Session 3B

TCP: Connection Termination

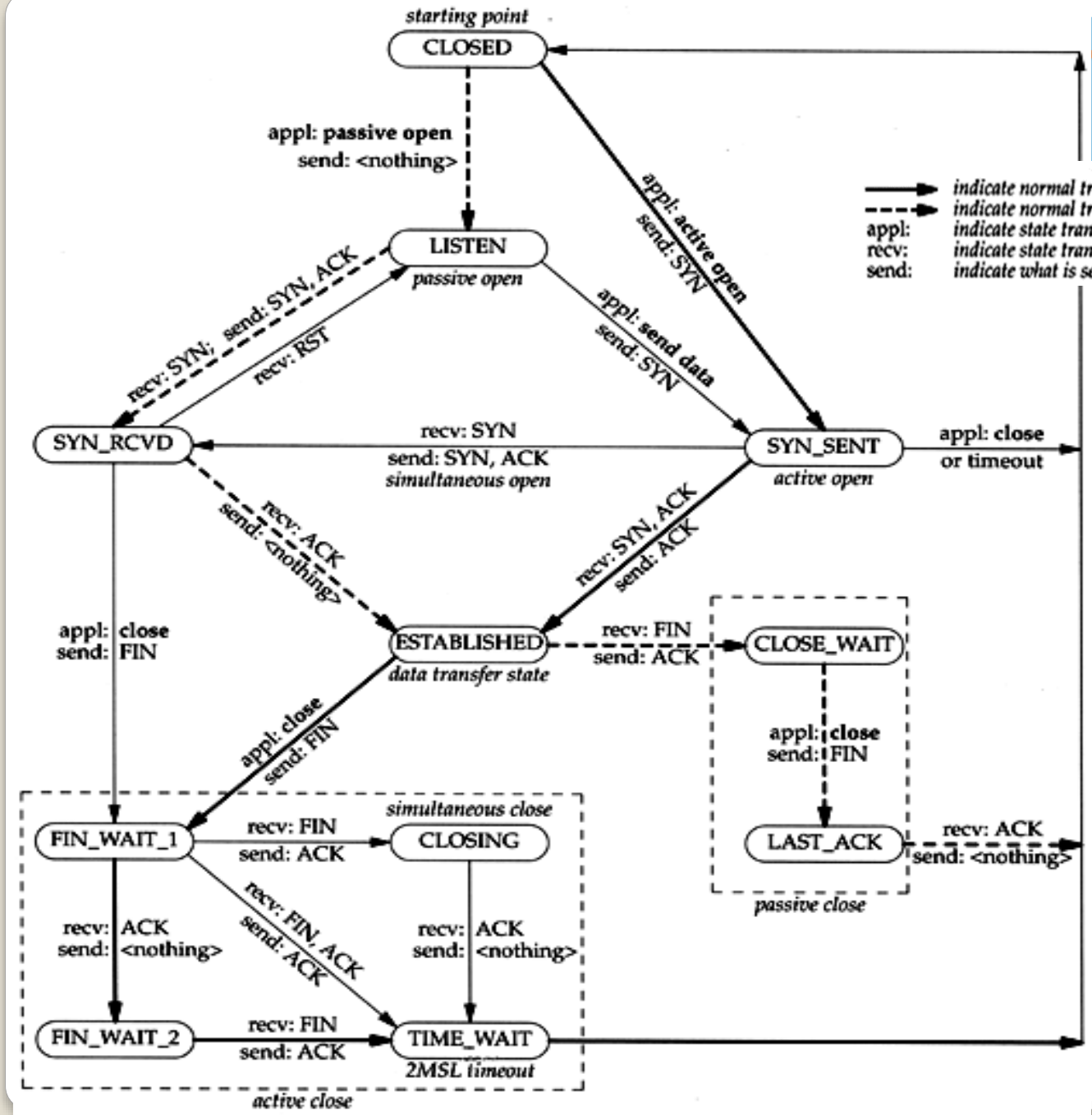
Mouli Sankaran

Session 3B: Focus

- Quiz 1: TCP Connection Establishment: Recap
- TCP Connection Termination
  - 3 Possible Combinations
  - 4th Combination
- Use of TIME_WAIT state
- Quiz 2 to 4

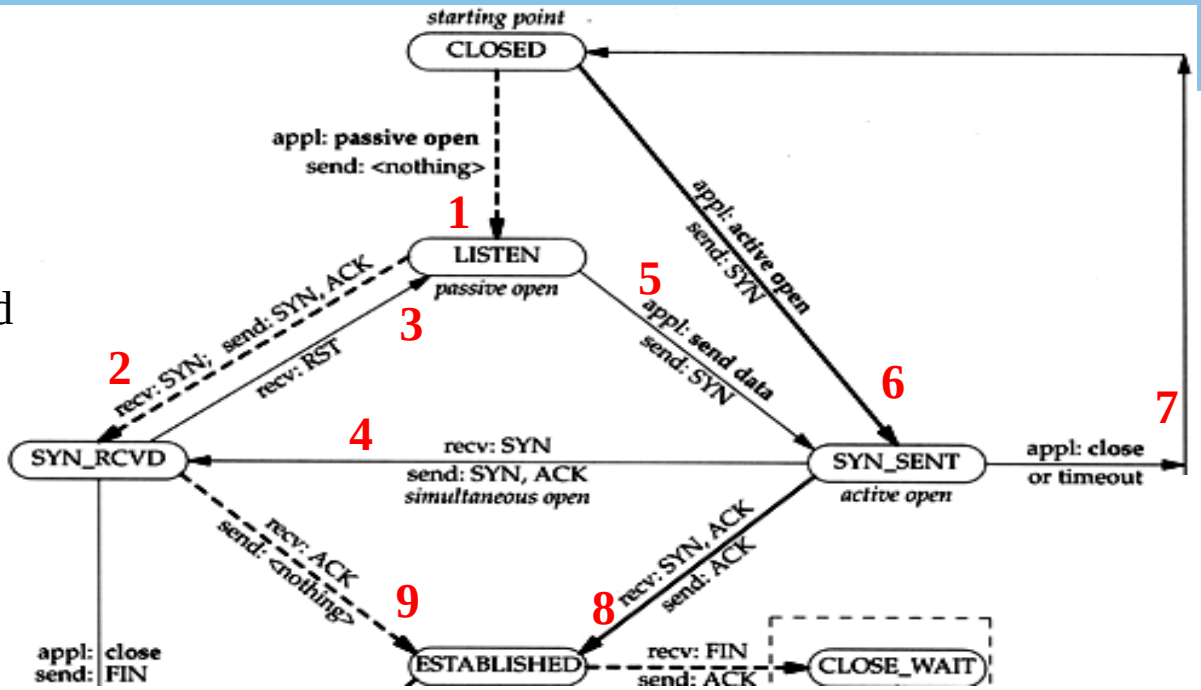
Course page where the course materials will be posted as the course progresses:

TCP: Complete State Transition Diagram



Quiz 1: TCP Connection Establishment- Recap

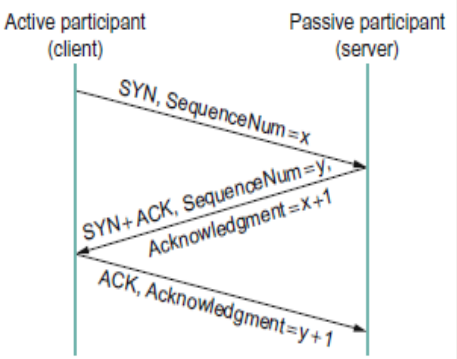
Abbreviations:
Appl: Application
Reqs: Requests
Rx: Receiving/Received
Tx: Transmitting



- Match the events causing the transitions (**fill in numbers**) shown above.

Event	Transition Number
Appl initiating a connection Req	6
Appl initiating to wait for connection Reqs	1
After Rx SYN from the Client	2

Event	Transition Number
After Rx SYN+ACK from the Server	8
After Rx ACK from the Client	9
After Tx SYN, if SYN from the other end Rx	4

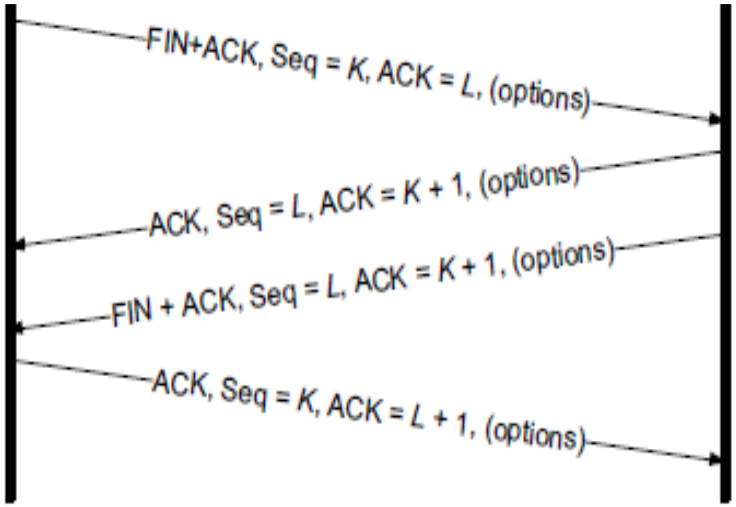


TCP Connection Termination

Reference: Ref1: TCP/IP Illustrated-Volume 1:
Chapter 13: TCP Connection Establishment and Termination

TCP: Connection Termination

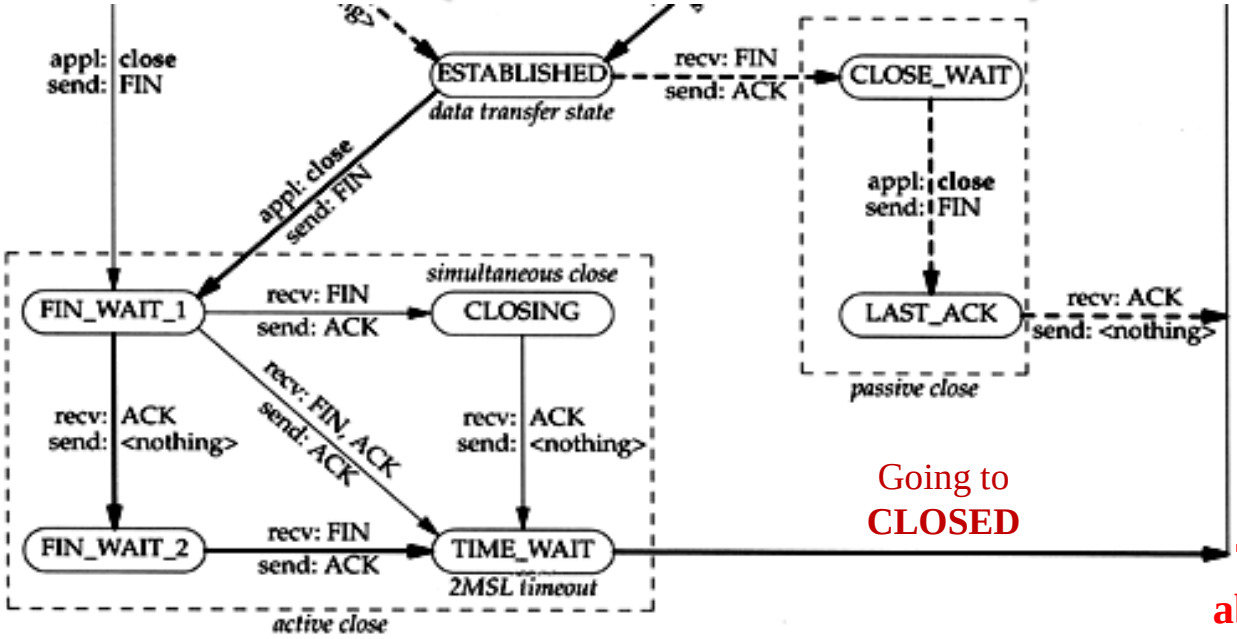
- The application process on both sides of the connection must independently close its half of (or part of) the connection.
- If only one side closes the connection, then this means it has no more data to send, but it is still available to receive data from the other side.

