计算机网络

# 第三次作业

1. **Explain precisely following abbreviations:**

**UDP: User Datagram Protocol（用户数据报协议）**

UDP是一种无连接的，位于网络模型的传输层的协议。它为网络应用程序提供了一种发送和接收封装的IP数据报的方式，而不需要建立稳定的连接。

**FDM: Frequency Division Multiplexing（频分复用）**

FDM是一种将一个通信信道分割成几个独立的部分，每个部分可以独立地传输信号的技术。这样，一个单一的通信信道就可以同时传输多个信号。

**ABR: Available Bit Rate（可用比特率）**

ABR是一种网络服务质量(QoS)策略，它使得数据流可以动态地调整其发送速率，以适应网络的变化情况。

**EFCI: Explicit Forward Congestion Indication（显式前向拥塞指示）**

EFCI 是一种网络拥塞控制机制，它允许网络设备在数据包级别标记出潜在的拥塞情况，以便收件人可以适当地调整其发送速率。

**AIMD: Additive Increase Multiplicative Decrease（加法增大，乘法减小）**

AIMD 是一种控制网络拥塞的算法，当网络没有拥塞时，它会逐渐增大窗口大小，而当检测到拥塞时，它则会减小窗口大小的一半。

**MSS: Maximum Segment Size（最大报文段长度）**

MSS 是 TCP 协议中的一个参数，它定义了 TCP/IP 协议栈应生成的最大 TCP 数据段大小。

## Which services can TCP and UDP provide?

UDP 和 TCP 最基本的责任是，将两个端系统间 IP 的交付服务扩展为运行在端系统上的两个进程之间的交付服务。将主机间交付扩展到进程间交付被称为运输层的多路复用与多路分解。UDP 和 TCP 还可以通过在其报文段首部中包括差错检查字段而提完整性检查。进程到进程的数据交付和差错检查是两种最低限度的运输层服务，也是UDP 所能提供的仅有的两种服务。UDP无需连接建立、无连接状态、分组首部开销小，关于发送什么数据以及何时发送的应用层控制更为精细。

TCP 为应用程序提供了几种附加服务，包括可靠数据传输、拥塞控制和流量控制。通过使用流量控制、序号、确认和定时器，TCP 确保正确地、按序地将数据从发送进程交付给接收进程。TCP 拥塞控制防止任何一条 TCP 连接用过多流量来淹没通信主机之间的链路和交换设备。TCP 力求为每个通过一条拥塞网络链路的连接平等地共享网络链路带宽，这可以通过调节 TCP 连接的发送端发送进网络的流量速率来做到。

1. **Write following TCP algorithms:**

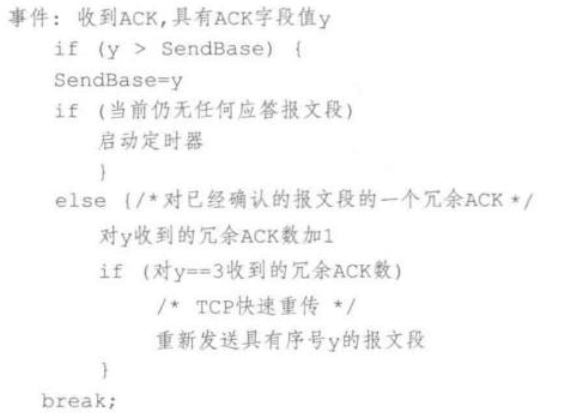
**可靠发送（Reliable sending）**

在TCP协议中，可靠发送主要依赖于“确认(Acknowledgement)与重传(Re-transmission)”机制。发送方在发送数据包后会启动一个时钟，在规定时间内如果没有收到接收方的ACK确认报文，则认为该数据包在传输过程中丢失，会进行重传。同时，发送方在接收到确认后，会更新发送窗口，继续发送后续的数据包。以下是一种简化的TCP发送方所执行的伪代码。



