CS168 (and EE122) TAs past and present give you….

# Transport Algorithms & Some Router Architecture

# Today

- Transport Algorithms
  - Stop-and-Wait
  - Go-Back-N
  - Selective Repeat
- Router Architecture
  - Longest-Prefix-Match
  - Head of Line Blocking

# Transport Algorithms

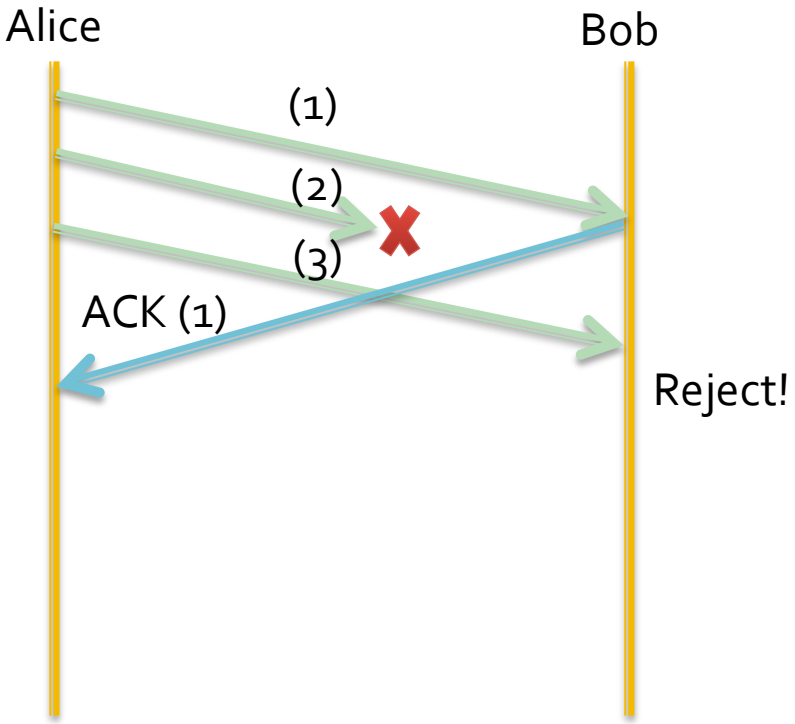
- Reliable Data Transmission
  - ARQ: Automatic Repeat reQuest
  - FEC: Forward Error Correction
  - We use ARQ for reliable transport. (Why?)
- Sliding Window
  - transmit window
  - receive window

# Transport Algorithms

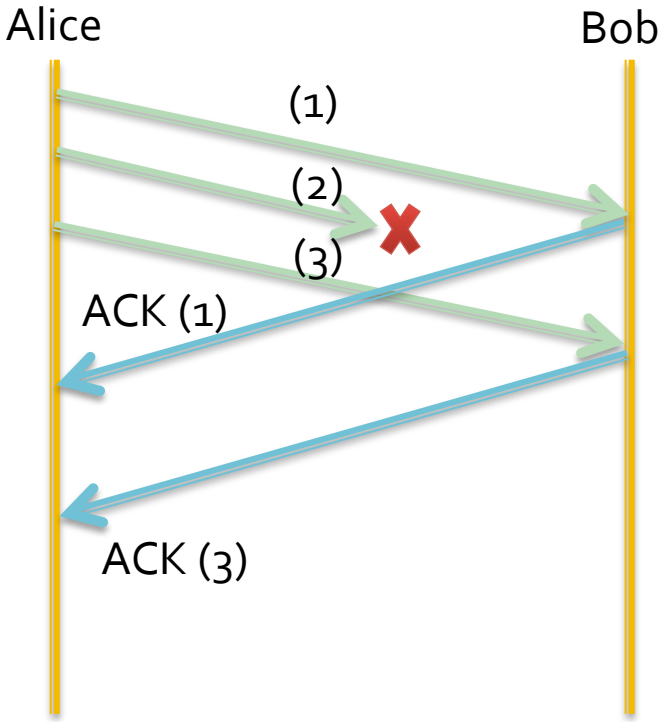
- ARQ Examples
  - Stop-and-Wait
    - tx window size: 1
    - rx window size: 1
  - Go-Back-N
    - tx window size: N
    - rx window size: 1
  - Selective Repeat
    - tx window size: N
    - rx window size: N

# Transport Algorithms

- Difference between Go-back-N and SR




Go-back-N

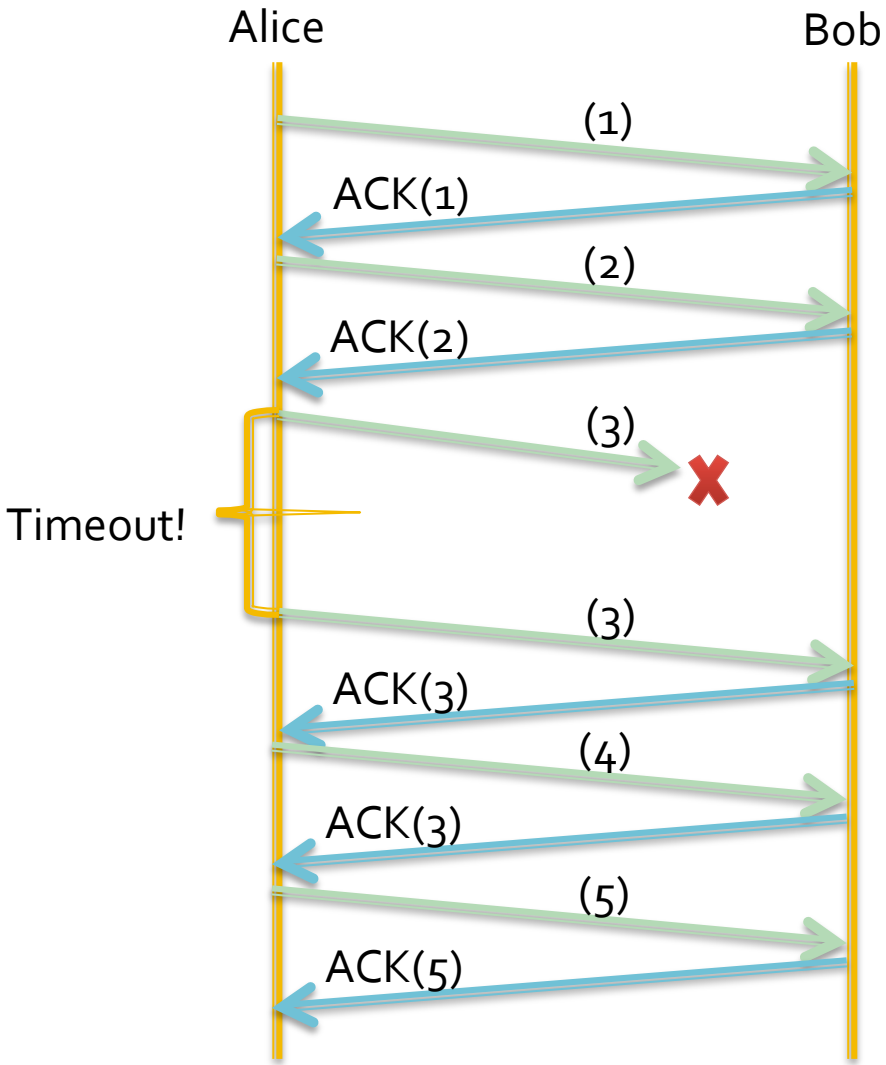



Selective Repeat

# ARQ in the Real World: TCP

- TCP Reliable Transmission
  - A variation of Go-Back-N scheme
  - Many TCP implementations will buffer correctly received but out-of-order segments.
- Selective Acknowledgement (SACK)
  - Selective Repeat
- Cumulative ACKs
  - The sequence number is in bytes.
  - Therefore each packet may contain a continuous range of sequence numbers `[seq, seq+len)`.
  - The receiver send an ACK `seq+len` for the packet, specifying it have received all data preceding `seq+len` bytes.
  - In TCP, an ACK specifies the sequence number of the next byte the receiver expects to receive.

