第六章 传输层

基本概念

服务：进程之间的通信、可靠数据传输、流量控制、拥塞控制

除了IP地址，进程通过端口号来标识自己 套接字socket

端口号是套接字标识的一部分

- 熟知端口：0～1023，由公共域协议使用
- 注册端口：1024～49151，需要向IANA注册才能使用
- 动态和/或私有端口：49152～65535，一般程序使用

创建套接字时，操作系统会分配端口号：客户端常采用自动分配，从动态端口里分配；服务器常使用指定端口号，公共域协议应使用熟知端口

传输层复用和分用

> 传输层基本服务：将主机间交付扩展到进程间交付，通过复用和分用实现

（发送端）**复用**:传输层从多个套接字收集数据，交给网络层发送

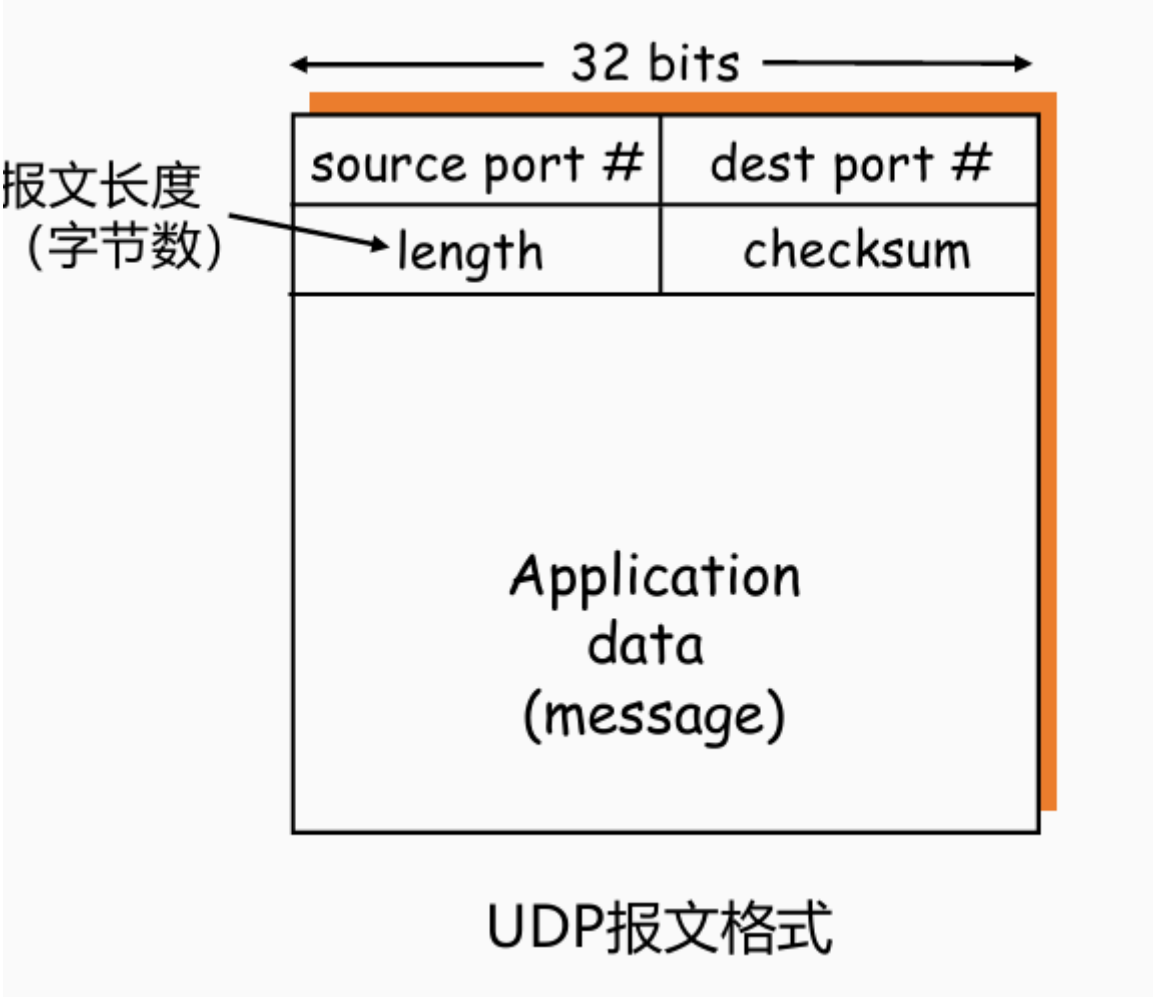
（接收端）**分用**:传输层将从网络层收到的数据，交付给正确的套接字

UDP

UDP提供的**服务**：进程到进程之间的报文交付；报文完整性检查（可选）：检测并丢弃出错的报文

UDP需要实现的**功能**：复用和分用、报文检错

报文段：头部8字节，载荷是上层传递下来的数据



校验和计算：要包含伪头、UDP头、数据；伪头取自IP头包括：源IP地址，目的IP地址、UDP的协议号(17)、UDP报文段总长度