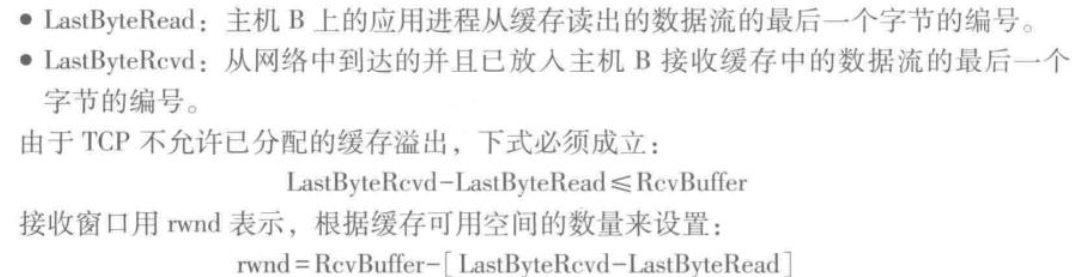
**可靠接收（Reliable receiving）**

可靠接收同样是基于“确认与重传”机制，它是由接收方来保证的。接收方在接收到数据包后，需要向发送方发送一个ACK确认报文。如果接收到的数据包在传输过程中出现了错误，则不发送ACK，此时发送方在时钟超时后会重新传送该数据包。但超时触发重传存在的问题之一是超时周期可能相对较长，这个问题可以通过快速重传即引入冗余ACK来解决。一旦收到 3 个冗余 ACK, TCP 就执行快速重传（fast retransmit）, 即在该报文段的定时器过期之前重传丢失的报文段。



**流量控制（Flow control）**

TCP 通过让发送方维护一个称为接收窗口 (receive window) 的变量来提供流量控制。接收方会告诉发送方其接收窗口的大小，这个大小是基于接收方当前可用的缓冲区大小。发送方在发送数据时，需要根据接收窗口的大小来控制发送的速度，以避免在接收方产生溢出。



**拥塞控制（Congestion control）**

拥塞控制是TCP协议为了防止网络阻塞而采取的一种控制策略，主要包括慢启动(slow-start)、拥塞避免(congestion-avoidance)和快速恢复(fast recovery)。当网络出现阻塞时，这三个算法能通过动态调整发送窗口的大小，控制数据的发送速度，从而避免网络的阻塞。慢启动和拥塞避免是 TCP 的强制部分，两者的差异在于对收到的 ACK 做出反应时增加cwnd 长度的方式。

在慢启动状态，cwnd 的值以 1 个 MSS 开始并且每当传输的报文段首次被确认就增加 1个MSS。这使得TCP 发送速率在慢启动阶段以指数增长。如果存在一个由超时指示的丢包事件，TCP 发送方将cwnd设置为1并重新开始慢启动过程，并将ssthresh置为拥塞窗口值的一半。再者，当 cwnd 的值等于ssthresh时，结束慢启动并且 TCP 转移到拥塞避免模式。最后一种结束慢启动的方式是，如果检测到3 个冗余ACK, 这时TCP 执行快速重传并进入快速恢复状态。

在拥塞避免模式，每个 RTT 只将 cwnd 的值增加一个 MSS。当出现超时时，TCP的拥塞避免算法行为相同。当收到 3 个冗余的 ACK 时，TCP 将 cwnd 的值减半, 并且将 ssthresh 的值记录为 cwnd 的值的一半，接下来进入快速恢复状态。

在快速恢复中，对于引起 TCP 进入快速恢复状态的缺失报文段，每当收到冗余的ACK，cwnd 的值增加一个 MSS。最终，当对丢失报文段的一个 ACK 到达时，TCP 在降低cwnd 后进人拥塞避免状态。如果出现超时事件，快速恢复在执行如同在慢启动和拥塞避免中相同的操作后，迁移到慢启动状态：当丢包事件出现时，cwnd 的值被设置为 1 个MSS, 并且 ssthresh 的值设置为 cwnd 值的一半。

