

第一部分:小测

第一章

- 1.常用的通信传输介质有哪些？它们之间的主要区别？
- (1)有线 双绞线、同轴电缆、光纤 无线
- (2)区别 带宽、误码率、传输距离、价格、频道及复用方式、是否支持移动通信等。
- 2.面向连接服务能提供什么服务？
- (1)可靠传输 (2)有序传输 (3)资源预留 (使用)
- 3.无连接分组交换与面向连接(虚电路/电路)交换的区别？
- (1)分组格式:前者完全端、目的地址:后者虚电路地址
- (2)路由由表:前者面向网络拓扑、转发时顺序查表路由由表:后者面向特定路径或源路由、转发基于索引查找路由由表。
- (3)可靠性、顺序性:前者无:后者有 (4)建立、维护连接:前者无:后者有
- 4.无连接服务的优点与缺点？
- 优点:无需知道网络状态(包括网络资源)或只需知道网络局部状态 缺点:无有不确定性(是否有满足服务的网络资源不确定、能否完成服务不确定)
- 5.分层网络体系结构的不足:
- 上层协议的性能依赖于下层协议
- 6.分组交换原理:
- (1)存储转发 (2)动态路由(包括每个分组自带源地址、目的地址、拆包重组、路由选择) (3)出错由端系统处理
- 7.若一个 WWW 文档中除有文本外,还有 6 个图像。试问使用 http/1.0 与 1.1 各需要建立几次 TCP 连接？
- 1.0:7 1.1:次
8. ADSL 传输速率(子频道)？其中数据通道数？若每个通道均采用 QAM-128 调制,数据通道总容量？
- 256.248.248~7+4k

- 9.假定要传送的报文共有 x(单位 bit),从源节点到目的节点共有 n 条链路,每条链路的传播延迟为 d(单位 s),链路带宽为 b(单位 bit/s),各链路包括(包括连接延迟及拆除)使用的控制帧(或信令)长度、在各节点的排队时延忽略不计。分组交换用的分组组头、分组长度分别为 h、p(单位 bit),分组在各节点的排队时延为 q(单位 s),试分析在何种条件下电路交换的时延要小于分组交换的总时延？电路交换总时延 D(c):
- (1) 建立时间 kd (2) 连接拆除时间 kd
- (3) 数据传输时间 x/b (4) 数据传播时间 x/b
- D(c)=3kd+x/b
- 分组交换总时延 D(p):
- (1) 单个分组传输时间(p+h)/b
- (2) 第 1 跳传输时间(x/p)/(p+h)/b(x/p 为分组个数)
- (3) 传输时间每 1 跳增加 1 个分组的传输时间,所以总的传输时间为 x/p*(p+h)/b+(k-1)*(p+h)/b
- (4) 排队时间 kq
- 5 传播时间 时间 kd D(p)=(x/p*(p+h)/b+(k-1)*(p+h)/b)+k*q
- 若 D(c)<D(p),则...

第三章

- 1.TCP 协议中 ACK 的作用。
- (1)建立连接、拆除连接 (2)差错控制(或可靠传送)
- (3)流量控制 (4)拥塞控制
- 2.TCP 连接的目标。
- (1)实现端到端通信 (2)实现可靠传送 (3)实现按序传送 (4)进行流量控制 (5)进行拥塞控制
- 3.实现 TCP 连接目标的主要机制。
- (1)通过传输层地址(端口号)实现端到端通信 (2)通过确认机制实现可靠传送 (3)通过接收方缓存实现按序传送 (4)流量控制(5)拥塞控制 (6)连接建立与拆除机制
- 4.在 TCP 连接中,客户端的初始号 215,客户打开通道,只发送一个携带有 200 字节的报文段,然后关闭连接。试问下面从客户端发出的各个报文段的序号分别是多少(1)SYN 报文段 (2)数据报文段 (3)FIN 报文段。
- 答:(1)215 (2)216 (3)216
- 5.在建立连接时,TCP 连接上发送一个长度为 32KB 的文件。发送端每次都发送一个最大长度的段(MSS),MSS 的长度为 4KB,接收端正确收到一个 TCP 段后立即给予确认。发送端的初始拥塞窗口门限为 16KB,假设发送端尽可能快地传输数据,即只要发送窗口允许,发送端就发送一个 MSS。
- 问:已知发生第一次超时后,发送端将拥塞窗口门限调整为 4KB,请向发生超时的时候,发送端的拥塞窗口是多大？此时发送端一共发送了多少数据？其中有多少数据被成功确认了？(2)发送端从未被确认的数据开始使用慢启动进行重传。假设此后不再发生超时,当文件全部发送完毕时,发送端的拥塞窗口是多大？
- (1)第一次超时发生时,发送端拥塞窗口大小=4KB*2=8KB 在新建立的 TCP 连接上,发送端采用慢启动开始发送,因此当第一次超时发生时,发送端已发送的数据量=1KB+2KB+4KB+8KB=15KB,此时,除最后一批 8 个 TCP 段未获确认外,之前发送的 TCP 段都被确认,因此成功确认的数据量为 7KB。(2)发送端采用慢启动重新开始发送,在拥塞窗口达到 4KB 时收到该数据量=1KB+2KB+4KB=7KB。然后进入拥塞避免阶段在收到全部 4 个 MSS 的确认后,拥塞窗口增至 5KB,相应地发送端发送了 5KB 数据:收到到全部 5 个 MSS 的确认后,拥塞窗口增至 6KB:收到全部 6 个 MSS 的确认后,拥塞窗口增至 7KB:此时刚刚好发完。因此,文件发送结束时,发送端的拥塞窗口大小为 7KB。
- 6.TCP 如何发送紧急数据？
- (1)紧急标志位 U(URG)置 1 (2)紧急数据置于 TCP 段数据(载荷)前部: (3)紧急指针指向紧急数据的最后一个字节。
- 7.TCP 接收方何种情形需要立即进行确认？
- (1)连续两个字节到达时,且前一个未确认 (2)收到失序段(序号比期望的序号大): (3)收到到失序段: (4)收到重复段。
- 8.滑动窗口协议中,GBN 与 SR 利用接收缓冲能力连续发送多个帧,令帧的传输时间(transmission time)=1(归一化)、传播时间(propagation time)=a,则链路的缓冲能力为?
- a(单向)或 2a(双向)

