topdown

2019年9月3日 23:31

说明：主要是《计算机网络：一种自顶向下的方法》前三章的读书笔记，比计算机网络课程要求的知识点略多。对应计算机网络期中考试范围。

1. 计算机网络和因特网
    a. 什么是因特网
        i. 主机（host）上有好多客人（应用程序），处于网络的边缘，所以也称为端系统（end system），它包括client和server
        ii. 网络边缘包括host和网关
        iii. 主机通过通信链路（communication link）或分组交换机（packet switch）连接在一起
            1) 通信链路类似高速公路，分组交换机类似交叉口
            2) 分组交换机包括路由器（router）、链路层交换机（link-layer switch）
            3) 因特网的标准文档称为RFC（request for comment）
        iv. 运行在不同端系统上的软件之间怎么发送数据呢?
            1) 使用套接字接口（socket interface）
    b. 接入网
        i. 将端系统连接到其边缘路由器（edge router）
        ii. 家庭接入
            1) DSL（digital subscriber line，数字用户线）
                a) 电话公司就是他的ISP
                b) 使用现有的电话线（双绞铜线），和电话公司中的本地中心局（CO）中的数字用户线接入复用器（DSLAM）交换数据
                c) 家里的调制解调器得到数字数据后将其转换成高频音，通过电话线传到CO后被DSLAM转换成数字信号
                d) 分配器将家里的数据信号和电话信号分隔开
                e) 电话线频段
                    i) 高速下行信道：50kHz 到 1MHz
                    ii) 中速上行信道：4kHz 到 50kHz
                    iii) 普通双向电话通道：0 到 4kHz
            2) 电缆
                a) 电缆因特网接入（cable Internet access）利用有线电视的基础设施
                b) 连接过程
                    i) 因特网连接到CMTS（cable modem termination system，电缆调制解调器端接系统）
                    ii) CMTS也是将模拟信号转换成数字信号
                    iii) CMTS通过光缆连接到光纤节点
                    iv) 光纤节点伸出若干同轴电缆
                    v) 家庭通过以太网端口连接这些电缆的支路
                c) 所以也被称为混合光纤同轴（hybrid fiber coax，HFC）系统
            3) 光纤到户（fiber to the home，FTTH）
                a) 从CO直接连一条光纤路径到家庭
                b) 直接光纤
                    i) CO到家就一条光纤
                c) 更一般的

i) 从CO出来的每一根光纤直到接近家庭的位置才分开来让好几个人连
ii) 体系结构方案
One. AON（active optical network，主动光纤网络）
Two. PON（passive optical network，被动光纤网络）
First. 被用于Verizon的FIOS服务中

4) 卫星链路
a) 很慢，但是在偏远地区没得选

iii. 企业（和家庭）接入
1) 以太网
a) 各种端系统包括服务器都和以太网交换机相连
b) 连接线路：双绞铜线
2) WiFi
a) 无线LAN

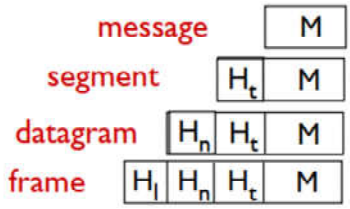
iv. 广域无线接入
1) 3G和LTE（long term evolution）
2) 移动设备使用了和无线电话相同的无线基础设施
3) 通过基站使用无线电频谱发送接收分组

v. 物理媒体：
1) 物理媒体的成本小，人力安装成本甚至可能比材料成本大几个数量级
2) 双绞铜线
a) 结构：两根绝缘铜线绞合，许多双绞线捆扎在一起
b) 用途：电话机、电话交换机，成为了LAN的主导解决方案
3) 同轴电缆
a) 两根绝缘铜线同心
b) 用途：电缆电视
4) 光纤
a) 优点：衰减低，难窃听
b) 缺点：各种发射、接收、交换机成本高
5) 陆地无线电信道
a) 有长中短距离三种
b) 短的可以用于头戴式耳机
6) 卫星无线电信道
a) 同步卫星、近地卫星

c. 网络核心
i. 分组交换
1) 机制：存储转发传输（store-and-forward transmission）
a) 任务：把 入分组 交换到 出链路
b) 时延：
i) L：报文长度；R：路由器速度
ii) 转发时延
One. 进来：L/R
Two. 全进来以后转发：L/R
Three. 所以：2L/R
Four. 如果有N条链路，N-1台路由器，则时间为
First. d=NL/R
iii) throughtput是平均值

ii. 排队时延（queuing delay）和分组丢失（丢包，packet loss）
1) 每台分组交换机都有输出缓存（output buffer），也称为输出队列（output queue）