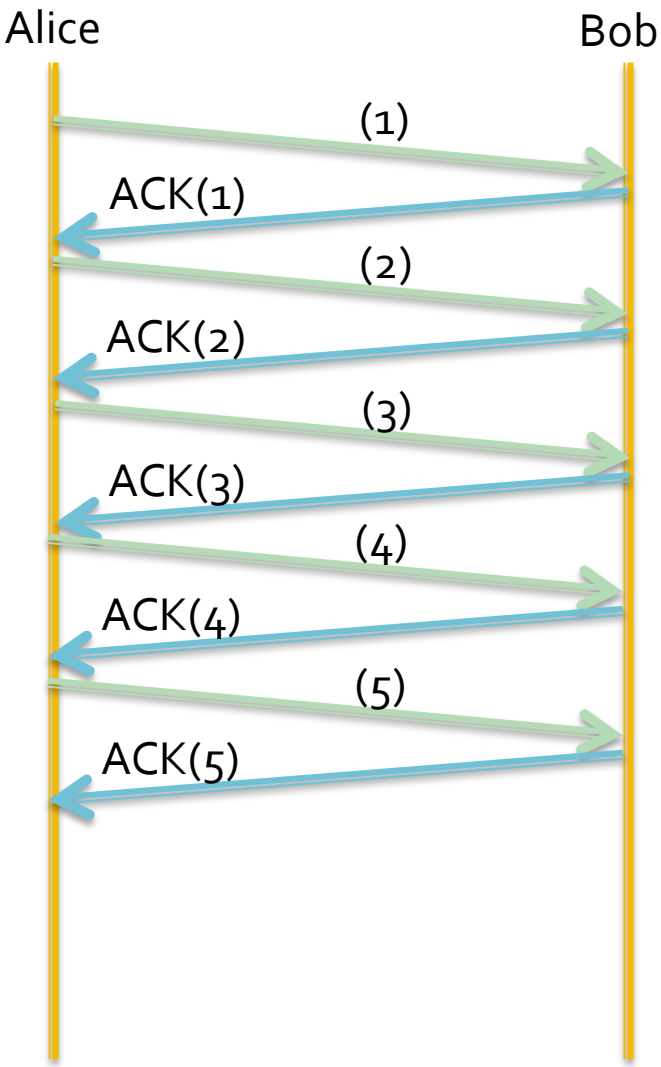
# (i)



# Transport Algorithms

- Tips and Tricks:
  - Always draw a timing diagram when trying to understand a new protocol!
  - If the homework/midterm asks for how long something takes, the timing diagram will help you calculate it.

# (ii)

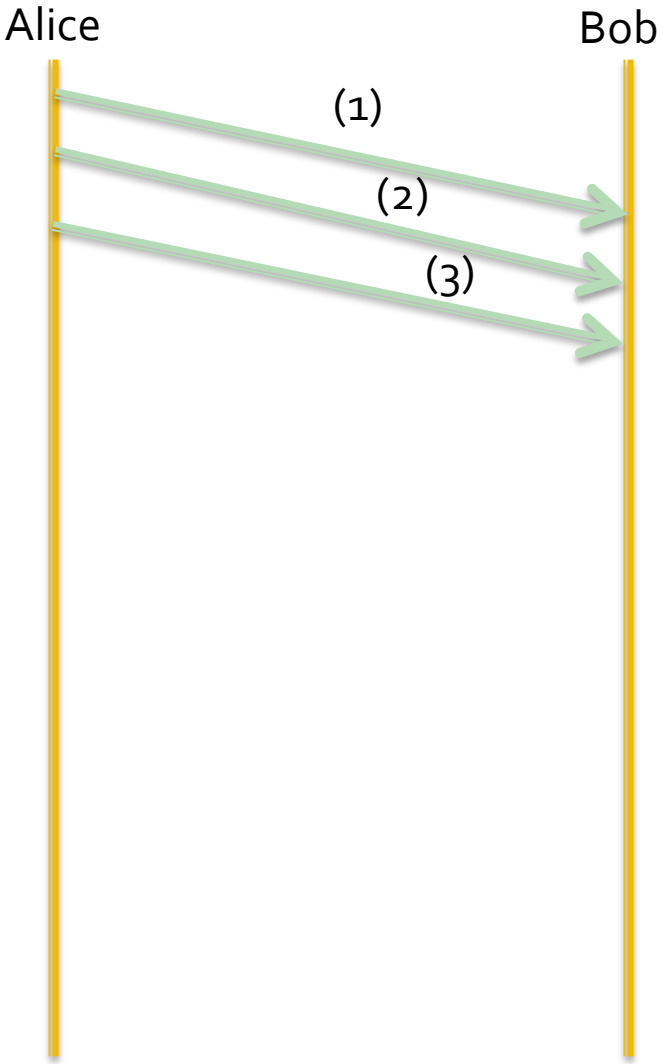



Transmission Delay for one packet: 2ms
Propagation Delay for one packet: 30ms

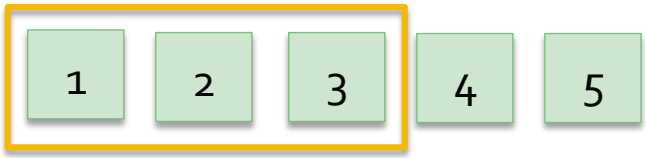
10 packets sent.

32 * 10 = 320 ms

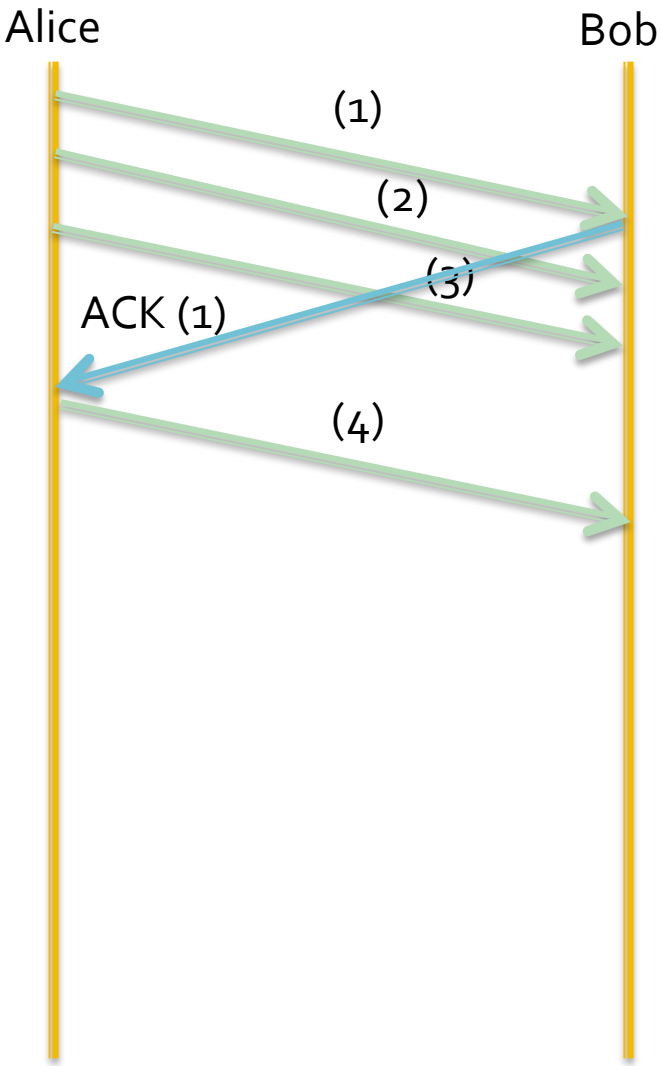
# (iii)



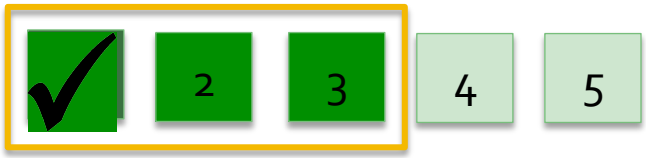

## Sliding Window



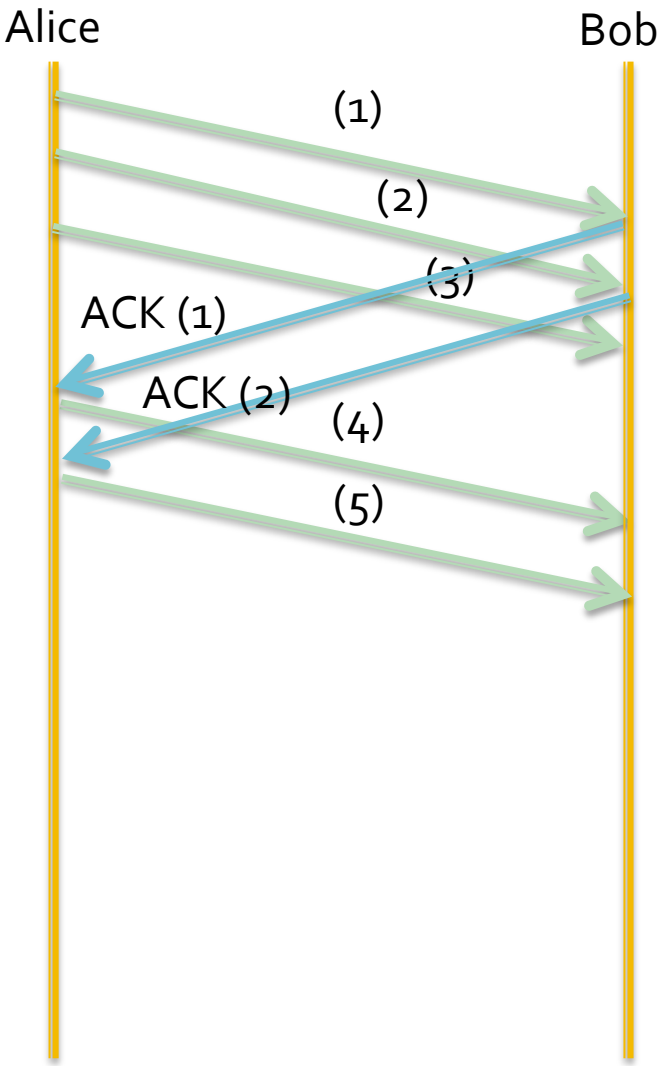
# (iii)




