

## Teoría de las comunicaciones

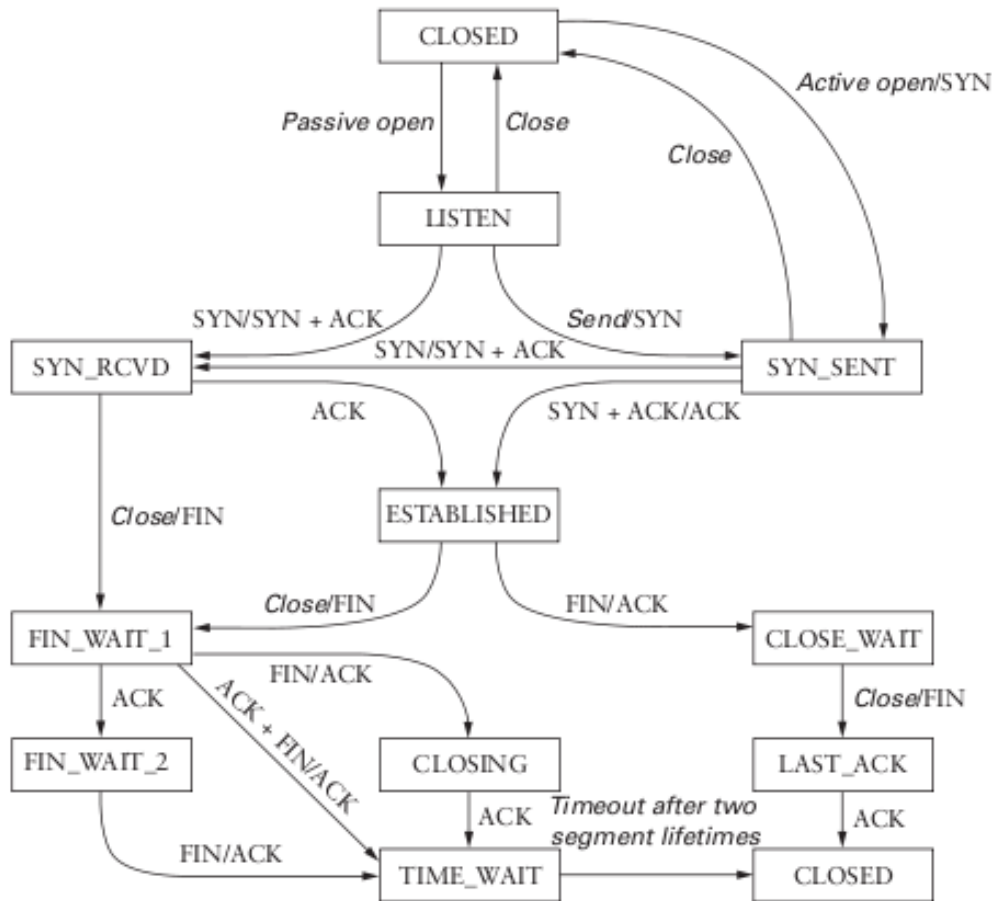
### Práctica 6: End to End

#### Temas

Multiplexación, conexiones, handshakes, estados de una conexión.

#### Definiciones

Máquina de estados de TCP:



#### Ejercicio 1

¿Qué es un puerto? ¿Qué relación guarda con la multiplexación que ofrece el nivel de transporte? ¿Qué sucedería si un protocolo de nivel de transporte no implementara este concepto? ¿Qué diferencias hay entre UDP y TCP con respecto a los servicios que ofrece? ¿En qué escenarios es preferible cada uno?

#### Ejercicio 2

Tanto a nivel de enlace como de transporte, se plantea la necesidad de garantizar comunicaciones confiables: que los datos recibidos representen exactamente toda la información transmitida y en el mismo orden. Para esto, en ambos niveles, se usan estrategias que son similares. Sin embargo, en la capa de transporte ciertos parámetros deben ir cambiando dinámicamente mientras que en la capa de enlace no.

- a. Discuta las estrategias que usan tanto las conexiones de nivel de enlace como las de nivel de transporte.
- b. Discuta las diferencias entre ambos niveles. ¿A qué se deben?

### Ejercicio 3

Suponga un protocolo de red que implementa un servicio orientado a conexión usando circuitos virtuales, garantizando la entrega en orden entre dos hosts, pero no realiza control de errores.