第二部分:知识点

第一章 概述

- 1.1 什么是因特网
1. 因特网的两种描述:
- 1 按松散的层次结构组织、并且遵循 TCP/IP 协议的 ISP 集合
- 2 为分布式应用提供通信服务的基础设施
2. 网络设备:称为主机或端系统,运行网络应用程序,通信链路同轴电缆、光纤、铜线、无线电磁频谱等,传输速率可为带宽。分组交换机:转发分组,包括链路和链路层交换机。
3. 主机间通过通信链路和分组交换机连接在一起。
- 2.1 网络边缘
- 1.物理媒体分类
- 设备之间通过物理媒体相连。分为导引型/非导引型
- 2 宽带入网:数字用户线 DSL(不对称和电缆光纤和同轴电缆)
- 3 以太网用户使用双绞线连接到以太网交换机连接
- 1.3 网络核心
- 1.分组交换
- 主机将应用程序划分成分组,交换主机在接收到整个分组后,才可以开始转发(存储-转发)。不考虑信号传播时间,一个分组经过 N 条链路的总耗时是(P+N-1)/R
- 当分组到达速率大于链路输出速率时,会在缓存中排队。若缓存满了,会丢包。
- 网络链路的重要功能:路由(生成转发表)、转发
- 分组交换原理:小测 1.6
- 优点: 更好的带宽共享,简单有效,适合突发数据,不需建立电路支持更多的端系统。能快速传输端系统产生的大量数据,按需要分配(不需预留)
- 缺点:可能产生严重拥塞(延迟、丢包) (不适合实时服务)
- 2.电路交换
- 通信前预留端-端资源,资源独占。不通信时资源闲置 优点:能在请求时间内端到端保持一个确定量的带宽
- 缺点:不考虑需求
- 3.ISP
- ISP 采用多层结构:局域网连接到区域 ISP,区域 ISP 间直连/用 IP 相连:连接到更高层 ISP
- 4.TDM(时分复用):时间划分为固定期间的帧,每个帧划分为固定数

- 量的时隙,每条电路分配有时隙。FDM(频分复用):每条电路分配有时隙,频段宽度为带宽。FDM 需要复杂的硬件(硬件来将信号转换到合适频带上。端对端电路传输时间与链路数量无关)
- 1.4 分組延遲
- 1.分組延遲的來源
- 节点处处理查错、确定输出链路/排队(在缓存中等待传输)/传输延迟(见 1.3.1,分组的发送时间)/传输延迟
- 最大吞吐量:路由能够转发分组的最大速率
- 流量强度:La/R [R 链路带宽][L:组长度] [a 分组到达速率]
- 上式<=1,越接近 1,延迟越高
- 1.5 协议层次
- 分层模型:显式的层次结构易于确定系统的各个部分及其相互关系/模块化使更新系统组件为容易
- 分层模型:一层可能允许多层功能
- 协议栈
- 应用层:支持各种网络应用 FTP,SMTP,HTTP
- 传输层:进程-进程的分组传输 TCP/UDP
- 网络层:源主机-目的主机的分组传输 IP/路由协议
- 链路层:相邻网络设备之间的分组传输 PPP/以太网
- 物理层:在物理媒体上传输比特/帧/数据/报文/报文/OSI 模型:应用层传输层增加了表示层(帧)与会话层。
- 表示层:解释交换数据信息,如压缩、解密等
- 会话层:提供数据交换定义,同时功能:建立检查点,提供恢复方案
- 封装路由:数据二层,链路层交换机二层,形式:首部+有效载荷字节

第二章 应用层

- 2.1 应用层协议原理
- 网络核心设备不在应用层起作用。
- 1.应用程序/程序/客户端、服务器架构:眼上需求(数据中心),成本,【百度谷歌搜索引擎】、对等架构 P2P;自扩展性,成本,安全性、可靠性、性能面临挑战【文件共享、无线电话视频】。
- 客户机发起通信的进程,需要与服务设备通信,不是实时在线通信,使用动态 IP 不与其他客户机直接通信
- 服务器等待联系的进程,总是在线的主机,具有永久 IP 地址。P2P 没有总是运行的服务器,任意一个端系统可连接通信,每个对等方可以请求服务也可以提供服务。对等方断连接,动态 IP
- 2.套接字
- 进程通过套接字收发报文,是应用层、传输层的接口,是应用程序与网络间的 API。为接收报文,每一个进程要有标识(端口号)。因为进程很多,不能用 IP 地址标识进程。HTTP 80 Mail 25
- 3.传输服务
- 传输层提供的服务种类:可靠性 吞吐量 定时 安全性。吞吐量和定时未运用在因特网
- TCP 面向连接的可靠传输,有流量控制、拥塞控制,不提供及时、最低带宽保证
- UDP 无连接,不可靠传输,不提供连接建立、可靠传输,流量控制、拥塞控制、及时性、最低带宽保证
- 应用层协议【HTTP SMTP Telnet FTP SIP RTP】
- 应用层协议定义了报文类型、报文语法、字段语义、进程发送/接收报文的标识符

2.2 Web、HTTP HTTP1.0 (RFC1945) HTTP1.1 (RFC2068)

- 网页由基本 HTML 文件、对象构成。HTML 文件引用对象。每一个对象有一个 URL
- HTTP1.0 (RFC1945) HTTP1.1 (RFC2068)
- 客户在 80 端口发起一个到服务器 xxx 的 TCP 连接(分别有一个套接字)——套接字套接字向服务器提供 HTTP 请求报文——服务器经套接字接受报文并封装成报文发送给客户端——断开 TCP
- HTTP1.0:非持久连接,一个连接一个对象【并行/非并行】
- HTTP1.1:持久连接,一个连接多个对象【流水线/非流水线】
- RTT (Round-Trip Time):一个小小分组从客户发送到服务器再返回客户的平均时间。
- 非持久: 2RTT+2nRTT
- 持久:非流水线: RTT+RTT+nRTT
- 流水线: RTT+RTT+RTT
- 3.报文格式:请求行,首行,回车,回车(表示结束)
- 上传文件 post 放在报文体内,使用实体体, get(放在 URL 内,实体体为头) HEAD(与 GET 类似,不返回对象) PUT DELETE
- 用表单表示的请求报文不是必须使用 POST
- 报文格式:状态行 首行行 实体体
- 状态行: 请求行 首行行 实体体

