Chapter 3

1. 传输层功能：为主机上的应用进程提供直接（端到端）的通信服务
2. 传输层网络层的关系：网络层提供主机之间的逻辑通信，传输层为运行在不同主机上的进程之间提供逻辑通信，依赖、强化网络层服务
3. 报文段(segment)：传输层分组。数据报（datagram）：网络层分组。

数据帧（frame）:链路层分组

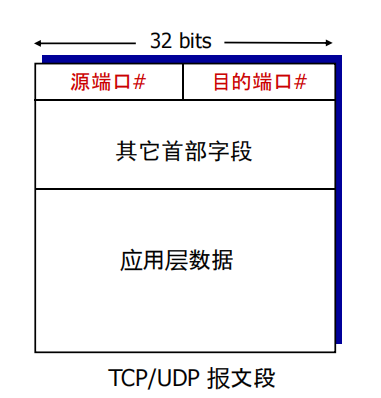
比特（位流）（bit）:物理层分组

1. 多路复用与分解

多路复用：从不同套接字收集数据块，并为每个数据块封装上首部信息，生成报文段，传递到网络层。

多路分解：将报文段中的数据交付到正确的套接字。即接收方运输层从报文段的多个字段中，识别出套接字，并将报文段定向到该套接字。

1. 报文段格式

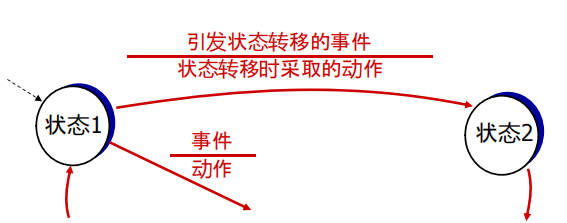
端口号：主机上的每个套接字分配一个端口号。16位（0～65535）。0～1023为周知端口号，保留给固定的应用程序。开发一个新应用时，需选择一个端口号。

自动为该套接字分配一个端口号。如 DatagramSocket mySocke = new DatagramSocket()直接为套接字指定一个特定的端口号。 如 DatagramSocket mySocke = new DatagramSocket (19157)

1. UDP主要应用：流式多媒体，DNS，SNMP
2. Checksum校验和



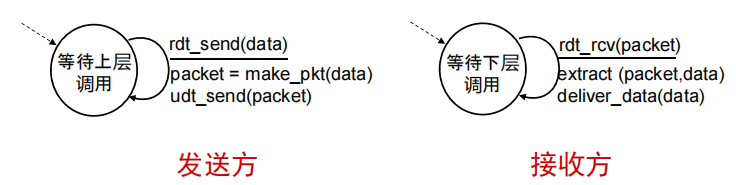
1. 有限状态机 (FSM)



9. 可靠数据传输rdt

rdt1.0可靠信道上的可靠数据传输

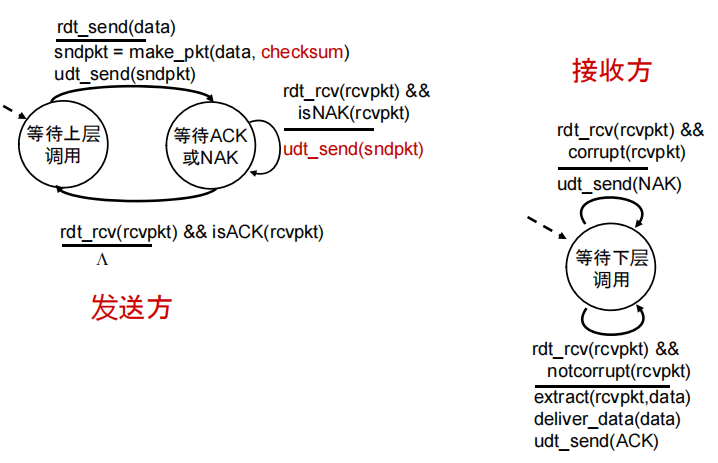
底层信道是可靠的，无比特错误，无丢包



发送方：将数据发向底层信道 从上层接受数据rdt\_send(data)，封装成分组，make\_pkt(data) 将分组发送到信道中udt\_send(packet)接收方：从底层信道读取数据 从底层信道接收一个分组rdt\_rcv(packet) ，解封取出数据extract(packet，data) ， 数据传给上层deliver\_data(data)