## Sliding Window

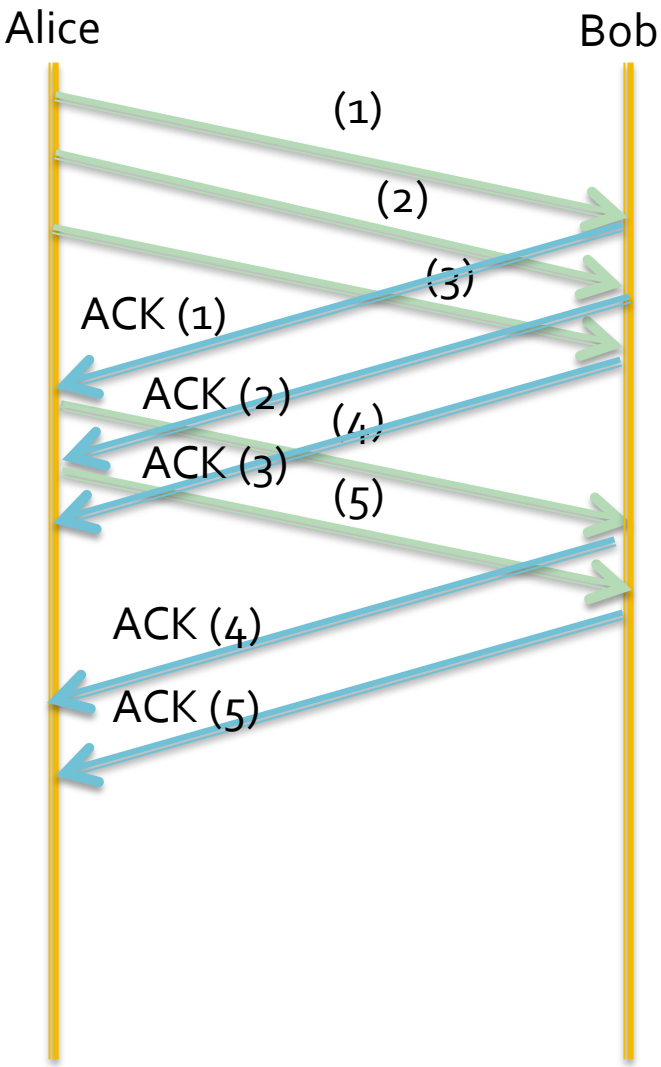


# (iii)

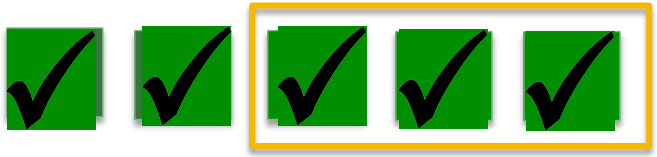



## Sliding Window

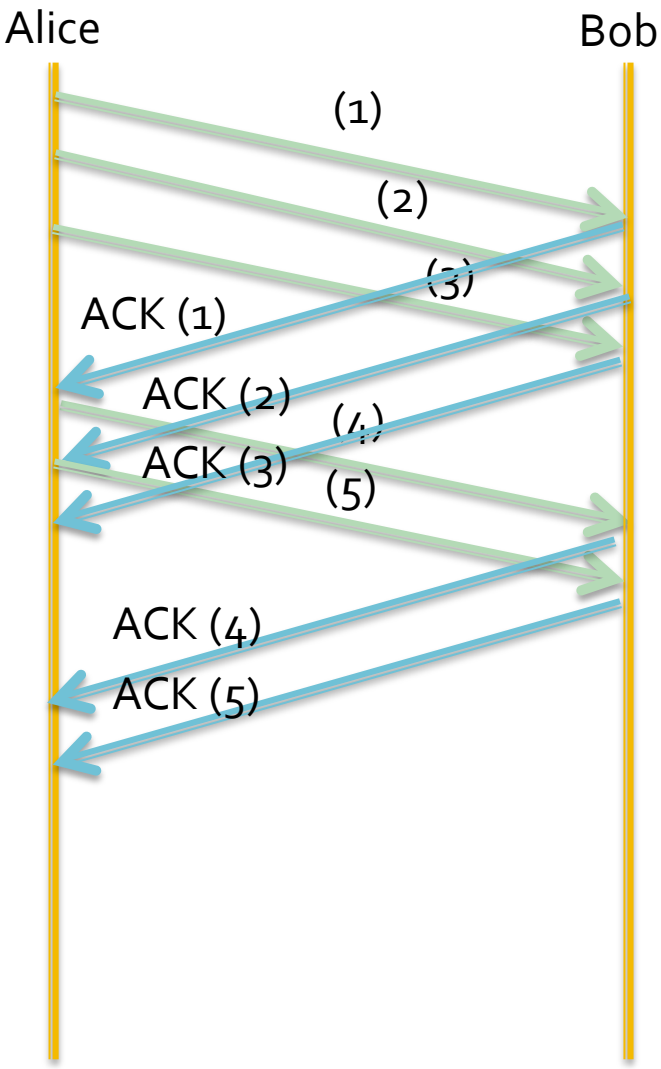
# (iii)




## Sliding Window

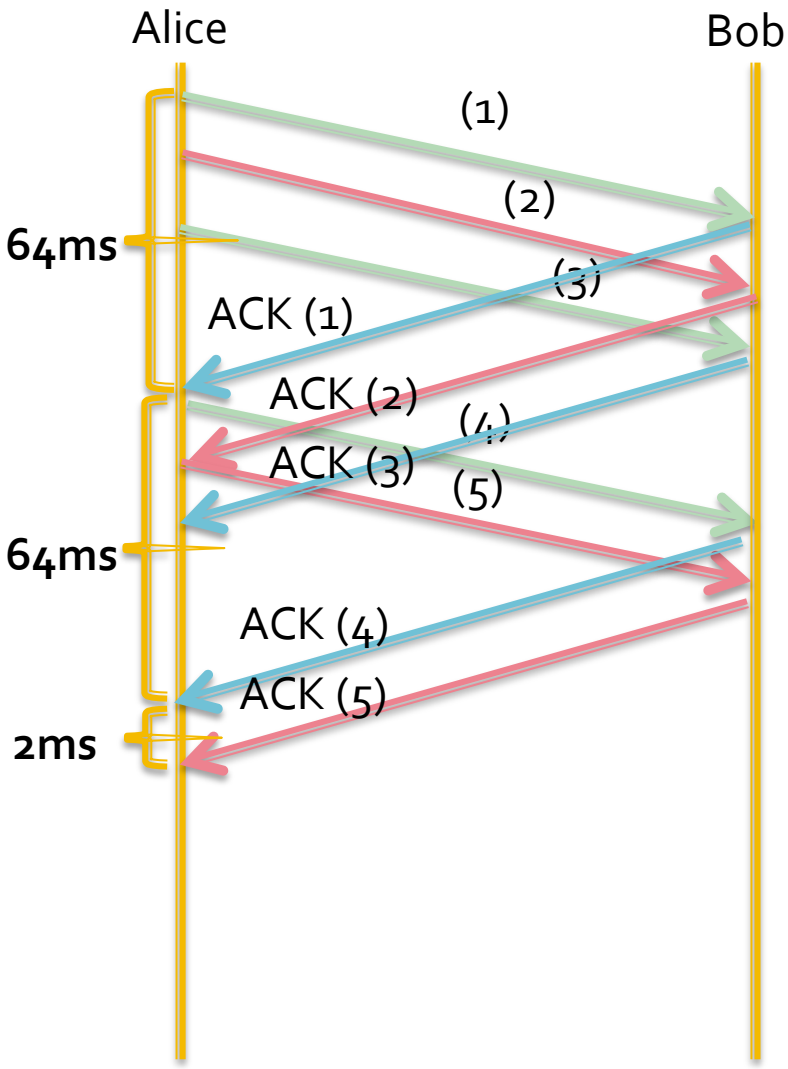


# (iii)




How long did it take for the LAST packet to be ACKed back to Alice?