- 4.cookie 站点对用户跟踪
- (1)响应报文 cookie 首行行 (2)请求报文 cookie 首行行 用户系统 cookie 文件 (4)后端数据库 [set-cookie 识别码]
- 存储位置:服务器端返回 ID 给客户/客户端文件中
- 5.web 缓存/代理服务器
- 既是客户端又是服务器,保存最近请求过的对象的副本。
- 减少响应时间,减少通信量,减少网络流量
- 6.命中:缓存满足需求的请求的比率
- 3.TCP (File Transfer Protocol) 20/21 端口
- FTP 采用两个并行的 TCP 连接传输文件:控制连接(端口 21)+数据连接(端口 20)是有状态连接。
- 21 一直保持,20 随文件传输结束而关闭
- 主动控制,数据连接原因:不会混数据与命令响应,简化协议设计和实现,在传输文件的过程中可以继续执行其它的操作,便于控制传输过程。
- 用关闭数据连接的方式结束传输:允许动态创建文件
- 2.4 电子邮件系统
- 1.三部分:用户代理、邮件服务器、简单邮件传输协议 SMTP
- 2 SMTP 端口号 25,使用 TCP 作为传输层协议,持续连接。
- 每台邮件服务器上就运行 SMTP 客户端也运行 SMTP 服务端。SMTP 不使用中间邮件服务器。
- A 代理-A 邮件服务器-B 邮件服务器-B 代理
- SMTP 与 HTTP 对比:
- (1)SMTP 是一个推协议:发送邮件服务器推到接受邮件服务器。
- HTTP 主要是一个拉协议:用户从 web 服务器拉取。(2)SMTP 要求报文发送了比特 ASCII 码进行编码,HTTP 无此限制。(3)SMTP 把所有报文对象直接放在一个报文中,HTTP 把对象封装到响应报文中
- 推送也可以用 HTTP,注意意图。
- 3 邮件访问协议 POP3(无状态) IMAP(有状态) HTTP(无状态)
- POP3 端口 110 TCP 连接,特许——事务处理——更新(quit)后,通常有“下载并删除”(不能重复读)模式和“下载并保留”模式。
- IMAP 可以创建远程文件夹,指派文件夹,维护用户状态,可以获取邮件的部分。
- 2.5 DNS 主机名-IP 地址转换
- (1)一个由分层的 DNS 服务器实现的分布式数据库 (2)一个使得主机名能查询数据库的应用层协议
- 1 提供服务:
- (1)允许拥有复杂主机名的主机具有一个或多个别名,提供与主机别名对应的规范主机名及 IP 地址 (2)提供邮件服务器的规范主机名及 IP 地址,允许用域名作为邮件服务器别名,邮件与主机名 (3)允许一个规范主机名对应一组 IP 地址,多个邮件服务器同名
- 2 响应报文<5128,采用 UDP,否则 TCP,端口 53
- 3 DNS 服务器类型:根/顶级域 TLD(.com)/权威 DNS 服务器
- 4 本地 DNS 服务器:代理作用,缓存(一定时间失效)
- 5 DNS 查询是:DNS 服务器间通信。迭代(8次)/递归(8次)
- 5 工作机制:应用层将主机名传给 DNS 应用层,DNS 客户向 DNS 服务器发送包含主机名的请求,最终收到回答包含主机名与 IP。

- 主机<=>本地<=>根、TLD、权威
- 主机<=>本地<=>根<=>根<=>根<=>权威

- 6 DNS 资源记录:RR: (name, value, type, TTL)
- Type=A: Name 是主机名 Value 是对应的 IP 地址 (1.b.foo.com.145.37.93.126A)
- Type=NS: Name:域, Value:域的权威 DNS 服务器主机名 (foo.com.dns.foo.com.NS)
- Type=CNAME: Name:别名, Value 是对应规范名 (foo.com.r1.b.foo.com.CNAME)
- Type=MX: Name:别名, Value:对应邮件服务器规范名 (foo.com.mail.b.foo.com.CNAME)
- 7 插入记录
- 向注册登记机构提供权威 DNS 的名字和 IP,NS 和 A 输入 TLD,并确保用于 Web 服务器的 A 和邮件服务器的是 MX 输入到权威 DNS。

第三章 传输层

- 3.1 概述和运输层服务
- 在应用程序看来,传输层提供了进程间逻辑通信。忽略通信过程,不同主机上的进程可以它们是直接相连的具体什么,由网络层管。
- 1 传输协议只工作在端系统上
- 发送方:将应用层报文封装成报文段,交给网络层发送。
- 接收方:从收到的报文段中取出载荷,交给应用层。
- 2 因特网的网络服务:
- 尽力而为的服务:网络层尽最大努力在主机间交付分组,不保证交付,不保证按序交付,不保证数据完整。
- 3 传输层不能提供的服务:延迟保证,带宽保证
- 4 传输层可以提供的服务:数据交付,差错检查,保证可靠交付的交付:TCP-不保证 UDP
- 3.2 多路复用与多路分解
- 1 多路复用:从多个套接字收集数据封装成报文段,传给网络层。
- 2 多路分解:将运输层报文段中的目的地址交付到正确的套接字
- 2 主机中每个套接字应分配一个唯一的标识,报文中特殊字段指示要交付的套接字。发送方传输层需在报文段中包含目的套接字标识。接收方传输层需将报文段中的目的套接字标识与本地套接字标识进行匹配,将报文段交付到正确的套接字
- 3 端口号是套接字标识的组成部分:是一个 16 比特的数 (0-65535), 0-1023 为熟知端口号,源/目的端口号
- 4 UDP 套接字的标识为二元组:(目的 IP 地址,目的端口号),有相同的 IP 和端口号则通过相同的目的套接字到相同的目的进程。源 IP 和端口要返回
- 5 每个 TCP 连接套接字与一个进程联系,为四元组(源 IP 地址,源端口号,目的 IP 地址,目的端口号),不同源的报文段通过不同的套接字
- 6 TCP 欢迎套接字 12000 端口。
- 7 连接套接字与进程并非是一一对应。
- 3.3 UDP
- 1 UDP 提供多路复用和多路分解(最基本的)、差错检测(但不尝试恢复)
- 2 不提供可靠/按序交付、延迟/带宽保证 不握手
- 3 UDP 检验和:前三个 16 比特字节的和取反码运算,溢出回卷(检验和+溢出)的 1。检验方法:四个 16 比特字相加全是 1
- 4 优点:没有拥塞控制和流量控制,可以尽可能快地发送报文,无须建立连接,协议简单,发送端和接收端不需要保持连接状态,分组首部开销小(UDP8B,TCP20B)
- 5 报文结构:源端口号,目的端口号、长度(首部+数据)、检验和,应用数据(应用层数据)
- 3.4 可靠数据传输(RDT)原理
- RDT1.0:底部信息完全可靠
- RDT1.0:比特差错:自动重传请求协议 ARQ,差错检测,接收方反馈 ACK/NAK,重传。缺点:【ACK/NAK 受损】
- 停等协议:收到 ACK 并离开当前状态才继续获取数据。
- RDT1.1:发送方发现受损就重传,为分组添加序号 0/1,发现序号相同就去丢【缺点:冗余分组】
- RDT2.2 无 NAK,用 ACK1/ACK0。NAK 换成上一个 ACK,发送端发现 ACK 序号不对就重传。
- RDT3.0:比特差错、丢包、启动定时器。如果超时前没收到 ACK/收到迟到的 ACK,重发。比特交替协议:必须等前定时超时时才处理。RDT3.0 是停等协议,太慢一个完了才能传下一个。需要流水线技术。利用(L/R)/(RTT+L/R)。增加序号范围,两端需要维护多个序号,返回 N 步/选择重传
- 回避 N 步 GoBackN (GBN):
- 发送方
- [0,base-1][base,next-1][next,base+N-1][base+N,+]
- 发送窗口 = 已发送未确认序号 + 未发送可用序号 =N
- TCP 序号按照字节计数,ACK 分组携带确认的分组的序号
- 使用累积确认:若 ACK 包含序号 q,表明 q 前的分组都正确收到,整体滑动发送窗口,使基序号 = q+1,视发送窗口是否空出/终止/定时器等。
- 发方只对基序号分组使用定时器,如果 ACK 不对,不管。若超时:发送方重传整个发送窗口
- 接收方:
- 只对正确收到的序号连续的一系列分组中最高的回 ACK,如果失序,就丢弃。重发前一个 ACK。
- 选择重传 (SR):
- 缓存失序分组, 避免不必要的重传
- 发送方:
- 与 GBN 类似,但不累计确认。如果 ACK 的是基序号,滑动发送窗口使基序号=最小未累计分组。
- 接收方:
- 一旦收到就 ACK,若失序,缓存。若收到的是基序号,滑动接收窗口,交付从 N 开始的若干连续分组。如果收到窗口前的冗余分组,说明发送方将超时或 ACK 丢包,发送 ACK (冗余)。
- 为使接收端不会将重复的分组当成新的分组,窗口 [0,N-1]和窗口 [N,2N-1]不能重叠。所以 N<=窗口中间的一半。
- 每个发送的分组需要一个定时器,以便被单独重发
- 3.5 TCP
- 1 特性:点到点,全双工,面向连接(握手)可靠有序,流水线式发送,流量控制,发送缓存,提供累计确认
- 无拥塞,无拥塞,非冗余,完全相同
- 2 最大传输单元,MTU:链路层最大长度 1500 字节
- 3 窗口比例因子:实际窗口大小=window size*2^n scale
- 4 TCP 报文段结构
- 源/目的端口号:多路复用/分解
- 序号:首字节在字节流中的序号,非报文段序号
- 窗口号:希望的下一字节序号,隐含累计确认
- 首部长度:32bits 为单位的首部长度,通常是 20 字节
- 检验和:紧急数据指针(指向最后一个字节)、数据段标志位:URG-紧急数据 PSH:立即交给上层 RST:不接受连接 SYN:建立连接 FIN:拆除连接
- 5 往返时间估计
- 每个时刻做一次为已发送未确认的估计 SampleRTT,重传不做,仅为传输一次的报文计算 SampleRTT,一旦获得 SampleRTT:做指数加权移动平均 EWMA
- EstimatedRTT=(1-α)DevRTT+αSampleRTT
- 估算 SampleRTT 偏离 EstimatedRTT 的程度:
- DevRTT=(1-β)DevRTT+β|SampleRTT-EstimatedRTT|
- α:推荐值为 0.125 β:推荐值为 0.25
- TimeoutInterval=EstimatedRTT+4*DevRTT
- TimeoutInterval 初始为 1,超时翻倍 SampleRTT 更新时才按照式更新。
- 6 传输机制:像一个带累计确认的 SR
- 接收方(延迟确认):
- 收到一个期待的报文段,且之前的报均已发过确认:推迟发送 ACK500ms 内若无下一个到来,发送 ACK
- 收到一个期待的报文段,且前一个被推迟确认:立即发送 ACK
- 收到一个失序的报文段:缓存 发送冗余 ACK
- 收到填满当前窗口的报文段:立即发送 ACK
- 推迟确认优点:减少通信量
- 缺点:延迟太大导致不必要的重传 RTT 估计不准确
- 发送方:
- 流水线式发送报文段
- 仅对最小序号未确认的报文段使用一个重传定时器(GBN)