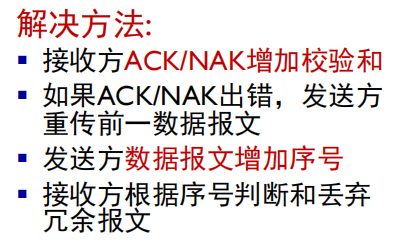
Rdt2.0

信道不完全可靠，有可能产生比特错，但不丢包采用了差错检测、肯定确认（ACK）与否定确认（NAK）及重传

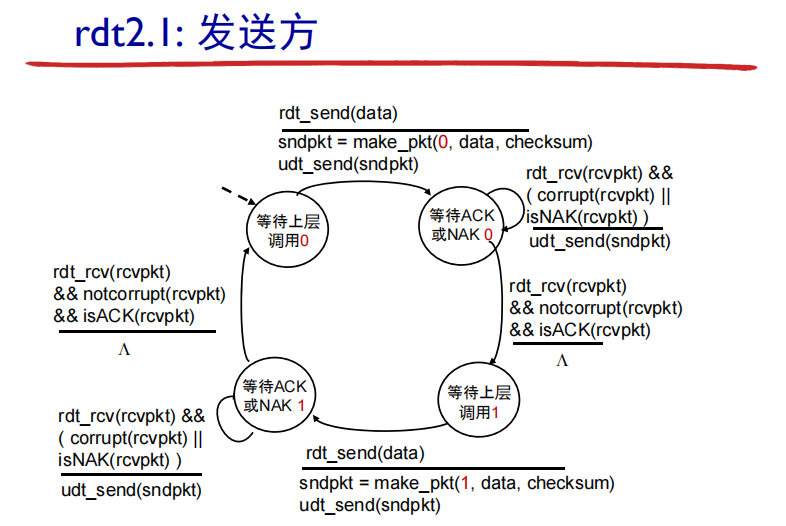
rdt2.0协议为停等协议，发送方发完一个分组后停止，等待对方回答。 

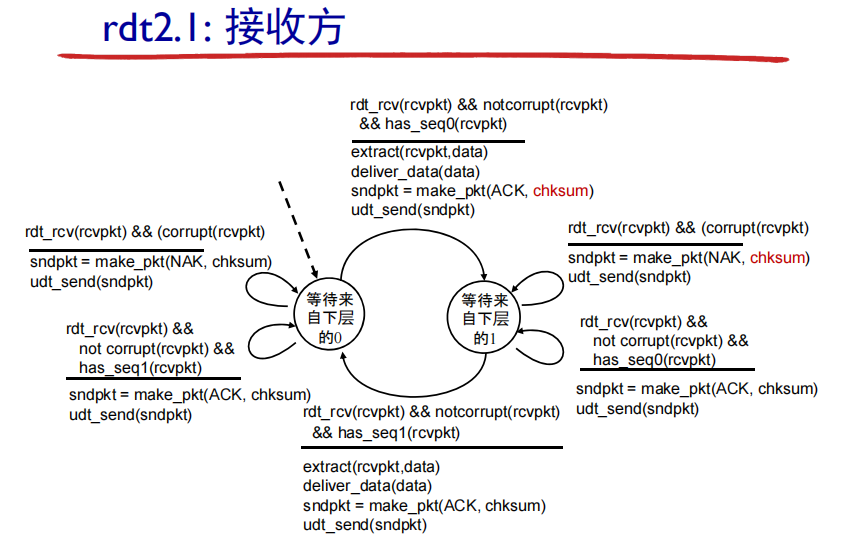
特点：可以解决位错

缺陷：若返回的ACK或NAK分组受损，发送方无法知道接收方是否正确接收了上一块数据



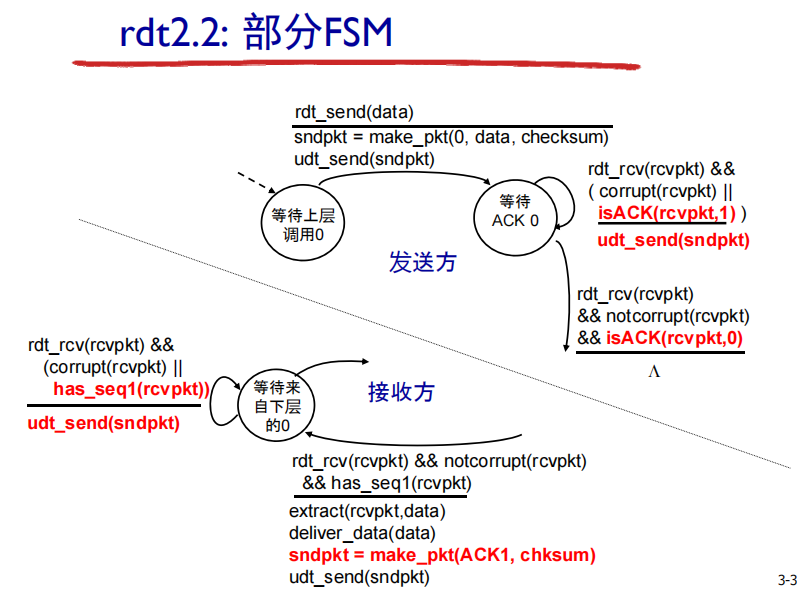
Rdt2.1

Rdt2.0的改进，可指出重复分组，指出正发送或准备接收的分组的序号。 



Rdt2.2

接收方收到受损的分组时，丢弃，不发送NAK，改为发送一个对前一个正确接收分组的ACK，发送方接收到对同一个分组的两个ACK（接收冗余ACK）后，可知道接收方没有正确接收到跟在被确认两次的分组后面的分组

