.1:4500    157.0.11.1:52843 SYN_RECV

cli# netstat -atun | grep 4500
tcp    0      1  157.0.11.1:52843 157.0.0.1:4500    SYN_SENT
```

a) ¿Qué paquetes llegaron y cuáles parecen que se están perdiendo en la red?
b) ¿Cómo quedará el estado en ambos lados si no logra establecerse la conexión?

14) Dado el diagrama de intercambio de segmentos de la figura 3 y suponiendo que se aplica el mecanismo de ARQ Go-Back-N numerando por bytes, responder:

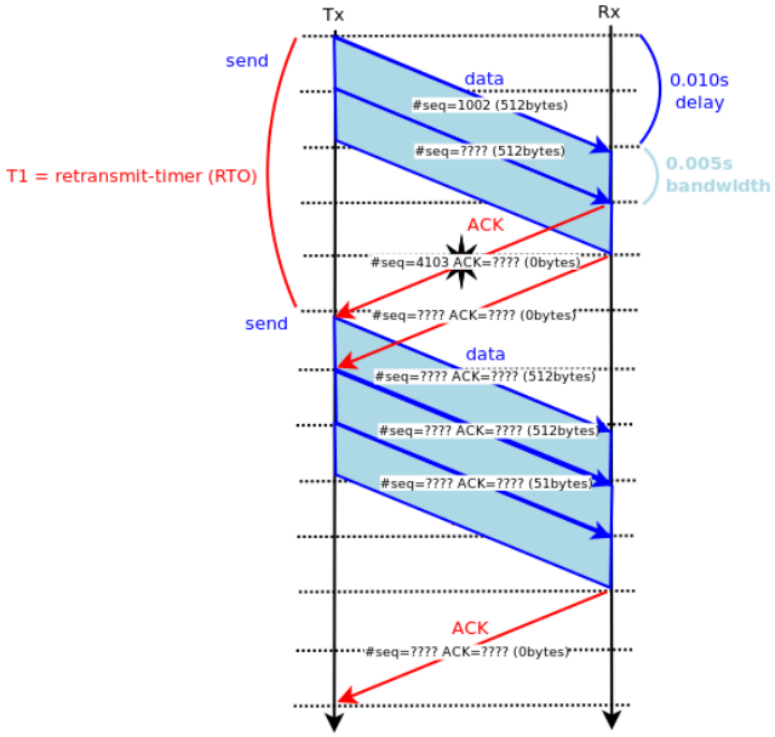

Figura 3: Control de Errores Go-Back-N

a) ¿Cuál es el ancho de banda digital (throughput) que se está obteniendo?

b) ¿Cuál debería ser el valor óptimo del T1 (RTO, Timer de retransmisión) suponiendo que el delay se mantiene constante? ¿Cómo se soluciona cuando el delay no es constante?
c) ¿Cuál sería un tamaño de ventana óptimo?
d) ¿Cuántos bytes lograron transferirse de forma efectiva?
e) Suponiendo que el primer ACK se pierde, completar los valores indicados con signos de interrogación.

15) El host A desea establecer una sesión TCP con el host B. A selecciona un ISN de 50430, MSS de 1400 bytes y un tamaño de ventana de 64KB; B, por su parte, tiene un ISN de 68900, MSS de 1000 bytes y un tamaño de ventana de 32KB. Responder:

a) ¿Cómo sería el intercambio de mensajes para establecer la sesión?
b) ¿Cuántos segmentos de tamaño máximo le puede enviar A a B sin esperar un ACK? ¿Y B a A? (Considere primero que no se aplica control de congestión tradicional y luego aplicándolo).

16) Se tiene una conexión TCP entre dos hosts, A y B. B ya recibió correctamente de A todos los bytes hasta el byte 225. A se conecta desde el port 1986 al port 22 de B. Responder:
a) ¿Qué valor indicará B en el campo ACK para reconocer esta condición? ¿Qué ports utilizará?
b) Si A le envía dos segmentos a B de 100 y 120 bytes respectivamente, ¿cuáles son los números de secuencia de los dos segmentos?
c) Si B envía un ACK por cada segmento recibido, ¿cuáles serían los números de ACKs seteados? ¿Y si envía uno solo por los dos segmentos?
d) Si el segundo segmento de datos (el de 120 bytes) arriba antes que el primer segmento, en el ACK del primer segmento recibido, ¿cuál es el valor del número de ACK?

17) Dada la sesión TCP de la figura 4 completar los valores marcados con un signo de interrogación.