- 仅在超时后重发引起超时(最小序号)的报文段(选择重传),收到重发的确认序号后,推迟发送窗口
- 为防止反复重传采用定时器策略,发送方每重传一个报文段,超时时间间隔就增大一倍(至一个规定的上限值)。
- SampleRTT 更新时才再次更新 interval
- 同一序号收到三次重传,记为丢包,快速重传
- TCP 中的 ACK 序号是期待的报文段第一个字节的序号
- 7 TCP 流量控制
- 接收方有个接收缓存,发送方调节速度使缓存不溢出
- 发送方维护接收窗口 rwnd(动态),接收方用于给发送方通告缓存可用空间
- LastByteRcvd 从缓存读的最后一个字节
- LastByteRead 从缓存读出的最后一个字节
- LastByteSent 发送的最后一个字节
- LastByteAcked 确认的最后一个字节
- LastByteRcvd = LastByteRead<=RcvBuffer
- rwnd = RcvBuffer-[LastByteRcvd - LastByteRead]
- LastByteSent-LastByteAcked<=rwnd
- 接收窗口为 0 时,发送方停发,但是数据还不能不传,发送方还知道窗口大小,此时发送方启动坚持定时器,发送零窗口探测报文段,从接收方的响应中获知窗口大小
- 8 连接管理
- 三次握手
- (1)客户 TCP 先向服务器端 TCP 发送 SYN 报文段,不包含应用层数据(SYN=1) 随机选择起始序号 client_init。
- (2)服务器 TCP 用 SYNACK 报文段响应(SYN),分配缓存变量,确认号 client_init+1。给出服务器选择的起始序号 server_init
- 不包含数据(服务器端分配缓存和变量)
- (3)客户用 ACK 报文段响应(SYN=0),分配缓存变量,确认号 server_init+1,可以包含数据。之后 SYN=0。
- 四次挥手
- 客户端:向服务器端发送 FIN=1,等待服务器确认 [FIN_WAIT1]
- 服务器:向客户端发送 ACK,确认其请求 [FIN_WAIT2]
- 服务器:向客户端发送 FIN=1,等待客户端确认 [TIME_WAIT]
- 客户端:向服务器发送 ACK,等待一段时间后结束,所有资源被释放,若此时 ACK 丢失,重传客户 ACK。
- 3.6 TCP 拥塞控制
- TCP 使用端到端拥塞控制机制:发送方根据自己感知的网络拥塞程度,限制其发送速度。IP 向端系统提供显示拥塞反馈。
- 拥塞窗口: cwnd
- LastByteSent-LastByteAcked<= min(rwnd, cwnd)
- 发送速度=cwnd/RTT 字节/秒
- 丢包:超时,3个冗余 ACK
- cwnd 初始为 1 个 MSS。每一个 RTT,速率翻倍。
- 慢启动: cwnd<=ssthresh, cwnd 每次加倍。
- 拥塞避免: cwnd=ssthresh, cwnd+1MSS。
- 丢包: cwnd=1, ssthresh=cwnd/2
- 三个 ACK: cwnd = cwnd/2+3MSS, ssthresh=cwnd/2
- 快速重传:收到三个 ACK 后认为丢包,立即重传报文文,之后进入拥塞避免。
- Reno 快速恢复:三次 ACK 之后 cwnd = cwnd/2+3MSS, ssthresh=cwnd/2,超时 cwnd=1, ssthresh=cwnd/2。
- Tahoe 无论超时还是三次 ACK, cwnd=1, ssthresh=cwnd/2