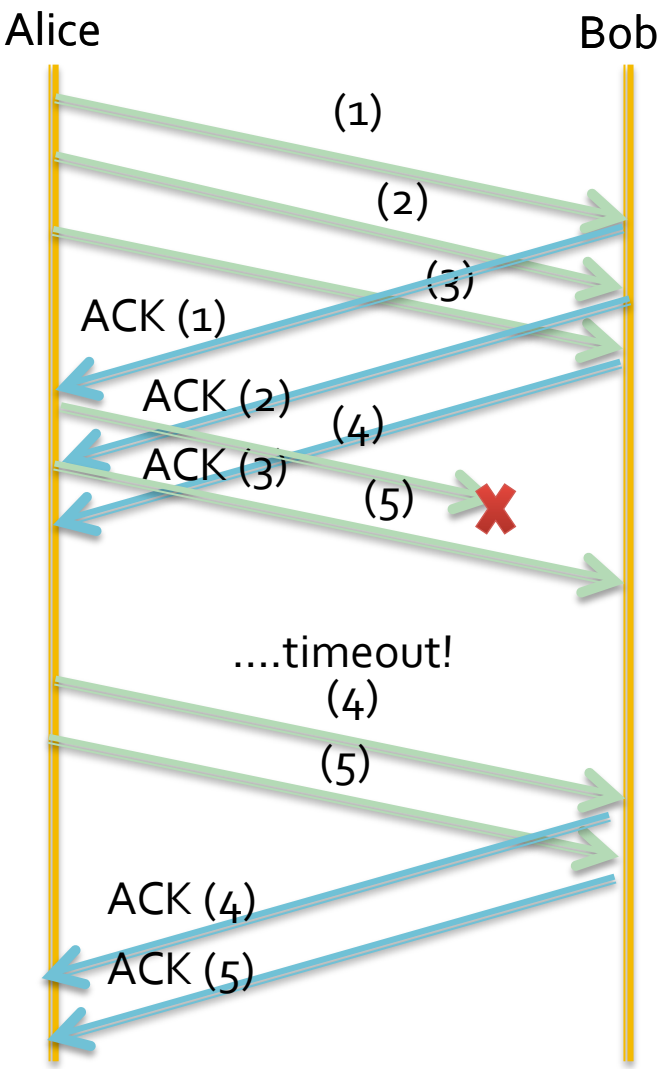
# (iii)




How long did it take for the LAST packet to be ACKed back to Alice?

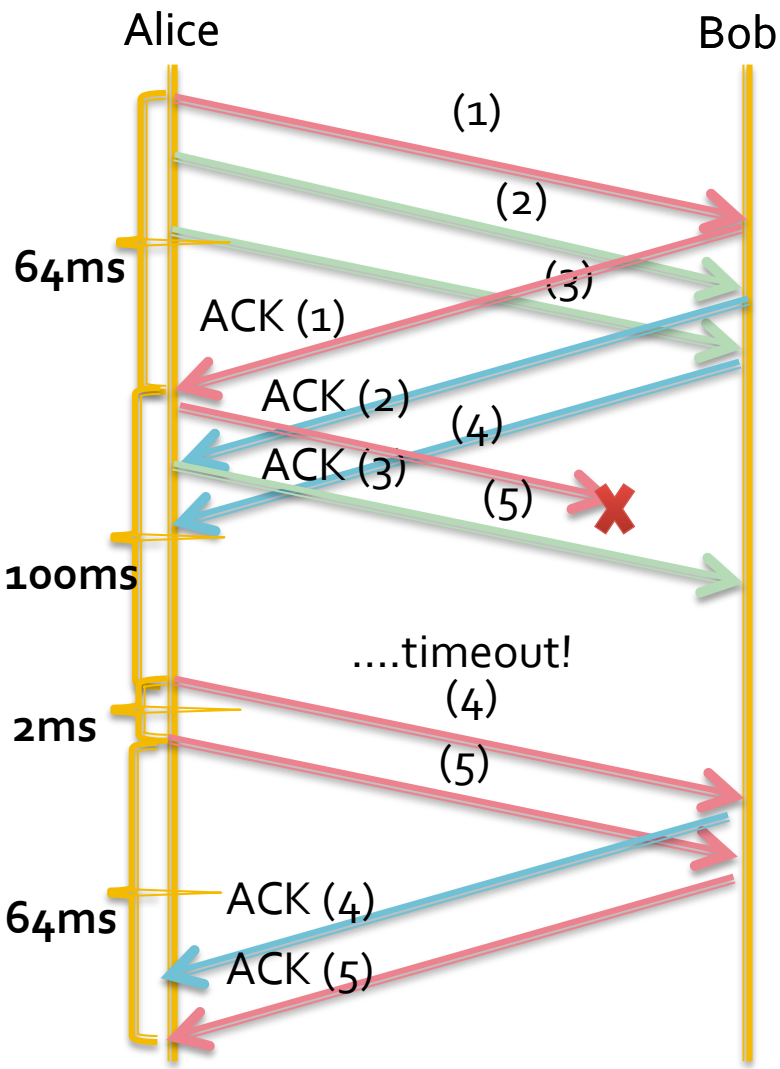
= 130ms

# (iv)




How long did it take for the LAST packet to be ACKed back to Alice?

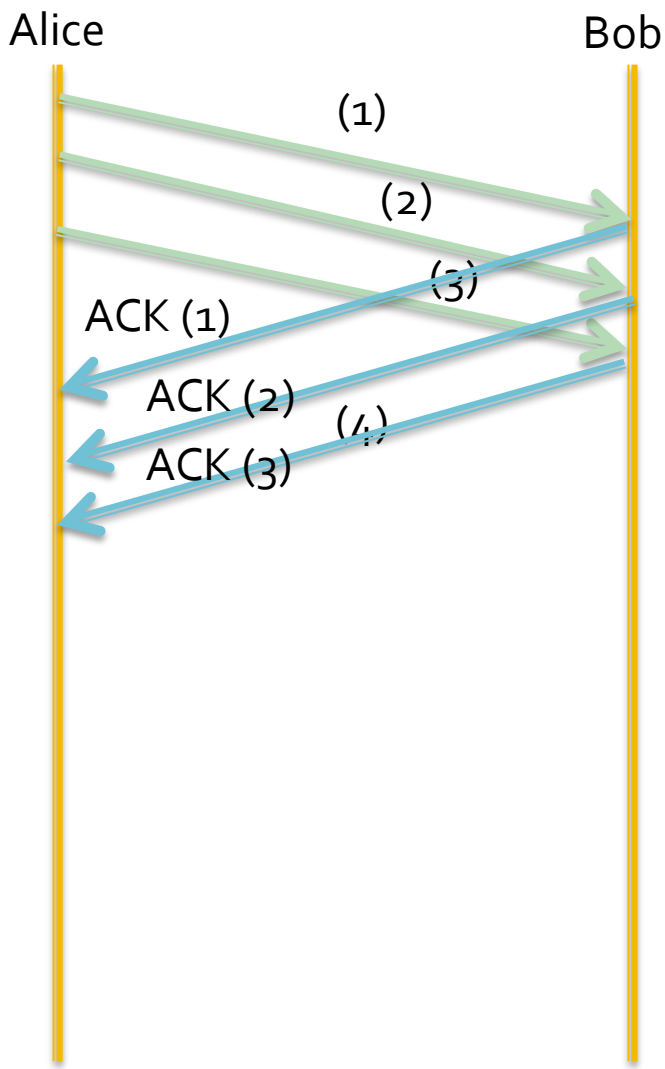
# (iv)




How long did it take for the LAST packet to be ACKed back to Alice?