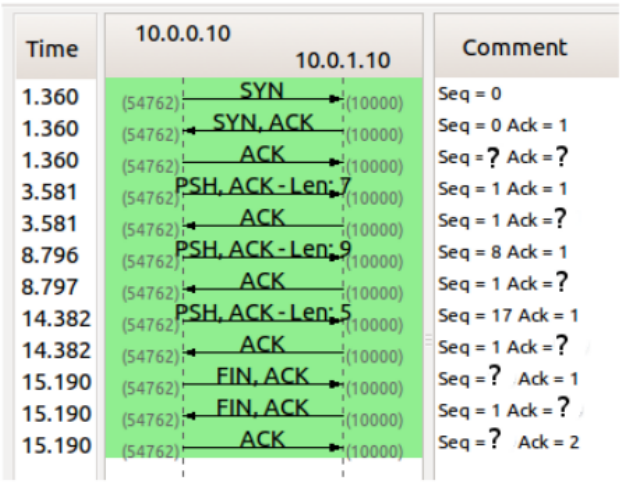

Figura 4: Sesión TCP

18) A partir de las capturas tcp-init.pcap y tcp-init2.pcap indicar qué opciones se intercambiaron en TCP. Indicar los ISN (Initial Sequence Number) de cada extremo y el port del cliente y del servidor. Investigar en la captura `tcp-init3.pcap` el intercambio del MSS (Maximum Segment Size) y comparar la diferencia con las anteriores, ¿a qué puede deberse?

19) Se tiene un enlace no congestionado con un RTT de 10ms. La ventana de recepción es de 12KB, y el cwnd = IW = 1 × M SS, MSS es de 1KB. Si el SSTHRESH iniciales de 16kB y el emisor envía datos continuamente, ¿cuánto tiempo le toma a la ventana utilizada, min(cwnd, rwnd), alcanzar su máximo? Hacer el diagrama.

20) Suponga que la ventana de congestión de TCP está seteada en 18KB y que, en ese momento, se vence el RTO de una transmisión. Si el tamaño máximo del segmento es 1KB, ¿qué tan grande será la ventana si las próximas 4 ráfagas de transmisión son exitosas?

21) Observar el gráfico 5, que ejecuta TCP Reno y responder:

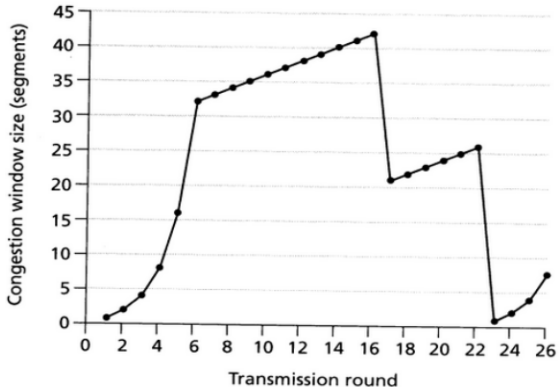

Figura 5: TCP Reno - Control de Congestión

a) ¿Cuáles son los intervalos en los que se ejecuta Slow-Start?
b) ¿Cuáles son los intervalos en los que se ejecuta Congestion Avoidance?

c) ¿Qué evento se produce en el momento 16? ¿Y en el 22?
d) ¿Cuál es el valor de la variable (SSTHRESH) SSThreshold inicialmente? ¿Y en los momentos 18 y 24?

22) A partir de las capturas **tcp*.pcap.gz** indicar en cuáles y cuándo se detecta que se activó el control de flujo y en cuáles el control de congestión.

23) NAT/NAPT: Dado el diagrama de la figura 6 indicar como quedaría la tabla de NAT del router n1 si este hace “overloading” con su única IP pública y se generan las dos conexiones TCP desde los hosts n2 y n3 hacia el servidor s4.