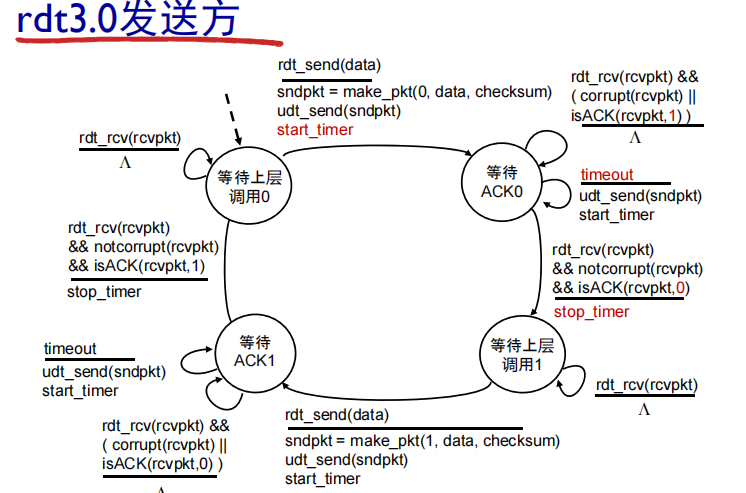


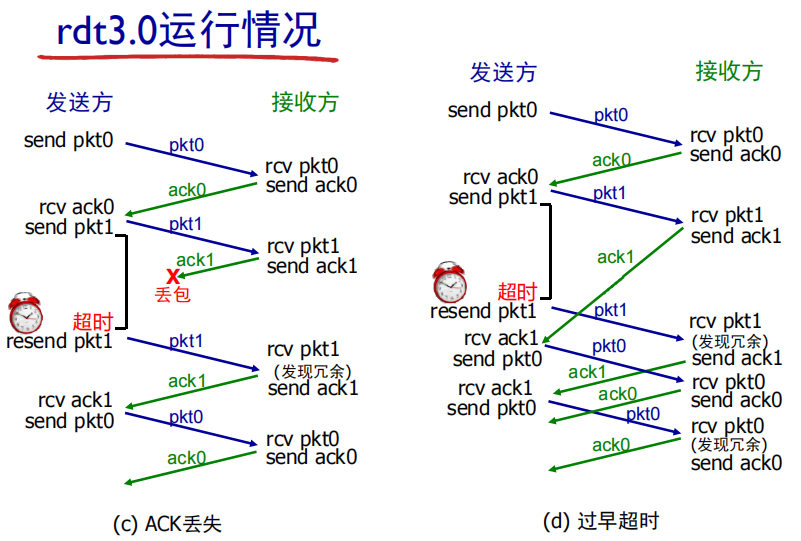
Rdt3.0

停等协议

具有比特差错的丢包信道上的可靠的数据传输

超时重发：由发送方负责检测丢包和恢复，即发送方发送一个数据分组后，等待一定时间，如果该段时间内没有收到ACK，则重传分组

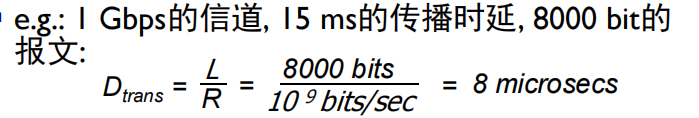
时间选择：通常为“从发送方发出分组到收到接收方应答所需的平均时间（RTT+接收方处理时延）”。 



10.停等协议的性能

rdt3.0正确，但性能差（停等协议）

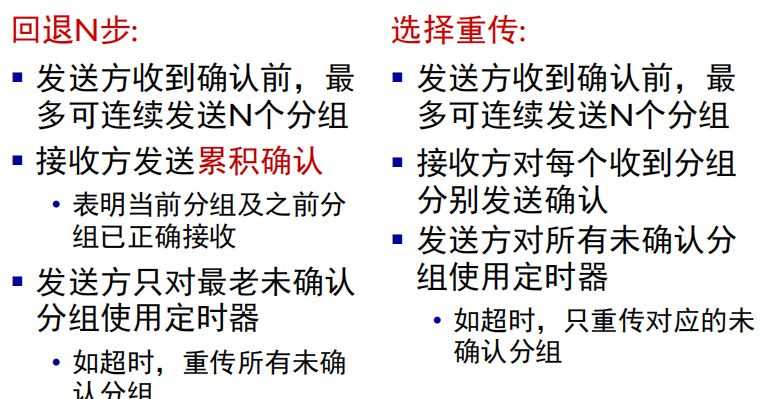
发送方(信道)利用率Usender ： 发送方将比特发送到信道的时间/发送时间



11. 流水线协议：发送方可连续发送多个报文，无需等待确认

流水线协议利用率：Usender=

类型：回退N步 (Go-Back-N, GBN协议)选择性重传 (Selective Repeat, SR协议)