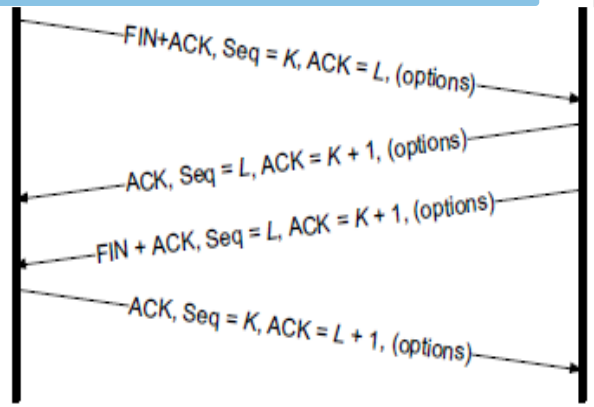


- This complicates the state-transition diagram because it must account for the possibility that the two sides invoke the close operator at the same time,
- As well as the possibility that first one side invokes close and then, at some later time, the other side invokes close.
- Thus, on any one side there are **three combinations** of **transitions** that get a connection from the ESTABLISHED state to the CLOSED state:

TCP: Connection Termination (3 Combinations)

Note: ACK is ignored if it is received for the same ACK no. which was already ACK'ed.





1. This side **1. Other side**

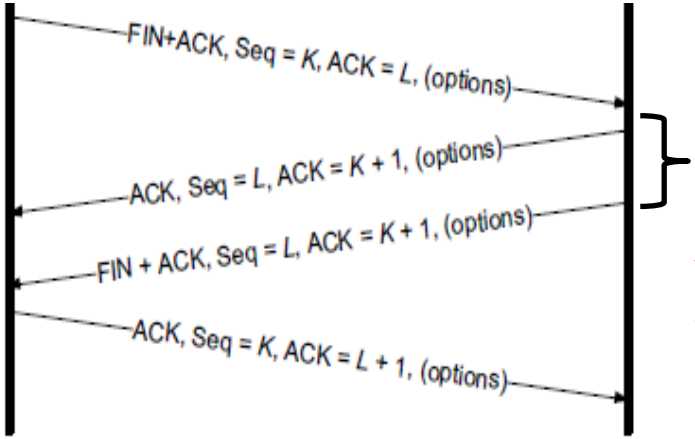
2. Other side **2. This side**

Going to **CLOSED**

Trace state transitions due to the above msg exchanges for "this side"

1. **This side closes first**: ESTABLISHED → FIN_WAIT_1 → FIN_WAIT_2 → TIME_WAIT → CLOSED.
   - Follow the state transitions in the state transition diagram above, along with the flow of messages, assuming "**This side**" on the **left**
2. **The other side closes first**: ESTABLISHED → CLOSE_WAIT → LAST_ACK → CLOSED.
   - Follow the state transitions in the state transition diagram above, along with the flow of messages, assuming "**Other side**" on the **left**

Quiz 2: Why is there a gap?



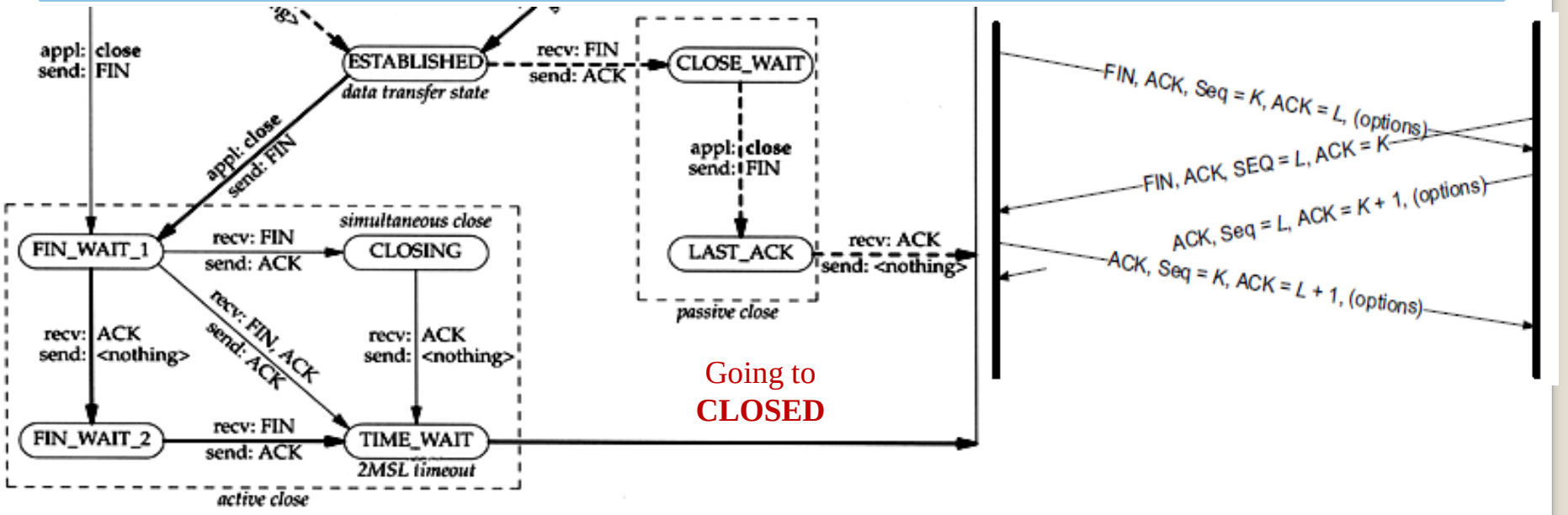
Why is this not done together (FIN+ACK)?

What is happening during this time gap on the side B?

