Part 5: Transport Protocols for Wireless Mobile Networks

1. Functionality of Transport Protocols

Transport layer resides between the application layer and the network layer and has the critical role of providing communication services directly to the application processes running on different hosts. There are mainly two transport layer protocols: UDP and TCP in the Internet. UDP provides unreliable, connectionless transport service, while TCP provides reliable, connection-oriented transport service.

[Please review the concepts of reliable/unreliable, connection-oriented/connectionless services by checking any networks textbook]

2. Introduction to TCP

In this course, we will mainly focus on the TCP protocol. TCP adds a great deal of functionality to the IP service it is layered over:

1) **Streams.** TCP data is organized as a stream of bytes, much like a file. The datagram nature of the network is concealed. A mechanism (the *Urgent Pointer*) exists to let out-of-band data be specially flagged.
2) **Reliable delivery.** Sequence numbers are used to coordinate which data has been transmitted and received. TCP will arrange for retransmission if it determines that data has been lost.
3) **Congestion control.** TCP will dynamically learn the delay/loss characteristics of a network and adjust its operation to maximize throughput without overloading the network.
4) **Flow control.** TCP manages data buffers, and coordinates traffic so its buffers will never overflow. Fast senders will be stopped periodically to keep up with slower receivers.

Sequence Numbers

TCP uses a 32-bit *sequence number* that counts bytes in the data stream. Each TCP packet contains the starting sequence number of the data in that packet, and the sequence number (called the *acknowledgment number*) of the last byte received from the remote peer. With this information, a sliding-window protocol is implemented. Forward and reverse sequence numbers are completely independent, and each TCP peer must track both its own sequence numbering and the numbering being used by the remote peer.

Window Size and Buffering

Each endpoint of a TCP connection will have a buffer for storing data that is transmitted over the network before the application is ready to read the data. This lets network transfers take place while applications are busy with other processing, improving overall performance.

To avoid overflowing the buffer, TCP sets a *Window Size* field in each packet it transmits. This field contains the amount of data that may be transmitted into the buffer. If this number falls to zero, the remote TCP can send no more data. It must wait until buffer space becomes available and it receives a packet announcing a non-zero window size.

Round-Trip Time Estimation

When a host transmits a TCP packet to its peer, it must wait a period of time for an acknowledgment. If the reply does not come within the expected period, the packet is assumed to have been lost and the data is retransmitted. The obvious question - How long do we wait? - lacks a simple answer. Over an Ethernet, no more than a few microseconds should be needed for a reply. If the traffic must flow over the wide-area Internet, a second or two might be reasonable during peak utilization times. If we're talking to an instrument package on a satellite hurtling toward Mars, minutes might be required before a reply. There is no one answer to the question - How long?

All modern TCP implementations seek to answer this question by monitoring the normal exchange of data packets and developing an estimate of how long is "too long". This

process is called Round-Trip Time (RTT) estimation. RTT estimates are one of the most important performance parameters in a TCP exchange, especially when you consider that on an indefinitely large transfer, *all* TCP implementations eventually drop packets and retransmit them, no matter how good the quality of the link. If the RTT estimate is too low, packets are retransmitted unnecessarily; if too high, the connection can sit idle while the host waits to timeout.

Flow Control

Flow control is a speed matching service- matching the rate at which the sender is sending to the rate at which the receiving application is reading. Please keep in mind that flow control is not congestion control. (Unfortunately, some careless authors use the term interchangeable in their papers or books.)

TCP provide flow control by having the sender maintain a variable called the "receive window." Informally, the receive window is used to give the sender an idea about how much free buffer space is available at the receiver.

Congestion Control

TCP makes assumption that when packet losses happen, there must be congestion in some intermediate routers. Based on this assumption, TCP throttles senders in the face of network congestion. TCP congestion control has four main parts: Slow Start (SS), Congestion Avoidance (CA), Fast Retransmit, and Fast Recovery.

The details of the above mechanisms should be fully understood. Please check Internet draft. (http://www.faqs.org/rfcs/rfc793.html).