- a. ¿Podrían dividirse los paquetes de una conexión a nivel de transporte por varios caminos de la red de circuitos virtuales?
- b. Si las conexiones del protocolo de red son simplex (i.e., sólo un host escribe y el otro lee), ¿podrían establecerse conexiones de transporte full-duplex (i.e., los dos leen y escriben)?
- c. ¿Sería necesario utilizar números de secuencia en los paquetes de la conexión de transporte?

### Ejercicio 4

Si tuviese que diseñar un sistema de monitoreo de red que deba registrar las conexiones TCP establecidas e identificar al cliente y servidor, utilizando para ello un único paquete dado que posee recursos limitados (no se puede mantener el estado de las conexiones), ¿qué paquete del three-way handshake utilizaría? (Suponiendo que se pueden distinguir todos los paquetes del mismo) ¿Por qué? Justifique su respuesta e indique por qué no es posible utilizar los otros paquetes.

### Ejercicio 5

Dada la siguiente secuencia de segmentos TCP, responder las preguntas que siguen:

#	ORIG	DEST	FLAGS	#SEQ	#ACK	LENGTH
1	1.2.3.4:5678	20.232.1.1:80	S	0	—	0
2	1.1.1.1:2222	3.3.3.3:4444	S	42	—	0
3	20.232.1.1:80	1.2.3.4:5678	SA	10000	1	0
4	1.2.3.4:5678	20.232.1.1:80	A	1	10001	0
5	3.3.3.3:4444	1.1.1.1:2222	SA	54321	43	0
6	1.2.3.4:5678	20.232.1.1:80	A	1	10001	50
7	1.1.1.1:2222	3.3.3.3:4444	A	43	54322	0
8	20.232.1.1:80	1.2.3.4:5678	A	10001	51	0
9	20.232.1.1:80	1.2.3.4:5678	A	10001	51	200
10	1.1.1.1:2222	3.3.3.3:4444	SAU	43	64334	0
11	20.232.1.1:80	1.2.3.4:5678	F	10201	—	0
12	1.2.3.4:5678	20.232.1.1:80	FA	51	10202	0
13	3.3.3.3:4444	1.1.1.1:2222	RA	54321	43	0
14	20.232.1.1:80	1.2.3.4:5678	A	10202	52	0

- a. ¿Cuántas conexiones TCP hay?
- b. Asocie cada segmento con cada una de las conexiones indicando el criterio usado para determinar a qué conexión pertenece un segmento.
- c. Detecte el cierre anómalo de una de las conexiones e indique una posible causa.
- d. Para la otra conexión, detalle los cambios de estado en cada extremo a lo largo de toda la comunicación.

### Ejercicio 6

En un Linux, abrir tres consolas:

- En una, ejecutar `watch netstat -an`.
- En otra, utilizar `netcat` para escuchar en el puerto 5555 de localhost. ¿Qué cambios se observan en la primera?
- En la restante, utilizar nuevamente `netcat` para conectarse a dicho puerto en localhost. ¿Qué cambios se observan ahora en la primera?
- Cerrar esta última conexión y analizar en qué estado se la marca en la primer consola.
- Repetir todo lo anterior usando sockets de Python para ambos procesos.

### Ejercicio 7

A continuación se presenta una lista de paquetes capturados.

#	Origen	Destino	Flags	Seq	ACK	Length
01	3.14.15.92:654	2.71.82.81:823	S	1	—	0
02	2.71.82.81:823	3.14.15.92:654	SA	1	2	0
03	3.14.15.92:654	2.71.82.81:823	A	2	2	0
04	3.14.15.92:654	2.71.82.81:823		2	—	300
05	2.71.82.81:823	3.14.15.92:654	A	2	302	0
06	3.14.15.92:654	2.71.82.81:823		302	—	1000
07	3.14.15.92:654	2.71.82.81:823		302	—	1000
08	2.71.82.81:823	3.14.15.92:654	A	2	1302	0
09	2.71.82.81:823	3.14.15.92:654	F	2	—	0
10	3.14.15.92:654	2.71.82.81:823	A	1302	3	0
11	3.14.15.92:654	2.71.82.81:823	F	1302	—	0
12	2.71.82.81:823	3.14.15.92:654	A	3	1303	0

- Detalle las transiciones de estados de la conexión TCP para cada uno de los hosts, a partir del segmento 9 en adelante.
- Indique qué segmentos se pueden usar para la estimación del RTT.