Side A **Side B**

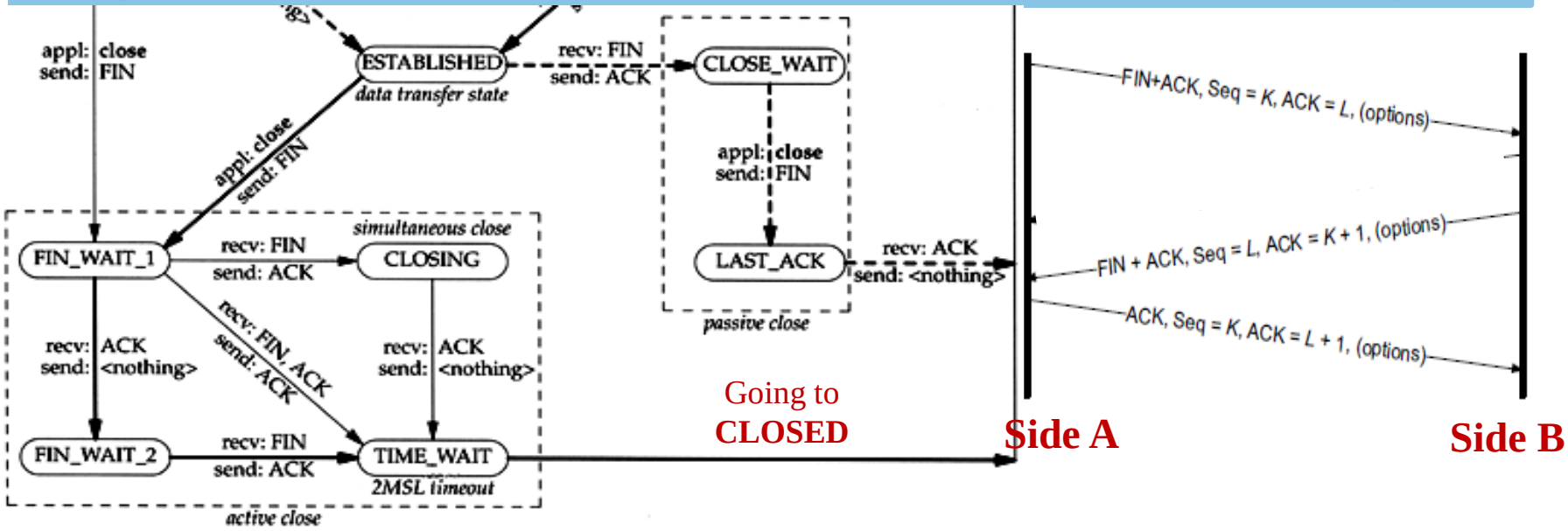
- What does Side B understand on receiving FIN+ACK from Side A?
- Application on Side A has sent a **Close** request to the TCP/IP stack, because it does not have any data to send to Side B and it wants to close the connection from its end.
- After sending ACK for the FIN from Side A, what does Side B do?
- Side B informs its application that Side A wants to close its part of the connection. The application on Side B has the liberty to decide either to keep its side of the connection and keep sending data, if it has, to Side A, or decide to close its connection as well, by sending a Close command to the protocol Stack.
- When does Side B send FIN+ACK to Side A?
- On receiving a Close command from the Application. Note that protocol stack has no authority to close the connection without the consent of the application ☺

TCP: Connection Termination (3rd Combination: Simultaneous Close)



3. **Both sides close at the same time:** ESTABLISHED → FIN_WAIT_1 → CLOSING → TIME_WAIT → CLOSED.
   - Follow the state transitions in the state transition diagram above, along with the flow of messages shown above for simultaneous close
4. **There is one more combination possible, which one? ☺**

TCP: Connection Termination (4th Combination: FIN+ACK from Other side)

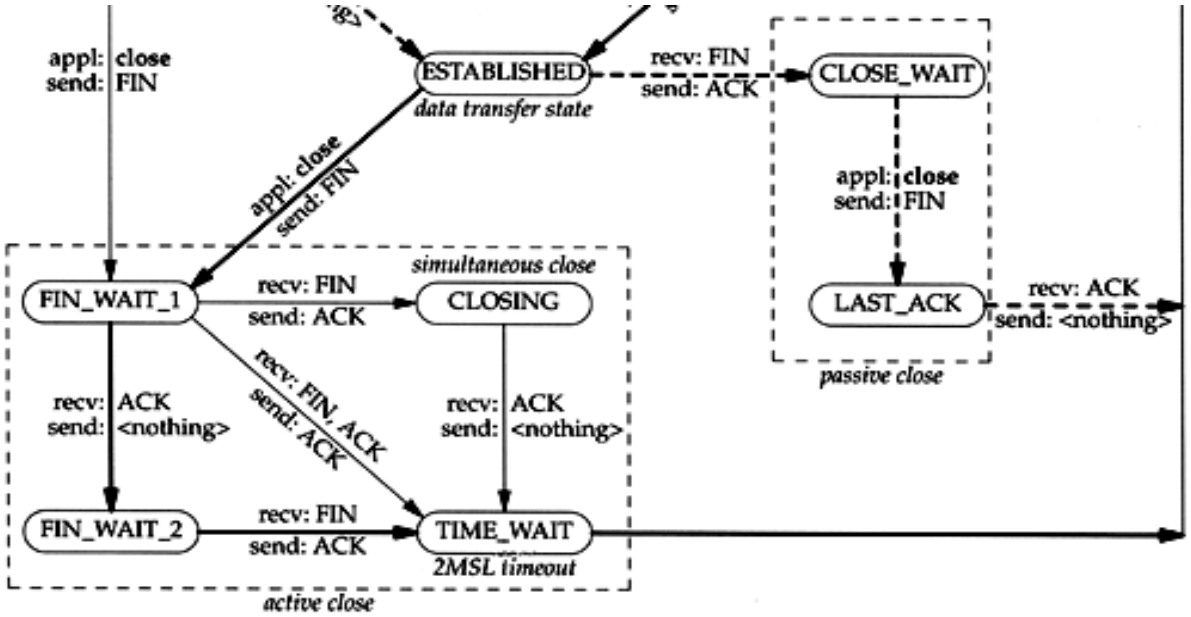


4. **Side A closes first and Side B also closes before sending ACK for FIN:** ESTABLISHED → FIN_WAIT_1 → TIME_WAIT → CLOSED.
   - Follow the state transitions in the state transition diagram above on Side A, along with the flow of messages shown above for the fourth combination, where FIN+ACK is sent from the other side B.