UDP的优点：

- 无建立连接产生的时延，无拥塞控制、流量控制导致的速度缓慢
- 报头开销小
- 协议处理简单

适合的应用：能容忍丢包但对时延敏感（流媒体），以单次请求/响应为主的应用（如DNS）

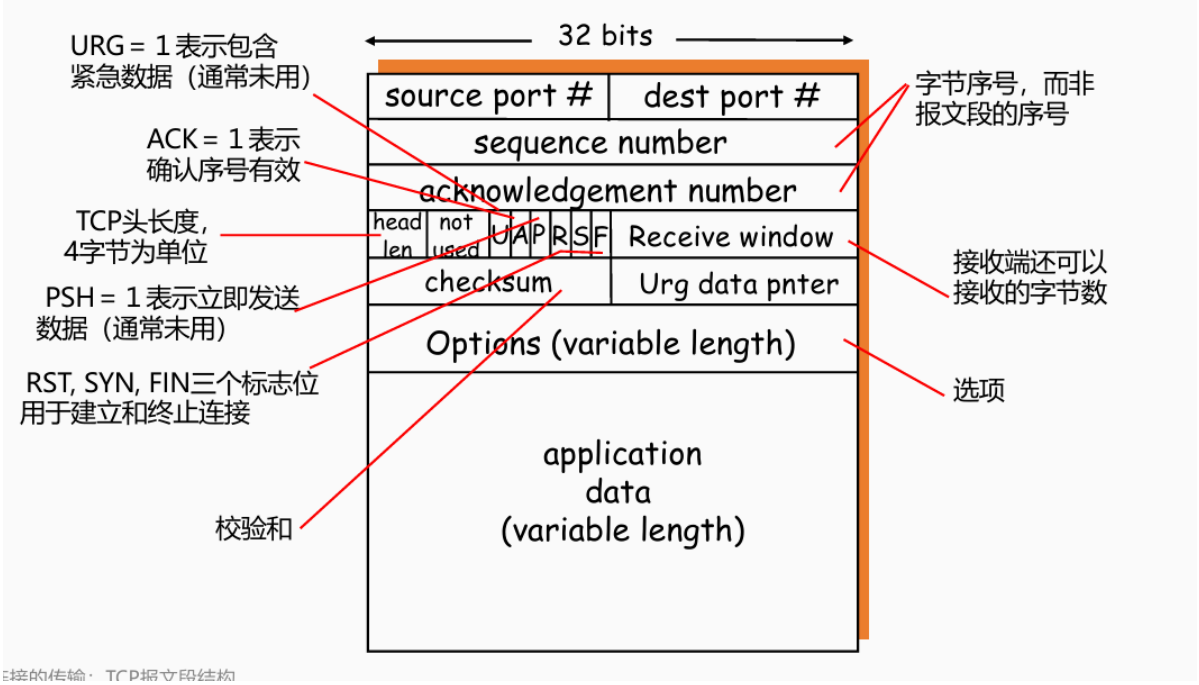
若应用要求基于UDP进行可靠传输，由应用层实现可靠性

TCP

点对点、全双工、可靠有序的字节流

需要的机制：建立连接、可靠数据传输、流量控制

报文段：头部4字节为单位，固定段20字节，可加扩展段；算上数据部分总长度不超过65535



发送序号是发送的数据载荷第一个字节在字节流中的编号，确认序号是最后一个的编号+1

TCP可靠数据传输

> 仅考虑可靠传输机制，且数据仅在一个方向上传输

接收方：

- 确认方式：采用累积确认，仅在正确、按序收到报文段后，更新确认序号；其余情况，重复前一次的确认序号（与GBN类似）
- 失序报文段处理：缓存失序的报文段（与SR类似）

发送方：

- 发送策略：流水线式发送报文段
- 定时器的使用：仅对最早未确认的报文段使用一个重传定时器（与GBN类似）
- 重发策略：仅在超时后重发最早未确认的报文段（与SR类似，因为接收端缓存了失序的报文段）