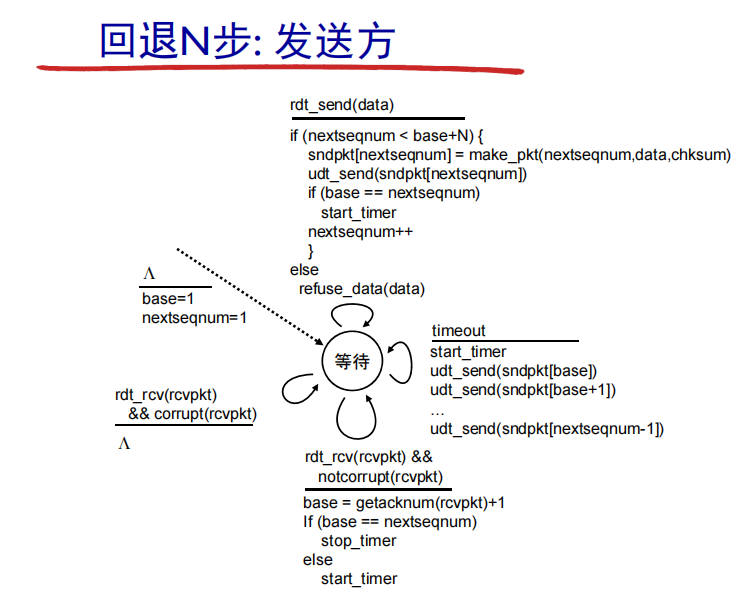


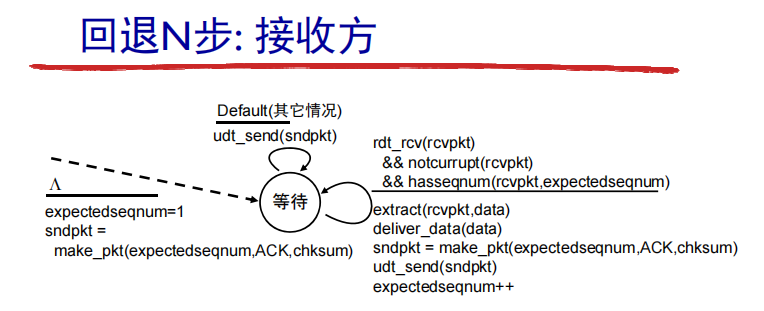
12.GBN（回退N步）/滑动窗口协议

序号、累积确认、检验和以及超时/重传

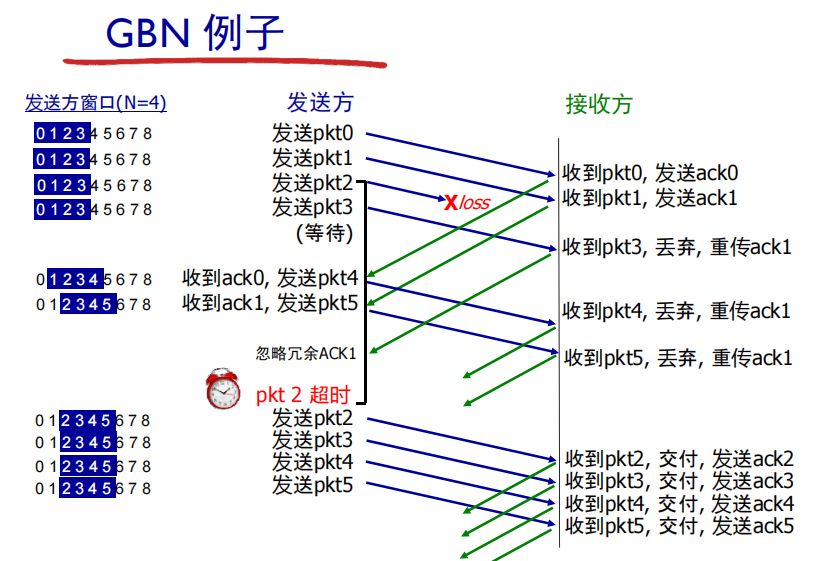
发送方：连续发送多个数据分组，停止等待收到确认ACK，继续发送后面分组；超时，未收到应答，从出错分组开始重发接收方：按序号接收数据分组正确：接收处理，发确认ACK；出错：将出错分组及后面分组均丢弃，不发任何应答。

如果分组序号字段的位数是k，则序号范围是[0，-1]





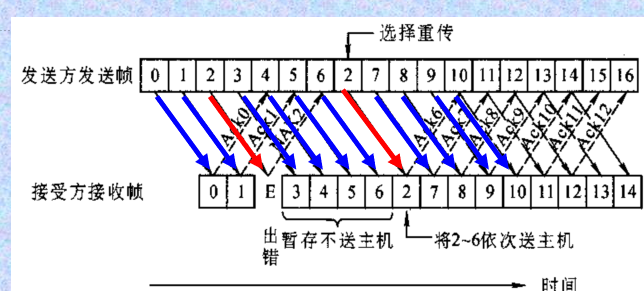
GBN缺点：可能需要很多的重传，当窗口大小和带宽时延积都很大时，堵塞管道，影响效率。



