HW1

Chenlin Liu, PID: A53284481

October 5, 2018

1 Peterson & Davie problem set

Problem 1.3

Since the RTT is the length of time it takes for a signal to be sent plus the length of time it takes for an acknowledgement of that signal to be received (from Wiki), we count the transfer without the time of transferring acknowledgement. The propagation time = RTT/2 = 25ms.

- (a). Latency = Propagation + Transmit + Queue + handshake
  Propagation = 25ms; Transmit = 1000KB/1.5Mbps $\approx$ 5333ms; handshake = 100ms
  Latency = 25ms + 5333ms + 100ms = **5457ms**.

- (b). Propagation = 25ms; handshake = 100ms; Transmit $\approx$ 5333ms;
  The sum of wait time after each packet = 999*50ms = 49950ms;
  Latency = 25ms + 5333ms + 100ms + 49950ms = **55408ms**.

- (c). handshake = 100ms; Transmit = 0
  Propagation = 1000KB/1KB/20*50ms = 2500ms;
  Latency = 2500ms + 100ms = **2600ms**.

- (d). handshake = 100ms; Transmit = 0
  $2^0 + 2^1 + 2^2 + ... + 2^m \geq 1000 \Rightarrow m \geq 9$; #RTT = 9;
  Latency = 100ms + 9*50ms = **550ms**.

Problem 1.5

- For 100 byte packets, propagation = 4km/(2×10^8m/s) = 2×10^{-5}s Assume the bandwidth is x Mbps, then 100byte/xMbps = 2×10^{-5}s
  x = **40Mbps**.

- For 512 byte packets
  Assume the bandwidth is x Mbps, then 512byte/xMbps = 2×10^{-5}s
  x = **204.8Mbps**.

Problem 1.13

- (a). RTT = 2$\times$385000km/(3×10^8m/s) $\approx$ 2566ms.

- (b). delay $\times$ bandwidth product = 2566ms $\times$ 1Gbps = 2.566 Giga-bits = 320.75MB.

- The delay $\times$ bandwidth product measures the maximum amount of data on this link. The link with a larger delay $\times$ bandwidth product has a better transmitting ability.
- Latency = Propagation + Transmit;
  Transmit = 25MB / 1Gbps = 0.2s = 200ms;
  Latency = 1283ms + 200ms = 1483ms.

Problem 1.20

- (a). Latency(A $\rightarrow$ S) = Propagation + Transmit = 20μs + 10000bits/100Mbps = 120μs;
  Latency(S $\rightarrow$ B) = Retransmitting + Propagation + Transmit = 35μs + 20μs + 100μs= 155μs;
  Total latency = 120μs + 155μs = 275μs.
- (b). Since there are 2 packets, an extra retransmitting delay should be added to the latency, thus the total latency equals to 275μs + 35μs = 310μs.

Problem 1.22

- 10,000 bytes is better. Firstly, comparing with 1,000 bytes, if the size of packets is too small, a 50-bytes overhead per packet seems very wasteful. Moreover, comparing with 20,000 bytes, if the size of packets is too large, the data loss caused by loss bytes will be unfortunately huge. Therefore, the packet size of 10,000 bytes is better.

Problem 1.23

Suppose: 6 point-to-point links and 5 switches. Assume the file is x*1000B, n = 1000x.

- (a). For circuit switching:
  Total size: x*1000B + 1KB(setup) = (x+1)*1000B
  For packet switching:
  Total size: x*1000B + x*1000B/1000B*24B(header) = x*1024B
  When circuit $\leq$ packet:
  (x+1)*1000B $\leq$ x*1024B $\Rightarrow$ x $\geq$ 42.667 $\Rightarrow$ n $\geq$ **42667**.
- (b). For circuit switching:
  Circuit setup: 1KB/4Mbps + 5*1ms(circuit setup) + 6*2ms(propagation) = 19ms
  Data transmit: x*1000B/4Mbps + 6*2ms(propagation) = (2x+12)ms
  Total time: (2x+31)ms
  For packet switching:
  Total data size including packet header: x*1000B/1000B*24B + x*1000B = x*1024B
  Total time: x*1024B/4Mbps + 6*2ms(propagation) + (5*x)ms(switch retransmitting) = (7.048x+12)ms
  When circuit $\leq$ packet:
  2x+31 $\leq$ 7.048x+12 $\Rightarrow$ x $\geq$ 3.764 $\Rightarrow$ n $\geq$ **3764**.
- (c).Assume the #switches is s, the bandwidth is b-Mbps, the ratio of packet size to packet header size is r.
  For problem (a):
  (x+1)*1000B $\leq$ x*1000B + x/r*1000B $\Rightarrow$ x $\geq$ r.
  Thus when x $\geq$ r, the result keeps.
  For problem (b):

Circuit switching: (8/b + 3s + 2)ms + (8x/b + 2s + 2)ms = (8(1+x)/b + 5s + 4)ms
Packet switching: (8/b*(1+1/r)x + s*x + 2s + 1)ms
(8(1+x)/b + 5s + 4) $\leq$ (8/b*(1+1/r)x + s*x + 2s + 1)
$\Rightarrow$ x $\geq \frac{8*(1/b)+3s+3}{8*(1/b)*(1/r)+s}$
Thus the result is very sensitive to the ratio r, but is not very sensitive to #switches and bandwidth.

- (d). This model is not very accurate of the relative merits of circuits and packets. It ignores some important For circuit switching, there may be other links using these switches, thus the link from source to destination could be or partly be occupied. For packet switching, it ignores FRAME time loss.

2 End-to-end argument

2.1 Description

End-to-end argument is the reasoning against low-level function implementation. The reason is that the function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible.

2.2 My opinions on end-to-end argument

I partly agree the end-to-end argument. The end-to-end argument can achieve a great trade-off between correctness and performance. For today's Internet, the TPC/IP is a well-known application of end-to-end argument. However, one limitation of end-to-end argument in today's Internet is that with the increasing amount of applications, the difficulties of implementation are also rising especially for mobile network. Moreover since nowadays IPv4 relies on address translation, it is impossible to implement "real" end-to-end network over IPv4. In the future, with the help of IPv6, devices can be assigned real IP address and thus a "real" end-to-end network can be implemented.

2.3 Three low-level functions

Continues bit transfer

For nowadays Internet, many types of data stream require continuity and orderliness, especially for online video and audio files. Normal packet switching and other methods cannot ensure that users on some online video website, such as Youtube, are able to continuously receive video data. Instead, an independent methods at higher level must control the continuity and orderliness of data.

Timeout-based retransmission

In TCP, a timeout-based retransmission is applied to ensure a reliable data transmission. Whenever a packet is sent, the sender sets a timer that is a conservative estimate of when that packet will be acked. If the sender does not receive an ack by then, it transmits that packet again. The timer function can only be implemented on higher level instead of lower communication subsystems. (ref: Wiki)

Error detection

Error bits of transmitted data come from several parts of the communication system, such as hardware faults of disks, software faults of file systems and bit loss over transmission links. The large amount of switches and routers make it much too inefficient to implement error detection mechanisms low-level systems such as file systems for every switches. Therefore, an end-to-end approach of error detection is necessary.