1. **Page 315 Review problems:**

**R3. How is a UDP socket fully identified? What about a TCP socket? What is the difference between the full identification of both sockets?**

一个UDP套接字完全由目标IP地址和目标端口号这两元组来标识，TCP套接字则由一个四元组来标识，包括源IP地址、源端口号、目标IP地址和目标端口号。这意味着TCP连接是源IP和端口以及目标IP和端口之间的连接。

两个套接字全标识的主要区别在于，尽管UDP只需要标识接收器（目标IP地址和端口），而TCP需要标识发送器（源IP地址和端口）和接收器（目标IP和端口）。这反映了TCP的面向连接性和UDP的无连接性。TCP在发送者和接收者之间建立清晰的连接，而UDP只是在不建立专用端到端连接的情况下发送数据。

**R5. Why is it that voice and video traffic is often sent over TCP rather than UDP in today’s Internet? (Hint: The answer we are looking for has nothing to do with TCP’s congestion-control mechanism.)**

音视频流量通常通过TCP而不是UDP发送，原因在于TCP提供了一种可靠的数据传输机制。与UDP相比，TCP保证了数据的完整性和按顺序传输，这对于音视频传输非常重要。丢失或混乱的数据包可能导致播放中断或图像质量下降。音频和视频数据常常需要完整和有序的传输，以保证播放的质量和连贯性。

**R14. True or false?**

1. **Host A is sending Host B a large file over a TCP connection. Assume Host B has no data to send Host A. Host B will not send acknowledgments to Host A because Host B cannot piggyback the acknowledgments on data.**

错误。即使Host B没有数据要发送给Host A，它也会向Host A发送确认信息，以告诉它已经收到了数据。

1. **The size of the TCP rwnd never changes throughout the duration of the connection.**

错误。TCP rwnd的大小在连接过程中可能会改变，以控制数据的传输速度并防止接收端缓冲区溢出。

1. **Suppose Host A is sending Host B a large file over a TCP connection. The number of unacknowledged bytes that A sends cannot exceed the size of the receive buffer.**

正确。A发送给B的未确认的字节数不能超过接收缓冲区的大小，否则，B可能会丢失一些数据。

1. **Suppose Host A is sending a large file to Host B over a TCP connection. If the sequence number for a segment of this connection is m, then the sequence number for the subsequent segment will necessarily be m + 1.**

错误。在TCP连接中，一个新的数据段的序列号等于上一个段的序列号加上上一个段中的字节数。

1. **The TCP segment has a field in its header for rwnd.**

正确。TCP段的头部有一个字段用于rwnd。

1. **Suppose that the last SampleRTT in a TCP connection is equal to 1 sec. The current value of TimeoutInterval for the connection will necessarily be >= 1 sec.**

错误。TimeoutInterval由EstimatedRTT和DevRTT计算得出，而EstimatedRTT和DevRTT又由SampleRTT以一定比率加权得出，因此当前的TimeoutInterval不一定大于最后一次的SampleRTT。

1. **Suppose Host A sends one segment with sequence number 38 and 4 bytes of data over a TCP connection to Host B. In this same segment the acknowledgment number is necessarily 42.**

错误。确认号是期待接收的下一个字节的编号。

1. **Problems:**

**P1. Suppose Client A requests a web page from Server S through HTTP and its socket is associated with port 33000.**

**a. What are the source and destination ports for the segments sent from A to S?**

**b. What are the source and destination ports for the segments sent from S to A?**

**c. Can Client A contact to Server S using UDP as the transport protocol?**

**d. Can Client A request multiple resources in a single TCP connection?**

a. 源端口；（A）33000 目标端口：（S）80（HTTP默认端口）

b. 源端口；（S）80（HTTP默认端口） 目标端口：（A）33000

c. 可以。HTTP/3.0协议基于QUIC协议和UDP协议，但相对来说由于UDP本身不能提供可靠数据传输，现有的设备一般采用TCP连接和HTTP/1.1或HTTP/2.0。

d. 可以。由于TCP是基于连接的数据传输协议，只要建立了一个TCP连接且连接没有断开，发送端和接收端可以在其中发送多个报文，以传输和请求多个资源。

**P4. Assume that a host receives a UDP segment with 01011101 11110010 (we separated the values of each byte with a space for clarity) as the checksum. The host adds the 16-bit words over all necessary fields excluding the checksum and obtains the value 00110010 00001101. Is the segment considered correctly received or not? What does the receiver do?**