### Ejercicio 8

Utilizar `nmap` para realizar un SYN scan al gateway de la red local. Mientras tanto, abrir Wireshark y observar cómo los paquetes son enviados.

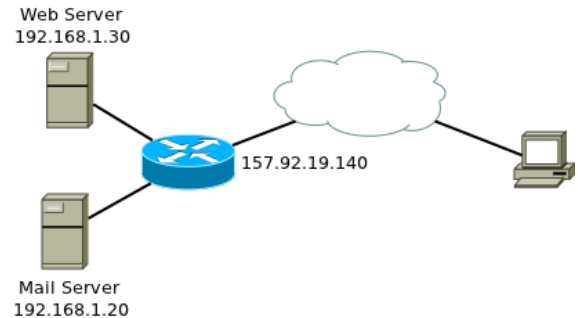
- ¿Usan el mismo puerto origen?
- ¿Qué ocurre con el tamaño de la ventana?
- ¿Y con el número de secuencia inicial?

## Ejercicios de Parcial

### Ejercicio 9

Dado el esquema de la figura:

- Describa las reglas que deberían configurarse en el router de manera tal que el host pueda tener acceso a los servicios que ofrecen el servidor Web (HTTP:TCP/80) y el servidor de Mail (SMTP:TCP/25).
- Suponga que el host envía un mail al servidor de Mail. En un momento dado, el host inicia la liberación de la conexión, pero el último ACK del handshake se pierde en la nube intermedia. Describa la secuencia de eventos hasta que se cierra la conexión.



### Ejercicio 10

El siguiente fragmento muestra parte de la salida de una ejecución de `netstat` en un host con dirección IP 192.168.1.104:

Proto	Local Address	Foreign Address	State
tcp	0.0.0.0:7777	0.0.0.0:*	LISTEN
tcp	192.168.1.104:48600	200.42.33.121:80	ESTABLISHED
tcp	192.168.1.104:34688	69.163.203.254:80	TIME-WAIT
tcp	192.168.1.104:45911	173.194.42.245:443	ESTABLISHED

En el transcurso de los cinco minutos siguientes se capturó el tráfico TCP entrante y saliente de dicho host. La siguiente traza corresponde a la porción final de este tráfico.

No.	Source	Destination	Info
1	192.168.1.247	192.168.1.104	27351 >7777 [SYN] Seq=0 Len=0
2	192.168.1.104	200.42.33.121	GET /index.html HTTP/1.1
3	173.194.42.245	192.168.1.104	443 >45911 [ACK] Seq=4562 Ack=2347 Len=0
4	192.168.1.104	192.168.1.247	7777 >27351 [SYN, ACK] Seq=0 Ack=1 Len=0
5	192.168.1.247	192.168.1.104	27351 >7777 [ACK] Seq=1 Ack=1 Len=0
6	200.42.33.121	192.168.1.104	80 >48600 [ACK] Seq=232 Ack=104 Len=0
7	200.42.33.121	192.168.1.104	HTTP/1.1 200 OK
8	173.194.42.245	192.168.1.104	443 >45911 [FIN, ACK] Seq=4562 Ack=2347 Len=0
9	192.168.1.104	173.194.42.245	45911 >443 [ACK] Seq=2347 Ack=4563 Len=0

- Indicar qué se obtendría si se volviese a correr `netstat` una vez transcurrido este tiempo.
- A partir del paquete 1, el proceso corriendo en 192.168.1.247:27351 está iniciando una conexión con nuestro host. ¿Cuál es el instante más temprano posible en el que el proceso receptor puede enviar datos a su contraparte?