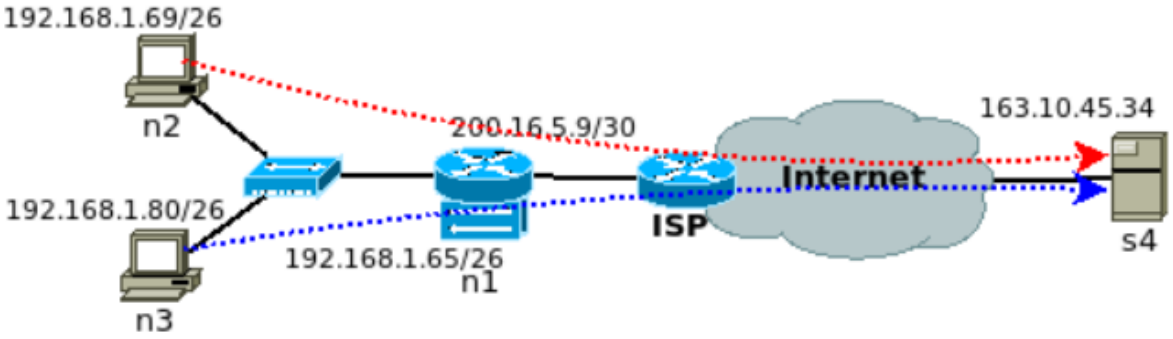

Figura 6: Diagrama de NAPT

24) Ejercicios UDP a entregar. Correr sobre la topología de la figura 7:
a) Levantar un servicio UDP con la herramienta nc (netcat) en el host n9 y enviar información desde n13 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 o similar con el servicio UDP echo y probar el cliente nc contra este servicio. Inspeccionar el estado. Ver de generar datos hacia el “servidor” desde más de un Nodo.
c) Enviar información desde n13 a n9 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 cómo usa estos mensajes (Ver ejercicio de IP con ruteo estático).
d) Para las pruebas anteriores capturar tráfico y ver el formato de los datagramas UDP y como se encapsulan en IP.

25) Ejercicios TCP a entregar. Correr sobre la topología de la figura 8 (continuación de lo que venían realizando en la topología en la práctica anterior):
a) En el nodo n9 levantar con el super daemon inetd algunos servicios extras, como tcp echo, discard y otros. Chequear los servicios TCP y UDP activos.
b) Desde el nodo n13 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 encabezados tcp.

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

d) Generar nuevas conexiones hacia el nodo n9 e inspeccionar los estados. Por ejemplo realizar varias conexiones simultáneas al servicio tcp echo desde el mismo origen y desde otros nodos.

e) Intentar generar conexiones a un puerto donde no existe un proceso esperando por recibir datos. ¿Cómo notifica TCP de este hecho (ver flags)?

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

g) Observando la captura indicar la cantidad de segmentos y los flags utilizados. ¿Con cuántos segmentos se cerró la conexión? ¿Existen otras variantes de cierre?

h) Hacer un diagrama de los segmentos intercambiados con los números de secuencia absolutos para una de las sesiones TCP (Se puede usar la herramienta wireshark u otra).

i) Alternativo: Realizar una conexión mediante nc indicando un puerto específico para el cliente. Luego cerrar la conexión desde el cliente e intentar abrirla nuevamente. ¿En qué estado está el socket? Investigar valor del 2MSL en la plataforma sobre la cual está haciendo los tests.

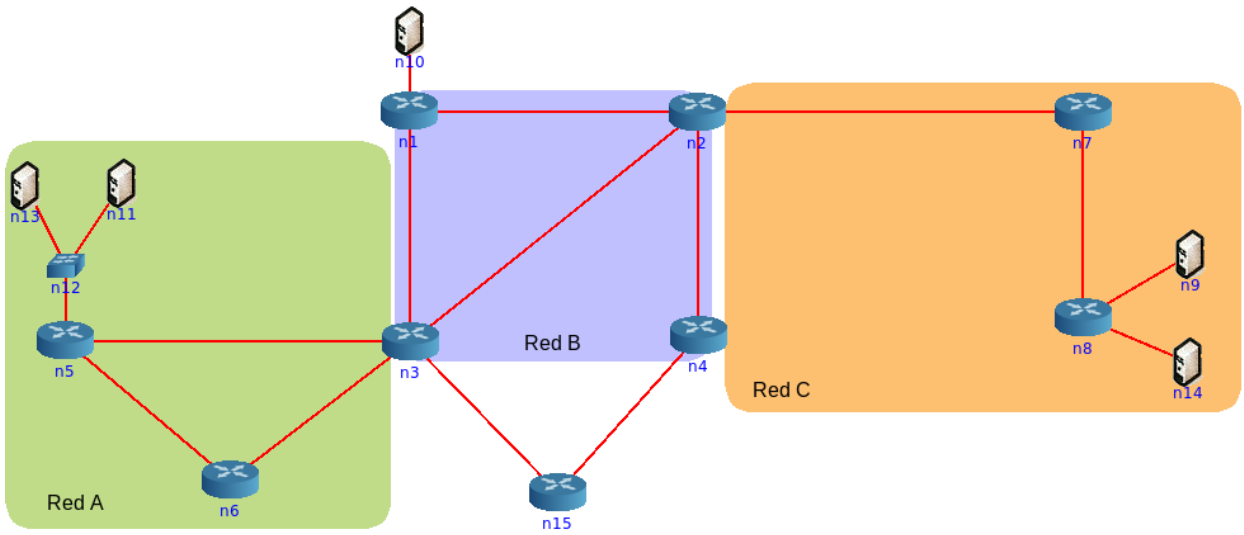

Figura 7: Diagrama de topología a entregar

Anexo I

```
# netstat -atun
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.1:7634          0.0.0.0:*               LISTEN
tcp        0      0 127.0.1.1:53            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:17500           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:445             0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:17600         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:17603         0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:2628          0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:139             0.0.0.0:*               LISTEN
tcp        1      0 163.10.41.9:34857       91.189.94.41:80         CLOSE_WAIT
tcp        0      0 163.10.41.9:41763       163.10.10.28:443        TIME_WAIT
tcp        1      1 163.10.41.9:40580       204.154.94.81:443       LAST_ACK