13.选择重传（SR）

发送方：连发多个数据分组，停止等待收到确认ACK，继续发送后面分组；超时,未收到应答，只重发出错分组。接收方：不按序号接收数据分组正确：接收、按顺序交付，发确认ACK；出错：丢弃该分组，以后正确分组放入缓存，当出错分组正确收到后，按顺序一起交付。

不按序接收。接收窗口大小 > 1，即只要序号落在接收窗口内的正确分组都可接收。



接收方对每个正确接收分组发送确认（包括失序分组）

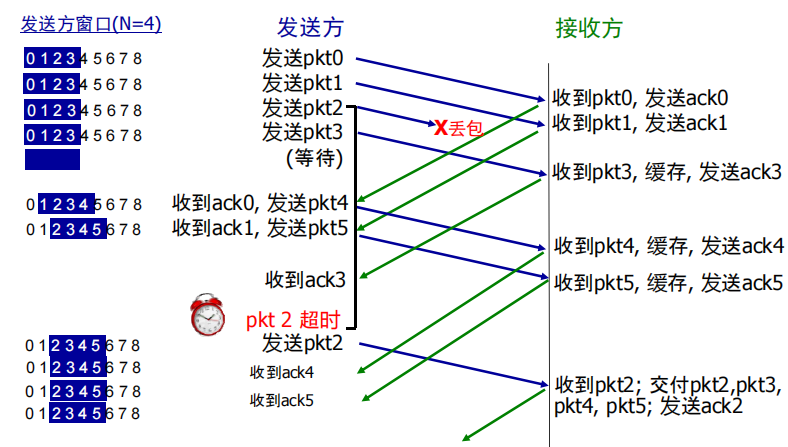
接收方缓存失序分组，实现最终按序交付，发送方只重传超时分组，对每个未确认分组使用定时器，双方都使用大小为N的缓存