- This combination is possible when the FIN is received from Side A, and the application on Side B also decides to close the connection before the ACK for the FIN is sent from Side B, then Side B sends FIN+ACK together to Side A

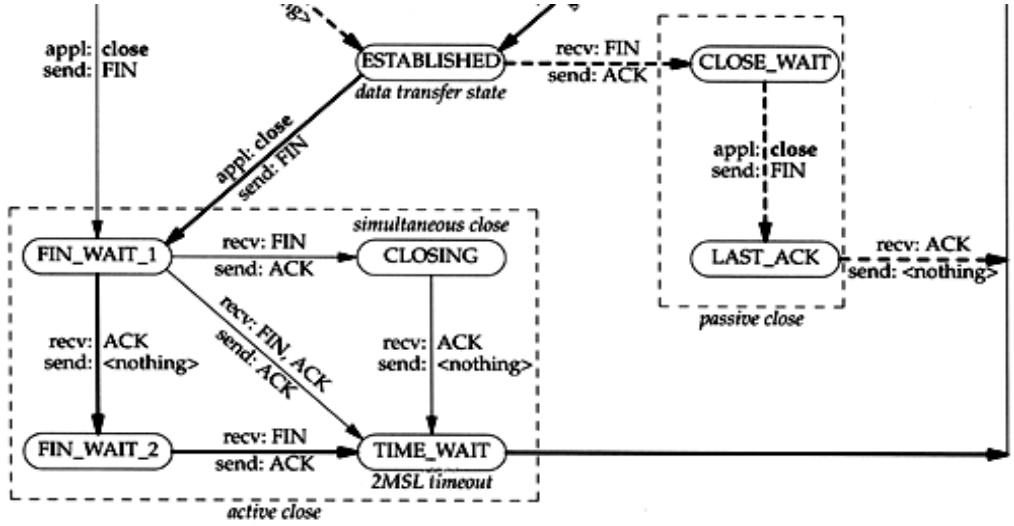
TCP Connection Termination: TIME_WAIT

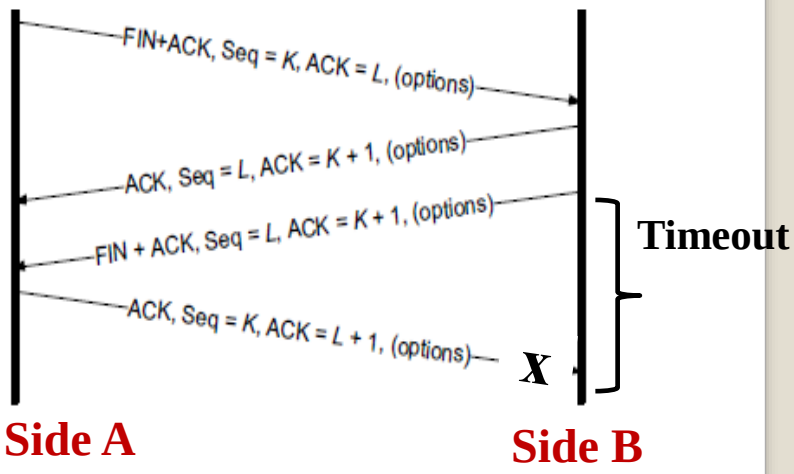
Note: 120 seconds is configurable based on the current behavior of Internet



- The main thing to recognize about connection teardown is that a connection in the **TIME WAIT** state cannot move to the **CLOSED** state until it has waited for **two times** the **maximum** amount of **time** an **IP datagram might live** in the **Internet** (i.e., **~120 seconds**).
- The reason for this is that, while the local side of the connection has sent an ACK in response to the other side's FIN segment, it does not know that the ACK was successfully delivered or not.

TCP Connection Termination: TIME_WAIT
Contd…





- As a consequence, the other side might retransmit its FIN segment, and this second FIN segment might be delayed in the network.
- If the connection were allowed to move directly to the CLOSED state, then another pair of application processes might come along and open the same connection (i.e., use the same pair of port numbers), and the delayed FIN segment from the **earlier incarnation** of the connection would immediately initiate the termination of the **later incarnation** of that connection.

TCP Connection Termination

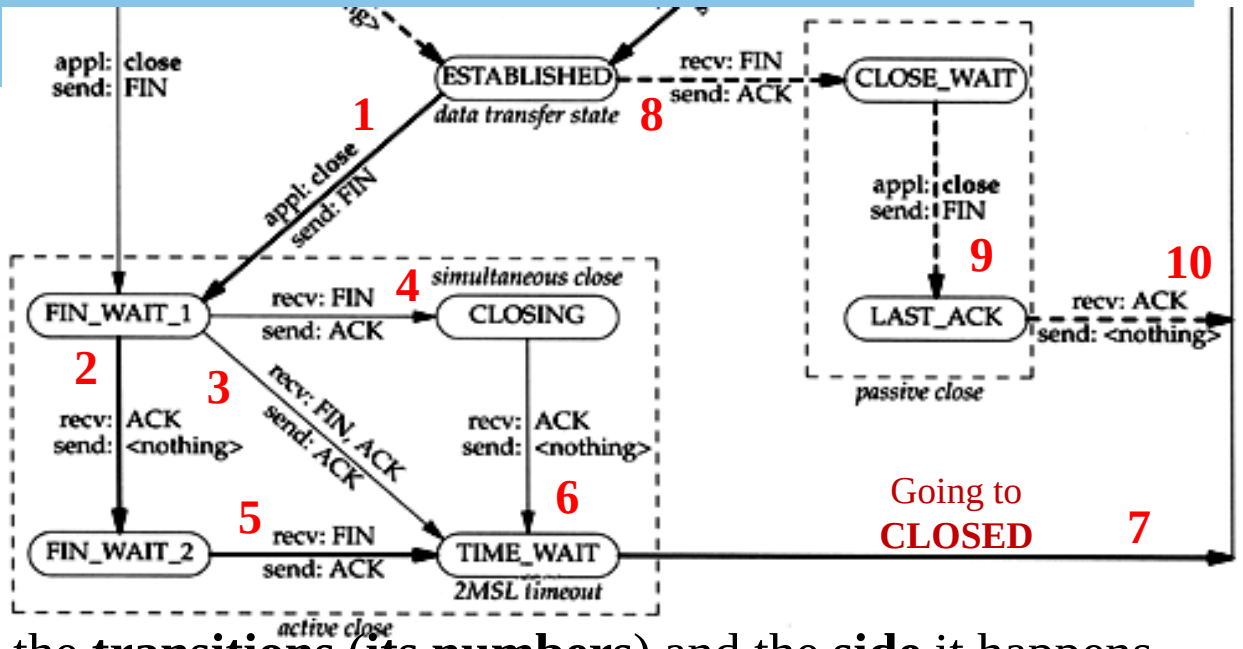
Quiz 2 to 4

Quiz 2: TCP Connection Termination

Abbreviation:
Appl: Application

Note: Consider State transitions of both **Side A** and **Side B**.

