Network Chapter#  20
TRANSPORT PROTOCOLS
Network Chapter # 20
TRANSPORT PROTOCOLS

Review Questions:

20.1. What addressing elements are needed to specify a target transport service (TS) user?

20.2. Describe four strategies by which a sending TS user can learn the address of a receiving TS user.
   (1) The TS user knows the address it wishes to use ahead of time. This is basically a system configuration function. For example, a process may be running that is only of concern to a limited number of TS users, such as a process that collects statistic on performance. From time to time, a central network management routine connects to the process to obtain the statistics. These processes generally are not, and should not be, well known and accessible to all.
   (2) Some commonly used services are assigned "well-known addresses.
   (3) A name server is provided. The TS user requests a service by some generic or global name. The request is sent to the name server, which does a directory lookup and returns an address. The transport entity then proceeds with the connection. This service is useful for commonly used applications that change location from time to time.
   (4) In some cases, the target user is to be a process that is spawned at request time. The initiating user can send a process request to a well-known address. The user at that address is a privileged system process that will spawn the new process and return an address.

20.3. Explain the use of multiplexing in the context of a transport protocol.
With respect to the interface between the transport protocol and higher-level protocols, the transport protocol performs a multiplexing/demultiplexing function. That is, multiple users employ the same transport protocol and are distinguished by port numbers or service access points. The transport entity may also perform a downward multiplexing function with respect to the network services that it uses.

20.4. Briefly describe the credit scheme used by TCP for flow control.
The credit scheme decouples acknowledgment from flow control. In a credit scheme, a segment may be acknowledged without granting new credit, and vice versa. For the credit scheme, each individual octet of data that is transmitted is considered to have a unique sequence number. In addition to data, each transmitted segment includes in its header three fields related to flow control: sequence number (SN), acknowledgment number (AN), and window (W).

20.5. What is the key difference between the TCP credit scheme and the sliding-window flow control scheme used by many other protocols, such as HDLC?
In a sliding-window scheme, acknowledgment and flow control are bound together. An acknowledgment results in a fixed additional credit being granted.

20.6. Explain the two-way and three-way handshake mechanisms.
Two-way: A connection establishment calls for the exchange of SYNs, a procedure sometimes referred to as a two-way handshake. Suppose that A issues a SYN to B. It expects to get a SYN back, confirming the connection.
Three-way: As part of connection establishment, each side acknowledges explicitly the other's SYN and sequence number.

20.7. What is the benefit of the three-way handshake mechanism?
It solves the duplicate SYN problem, in which an obsolete SYN arrives after the close of a connection.

20.8. Define the urgent and push features of TCP.
**Data stream push:** Ordinarily, TCP decides when sufficient data have accumulated to form a segment for transmission. The TCP user can require TCP to transmit all outstanding data up to and including that labeled with a push flag. On the receiving end, TCP will deliver these data to the user in the same manner. A user might request this if it has come to a logical break in the data.

**Urgent data signaling:** This provides a means of informing the destination TCP user that significant or "urgent" data is in the upcoming data stream. It is up to the destination user to determine appropriate action.

20.9. What is a TCP implementation policy option?
The TCP standard provides a precise specification of the protocol to be used between TCP entities. However, certain aspects of the protocol admit several possible implementation options. These options are defined in the TCP standard. Although two implementations that choose alternative options will be interoperable, there may be performance implications.

20.10. How can TCP be used to deal with network or internet congestion?
The TCP flow control mechanism can be used to recognize the onset of congestion (by recognizing increased delay times and dropped segments) and to react by reducing the flow of data. If many of the TCP entities operating across a network exercise this sort of restraint, internet congestion is relieved.

20.11. What does UDP provide that is not provided by IP?
UDP provides the source and destination port addresses and a checksum that covers the data field. These functions would not normally be performed by protocols above the transport layer. Thus UDP provides a useful, though limited, service.

**Problems:**

20.1 It is common practice in most transport protocols (indeed, most protocols at all levels) for control and data to be multiplexed over the same logical channel on a per-user-connection basis. An alternative is to establish a single control transport connection between each pair of communicating transport entities. This connection would be used to carry control signals relating to all user transport connections between the two entities. Discuss the implications of this strategy?
A single control channel implies a single control entity that can manage all the resources associated with connections to a particular remote station. This may allow more powerful resource control mechanisms. On the other hand, this strategy requires a substantial number of permanent connections, which may lead to buffers or state table overhead.