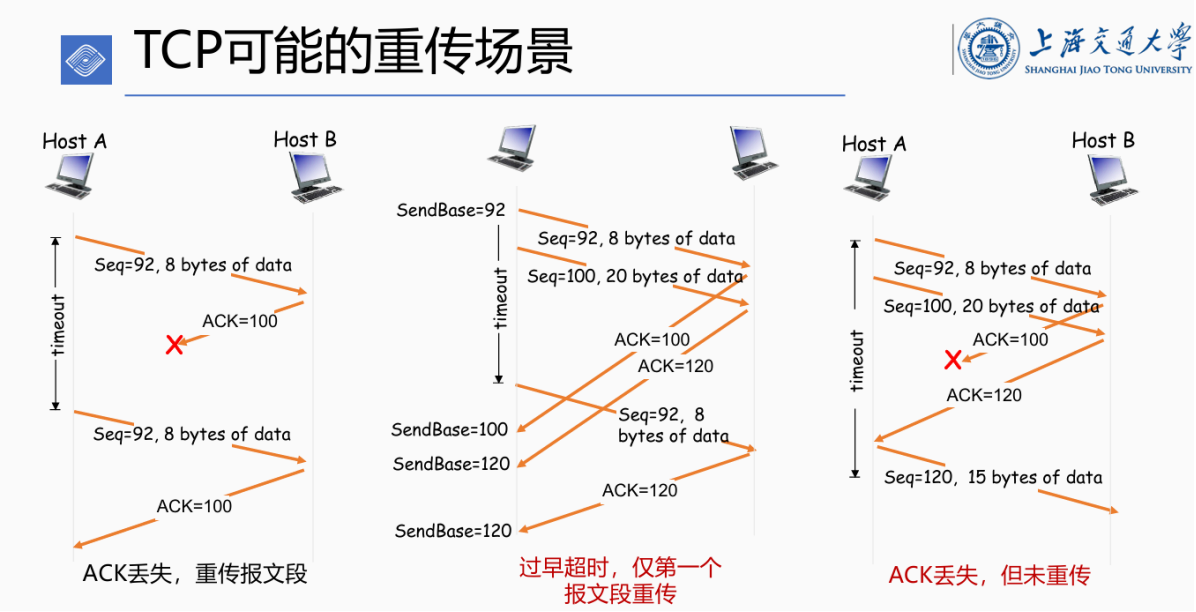
TCP发送方要处理的事件

上海交通 SHANGHAI JIAO TONG

➢**收到应用数据：**
- 创建并发送TCP报文段
- 若当前没有定时器在运行（没有已发送、未确认的报文段），启动定时器

➢**超时：**
- 重传包含最小序号的、未确认的报文段
- 重启定时器

➢**收到ACK：**
- 如果确认序号大于基序号（已发送未确认的最小序号）：
  - 推进发送窗口（更新基序号）
  - 如果发送窗口中还有未确认的报文段，启动定时器，否则终止定时器



只使用一个定时器，避免了超时设置过小时重发大量报文段

利用流水式发送和累积确认，可以避免重发某些丢失了ACK的报文段

超时值的设置（即RTT的估计）：

- EstimatedRTT = (1-α)*EstimatedRTT+α*SampleRTT；α=0.125是常用的值
- DevRTT = (1-β)*DevRTT + β*|SampleRTT-EstimatedRTT| $\beta=0.25$
- TimeoutInterval = EstimatedRTT + 4*DevRTT

快速重传：由于按照超时重传太慢，快速重传法规定当发送方收到对同一序号的3次重复确认时，立即重发包含该序号的报文段

为减小通信量，TCP允许接收端推迟确认：接收端可以在收到若干个报文段后，发送一个累积确认的报文段

TCP接收端的事件和处理

接收端事件	接收端动作
收到一个期待的报文段，且之前的报文段均已发送过确认	推迟发送确认，在500ms时间内若无下一个报文段到来，发送确认
收到一个期待的报文段，且前一个报文段被推迟确认	立即发送确认（估计RTT的需要）
收到一个失序的报文段（序号大于期待的序号），检测到序号间隙	立即发送确认（快速重传的需要），重复当前的确认序号
收到部分或全部填充间隙的报文段	若报文段始于间隙的低端，立即发送确认（推进发送窗口），更新确认序号