tcp        0      0 163.10.41.9:44625       54.192.225.90:443       ESTABLISHED
tcp       38      0 163.10.41.9:43850       54.192.226.8:443        CLOSE_WAIT
tcp        1      0 163.10.41.9:36892       162.213.33.241:80       CLOSE_WAIT
tcp        0      0 163.10.41.9:45182       108.160.167.155:443     ESTABLISHED
tcp        0      0 163.10.41.9:35809       64.233.186.132:443      ESTABLISHED
tcp       38      0 163.10.41.9:37641       108.160.172.193:443     CLOSE_WAIT
tcp        1      0 163.10.41.9:44016       162.213.33.242:80       CLOSE_WAIT
tcp       38      0 163.10.41.9:33074       54.230.227.83:443       CLOSE_WAIT
tcp       38      0 163.10.41.9:57093       107.21.218.91:443       CLOSE_WAIT
tcp       38      0 163.10.41.9:50962       54.230.225.130:443      CLOSE_WAIT
tcp        0      0 163.10.41.9:41769       163.10.10.28:443        TIME_WAIT
tcp        0      0 163.10.41.9:36768       64.233.186.106:443      ESTABLISHED
tcp        0      0 163.10.41.9:41777       163.10.10.28:443        ESTABLISHED
tcp        1      0 163.10.41.9:44018       162.213.33.242:80       CLOSE_WAIT
tcp       38      0 163.10.41.9:60490       54.83.193.58:443        CLOSE_WAIT
tcp        0      0 163.10.41.9:33076       64.233.186.188:5228     ESTABLISHED
tcp        1      0 163.10.41.9:44004       162.213.33.242:80       CLOSE_WAIT
tcp       38      0 163.10.41.9:53031       54.230.227.249:443      CLOSE_WAIT
tcp        0      0 163.10.41.9:49591       162.159.246.178:443     ESTABLISHED
tcp        1      0 163.10.41.9:44017       162.213.33.242:80       CLOSE_WAIT
tcp        0      0 163.10.41.9:52814       199.16.158.17:443       ESTABLISHED
tcp        0      0 163.10.41.9:34323       199.96.57.6:443         TIME_WAIT
tcp        1      0 163.10.41.9:40581       204.154.94.81:443       CLOSE_WAIT
tcp       70      0 163.10.41.9:34097       91.189.92.10:443        CLOSE_WAIT
tcp        0      0 163.10.41.9:38944       163.10.0.111:993        ESTABLISHED
tcp        0      0 163.10.41.9:38943       163.10.0.111:993        ESTABLISHED
tcp       38      0 163.10.41.9:57172       108.160.172.1:443       CLOSE_WAIT
tcp        0      0 163.10.41.9:40564       163.10.0.195:443        ESTABLISHED
tcp       38      0 163.10.41.9:38935       108.160.172.193:443     CLOSE_WAIT
tcp        1      0 163.10.41.9:44019       162.213.33.242:80       CLOSE_WAIT
tcp       38      0 163.10.41.9:37373       199.47.217.97:443       CLOSE_WAIT
tcp        1      0 163.10.41.9:44003       162.213.33.242:80       CLOSE_WAIT
tcp       38      0 163.10.41.9:57732       108.160.172.236:443     CLOSE_WAIT
tcp        1      0 163.10.41.9:44023       162.213.33.242:80       CLOSE_WAIT
tcp        0      0 163.10.41.9:49403       64.233.190.138:443      ESTABLISHED
tcp       38      0 163.10.41.9:56534       199.47.217.65:443       CLOSE_WAIT
```

```
tcp        0      0 163.10.41.9:40566       163.10.0.195:443        ESTABLISHED
tcp        0      0 163.10.41.9:38933       163.10.0.111:993        ESTABLISHED
tcp        0      0 163.10.41.9:38942       163.10.0.111:993        ESTABLISHED
tcp        0      0 163.10.41.9:40565       163.10.0.195:443        ESTABLISHED