第三部分:田野野生小测题

- 1.1 一条分组交换链路,带宽 10Mbps,现有 20 位用户先后希望建立并保持通信连接,每位用户产生 1Mbps 的数据流量:
- 2.0 位用户均建立连接,数据通信产生较多丢包和较大时延
- 1.2 下面那种不是 ISP 的连接方式 (a)
- (a) 双方连接内容提供网络实现连接
- (b) 通过客户-提供商方式连接
- (c) 双方接入因特网交换机 (XP) 实现连接
- (d) 双方通过点对点 (Peering) 方式连接
- 1.3 以下哪些协议工作在主机/3 层路由器/2 层交换机上？
- 主机:应用/数据/链路/网络/物理层
- 2.进程 A 想数据搬到另一台主机上的 B 进程,如何识别 B？
- IP 地址+端口号
- 2.2 SSL 工作在上层应用
- HTTP/1.0 一次发一个对象
- P2P 有服务器进程和客户端进程,但是每个主机既是服务器又是客户端
- HTTP 是无状态服务,FTP 是有状态服务
- 3.2 仿访百度,哪个 DNS 服务器你一定会遇到？
- 百度的 localDNS 服务器
- 3.1 TCP/UDP 没有提供的服务:
- 延迟保证/带宽保留
- 3.2 访问百度:假设 q=32 给 B 发了 8Byte A 回 ACK40 然而 B 没收到 A 的 ACK 又给 B 发了 20A 回 ACK60
- 3.3 TCP 接收窗口在连接持续的过程中可能会变化。
- 用户 A 的未确认比特不能超过接受缓冲区大小。
- TCP 中 A 发了序号 m,那么下一个序号不是 m+1
- TCP 段中有窗口大小
- TimeoutInterval=SampleRTT
- A 的 seq 是 38,发了 4byte 回的回 ACK 应该是 42,但是 A 的 seq 和 B 的 seq 没发, B 是按照 A 的 ACK 回复的。

第四部分:小测题补充

- 1.TCP 如何发送紧急数据？
- 答: (1)紧急标志位 U(URG)置 1;
- (2)紧急数据置于 TCP 段数据(载荷)前部;
- (3)紧急指针指向紧急数据的最后一个字节。
- 2.TCP 接收方何种情形需要立即进行确认？
- 答: (1)连续两个字节到达时,且前一个未确认;
- (2)收到失序段(序号比期望的序号大);
- (3)收到重复段。
3. Alice 从她的终端登陆到公司的文件服务器上下载了 4 个文件。请问 Alice 的终端和公司的文件服务器之间总共建立了几条 TCP 连接?这些 TCP 连接分别用来传输什么?
- 答: 5 条连接。一条控制连接,用于传输命令和响应。四条数据连接,每条连接用于传输一个文件。
- 4.在某个时刻, Alice 的邮件服务器和 Bob 的邮件服务器之间需要交换一批邮件,这两个邮件服务器之间需建立几条 TCP 连接？
- 一条 TCP 连接。
5. 以下哪些应用层协议可能会被用来传输一个邮件报文: HTTP, FTP, SMTP, POP, DNS?
- HTTP, SMTP, POP
6. Alice 向 Bob 发送了一封邮件, Bob 阅读了这封邮件。这封邮件在 Alice 的用户代理、Bob 的用户代理、Alice 的邮件服务器、Bob 的邮件服务器中停留过。请按照顺序列出这封邮件经过的对方,经过邮件服务器时指出是进入发队列还是进入信箱。
- 邮件从 Alice 的用户代理到到达 Alice 的邮件服务器,进入发队列;然后到达 Bob 的邮件服务器,进入 Bob 的邮箱,最后到达 Bob 的用户代理。
- 7.如果 TCP 服务器支持 n 个并发连接,每个连接来自不同的客户机主机, TCP 服务器将需要多少个套接字?这些套接字是怎么分配的?
- 服务器需要 (n+1) 个套接字。一个套接字用于监听来自客户的连接请求;其余 n 个套接字,每个用于和一个客户进程进行通信。
- 8.是非判断题:
- 1) 假设主机 A 通过一条 TCP 连接到主机 B 发送一个大文件,如果某个报文段的序号为 m,则其后续报文段的序号必是 m+1。(×)
- 2) 假设主机 A 通过一条 TCP 连接到主机 B 发送一个序

- 号) 为 38、包含 4 个数据字节的报文段, 则主机 B 对该报文段的确认号必定是 42。(×)
- 3) 假设主机 A 通过一个 TCP 连接向主机 B 发送一个大文件, 主机 A 已发送但未被确认的字节数不会超过接收缓存的大小。(√)
- 4) 在 TCP 连接的持续过程中, TCP 头中的 rwnd 不会变化。(×)

第二章补充

- 1.计算机网络向用户提供的最重要的功能、连通性、共享
- 2.不同的网络设计什么为区分：所提供的服务
- 3.服务以什么为区分：延迟、带宽、丢失率、节点数、服务接口、可靠性、单播/多播、延迟、消息/字节流
- 4.路由 (route, path)：分组从发送终端到接收终端经过的通信链路及分组交换机的序列。
- 5.ISP：由分组交换机和通信链路组成的网络；为终端提供接入因特网的服务
- 6.因特网提供的通信服务包括：源主机到目的主机的可靠数据交付尽力而为的 (不可靠的) 数据交付
- 7.协议定义了通信实体之间交换的报文的格式和次序, 以及在报文发送、接收或收到其它事件后采取的动作。
- 8.因特网生态系统：接入 ISP 区域 ISP 第一层 ISP 存在点 PoP: 低层 ISP 接入高层 ISP 的地方 多播：一个 ISP 可以与两个或多个上层 ISP 连接 对称：相同层次上的一对 ISP 直接相连, 不进行计算 因特网交换点 IXP: 多个 ISP 在这里共同对称
- 9.节点延迟：dproc = 处理延迟 典型值为几个微秒或更低
- dqueue = 排队延迟 取决于拥塞程度
- dtrans = 传输延迟 微秒-毫秒 低速链路上较大
- dprop=传播延迟 几微秒-几百毫秒, 长距离链路上较大
- 10.为什么分层? 易于处理复杂系统 显式的层次结构易于确定系统的各个部分及其相互关系; 模块化简化了系统的维护和升级; 改变某层服务的实现对于其它层次是透明的
- 11.SMTP 使用持久连接：可以在一条 TCP 连接上传输多个报文一个方向的报文传输结束后, 可以在另一个方向上传输报文
- 12.Q: 能将用户信息放在本地终端吗?
- A: 不能, 用户终端不可能一直在因特网上
- Q: 能用 SMTP 从邮件服务器中获取邮件吗?
- A: 不能, SMTP 是一个“推”协议, 只能将邮件从用户代理推送到其邮件服务器
- 13.邮件访问协议：可以从服务器获取邮件的协议有: POP IMAP HTTP
14. DNS 是因特网的核心功能, 但实现为一个应用层服务：使用客户-服务器模式运行在端系统之间; 利用传输层协议传输报文使用者不是用户, 而是其它应用程序
15. Q: 为什么不使用集中式的 DNS?
- 单点失效; 流量集中: 单个 DNS 服务器需处理全部查询; 响应时间长: 远距离的集中式数据库; 需要维护庞大的数据库