iii. 转发表和路由选择协议
    1) 路由器的forward实际上可能是不同的
    2) 因特网具有一些特殊的路由选择协议（routing protocol）
iv. 电路交换（circuit switching）
    1) 有预留的路线，适合比较稳定的场景
    2) 频分复用（frequency-division multiplexing，FDM）
        a) 频段的宽度称为带宽
    3) 时分复用（time-division multiplexing，TDM）
        a) 时间被划分成固定的帧，每一个帧中有若干个time slice，称为时隙
v. 分组交换和电路交换的比较
    1) 分组交换不适合实时服务
    2) 分组交换更简单、有效，成本低
    3) 目前的趋势是朝着分组交换方向发展
vi. 网络相连
    1) 底层ISP向顶层ISP付费并于其通过 PoP 相连，一个底层ISP可以和多个顶层ISP相连，称为多宿（multi-home）
    2) 顶层ISP之间互联
    3) 内容提供商与顶层ISP相连时要付费，但是可以与底层ISP通过IXP（Internet exchange point，互联网交换点）互连而不用向其付费
    4) 很多ISP在IXP处对等（peer）互免
    5) 还有一些区域ISP，比如中国的省级ISP
vii. 时延综述
    1) 节点处理（nodal processing）时延
        a) 运用汉明码等看看有没有比特级别的错误，错了就丢弃（drop）这个包
        b) 检查分组的首部确定这个分组的去处
        c) 微秒级别，甚至更低
    2) 排队（queuing） 时延
        a) 流量强度（traffic intensity）
    3) 传输（transmission）时延
        a) 毫秒到微秒级别
    4) 传播（propagation）时延
    5) 合称为 节点总时延（total nodal delay）
    6) 还有其他时延，比如WiFi可能会有意地降低速度
viii. 吞吐量
    1) 瞬间吞吐量（instantaneous throughput）
    2) 平均吞吐量（average throughput）
        a) 服务器到客户的文件传输吞吐量是 min{R1， R2，R3，……}，即它是bottleneck link（瓶颈链路）的传输速率
        b) 今天因特网中的瓶颈链路通常是接入网
    3) 一个粗大的链路可能会因为使用者过多而成为瓶颈链路
ix. 协议分层
    1) 各层的协议加起来被称为协议栈
    2) 分层的缺点：
        a) 冗余较低层的功能，如：纠错
        b) 某些功能可能仅仅需要在其他某层中出现的信息，这违背了分层的目的
        c) 报文、报文段、数据段、帧

i) 

x. 计算机安全
   1) botnet：僵尸网络
   2) malware
      a) 蠕虫（worm）：
         i) 无需明显的用户交互就能进入设备
      b) virus：
         i) 需要用户的action

2. 应用层
   a. 应用层协议原理
      i. 网络核心设备主要在网络层及下面起作用
      ii. 流量密集的服务很多是用P2P体系结构，在这种结构中服务器和客户机有时会mix
   b. 进程通信
      i. 同一台主机上的进程通信由操作系统负责，不同的主机的进程通信由计网负责
      ii. 在P2P中，发起通信的被视为客户机，等待联系的是服务器
      iii. 进程通过套接字（socket）收发网络中的报文
      iv. 应用程序开发者对运输层的控制仅限于
         1) 选择运输层协议
         2) 也许能设置一些运输层参数，比如：最大缓存、最大报文长度
   c. 进程寻址
      i. 想发信息，需要定义
         1) 接收主机的地址
         2) 接收主机的接收进程，即接收套接字。即目的地端口号
            a) 比如邮件的端口号是25，Web是80
   d. 运输层协议比较
      i. TCP：稳定、安全、吞吐量小
      ii. UDP：不握手、发送端可以选择以任意速率发送
         1) 许多防火墙被配置成阻挡UDP流量，所以因特网电话一般用TCP作为备份，如果UDP失败了，就使用TCP
   e. TCP安全
      i. TCP、UDP都没有加密机制
      ii. TCP的加强版：安全套接字层（secure sockets Layers，SSL）
   f. Web和HTTP
      i. 概论
         1) 浏览器就是客户
         2) 页面就是由各种对象构成的
         3) TCP提供可靠服务，一定会将双方的报文传递成功
         4) HTTP是无状态协议，不存储关于该客户的状态信息，哪怕同一个客户反复请求报文，也会反复发送
         5) RTT（round-trip-time，往返时间）
            a) 报文从客户到服务器再返回客户
      ii. 非持续连接（non - persistent connection）
         1) 举例：客户请求一个有一个HTML基本文件和10个JPEG图像的页面
            a) 客户先建立连接，然后先传基本文件，TCP确认客户收到了完整地收到后，才会中断传输
            b) 再一个个传图像，每次都中断连接