3. Problems of TCP over Wireless Networks

The wired networks are relatively reliable as compared to the wireless links and so TCP assumes congestion to be the main cause of any packet loss, and invokes congestion control measures at the source. Wireless links (part of a heterogenous network) bring some serious issues with them:

1) Bit Error Rate (BER): Wireless hosts use radio transmission or infrared wave transmission for communication. The BER of wireless links is typically higher than that of wired networks. Also the wireless environment changes quickly, and so the BER also varies by a large amount.
2) Bandwidth: Wireless links offer very less bandwidth as compared to the wired links. Optimum use of available bandwidth is a major issue that has to be taken care of.
3) Round Trip Time (RTT): The wireless media exhibits longer latencies than wired media. This affects overall throughput and increases interactive delays. Different versions of TCP pour data in the network depending on the incoming acknowledgements, which depends on RTT. Thus networks (especially WANs) with high bandwidth-delay product are severely affected.
4) Mobility: Addition of mobile devices introduces huge amount of indeterminate mobility in rather a stationary network. This tends to introduce some amount of instability in existing network topology.
5) Power consumption: Mobile host has limited power and smaller processing capacity as compared to base stations, which tend to introduce inefficiency in the network. Solutions that take power consumption into account have a clear-cut advantage over the otherwise designed solutions.

The fundamental problem is the *underestimation of bandwidth* by the network endpoints which results in reduced application layer performance, reductions in throughput and unacceptable delays. When this happens the applications don't get their fair share of the bottleneck link's bandwidth. The reason is that any packet loss in wireless networks, is mistakenly taken as congestion by TCP, thereby activating the slow-start algorithm and reducing the window size to one segment. It also resets the retransmission timer to a back off interval that doubles with each successive timeout.

4. Possible Solutions to Improve TCP over Wireless Networks

This part does not intend to provide a thorough list of solutions. As stated in the class, this research area is pretty crowded: a lot of people are digging on this small land. It is impossible to list all of them. Occasionally, some brilliant idea pops up although most work only provides small variance over existing solutions.

Various solutions can be classified in the following way:

1) **Pure Link-level Approaches:** These approaches aim at hiding the unwanted characteristics of the wireless links from the higher layers. These schemes almost solve the problems caused by the wireless links but a critical factor is the determination of the link-level timeout value. Also these schemes need to be backed by some other techniques as they by themselves cannot completely avoid the sender from timing out and invoking the congestion control mechanisms. In this approach, reliable link-level protocols are implemented on the wireless link which perform local retransmissions to improve the reliability of communication independent of the higher-level protocols.
2) **Soft-state Transport Layer Caching Approaches:** These approaches maintain a "soft" state, i.e., it is not crucial for the end-to-end connection. These are aware of the transport layer and use caching as a technique to save the sender from unnecessary invocation of the congestion control mechanism. The disadvantage here is that they require changes at the intermediate node (base station) and optionally at the mobile host. They fail in the presence of encryption due to the intermediate node's dependence on the interpretation of the transport header.
3) **Soft-state Cross Layer Signaling Approaches:** These approaches make the transport layer sender aware of the wireless link. This can be achieved by having the link or the network layer inform the transport layer sender about specific events so that it can adapt accordingly. The advantage here is that they cleanly separate the congestion losses from the error losses but this may

not be possible if the network is extremely congested in which case it is not possible for the network to inform the sender through any kind of signaling. They also involve changes at some or all of the intermediate nodes and at the transport layer of the sender's protocol stack for taking appropriate actions in response to the signals received from the network.

4) **Hard-state Transport Layer Approaches:** These solutions encompass all forms of splitting and the end-to-end semantics may be sacrificed. The advantage of these approaches is that the wireless link is completely shielded from damage loss. But they pay a heavy price in terms of maintaining the state that is crucial to the end-to-end connection.
5) **Pure End-to-end Transport Level Approache**s: These solutions try to solve the problem solely on the end-to-end basis. The advantage of these approaches is that there is no change or extra logic needed at the base station or the any intermediate nodes. But the transport layer of the stack at the sender needs to be modified.

[More examples for each category]

5. Last Words

As you have seen, the performance of TCP over wireless networks is poor since originally TCP is not intended for such networks. Probably a perfect solution is to re-design a totally new transport protocol for wireless mobile networks. Nevertheless, designing new protocols is practically difficult, not because of technique reasons, but because of the fact that we have to give up existing legacy applications- the majority of Internet applications use TCP and our main information source is rooted to the Internet. Ubiquitous information access provides us with more convenience and flexibility, but we have to rely on the interconnection between the Internet and wireless networks. In fact, we are bearing the same pain in other networking areas. For example, providing QoS guarantee over the Internet is hard. It appears that the brilliant idea of packet switch at 60s last century has become dog food that we have to eat.