

A

1

Setting the pin numbering mode – bcm or board numbering

2

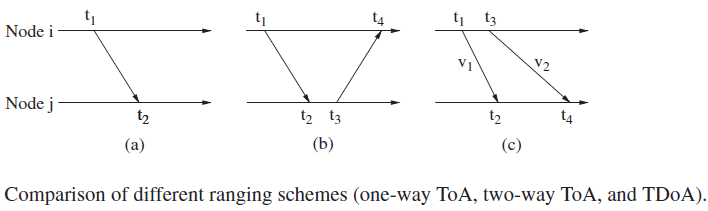
By default, TCP sends out N segments immediately, and then starts waiting. N is defined by congestion-window or receive-window, whichever is smaller. Once receives gets a cumulative ACK of *m* segments (m < N), it sends out another *m* segments. So effectively, N packets are always in-flight (sent, waiting for acknowledgement). Purpose of this mechanism is to increase the throughput as high as possible.

IoT devices do not have enough memory to maintain a window of size N. Also, throughput is not a concern in many IoT applications. Therefore, the sender uses a window size of 1, which makes it a stop-and-wait methodology.

3

Answers will vary, see L1p, slides 6-12.

4

t1 = 5, t4 = 5.3, (t3–t2) = 0.1

distance = = 0.1

B

1

Problem is that cleanup() call will never run if the program crashes half way though due to an exception.

2

TCP ACKs are not immediately sent, receiver waits 200ms for another segment to arrive so that it can send a cumulative ACK for both. If no other segment arrives within 200ms, the first segment is ACKed. Purpose of this scheme is to halve the number of ACKs sent in a busy TCP connection.

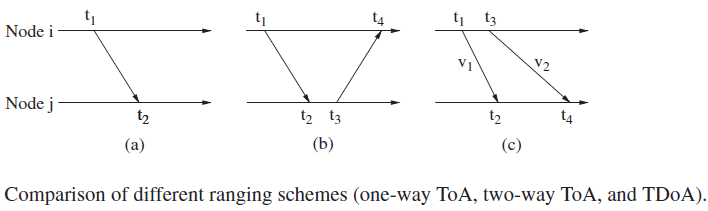
In IoT, most senders operate in stop-and-wait fashion, by not sending the next segment unless previous ACK has arrived. Delayed ACKs will significantly slow down the data rate in such scenario.

3

Sensor networks often monitor critical infrastructure or carry sensitive information, making them desirable targets for attacks.

OR

Conventional security techniques are often not feasible due to their computational, communication, and storage requirements.

4

Refer to figure, the wait time (t3-t1) is given to be zero. Distance is simply the difference of speeds multiply difference in arrival time

Distance = (350–300) × (6.5–6)