在UDP中，如果报文段被正确传输，全部的16比特字节（包括检验和）加在一起的结果应该是11111111 11111111。在本例中，接收端计算得到的和是01011101 11110010 + 00110010 00001101 = 10001111 11111111与预期不符，因此该报文段的传输不正确。UDP对差错恢复无能为力，接收端将丢弃这段报文或将其交给应用程序、给出警告。

**P28. Host A and B are directly connected with a 100 Mbps link. There is one TCP connection between the two hosts, and Host A is sending to Host B an enormous file over this connection. Host A can send its application data into its TCP socket at a rate as high as 120 Mbps but Host B can read out of its TCP receive buffer at a maximum rate of 50 Mbps. Describe the effect of TCP flow control.**

在本案例中，主机A可以以120Mbps的速度将应用数据读取进TCP套接字，但受限于主机A和主机B之间的100Mbps连接，TCP拥塞控制将处理和控制数据的发送速度。然而，主机B读取接收缓存的速度仅为50Mbps，这显著小于数据的发送速度，此时TCP进行流量控制以避免数据丢失。主机B需要将当前的rwnd（rwnd = RcvBuffer-[LastByteRcvd-LastByteRead]，代表了主机B的接收缓存中剩余的可用空间）在TCP报文中发送给A，而主机A将在其发送中严格控制LastByteSent-LastByteAcked <= rwnd，即当前已经发出的数据不至于使主机B的缓存溢出。如果主机B的rwnd为0，主机A将继续发送只有一个字节数据的报文段，直到主机B的缓存被清空且rwnd非零。根据以上的流量控制策略，TCP保证的数据传输不会因为处理速率差异而丢失。

**P32. Consider the TCP procedure for estimating RTT. Suppose that α = 0.1. Let SampleRTT1 be the most recent sample RTT, let SampleRTT2 be the next most recent sample RTT, and so on.**

**a. For a given TCP connection, suppose four acknowledgments have been returned with corresponding sample RTTs: SampleRTT4, SampleRTT3, SampleRTT2, and SampleRTT1. Express EstimatedRTT in terms of the four sample RTTs.**

**b. Generalize your formula for n sample RTTs.**

**c. For the formula in part (b) let n approach infinity. Comment on why this averaging procedure is called an exponential moving average.**

a. 四个报文相继到达时，EstimatedRTT将经历以下变化：

EstimatedRTT = SampleRTT4

EstimatedRTT = 0.9 \* SampleRTT4 + 0.1\* SampleRTT3

EstimatedRTT

= 0.9\*（0.9 \* SampleRTT4 + 0.1\* SampleRTT3）+ 0.1\*SampleRTT2

= 0.81\* SampleRTT4 + 0.09\* SampleRTT3 + 0.1\* SampleRTT2

EstimatedRTT

= 0.9\*（0.81\* SampleRTT4 + 0.09\* SampleRTT3 + 0.1\* SampleRTT2）+ 0.1\*SampleRTT1

= 0.729\* SampleRTT4 + 0.081\* SampleRTT3 + 0.09\* SampleRTT2 + 0.1\*SampleRTT1

b. EstimatedRTT = (0.9)^(n-1) \* SampleRTTn + (0.9)^(n-2) \* 0.1 \* SampleRTTn-1 + … + 0.9 \* 0.1 \* SampleRTT2 + 0.1\*SampleRTT1

=

c. 当n趋于无穷大时，对整个式子造成的影响将非常非常小并接近于0，且对式子造成的影响随着i的增大而指数级下降。因此这个平均被称为指数移动平均。