TCP流量控制

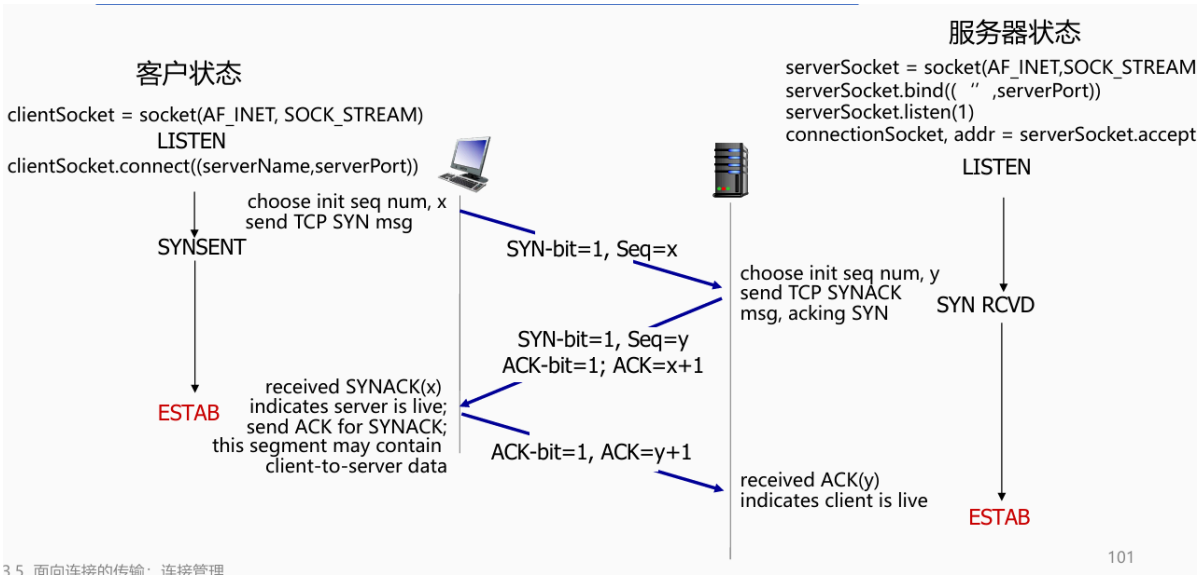
流量控制：发送端TCP通过调节发送速率，不使接收端缓存溢出

接收方将RcvWindow放在报头中，向发送方通告接收缓存的可用空间；发送方限制未确认的字节数不超过接收窗口的大小

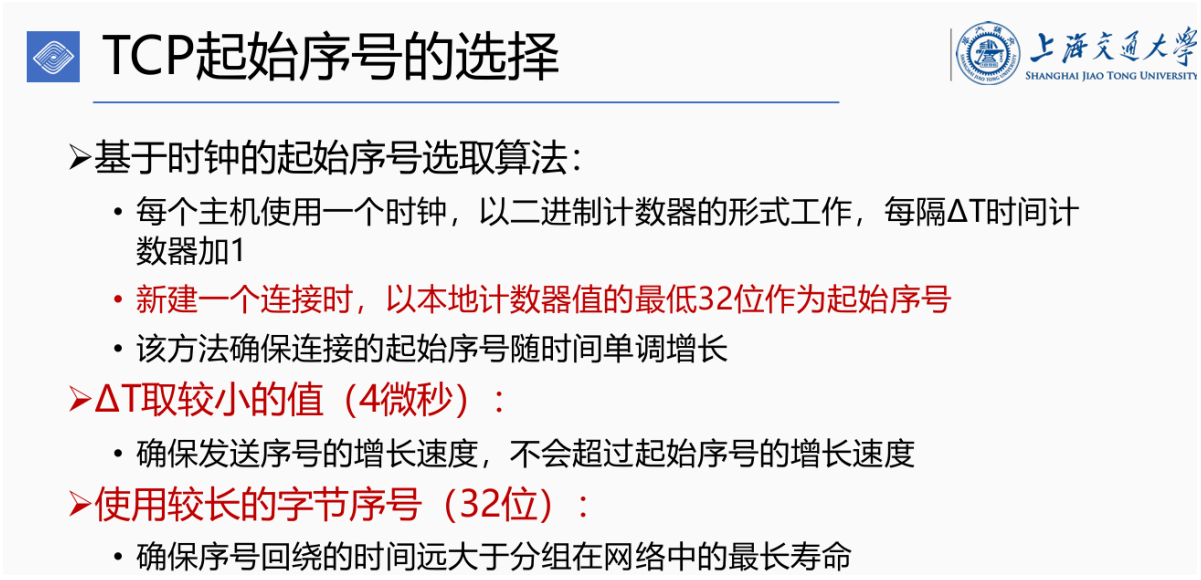
发送方收到“零窗口通告”后，维护一个坚持定时器，超时则发送一个“零窗口探测”报文段，让接受方发送它更新后的接收窗口