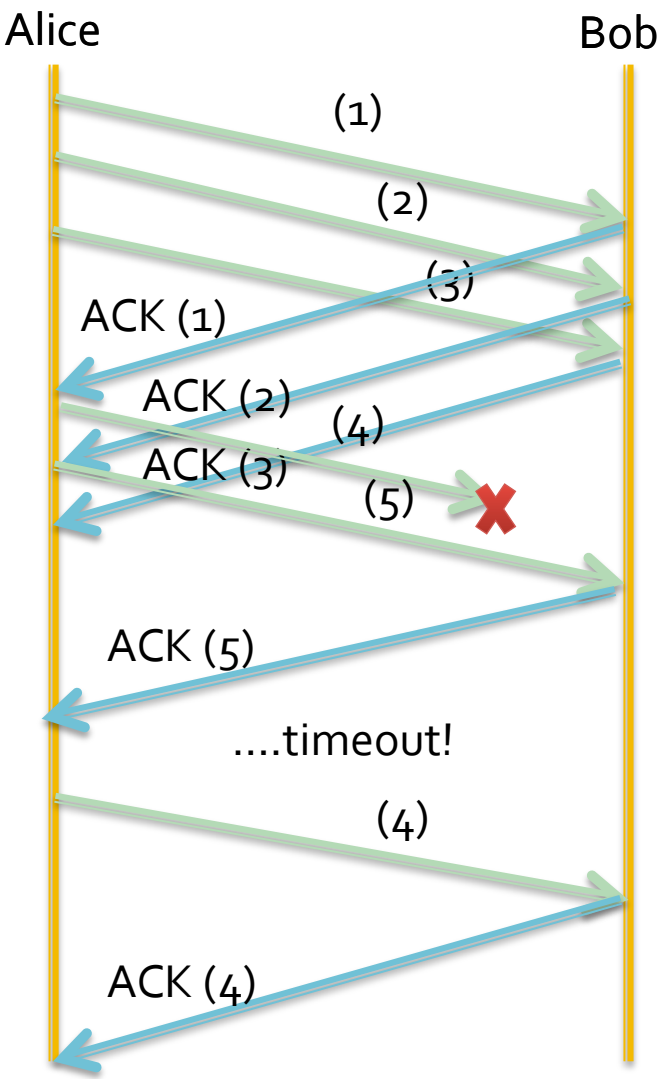
= 230ms

# (v)



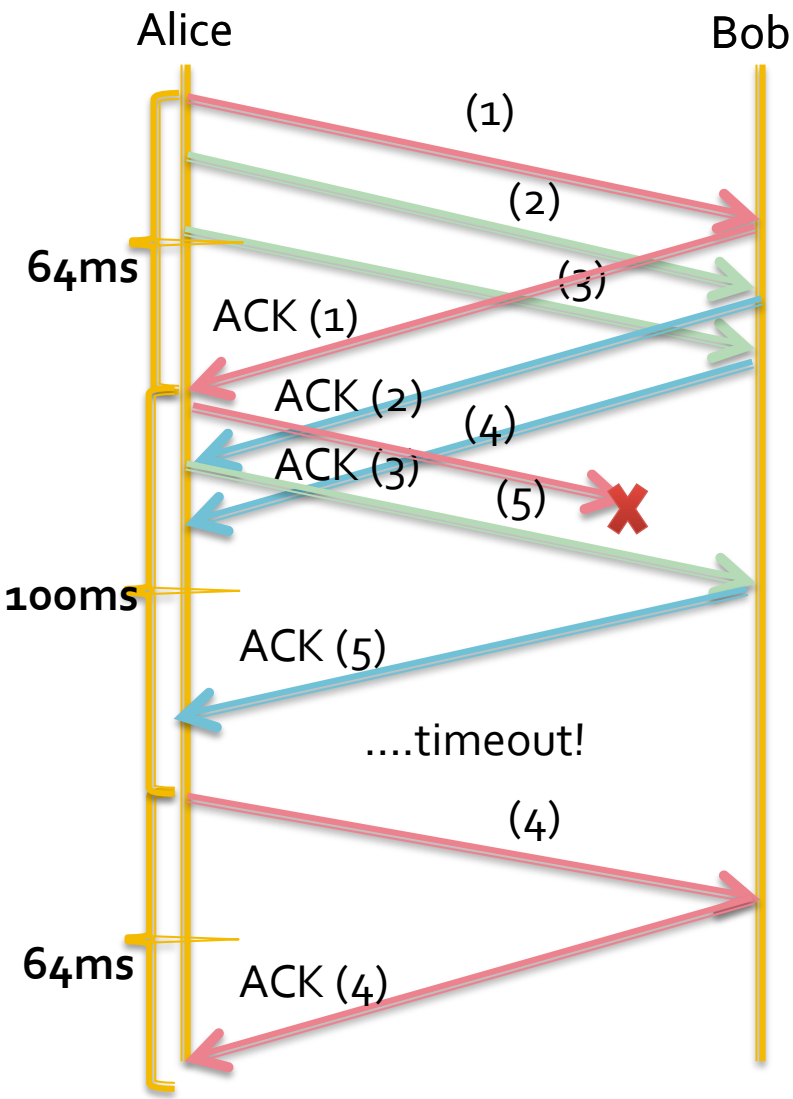

How long did it take for the LAST packet to be ACKed back to Alice?

# (v)




How long did it take for the LAST packet to be ACKed back to Alice?

# (v)




How long did it take for the LAST packet to be ACKed back to Alice?

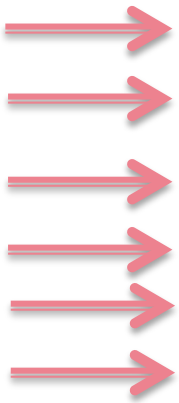
= 228ms.

# Longest Prefix Match

- Tips and Tricks:
  - Convert everything to binary first.
  - Always match from left-to-right.
  - Looking for the longest exact match starting from the first bit.

# 2)

| Prefix | Port |
|---|---|
| 8.0.0.0/8 | 1 |
| 9.0.0.0/8 | 2 |
| 8.192.192.0/18 | 2 |
| 8.192.128.0/18 | 1 |
| 8.128.0.0/16 | 3 |
| 8.192.0.0/16 | 4 |
| Default | 5 |



00001000 00000000 00000000 00000000

00001001 00000000 00000000 00000000

00001000 11000000 11000000 00000000

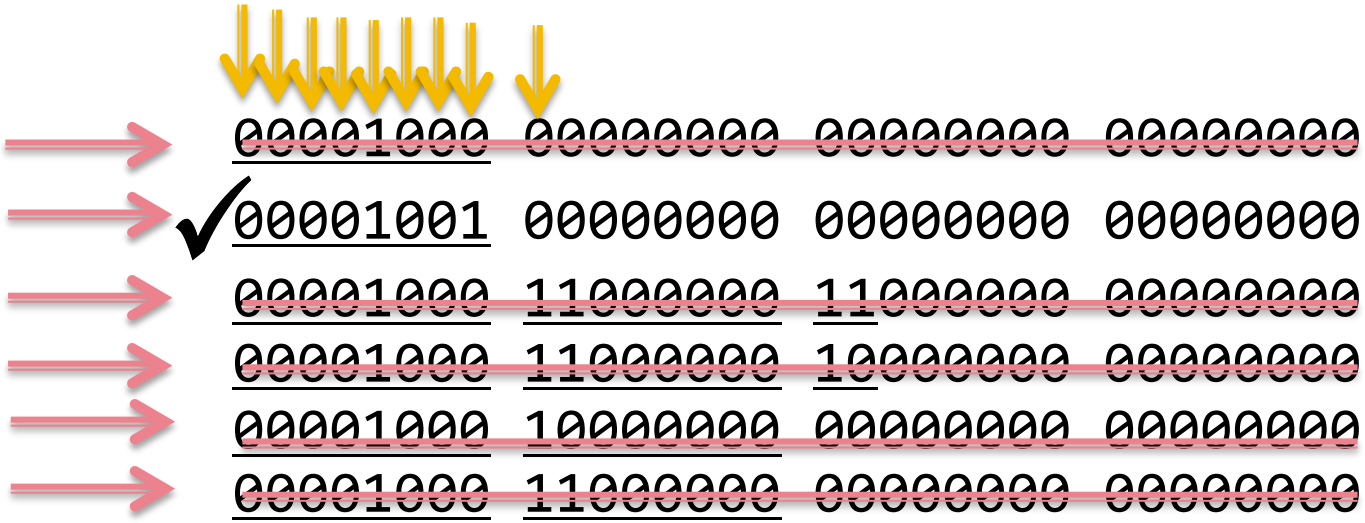
00001000 11000000 10000000 00000000

00001000 10000000 00000000 00000000

00001000 11000000 00000000 00000000

# 2i)

| Prefix | Port |
|---|---|
| 8.0.0.0/8 | 1 |
| 9.0.0.0/8 | 2 |
| 8.192.192.0/18 | 2 |
| 8.192.128.0/18 | 1 |
| 8.128.0.0/16 | 3 |
| 8.192.0.0/16 | 4 |
| Default | 5 |

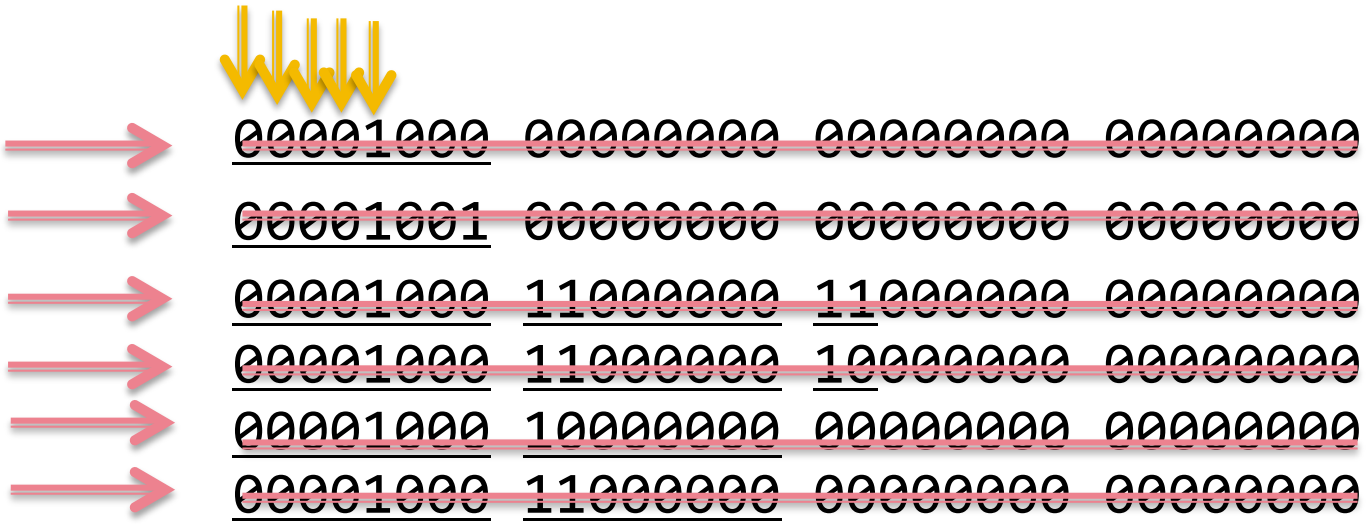



9.8.4.56 00001001 000011000 00000100 00111000

2

# 2ii)

| Prefix | Port |
|---|---|
| 8.0.0.0/8 | 1 |
| 9.0.0.0/8 | 2 |
| 8.192.192.0/18 | 2 |
| 8.192.128.0/18 | 1 |
| 8.128.0.0/16 | 3 |
| 8.192.0.0/16 | 4 |
| Default | 5 |