- 域名和标识号
- 树上每一个节点都有一个标识号 (最多 63 个字符), 根标识号是空字符串。树中一个特定的节点以及该节点下所有的节点构成一个域。某个域的名字表示为从该域开始向上直到树根 (为空的) 的标识号序列, 标识号之间用句点隔开。域名的任一后缀也是一个域。
- 顶级域：顶级域分为组织域、国家域和反向域三种。
- 组织域：美国国内及一些国际组织使用。
- 国家域：使用二字符的国家代码, 每个国家对应一个。
- 反向域：域名为arpa, 用来把一个 IP 地址映射为名字。

- DNS: 根域名称服务器
- 全球有 13 个根服务器; 根服务器知道所有顶级域 (TLD) 服务器的 IP 地址
- TLD 服务器, 权威服务器
- 顶级域 (TLD) 服务器: 每个 TLD 服务器负责一个顶级域; 知道其所有二等级域的域名服务器地址
- 权威 DNS 服务器: 机构的 DNS 服务器, 提供机构内服务器 (如 Webmail) 的主机名-IP 地址映射; 提供一个主域名服务器, 一个或多个辅助域名服务器; 可由机构维护, 也可委托 ISP 维护
- 本地 DNS 服务器

- 严格来说不属于 DNS 服务器的层次结构; 每个 ISP 都有一台本地 DNS 服务器, 也称“默认”的 DNS 服务器; 解析器的 DNS 查询报文实际上发送给本地 DNS 服务器, 本地 DNS 服务器起着代理的作用, 负责将 DNS 查询报文发送到 DNS 层次结构中, 并将查找结果返回给解析器。

17. DNS zone
- 整个 DNS 名字空间被划分成一些不重叠的区域, 称为 DNSzone, 每个 zone 包含域名树的一部分; 在管理上, 每个 DNS zone 代表一个权威域的边界。
18. DNS 缓存
- 每当收到一个响应报文, DNS 服务器将报文中的映射信息缓存在本地。DNS 服务器首先使用缓存中的信息响应查询请求 DNS 缓存中的映射在一定时间后被丢弃并特别地, 本地 DNS 服务器通常会缓存 TLD 服务器的 IP 地址, 因而很少去访问根服务器
- 19.应用通过 socket API 可以调用两种传输服务不可靠的数据报服务 (UDP 协议) 可靠的字节流服务 (TCP 协议)
20. UDP
- 报文传输服务
- 由于没有建立管道, 应用程序发送每个报文必须给出远端进程地址
- 服务器使用同一个进程和一个套接字为所有客户服务, 一次请求-响应完成一次服务
- TCP
- 字节流传输服务
- 由于建立了管道, 应用程序只需向套接字中写入字节序列, 不需指出远端进程地址
- 服务器为每个客户单独生成一个套接字和一个新进程, 允许双方长时间通信

第三章补充

- 1.网络层提供主机之间的逻辑通信
- 传输层: 提供进程之间的逻辑通信, 依赖并增强网络层服务
- 2.UDP 套接字的端口号是怎么分配的?
- 为套接字自动分配端口号: 端口号通常从 1024-65535 中分配, 客户通常使用这种方法使用指定端口号创建套接字: 实现公共域地址的服务器应分配众所周知的端口号, 服务器通常采用这种方法
- 3.无连接多路分解
- 当接收方传输层收到一个 UDP 报文时:
- 检查报文中的目的端口号
- 将 UDP 报文交付给具有该端口号的套接字
- 具有相同 (目的 IP 地址, 目的端口号) 的 UDP 报文被交付给同一个套接字, 与源 IP 地址及源端口号无关
- 报文中的 (源 IP 地址, 源端口号) 被接收进程用来发送响应报文
- 4.创建 TCP 套接字
- 服务器在 port=x 创建一个欢迎套接字;
- 客户 A 创建一个与欢迎套接字通信的客户套接字 (假设自动分配端口号 y):
- 服务器在收到客户 A 的连接请求后创建一个连接套接字:
- 该连接套接字只与客户 A 的套接字通信, 即只接收具有以下四元组的报文段:
- 源 IP 地址 = 客户 A 的 IP 地址
 - 源端口号 = y
 - 目的 IP 地址 = 服务器的 IP 地址
 - 目的端口号 = x
- 不同的客户进程与服务器上不同的连接套接字对应
- 5.面向连接的 mult路分解
- 服务器主机可同时支持很多连接套接字
- 每个连接套接字与一个进程相联系, 并由以下四元组标识: 源 IP 地址 源端口号 目的 IP 地址 目的端口号
- 服务器使用该四元组将报文段交付到正确的连接套接字
- 6.为什么需要 UDP?
- 没有建立连接的延迟
- 协议简单: 发送端和接收端不需要保存连接状态
- 报文开销小
- 没有拥塞控制和流量控制: UDP 可以尽快地发送报文
- 若需要利用 UDP 可靠传输的应用层实现可靠性
7. rdt3.0: 可能产生比特错误和丢包的底层信道需要两项新技术:

- 检测丢包
- 从丢包中恢复
- 检测丢包:
- 若发送方在“合理”的时间内未收到 ACK, 认为丢包 (需要定时器)
- 从丢包中恢复:
- 发送方重发当前分组
- 接收方使用分组序号检测重复分组
8. 发送方窗口和接收方窗口
- 发送方窗口
- 限制已发送未确认的序号
- 当收到基序号的 ACK 时, 滑动发送窗口
- GBN: 已发送已确认和已发送未确认的序号不交织 (采用累积确认)
- SR: 已发送已确认和已发送未确认的序号可能交织在一起
- 接收方窗口
- 限制可以接收的分组序号
- 接收窗口之外的分组被丢弃
- GBN: 接收窗口只包含当前期待接收的分组序号
- SR: 接收窗口包含以下三种序号:
- 期待但未收到
- 失序 (已缓存) 且已确认
- 可接受
9. TCP 协议在减少重传方面的设计:
- 利用流水式发送和累积确认, 可避免发生某些丢失了 ACK 的报文段
- 只使用一个重传定时器, 可避免超时时间值小时大量报文的重复

10. TCP 确认的二义性
- 估计 RTT 的问题:
- TCP 是对接收到的数据而不是对携带数据的报文段进行确认因此 TCP 的确认是二义性的。
- 解决方法:
- 忽略有二义性的确认, 只对一次发送成功的报文段测量 SampleRTT, 并据此更新 EstmitedRTT。
- 当 TCP 重传一个段时, 停止测量 SampleRTT。
- 定时器补偿
- 简单忽略重传报文段的问题:
- 重传意味着超时设置偏小了, 需加大。
- 但若简单忽略重传报文段 (不估计、不更新 RTT), 则超时设也不会更新, 形成反复重传的局面。
- 解决方法:
- 采用定时器补偿策略, 发送方每重传一个报文段, 超时值就增大。
- 若连续发生超时事件, 超时值呈指数增长 (至一个规定的上限值)。