TCP连接管理

建立连接：三次握手

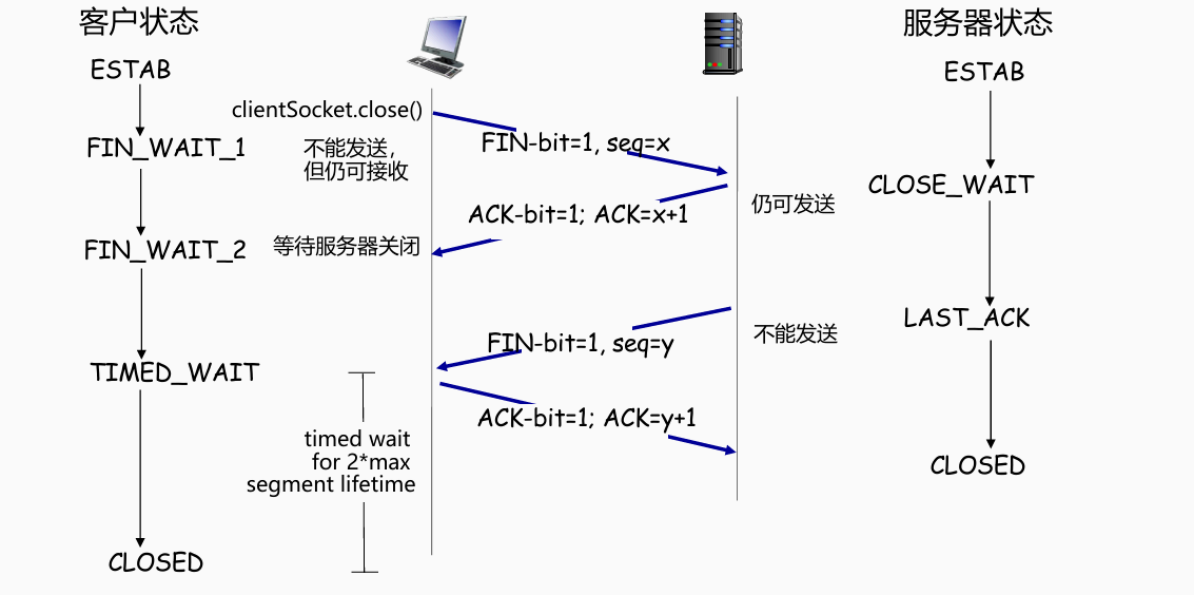


必须避免新、旧连接上的序号产生重叠



SYN洪泛攻击：攻击者采用伪造的源IP地址，向服务器发送大量的SYN段，却不发送ACK段；服务器为维护一个巨大的半连接表耗尽资源，导致无法处理正常客户的连接请求，表现为服务器停止服务

释放连接：四次挥手



TCP拥塞控制

流量控制与拥塞控制的异同：

- 流量控制：限制发送速度，使不超过接收端的处理能力
- 拥塞控制：限制发送速度，使不超过网络的处理能力

拥塞的后果：丢包：由路由器缓存溢出造成；分组延迟增大：链路接近满载造成

常用方法：网络辅助的拥塞控制、端到端拥塞控制（TCP）

TCP拥塞控制：

发送方根据丢包事件来判断拥塞（超时、三次重复确认）

TCP发送端的事件与动作

State	Event	TCP Sender Action	Commentary
慢启动 (SS)	收到新的确认	cwnd = cwnd + MSS, If (cwnd > ssthresh) set state to "Congestion Avoidance"	每经过一个RTT，cwnd加倍
拥塞避免 (CA)	收到新的确认	cwnd = cwnd+MSS * (MSS/cwnd)	每经过一个RTT，cwnd增加一个MSS
SS or CA	收到3个重复的确认	ssthresh = cwnd/2, cwnd = Threshold+3 MSS, Set state to "Congestion Avoidance"	cwnd减半，然后线性增长
SS or CA	超时	ssthresh = cwnd/2, cwnd = 1 MSS, Set state to "Slow Start"	cwnd降为一个MSS，进入慢启动
SS or CA	收到一个重复的确认	统计收到的重复确认数	cwnd 和 ssthresh 都不变