tcp        1      0 163.10.41.9:60928       91.189.94.25:80         CLOSE_WAIT
tcp        1      0 163.10.41.9:44021       162.213.33.242:80       CLOSE_WAIT
tcp       38      0 163.10.41.9:40911       108.160.173.65:443      CLOSE_WAIT
tcp        0      0 163.10.41.9:34322       199.96.57.6:443         TIME_WAIT
tcp        0      0 163.10.41.9:41738       163.10.10.28:443        TIME_WAIT
tcp        0      0 163.10.41.9:38941       163.10.0.111:993        ESTABLISHED
tcp        0      0 163.10.41.9:34321       199.96.57.6:443         TIME_WAIT
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:631                 :::*                    LISTEN
tcp6       0      0 :::445                  :::*                    LISTEN
tcp6       0      0 :::139                  :::*                    LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
tcp6       0      0 2800:340::643d:a8:56651 2800:240:c:5::be62::443 ESTABLISHED
tcp6       0      0 2800:340::643d:a8:51075 2800:3f0:4003:c01:::443 ESTABLISHED
tcp6       0      0 2800:340::643d:a8:45752 2800:3f0:4003:c01:::443 ESTABLISHED
tcp6       0      0 2800:340::643d:a8:36742 2800:3f0:4003:c01:::443 ESTABLISHED
tcp6       0      1 2800:340::643d:a8:46816 2a03:2880:f000:1:f:3333 SYN_SENT
tcp6       0      1 2800:340::643d:a8:59903 2800:3f0:4003:c00:::442 SYN_SENT
tcp6       0      0 2800:340::643d:a8:43912 2a03:2880:f000:1:fa:443 ESTABLISHED
tcp6       1      0 ::1:50787               ::1:631                 CLOSE_WAIT
tcp6       0      1 2800:340::643d:a8:46322 2a03:2880:f000:1:fa:123 SYN_SENT
udp        0      0 127.0.1.1:53            0.0.0.0:*
udp        0      0 0.0.0.0:68              0.0.0.0:*
udp        0      0 0.0.0.0:69              0.0.0.0:*
udp        0      0 163.10.41.9:123         0.0.0.0:*
udp        0      0 127.0.0.1:123           0.0.0.0:*
udp        0      0 0.0.0.0:123             0.0.0.0:*
udp        0      0 163.10.41.63:137        0.0.0.0:*
udp        0      0 163.10.41.9:137         0.0.0.0:*
udp        0      0 0.0.0.0:137             0.0.0.0:*
udp        0      0 163.10.41.63:138        0.0.0.0:*
udp        0      0 163.10.41.9:138         0.0.0.0:*
udp        0      0 0.0.0.0:138             0.0.0.0:*
udp        0      0 0.0.0.0:41409           0.0.0.0:*
udp        0      0 0.0.0.0:631             0.0.0.0:*
udp        0      0 0.0.0.0:17500           0.0.0.0:*
udp        0      0 0.0.0.0:5353            0.0.0.0:*
udp        0      0 0.0.0.0:5353            0.0.0.0:*
udp        0      0 0.0.0.0:43220           0.0.0.0:*
udp6       0      0 2800:340::643d:a8:56356 2800:3f0:4003:c01:::443 ESTABLISHED
udp6       0      0 :::40484                :::*
udp6       0      0 :::44618                :::*
udp6       0      0 2800:340::643d:a8:49210 2800:3f0:4003:c01:::443 ESTABLISHED
udp6       0      0 2800:340::643d:a8:57410 2800:3f0:4003:c01:::443 ESTABLISHED
udp6       0      0 2800:340::76e5:bff::123 :::*
udp6       0      0 fe80::76e5:bff:fe09:123 :::*
udp6       0      0 2800:340::643d:a8b6:123 :::*
udp6       0      0 ::1:123                 :::*
udp6       0      0 :::123                  :::*
```