- Karn 算法
- Karn 算法结合使用 RTT 估计值和定时器补偿策略确定超时值: 使用 EstimatedRTT 估计初始的超时值
- 若发生超时, 每次重传时定时进行补偿, 直到成功传输一个报文段为止。
- 若收到上层应用数据、或某个报文段没有重传就被确认了, 用最近的 EstimatedRTT 估计超时值。

11. 推迟确认
- 推迟确认的优点:
- 可以减少通信量
- 推迟确认的缺点:
- 当延迟太大时, 会导致不必要的重传
- 推迟确认造成 RTT 估计不准确
- TCP 协议规定:
- 推迟确认的时间最多为 500ms
- 接收方至少每隔一个报文段使用正常方式进行确认
12. TCP 使用 GBN 是 SR?
- Go-Back-N
- 接收方:
- 不缓存失序的分组
- 使用累积确认
- 对失序分组发送重复 ACK
- 发送方:
- 超时后重传从基序号开始的所有分组
- 仅维护基序号和下一个序号

- TCP
- 接收方:
- 缓存失序的报文段
- 使用累积确认
- 对失序报文段发送重复 ACK
- 发送方:
- 超时后仅重传最早未确认的报文段
- 仅维护基序号和下一个序号
- 选择重传
- 接收方:
- 缓存失序的分组
- 单独确认每个正确收到的分组
- 发送方:
- 仅重传未被确认的分组
- 修改的 TCP [RFC2018]
- 接收方:
- 缓存失序的报文段
- 在 SACK 选项头中给出收到的非连续数据块的上下边界
- 发送方:
- 仅重传接收方缺失的数据

- 13.零窗口探测
- 接收窗口为 0 时, 发送方必须停止发送。
- 问题：接收方如何通告增大了的接收窗口?
- 触发 TCP 传输的条件:
- 应用程序调用, 超时, 收到数据 (发送 ACK)
- 对于单向传输中的接收方, 只有第三个条件能触发传输

- TCP 协议规定:
- 发送方收到零窗口通告后, 可以发送零窗口探测报文段, 以触发一个包含接收窗口的响应段
- 发送端收到窗口通告时, 启动一个坚持定时器
- 定时器超时后, 发送端发送一个零窗口探测报文段 (包含前一个报文段的最后一个字节)
- 接收端在响应的报文段中通告当前接收窗口的大小
- 若发送端仍收到零窗口通告, 重新启动坚持定时器
- 14.接收方启发式策略
- 接收端避免糊涂窗口综合症的策略:
- 通告零窗口之后, 仅当窗口大小显著增加之后才发送更新的窗口通告。
- 什么是显著增加: 窗口大小达到缓存空间的一半或者一个 MSS, 取两者的较小值。
- TCP 执行该策略的做法:
- 当窗口大小不满足以上策略时, 推迟发送确认 (但最多推迟 500ms, 且至少每隔一个报文段使用正常方式进行确认), 寄希望于推迟时间隔内有更多数据被消费
- 仅当窗口大小满足以上策略时, 再通告新的窗口大小

- 发送方启发式策略
- 发送方避免糊涂窗口综合症的策略:
- 发送方应积累足够多的数据再发送, 以防止发送太短的报文段。
- 问题: 发送方应等待多长时间?
- 如等待时间不够, 报文段会太短
- 如等待时间过久, 应用程序的延时会长
- 更重要的是, TCP 不知道应用程序会不会在最近的将来生成更多的数据

- Nagle 算法
- 在新建连接上, 当应用数据到来时, 组成一个 TCP 段发送 (哪怕只有一个字节)
- 在收到确认之前, 后续到来的数据放在发送缓存中
- 当数据量达到一个 MSS 或上一次传输的确认到来 (取两者的较小时间), 用一个 TCP 段将缓存的字节全部发送
- Nagle 算法的优点:
- 适应网络延时, MSS 长度及应用速度的各种组合
- 常规情况下不会降低网络的吞吐量
- 15.流量控制与拥塞控制:
- 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

- 16.拥塞的代价
- 拥塞造成:
- 丢包 (缓存溢出) 分组延迟增大 (链路接近满载)
- 检测丢包:
- 大量网络资源用于: 重传丢失的分组 (不必要地) 重传延迟过大的分组 转发最终被丢弃的分组
- 结果: 网络负载很重, 但网络吞吐量很低。
- 17.ATM 使用网络辅助的拥塞控制, 路由器向端系统反馈。
18. TCP 的平均吞吐量是多少? (忽略慢启动阶段)
- 令 W=丢包发生时的 CongWin, 此时: throughput = W/RTT
- 刚发生丢包时, CongWin=W/2, 此时: throughput=W/2RTT。
- 假设在 TCP 连接的生命期内, RTT 和 W 几乎不变, 则: Average throughput=0.75 W/RTT
- 19.吞吐量与丢包率 L 的关系: 1.22*MSS/(RTT*L*0.5)

小测试补充

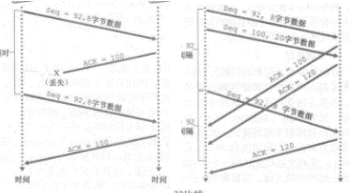
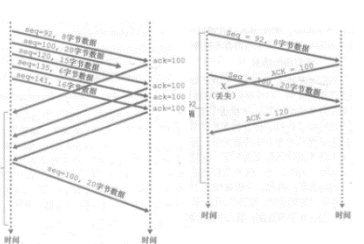
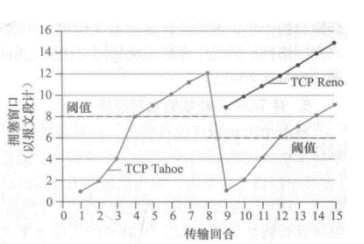
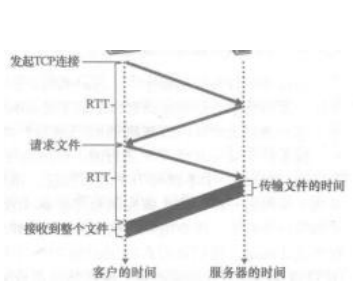
1. 发送方 TCP 的基序号 SendBase 和接收方缓存中的 LastByteRcvd 之间的关系为 (A)
- (A) LastByteRcvd \geq SendBase-1 (B) LastByteRcvd \geq SendBase (C) 不能确定
- 2.假设发送方 TCP 收到了确认序号 y (表示 y 之前的字节均已正确收到), 则 y 与接收方缓存中的 LastByteRcvd 之间的关系为 (B)
- (A) LastByteRcvd = y-1 (B) LastByteRcvd \leq y-1 (C) 不能确定
- 3.主机 A 向主机 B 发起一个 TCP 连接, 假设主机 A 和主机 B 选择的起始序号分别为 70 和 90, 将下表中三次握手交换的报文段的相关信息填写完整。
- | 报文段 | SYN flag | ACK flag | Seq number | Ack number |
|-----|----------|----------|------------|------------|
| 1 | 1 | 0 | 70 | -- |
| 2 | 1 | 1 | 90 | 71 |
| 3 | 0 | 1 | 71 | 91 |