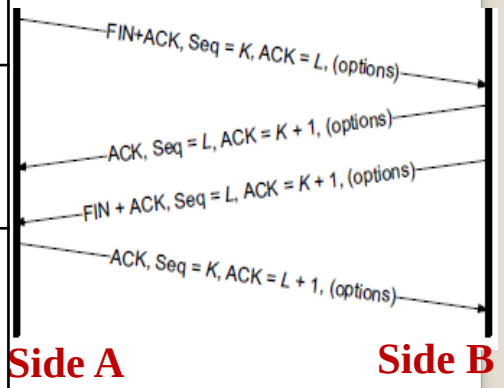
Assume there was a TCP connection between **Side A** and **Side B** already.



- Match the events causing the **transitions** (**its numbers**) and the **side** it happens.

Event	Side	Num
The **transition** that would **trigger** a **close** notification to **appl**.	**B**	**8**
The **transition** due to the **appl** closing **later**	**B**	**9**
The **transition** that takes a **side** to **TIME_WAIT**	**A**	**5**

Event	Side	Num
The **transition** that takes a **side** to **CLOSED** without **TIME_WAIT**.	**B**	**10**
The **transition** triggered on the **side,** which receives **ACK last**	**B**	**10**
The **transition** that takes a **side** to **CLOSED** after **TIME_WAIT**	**A**	**7**



Message transactions are shown here,

Quiz 3: TIME_WAIT

- Why is there a **TIME_WAIT** state **only** on the **active close** side of the transition and **not on both sides** and why is it on the **Active close** side and **not passive close**? Choose **all the correct options**.

A. It is not possible to make both sides to wait.

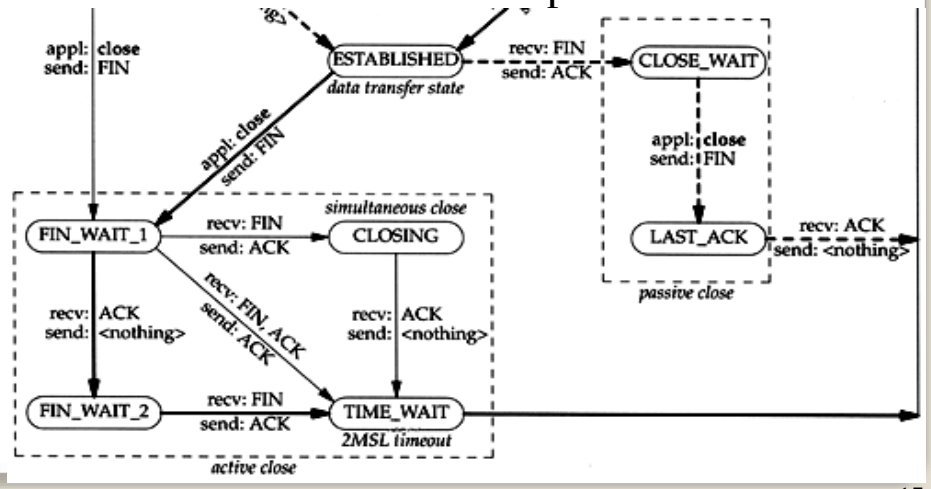
B. It is sufficient to make only one side to wait because it is done to make sure a new connection between the same set of machines and the ports happens only after a fixed delay (2 MSL)

C. It is not correct to make both sides to wait because it won't be possible to establish a connection between the same set of machines and ports later on.

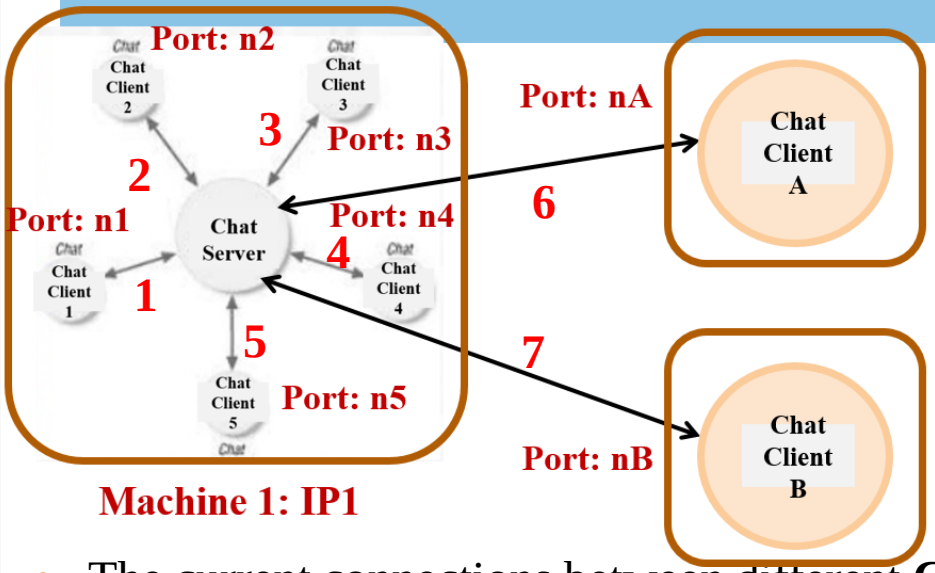
D. Though it is sufficient to introduce delay on any side, active side is chosen because that is most likely to initiate a new connection with the same machine and the port later on.

ANS: B and D

MSL: Maximum Segment Lifetime



Quiz 4: Multi-chat Application



Note: IP and port nums are given here.

Let us number each connection from 1 to 7.