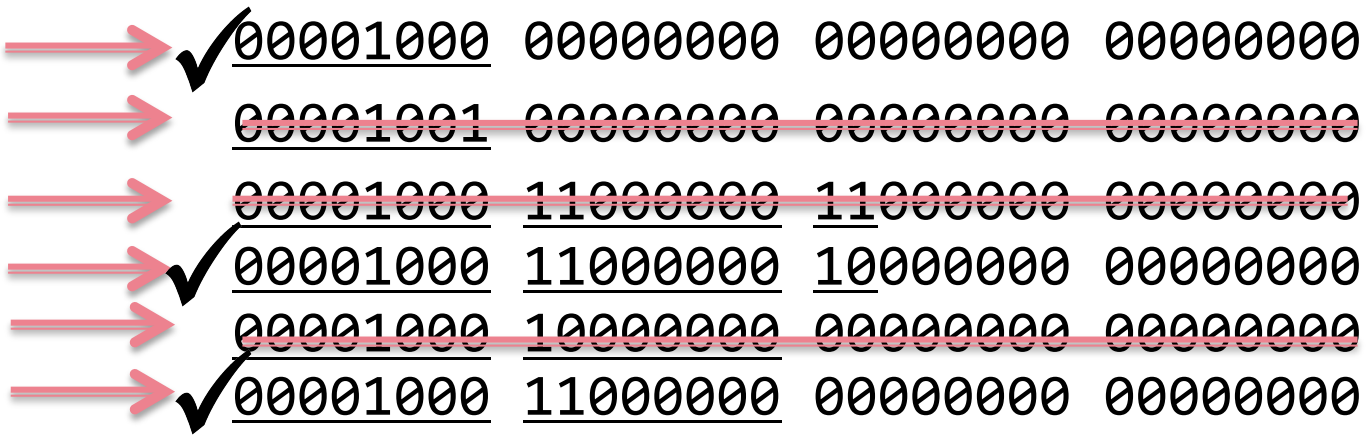



7.63.23.5 00000111 00111111 00010111 00000101

5

# 2iii)

| Prefix | Port |
|---|---|
| 8.0.0.0/8 | 1 |
| 9.0.0.0/8 | 2 |
| 8.192.192.0/18 | 2 |
| 8.192.128.0/18 | 1 |
| 8.128.0.0/16 | 3 |
| 8.192.0.0/16 | 4 |
| Default | 5 |

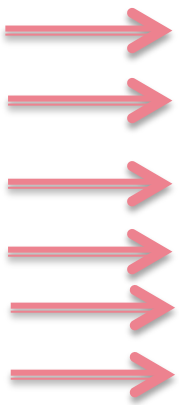



8.192.130.43 00001000 11000000 10000010 00101011

1

# 2iv)

| Prefix | Port |
|---|---|
| 8.0.0.0/8 | 1 |
| 9.0.0.0/8 | 2 |
| 8.192.192.0/18 | 2 |
| 8.192.128.0/18 | 1 |
| 8.128.0.0/16 | 3 |
| 8.192.0.0/16 | 4 |
| Default | 5 |



00001000 00000000 00000000 00000000

00001001 00000000 00000000 00000000

00001000 11000000 11000000 00000000

00001000 11000000 10000000 00000000

00001000 10000000 00000000 00000000

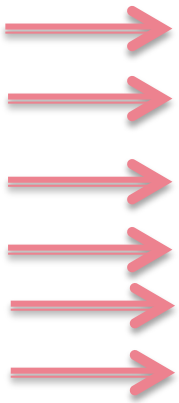
00001000 11000000 00000000 00000000

8.178.54.3 00001000 10110010 00110110 00000011

1

# 2v)

| Prefix | Port |
|---|---|
| 8.0.0.0/8 | 1 |
| 9.0.0.0/8 | 2 |
| 8.192.192.0/18 | 2 |
| 8.192.128.0/18 | 1 |
| 8.128.0.0/16 | 3 |
| 8.192.0.0/16 | 4 |
| Default | 5 |



00001000 00000000 00000000 00000000

00001001 00000000 00000000 00000000

00001000 11000000 11000000 00000000

00001000 11000000 10000000 00000000

00001000 10000000 00000000 00000000

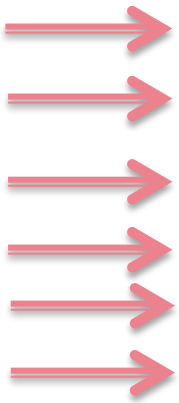
00001000 11000000 00000000 00000000

8.192.200.14 00001000 11000000 11010000 00001110

2

# 2vi)

| Prefix | Port |
|---|---|
| 8.0.0.0/8 | 1 |
| 9.0.0.0/8 | 2 |
| 8.192.192.0/18 | 2 |
| 8.192.128.0/18 | 1 |
| 8.128.0.0/16 | 3 |
| 8.192.0.0/16 | 4 |
| Default | 5 |



00001000 00000000 00000000 00000000

00001001 00000000 00000000 00000000

00001000 11000000 11000000 00000000

00001000 11000000 10000000 00000000

00001000 10000000 00000000 00000000

00001000 11000000 00000000 00000000

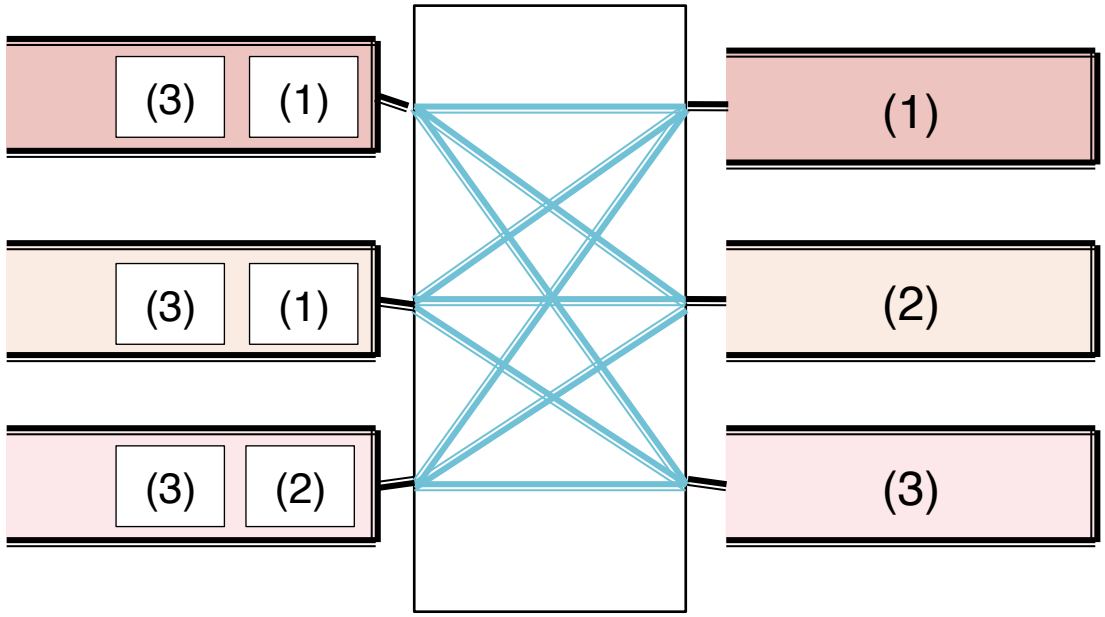
8.0.192.0 00001000 00000000 11000000 00000000