### Ejercicio 11

Desde un host con dirección 192.168.0.1 se captura el siguiente tráfico:

No.	Time	Source	Destination	Info
1	5.487931	192.168.0.100	157.92.27.21	60205 > http [SYN] Seq=0 Len=0
2	5.505790	157.92.27.21	192.168.0.100	http > 60205 [SYN, ACK] Seq=0 Ack=1 Len=0
3	5.505848	192.168.0.100	157.92.27.21	60205 > http [ACK] Seq=1 Ack=1 Len=0

```
4  5.506098  192.168.0.100  157.92.27.21  GET /materias/tc/2012/2c/index_html HTTP/1.1
5  5.532603  157.92.27.21   192.168.0.100  http > 60205 [ACK] Seq=1 Ack=568 Len=0
6  5.930460  157.92.27.21   192.168.0.100  [TCP segment of a reassembled PDU]
7  5.930494  192.168.0.100  157.92.27.21  60205 > http [ACK] Seq=568 Ack=1449 Len=0
8  5.930573  157.92.27.21   192.168.0.100  [TCP segment of a reassembled PDU]
9  5.930576  192.168.0.100  157.92.27.21  60205 > http [ACK] Seq=568 Ack=2897 Len=0
```

- Describir los cambios de estado de ambos extremos de la conexión.
- Explicar en qué estado quedan ambos extremos de la conexión si un software en el host 192.168.0.1 (gateway de 192.168.0.100) reemplaza maliciosamente el segmento 9 por el siguiente:

```
192.168.0.100  157.92.27.21  60205 > http [RST, ACK] Seq=568 Ack=2897 Len=0
```

## Ejercicio 12

Consideremos un host con dirección IP 10.80.1.10. En el instante  $t_0$ , una ejecución de **netstat** en este host arroja lo siguiente:

Proto	Local Address	Foreign Address	State
tcp	0.0.0.0:8080	0.0.0.0:*	LISTEN
tcp	10.80.1.10:35336	157.92.27.21:80	SYN-SENT
tcp	10.80.1.10:51247	200.42.93.137:80	TIME-WAIT
tcp	10.80.1.10:24592	98.139.183.24:80	ESTABLISHED

Al cabo de un momento, en el instante  $t_1$  se vuelve a correr **netstat** y esta vez la salida es así:

Proto	Local Address	Foreign Address	State
tcp	0.0.0.0:8080	0.0.0.0:*	LISTEN
tcp	10.80.1.10:35336	157.92.27.21:80	CLOSE-WAIT
tcp	10.80.1.10:8080	200.11.54.101:32154	ESTABLISHED
tcp	10.80.1.10:24592	98.139.183.24:80	ESTABLISHED

- Proponer un intercambio de paquetes TCP entre el host dado y los demás de manera tal que, partiendo de  $t_0$ , la ejecución de **netstat** en  $t_1$  sea tal como se describe arriba.
- Si se sabe que el último paquete enviado al host 200.42.93.137 se dio en  $t_0$ , indicar una cota inferior para  $t_1$  expresada en términos de  $t_0$ .

## Ejercicio 13

La técnica de port knocking permite abrir puertos dinámicamente en un sistema operativo a partir de sucesivos intentos de conexión a una secuencia de puertos TCP preestablecidos. Supongamos que un host con dirección IP 192.168.0.100 implementa esta técnica y la secuencia de puertos elegida para abrir el puerto 2000 es 5000, 10500, 20000. En otras palabras, esto significa que cualquier host que desee establecer una conexión al puerto 2000 de 192.168.0.100 debe primero generar una secuencia de paquetes a esos puertos, que asumimos cerrados, y en ese exacto orden. Una vez detectado esto, el sistema operativo del host procederá a abrir el puerto 2000.

- Supongamos que un host con dirección IP 192.168.10.10 desea conectarse al puerto 2000 de 192.168.0.100, pero todos los paquetes que envía luego del port knocking se pierden en la red. Indicar qué eventos ocurren desde el punto de vista de 192.168.10.10 desde el momento de iniciar el proceso de conexión.
- Supongamos ahora que el port knocking tuvo éxito y el puerto 2000 está abierto. Proponer un intercambio de a lo sumo ocho paquetes TCP entre ambos hosts que permita que 192.168.10.10 envíe el mensaje 'HOLA' y 192.168.0.100 le responda 'HOLA'. Además, ambos hosts deben atravesar en algún momento el estado CLOSING.

## Bibliografía

**Computer Networks: A systems approach. 3ra Edición.** *Peterson & Davie*. Capítulo 5: End-to-End Protocols (sección 5.1 y 5.2).

**RFC 793: Transmission Control Protocol.**