发送窗口(N)：base指向第一个未确认分组；收到确认号

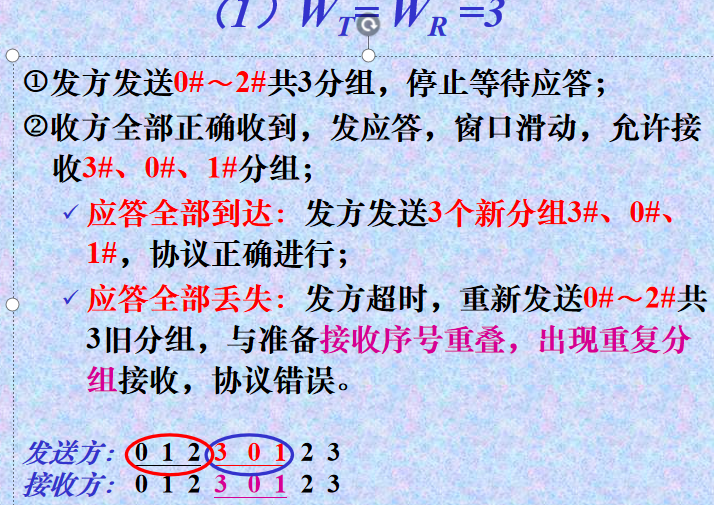
=base，窗口滑动，使base指向第一个未确认分组

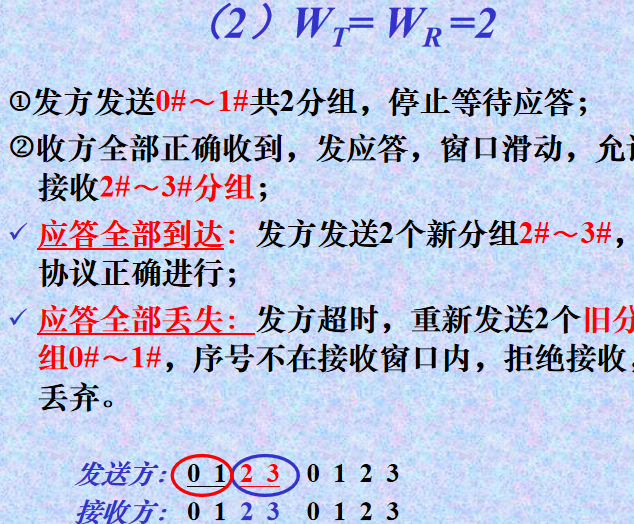
接收窗口(N)：base指向第一个未接收；收到分组编号

=base，窗口滑动，使base



若序号位数k位，SR协议，发送窗口和接收窗口尺寸最大是，不是-1，即序号空间一半。





14．三种协议比较

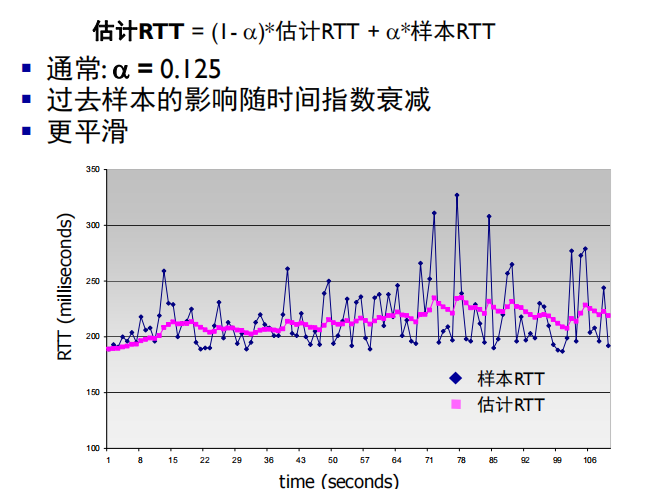


15. 面向连接的传输TCP

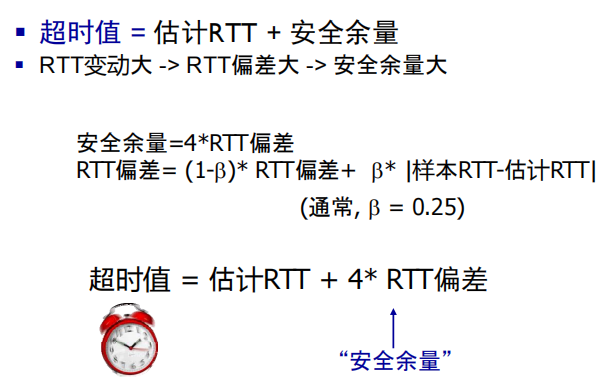
采用：差错检测、重传、累积确认、定时、序号和确认序号。

* 面向连接：数据交换前，发送方与接收方进行握手(交换控制信息)
* 全双工数据：同一连接上的双向数据流
* 点对点：一个发送方和一个接收方之间的连接
* 多播：一个发送方同时传给多个接收方。
* 流量控制：控制发送方速度，使得接收方来得及接收(避免缓存溢出)，调整N的大小
* 拥塞控制：网络拥塞时遏制发送方速率。

16. TCP 往返时延



* TCP 的超时值



17. 拥塞控制原理

太多的源发送太多太快的数据，使网络来不及处理。表现:丢包 （路由器缓冲区溢出） 长时延 （路由器缓冲区中排队）

网络拥塞代价：当分组到达速率接近链路容量时，分组排队时延非常大，发送方必须执行重传以补偿因为缓存溢出而丢弃的分组，发送方在遇到大时延时所进行的不必要重传会引起路由器转发不必要的分组拷贝，被丢弃的分组，其上游路由器的转

发成为“无用功”

AIMD（加增倍减算法）