20.2 The discussion of flow control with a reliable network service referred to a backpressure mechanism utilizing a lower-level flow control protocol. Discuss the disadvantages of this strategy.
It is relatively sluggish and may unnecessarily stress the network layer.
20.4 Someone posting to comp.protocols.tcp-ip complained about a throughput of 120 kbps on a 256-kbps link with a 128-ms round-trip delay between the United States and Japan, and a throughput of 33 kbps when the link was routed over a satellite.

a. What is the utilization over the two links? Assume a 500-ms round-trip delay for the satellite link.

For the first case, the utilization is $\frac{120}{256} = 47\%$.

For the second case (satellite), the utilization is $\frac{33}{256} = 13\%$.

b. What does the window size appear to be for the two cases?

\[
ER = \frac{W}{RTT} \\
W = ER \times RTT
\]

For the first case $W = (120 \times 10^3) \times (128 \times 10^{-3}) = 15360$ bits = 1920 bytes

For the second case $W = (33 \times 10^3) \times (500 \times 10^{-3}) = 16500$ bits = 2063 bytes

c. How big should the window size be for the satellite link?

We want $ER = \frac{256 \times 10^3}{500 \times 10^{-3}} = \frac{W}{500 \times 10^{-3}}$

$W = 128,000$ bits = 16,000 bytes.

20.6 With a reliable sequencing network service, are segment sequence numbers strictly necessary? What, if any, capability is lost without them?

No. They do make it easier to implement flow control in a manner that is extensible to unreliable and/or nonsequencing networks.

20.7 Consider a connection-oriented network service that suffers a reset. How could this be dealt with by a transport protocol that assumes that the network service is reliable except for resets?

When a reset occurs, the transport entity may have a number of outstanding segments that have not been acknowledged. The entity does not know if they were received by the other side before the network connection went down. This uncertainty must be resolved during the resynchronization procedure.

20.8 The discussion of retransmission strategy made reference to three problems associated with dynamic timer calculation. What modifications to the strategy would help to alleviate those problems?

There is no good solution if the delay experienced has high variance. One approach is to use an exponential decay smoothing algorithm discussed in this chapter.

20.9 Consider a transport protocol that uses a connection-oriented network service. Suppose that the transport protocol uses a credit allocation flow control scheme, and the network protocol uses a sliding-window scheme. What relationship, if any, should there be between the dynamic window of the transport protocol and the fixed window of the network protocol?

This will depend on whether multiplexing or splitting occurs.

- If there is a one-to one relationship between network connections and transport connections, then it will do no good to grant credit at the transport level in excess of the window size at the network level.
- If one transport connection is split among multiple network connections (each one dedicated to that single transport connection), then a practical upper bound on the transport credit is the sum of the network window sizes.
- If multiple transport connections are multiplexed on a single network connection, their aggregate credit should not exceed the network window size.

Furthermore, the relative amount of credit will result in a form of priority mechanism.
20.10 In a network that has a maximum packet size of 128 bytes, a maximum packet lifetime of 30 s, and an 8-bit packet sequence number, what is the maximum data rate per connection? A sender may not send more than 256 packets; that is,

\[
\text{# of bits} = 256 \times 128 \times 8 = 262,144 \text{ bits}
\]

The data rate = 262,144 bits / 30 sec = 8738 bps.

20.14 What happens in Figure 20.3 if a SYN comes in while the requested user is in CLOSED? Is there any way to get the attention of the user when it is not listening? The transport entity could interrupt the user to notify it of a pending request. The user could then move into the LISTEN state. An alternative would be to implement an Accept command, which would allow the user to move to ESTAB directly. The transport entity could also queue the request or discard it.

20.15 In discussing connection termination with reference to Figure 20.8, it was stated that in addition to receiving an acknowledgement of its FIN and sending an acknowledgement of the incoming FIN, a TCP entity must wait an interval equal to twice the maximum expected segment lifetime (the TIME WAIT state). Receiving an ACK to its FIN assures that all of the segments it sent have been received by the other side. Sending an ACK to the other side’s FIN assures the other side that all its segments have been received. Give a reason why it is still necessary to wait before closing the connection. The connection is held in limbo to allow all connection messages (e.g., late duplicates) that may still exit in the network to arrive or be discarded. If any messages arrive, TCP will know that they belong to a defunct connection and will discard them.

