

运输层(传输层)

应用进程无需考虑物理细节

为不同主机上应用进程之间提供了逻辑通信

运输协议的服务受限于网络层的服务模型

即使，运输层协议也能提供某些服务。

应用层 → 报文

运输层 → 报文段

网络层 → 数据报

UDP 用户数据报协议

不靠、无连接
数据报

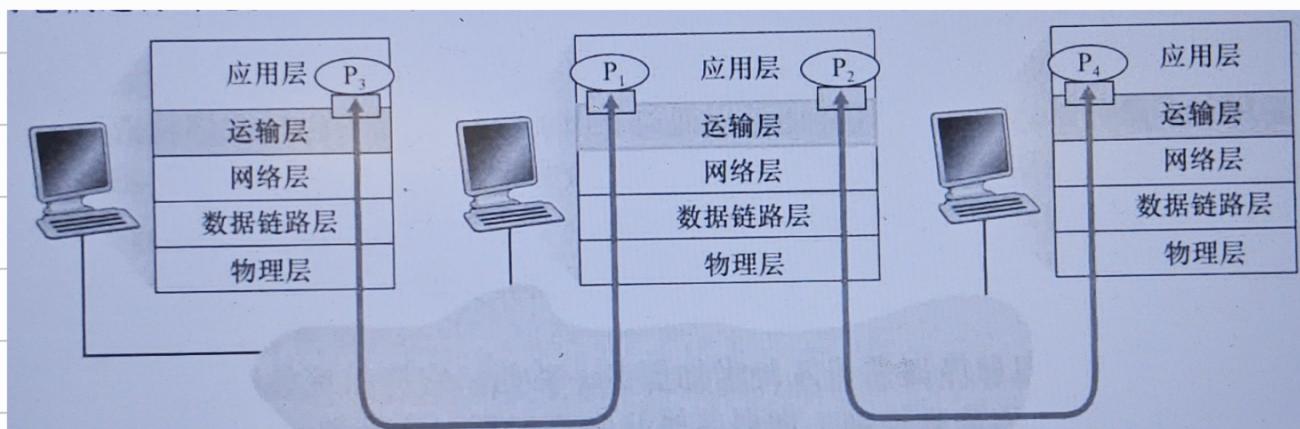
TCP 传输控制协议

可靠、面向连接
报文段

⇒ 报文段

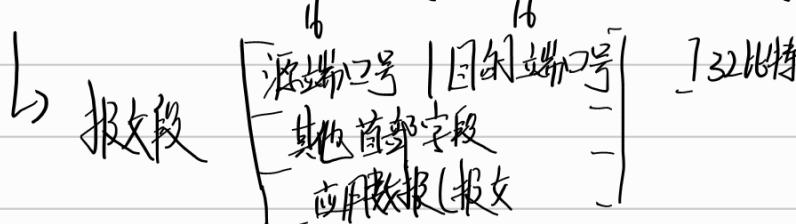
将IP不可靠变成了可靠(流量控制, 序号, 检查和)

多路复用、多路分解



将运输层报文段中的数据交付到正确的套接字的工作 多路分解。

在主机不同套接字中收集数据块，并为每个数据块封装首部信息，变为报文段送到网段 多路复用



端口号 0-65535

↑ 周知端口号 0-1023

无连接的多路复用和多路分解 UDP

UPP套接字

七一小组

UDP报文具有 源端口号 / 目的端口号

→ 客户端常用临时端口，服务器则会“端口翻转”

(不连接，服务器不知道对方是谁)

面向连接必须使用多路复用和多路分解 TCP

TCP套接字

四元组

源IP地址 源端口号 目的IP地址 目的端口号

三次握手会建立一个[连接表项]

区别就是有上下文。

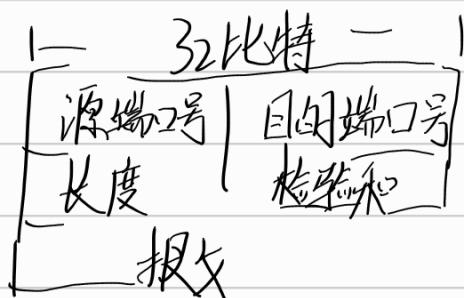
且绑定了端口。

UDP报文结构

长度：UDP报文字节数(包括首部)

校验和：报文中16比特字进行校验运算

报文遇到问题被回包



端到端原则

可靠数据传输协议

在不可靠的底层信道之上，通过一系列机制保证数据从发送方到接收方正确完整有序

rdt 协议演进

rdt1.0：假设信道完美可靠，不需要任何机制。

rdt2.0：底层信道可能产生比特错误。

↳ 引入差错检测（校验和）

↳ 接收方返回 ACK(正确接收)或 NAK(检测到错误)

↳ 发送方收到 NAK，重传该分组

rdt 2.1/2.2：底层信道不仅可能导致比特错误，还能使 ACK/NAK 出错，缺

↳ 每个分组打上序号

↳ 接收方可通过序号识别当前分组是否是新到还是重传的

↳ rdt 2.1：使用 ACK 和 NAK，并通过序号区分正确/错误

↳ rdt 2.2：仅使用 ACK，接收方只对正确的分组返回 ACK，发送方重传

rdt 3.0：底层信道可能会丢包(分组或 ACK/NAK 都丢)

↳ 在序号机制基础上，发送方增加定时器

↳ 在超时时间内未收到预期 ACK → 重传

↳ 发送方在未确定前只能发送一个分组，效率低(停等协议)

↖ 问题

流水线协议：

解决：允许发送方在未确定的情况下发送多个分组(流水线传输)

Go-BACK-N：发送方可连续发送窗口大小范围内的分组，若有丢失，接收方只确定最后一个正确的分组，发送方从此错点重传。

Selective Repeat：接收方每个分组确认缓存，发送方只发送无错。