1

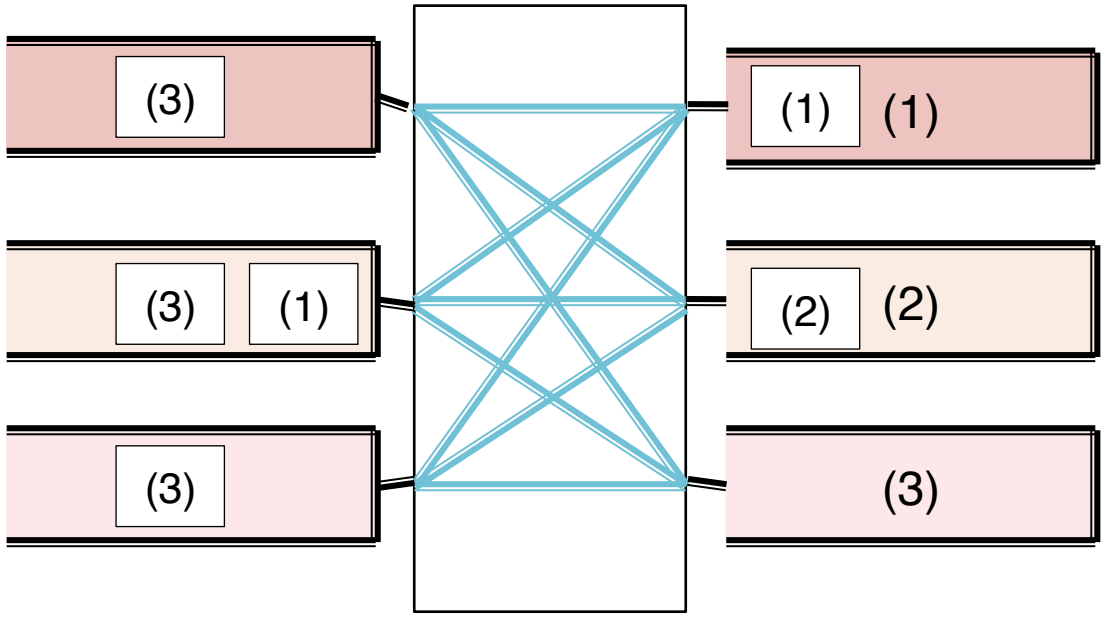
# Head of Line Blocking

- Tips and Tricks:
  - Draw out your queues with the packets in them.
  - Cross out all of the packets that can go across the interconnect in a single round.
  - Mark "1"
  - Repeat with the remaining packets, marking 2 after you've "scheduled" one round.
  - Repeat again: 3, 4, etc until all packets are gone.

# 3i)

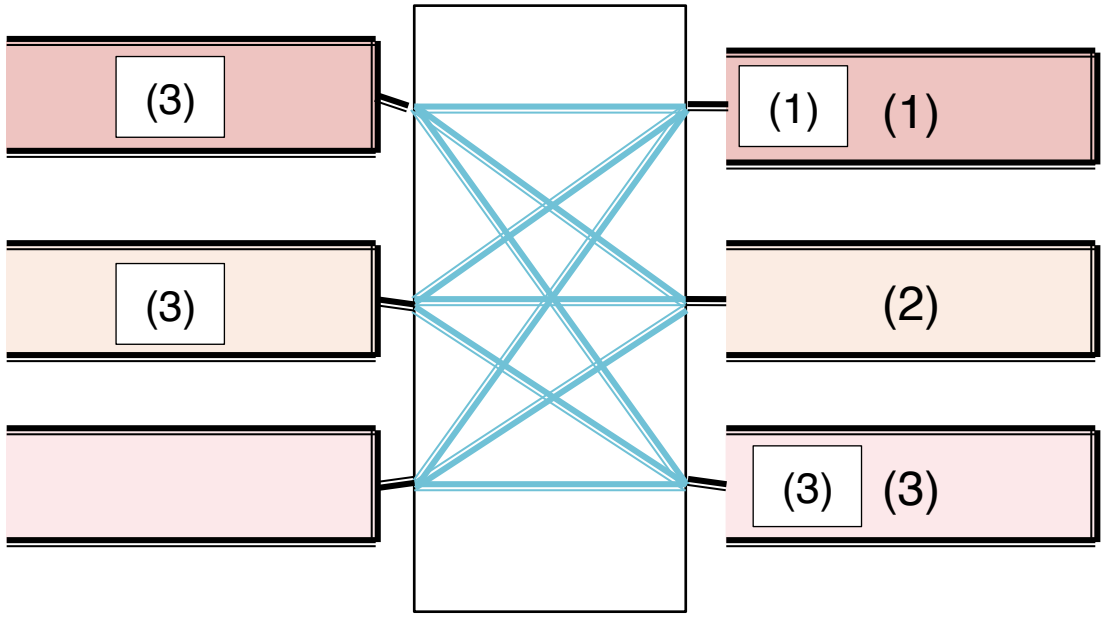


# 3i)



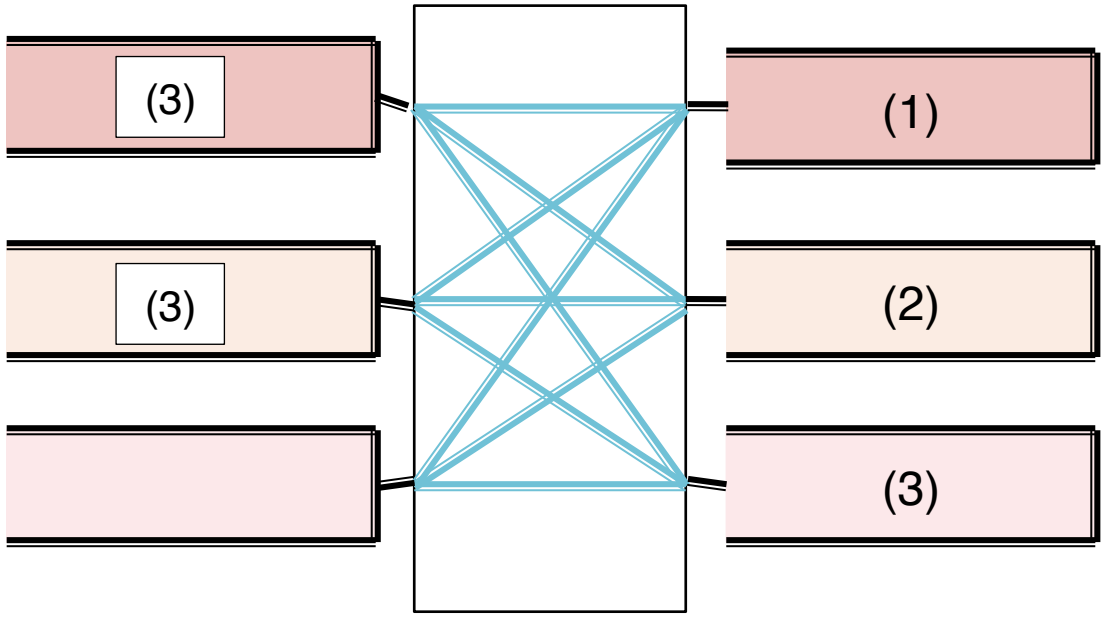

(1)

# 3i)



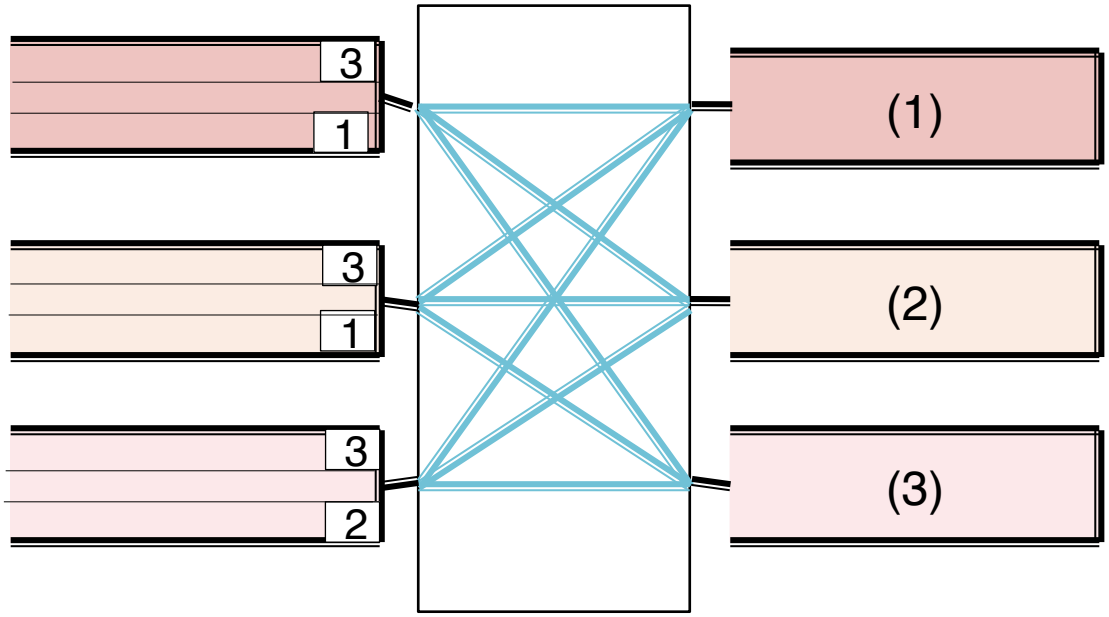

(1,2)