Note: Recall the **Server Port ID** was: **9001**

Let us call **loopback IP address 127.0.0.1** as **LBIP**

Note: The OS takes care of allotting unique Client ports.

- The current connections between different **ChatClients** and the **ChatServer** are shown above. Answer the questions and fill in the relevant values below.

1. **How many unique sockets are created here?** **ANS1: 7**

2. **How many of the connections are in loopback mode?** **ANS2: 5 (1 to 5)**

3. **Fill in the values of each socket values of the connections in the table:**

Note: You can notice that for all the **sockets 4-tuple values** are **unique**, with the **server**.

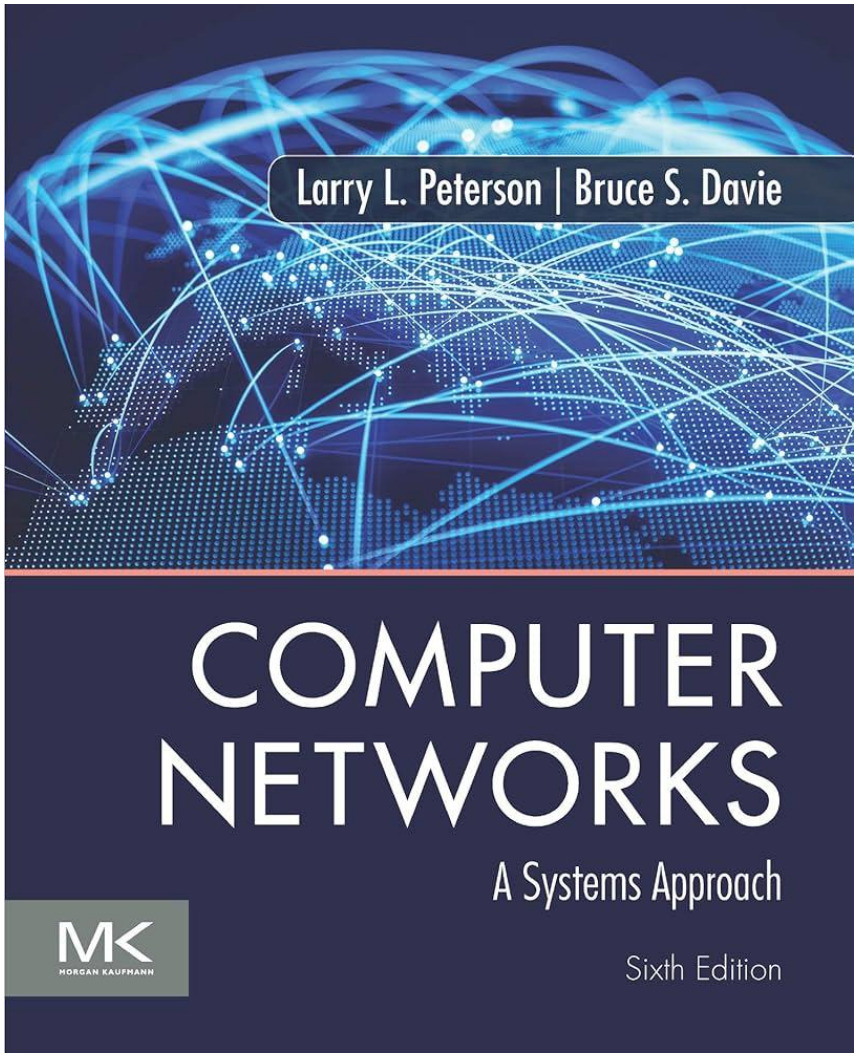
Connections	Server IP	Server Port	Client IP	Client Port
1	LBIP	9001	LBIP	n1
2	LBIP	9001	LBIP	n2
3	LBIP	9001	LBIP	n3
4	LBIP	9001	LBIP	n4
5	LBIP	9001	LBIP	n5
6	IP1	9001	IPA	nA
7	IP1	9001	IPB	nB

Session 3B: Summary

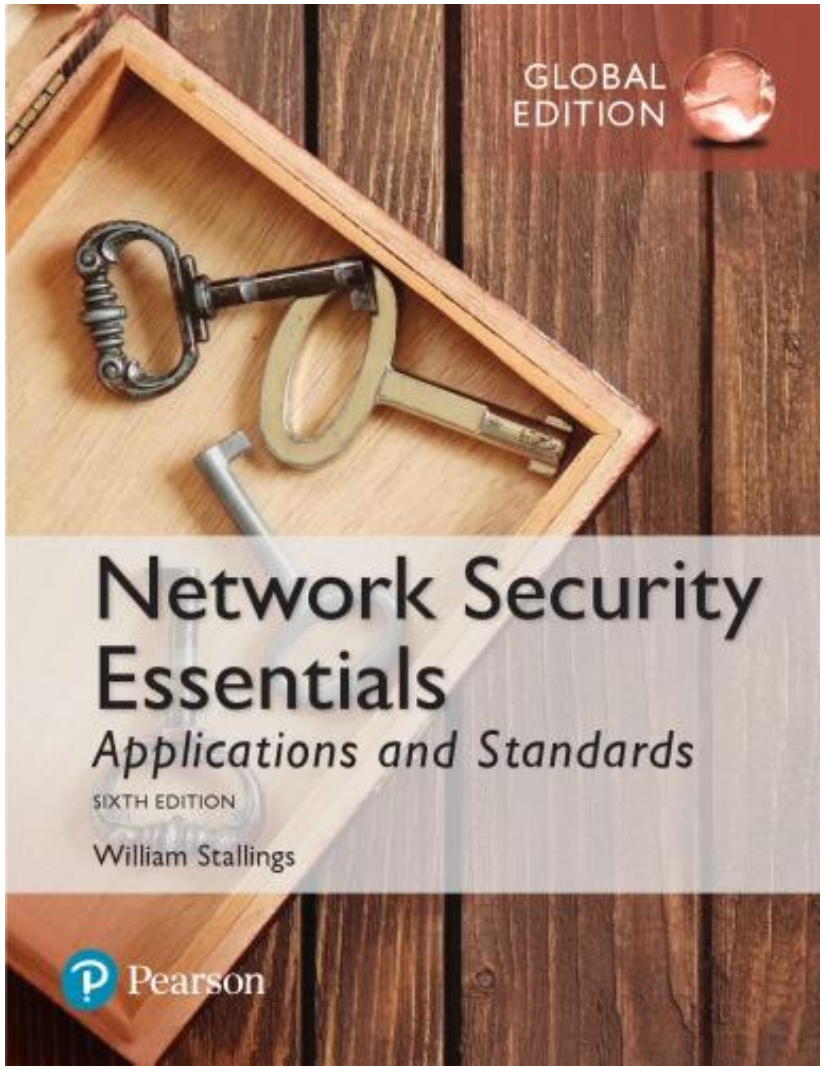
- Quiz 1: TCP Connection Establishment: Recap
- TCP Connection Termination
  - 3 Possible Combinations
  - 4th Combination
- Use of TIME_WAIT state
- Quiz 2 to 4