20.17 Suppose the round-trip time (RTT) between two hosts is 100 ms, and that both hosts use a TCP window of 32 Kbytes. What is the maximum throughput that can be achieved by means of TCP in this scenario?

TCP Throughput \( \leq \) TCP Window / RTT
TCP Throughput \( \leq \) 32Kbyte / 100ms = 320 Kbytes/sec.

The maximum throughput TCP can achieve in this scenario is 320 Kbytes/sec.

20.18 Suppose two hosts are connected with each other by a means of a 100 Mbps link, and assume the round-trip time (RTT) between them is 1 ms. What is the minimum TCP window size that would let TCP achieve the maximum possible throughput between these two hosts? (Note: Assume no overhead.)

TCP Window \( \geq \) Bandwidth \times RTT
TCP Window \( \geq \) 100Mbps \times 1ms

20.21 Jacobson’s congestion control algorithm assumes most packet losses are caused by routers dropping packets due to network congestion. However, packets may be also dropped if they are corrupted in their path to destination. Analyze the performance of TCP in a such lossy environment, due to Jacobson’s congestion control algorithm.

In such a lossy environment, packets would be dropped because of packets being corrupted, rather than as a result of network congestion. While it makes sense to reduce the rate at which packets are sent when congestion occurs, there is no reason to reduce the rate at which packets are sent when packets are dropped due to errors. Thus, Van Jacobson’s congestion control algorithm will cause TCP to perform badly in such a lossy environment.
20.22 One difficulty with the original TCP SRTT estimator is the choice of an initial value. In the absence of any special knowledge of network conditions, the typical approach is to pick an arbitrary value, such as 3 seconds, and hope that this will converge quickly to an accurate value. If this estimate is too small, TCP will perform unnecessary retransmissions. If it is too large, TCP will wait a long time before retransmitting if the first segment is lost. Also, the convergence may be slow, as this problem indicates.

a. Choose \( \alpha = 0.85 \) and \( \text{SRTT}(0) = 3 \) seconds, and assume all measured RTT and no packet loss. What is \( \text{SRTT}(19) \)? *Hint:* Equation (20.3) can be rewritten to simplify the calculation, using the expression
\[
\text{SRTT}(n) = \alpha \times \text{SRTT}(0) + (1 - \alpha) \times \text{RTT} \times (\alpha^{n-1} + \alpha^{n-2} + \ldots + 1)
\]
\[
= \alpha \times \text{SRTT}(0) + (1 - \alpha) \times \text{RTT} \times (\alpha^n)/(1 - \alpha)
\]

\( \text{SRTT}(19) = 1.1 \text{ sec} \)

b. Now let \( \text{SRTT}(0) = 1 \) second and assume measured RTT values = 3 seconds and no packet loss. What is \( \text{SRTT}(19) \)?
\( \text{SRTT}(19) = 2.9 \text{ sec} \)

In both cases, the convergence speed is slow, because in both cases, the initial \( \text{SRTT}(0) \) is improperly chosen.

20.27 A TCP entity opens a connection and uses slow start. Approximately how many round-trip times are required before TCP can send \( N \) segments.
TCP initializes the congestion window to 1, sends an initial segment, and waits. When the ACK arrives, it increases the congestion window to 2, sends 2 segments, and waits. When the 2 ACKs arrives, they each increase the congestion window by one, so that it can send 4 segments. In general, it takes \( \log_2 N \) round trips before TCP can send \( N \) segments.

20.28 Although slow start with congestion avoidance is an effective technique for coping with congestion, it can result in long recovery times in high-speed networks, as this problem demonstrates.

a. Assume a round-trip time of 60 ms (about what might occur across a continent) and a link with an available bandwidth of 1 Gbps and a segment size of 576 octets. Determine the window size needed to keep the pipe full and the time it will take to reach that window size after a timeout using Jacobson’s approach.
\[
W = (10^9 \times 0.06)/(576 \times 8) \approx 13,000 \text{ segments}
\]

b. Repeat (a) for a segment size of 16 Kbytes.
\[
W = (10^9 \times 0.06)/(16,000 \times 8) \approx 460 \text{ segments}
\]