# 3i)

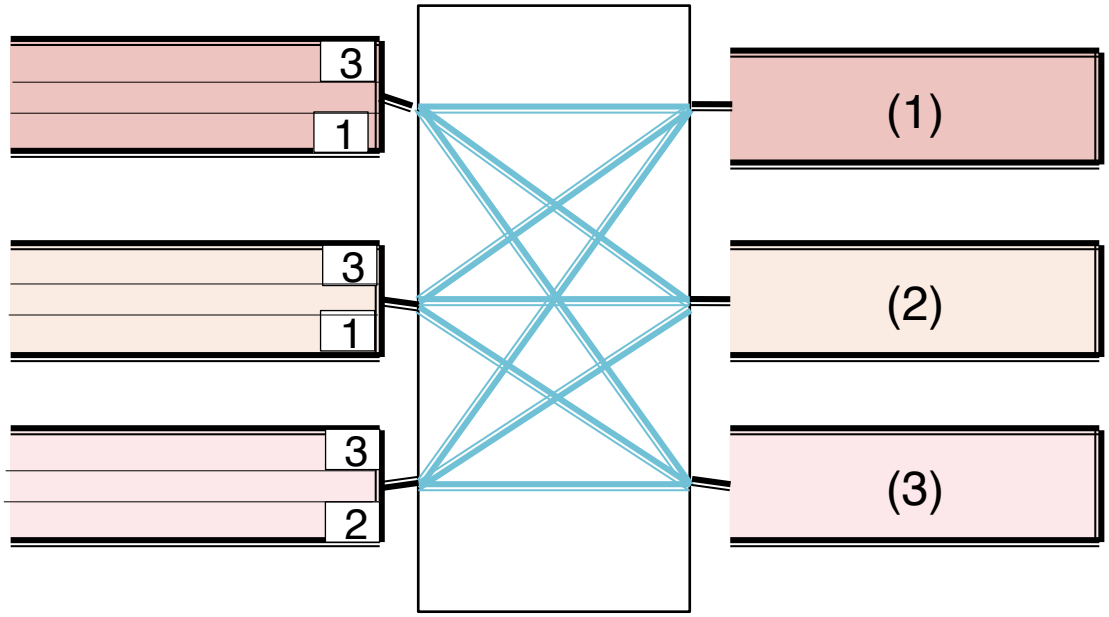



(1,2,3,4)

# 3ii)

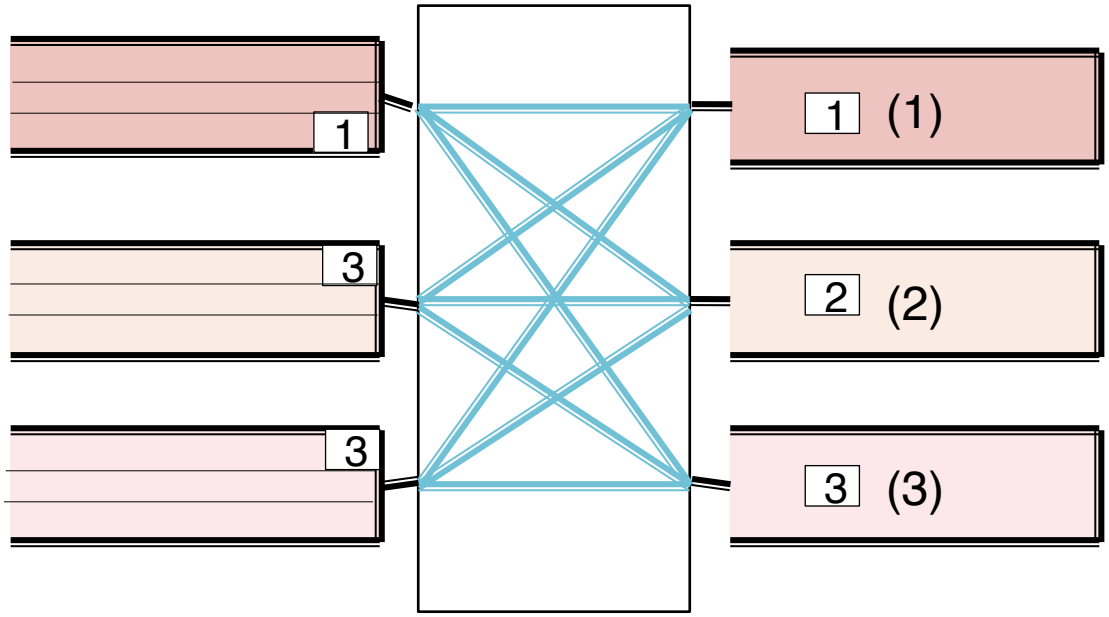


# 3iii)



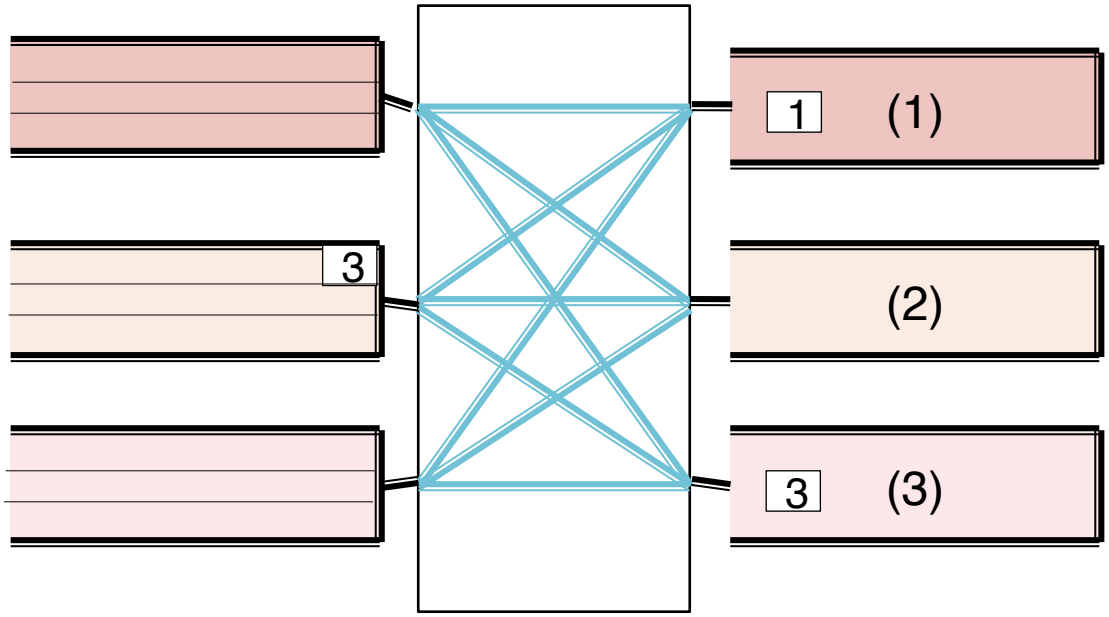

(1)

# 3iii)



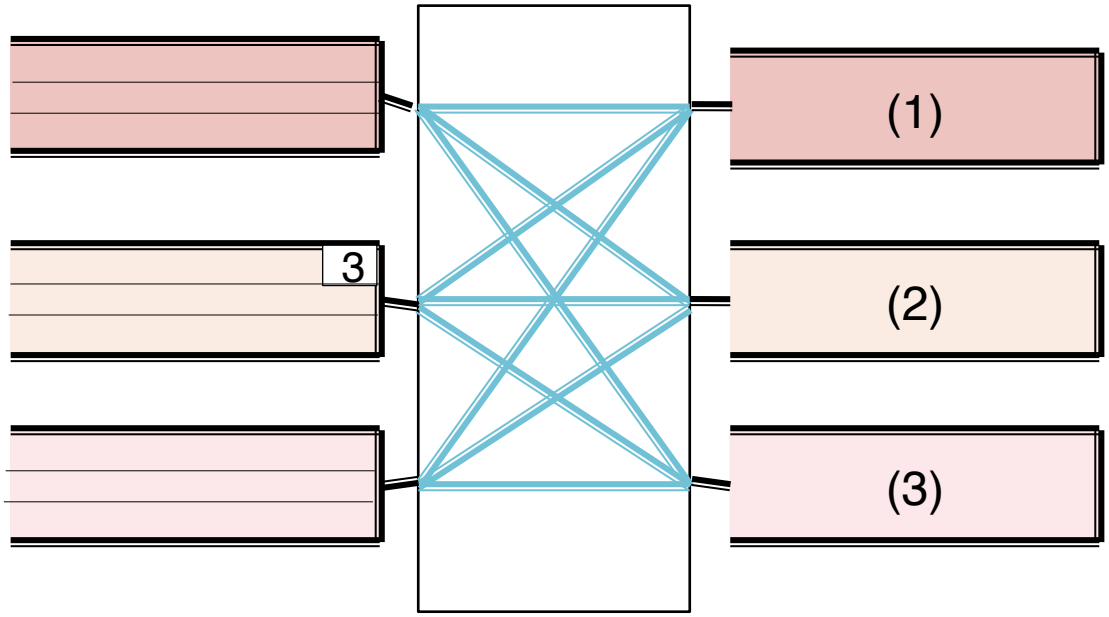

(1)

# 3iii)




(1,2)

# 3iii)




(1,2,3)