Textbooks

Textbook 1

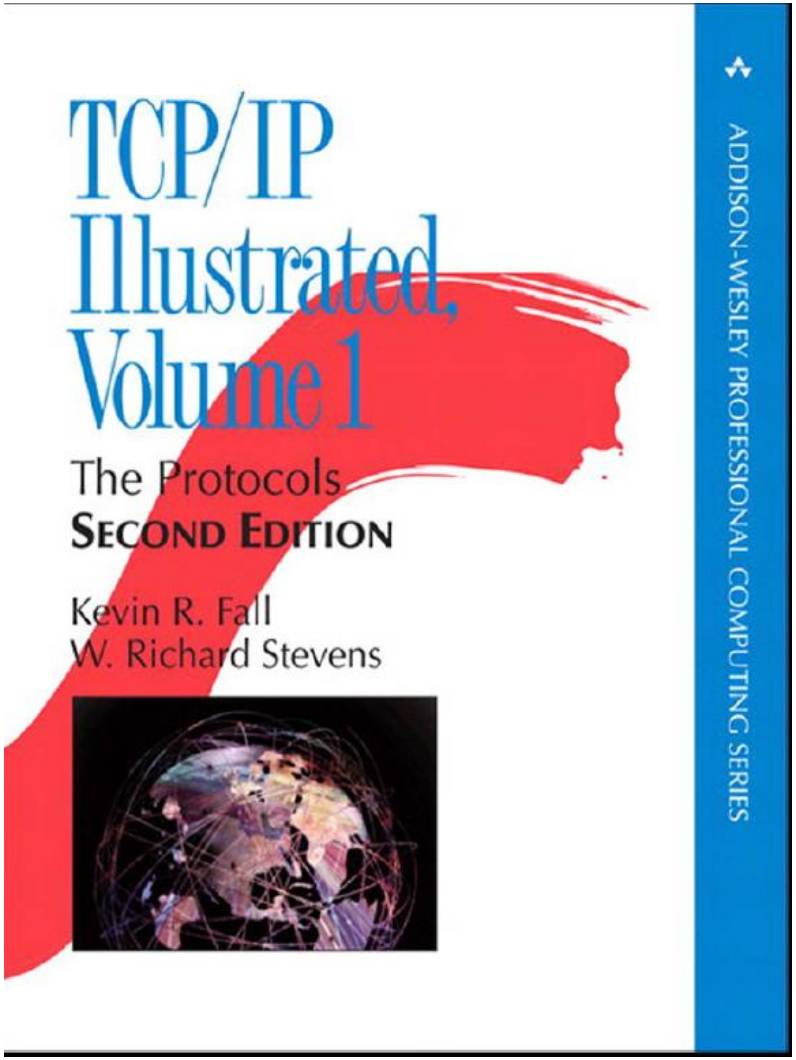


Textbook 2



References

Ref 1



Ref 2

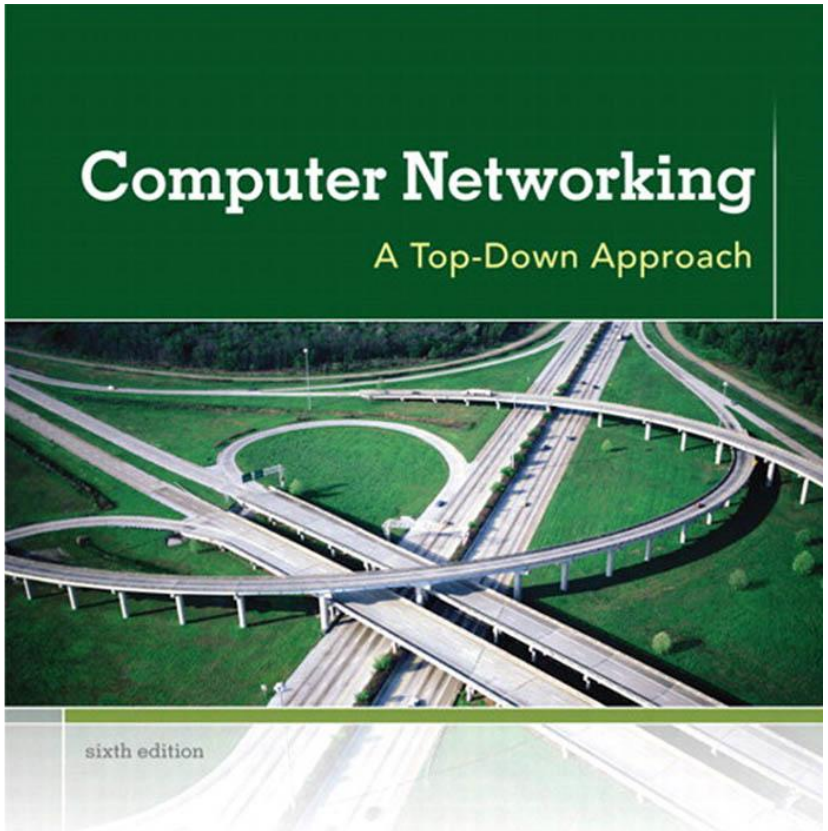
TCP Congestion Control: A Systems Approach

TCP Congestion Control: A Systems Approach

Peterson, Brakmo, and Davie

References

Ref 3



Ref 4