4. TCP 用于流量控制的窗口是 接收窗口, 用于拥塞控制的窗口是 拥塞窗口。
- 5.假设主机 A 在一条 TCP 连接上发送了一大批数据, 然后在 t1 时刻获得空闲 (因为没有更多的数据需要发送)。在相对较长的一段时间空闲后, 在 t2 时刻又有一大批数据需要发送。你认为此时主机 A 应当使用 t1 时刻的 CongWin 和 Threshold, 还是应当使用慢启动发送数据? 为什么?
- 答: 应使用慢启动发送数据。从愿意来看, t1 时刻的 CongWin 和 Threshold 可能较大。经过了相对较长的一段时间后, 网络状态可能发生了变化。此时应使用慢启动逐渐提高发送速度, 以免一下子发送大量数据引起网络拥塞。
- 6.写出至少三种接入网络的技术。对于每一种接入网技术, 指出使用的传输媒体是什么。

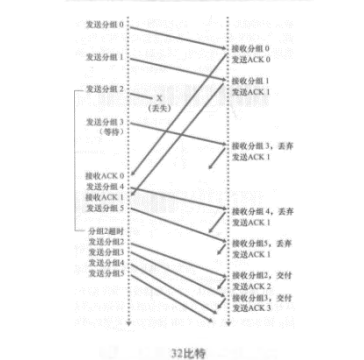
- 接入网技术: DSL, HFC, 以太网, WiFi, 3G, …… (写出三种即可) DSL 使用双绞线, HFC 使用光纤和同轴电缆, 以太网使用双绞线或光纤, WiFi 和 3G 使用电磁波
- 7.分组在交换网络中要经历哪四种延迟? 哪种延迟的变化范围最大? 什么情况下会出现丢包
- 四种延迟: 处理延迟, 排队延迟, 传输延迟, 传播延迟
- 排队延迟的变化范围最大。
- 当分组到达交换机设备时, 若输出链路的缓冲队列满, 发生丢包。
- 8.从高到低列出因特网协议栈的五个层次。主机上运行哪些层次? TCP 协议运行在哪个层次? IP 协议运行在哪个层次? HTTP 协议运行在哪个层次?
- 因特网协议栈的五个层次: 应用层, 传输层, 网络层, 数据链路层, 物理层
- 主机上运行全部五个层次
- TCP 运行在传输层
- IP 运行在网络层
- HTTP 运行在应用层

- 9.分组交换和电路交换中的同步时分复用 (TDM), 都是让用户轮流使用链路, 它们之间的区别是什么?
- 分组交换: 用户使用链路的方式不固定
- TDM: 用户使用链路的方式固定
10. A 和 B 两个终端通过一台分组交换机连接到一起。两段链路的数据速率分别为 R1 和 R2, 忽略信号传播时间。A 向 B 连续发送 2 个长度为 L 的分组。假设接收端上没有其它分组传输, 请问从 A 开始发送到 B 完整收到 2 个分组, 中间经过了多长时间? (提示: 分 R1 \leq R2 和 R1>R2 两种情况考虑)
- 若 R1 \leq R2, T = 2L/R1 + L/R2
- 若 R1>R2, T = L/R1 + 2L/R2
- 11.服务器通过包含两条链路的路径, 向客户端传输两个数据包, 每个长度为 L, 两条链路的传播时延均为 dprop。
- (1) 如果第一条链路是整个路径的瓶颈, 即 R<Rc, 服务器“背靠背”传输数据, 即完成第一个数据包传输后立即开始传输第二个数据包。问客户端收到两个数据包的第一个 bit 之间间隔多长时间?
- (2) 如果第二条链路是整个路径的瓶颈, 则 Rc<Rs。服务器完成第一个数据包的传输后等待 T, 再传输第二个数据包, 问 T 满足什么条件, 第二个数据包在路由器中不会排队等待?
- (1) L/RS
- (2) 第二个数据包到达路由器的时间: L/RS + dprop + T + L/RS
- 第一个数据包离开路由器的时间: L/RS + dprop + L/RC
- 需满足 L/RS + dprop + T + L/RS > L/RS + dprop + L/RC
- 所以, T > L/RC - L/RS
- 12.假设您使用 Web 浏览器单击一个链接以检索引用同一服务器上的大量非小对象的 HTML 网页。Web 服务器和本地主机之间的 RTT 是 RTT0。忽略传输时间, 会花费多少时间
- a. Non-persistent HTTP with no parallel TCPconnections?
- 2 \times RTT0+8 \times 2 \times RTT0
- b. Non-persistent HTTP with the browser configured for 5 parallel connections?
- 2 \times RTT0+2 \times 2 \times RTT0
- c. Persistent HTTP?
- 2 \times RTT0+RTT0

- 13.Which one of the following statements is wrong(multiple choices) (a)(c)(d)
- (a) SSL is at the transmission layer.
- (b) HTTP/1.0 retrieves one object with one TCPconnection.
- (c) There is no server process and client process forP2P network architecture.
- (d) Both HTTP and FTP protocols are stateless.
14. Suppose you issues a DNS query on the name “www.baidu.com” from USTC campus network, which of the following DNS servers are definitely not involved in the name resolution process? (c)
- (a) Root DNS server
- (b) “com” TLD server
- (c) Local DNS server of Baidu company
- (d) Local DNS server of USTC
- (e) Authoritative server of the “baidu.com” domain
- 15.Consider the case of Web cache as in the figure .The institutional network users issue 16 requests per second, the averaged object size is 1.2M bits. What is the minimum cache hit ratio so that the access link’s utilization ratio is below 100%?
- Ans: suppose hit ratio is p
- 16 \times 1.2 \times (1-p) < 15 Mbps
- p > 21.875%
16. 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. (False)
17. 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. (True)
18. The TCP segment has a field in its header for rwnd.(True)
19. Suppose that the last SampleRTT in a TCP connection is equal to 1 sec. The current value of Timeout Interval for the connection will necessarily be \geq 1 sec. (False)



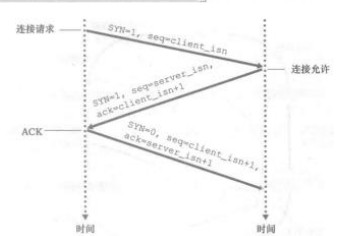
源端口号	目的端口号
序号	接收窗口
确认号	紧急数据指针
选项	数据



源端口号	目的端口号
长度	检验和
应用数据 (报文)	

图 3-7 UDP 报文段结构

标识符	标志	
问题R数	回答R数	12 字节
问题R数	附加R数	
问题 (问题的变量数)		查询的名称和类型字段
回答 (资源记录的变量数)		对查询的响应中的RR
权威 (资源记录的变量数)		权威服务器的记录
附加信息 (资源记录的变量数)		可被使用的附加“有帮助的”信息



PB19010450 和泳毅