2) 大部分浏览器打开5~10个并行的TCP连接，但是可以设置成1然后强行串联
3) 这样，每一个对象都要使用两个RTT
    a) 客户发申请，服务器确认申请，客户发所需的链接，客户确认收到了

iii. 持续连接（persistent connection）
1) HTTP默认使用流水线式的持续连接，服务器可以一个个发送，不必等客户确认
2) 同一个客户向同一个服务器请求多个页面时，甚至可以在单个TCP上进行

iv. HTTP报文
1) 报文使用ASCII编写
2) 这个报文的每一行都可以扩展成若干行
3) 请求报文
    a) 当你在浏览器中输入www.someschool.edu/somedir/page.html
    b) 发送的请求报文：

        Get /somedir/page.html HTTP/1.1
        Host: www.someschool.edu
        Connection: close
        User-agent: Mozilla/5.0
        Accept-language: fr

    c) 请求行（require line）
        i) 方法字段
        One. 包括Get、POST、HEAD、PUT、DELETE
        Two. 一般使用GET
        ii) URL字段
        iii) HTTP版本字段
    d) 首部行（head line）
        i) 在请求行后面所有行
        ii) close是说让服务器发送并且确认客户收到了之后关闭连接
        iii) user-agent指发送的版本，不同浏览器的版本不一样
        iv) 接收语言是法语（如果这个服务器有这个语言版本的话），默认是英语
    e) 一般首部行后面会有空行，空行后面有实体行，但是在Get类型的报文中没有
    f) POST
        i) 用搜索引擎键入关键词搜索（提交表单）时，一般使用POST方法，这样的话实体行中会有键入的关键词
    g) URL method
        i) 使用搜索引擎时仍然可以使用GET，就是URL就改变，加上关键词
        ii) 比如填写mokey banana ，报文可能就是 www.somesite.com/animalsearch? mokey&banana
    h) HEAD
        i) 服务器会用一个HTTP报文进行响应，但是不会返回请求对象
        ii) 经常用来调试跟踪
    i) PUT
        i) 用户上传对象到服务器上
    j) DELETE
        i) 用户删除服务器上的对象
4) 响应报文
    a) 状态行（status line）
        i) 协议版本：HTTP/1.1
        ii) 状态码：
        One. 200 OK

Two. 301 moved permanently：新的URL会在响应报文的首部行location中，然后client 会自动获取新的URL

Three. 400 Bad request：一个通用差错代码，指示该请求不能被服务器理解

Four. 404 not found：该文档不在服务器上

Five. 505 HTTP version not supposed：服务器不支持请求报文用的HTTP版本

iii) 相应状态：OK

b) header line（首部行）

i) connection

ii) date

iii) last-modified

iv) content length

v) content type

c) entity body（实体体）

i) 数据

5) cookie：

a) client发一个请求报文后，server创建一个ID并且发给client，于是client的cookie中就多出一行，然后服务器就可以识别了

6) web缓存器（Web cache）

a) 也称为代理服务器（proxy server）

b) 过程

i) 浏览器创建一个到Web缓冲器的TCP链接，向它发送一个HTTP请求

ii) Web缓冲器看看自己有没有存储该对象的副本，如果有就发，如果没有就向原网站要一份存储起来再发个浏览器

c) 注意：

i) web缓冲器在这里既是浏览器的服务器又是原网站的客户

ii) 它通常是由ISP购买安装的，比如一个大学可能就有一个

d) 优点：

i) 快、省通信量=省钱

7) 条件GET（conditional GET）方法

a) 目的：确保缓冲器中的对象副本能够及时更新

b) 形式：含有“If-Modified-Since”首部行的GET报文

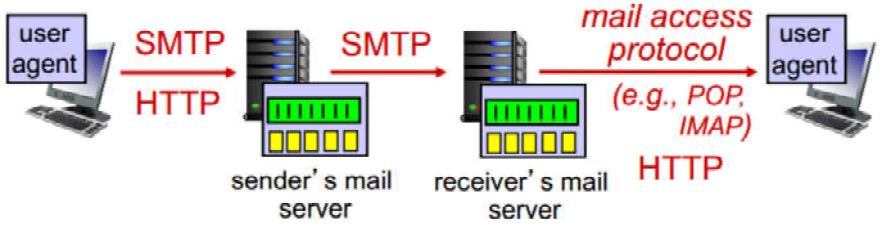
i) GET 对象路径 HTTP/1.1

Host：网站域名

If-Modified-Since：时间

c) 时间：在每一次浏览器请求这个对象的时候，缓存器都会发一个条件请求报文，如果服务器回复一个304 Not Modified，就说明没有修改

g. 电子邮件

i. 

ii. SMTP（simple mail transfer protocol，简单邮件传输协议）

1) 很久以前就出现了

2) 是一个推协议（push protocol），把发送的文件推向接收邮件服务器

3) 要求邮件报文主体部分的体部分也用ASCII表示，所以邮件中的多媒体数据需要encode和decode

4) SMTP不使用中间邮件服务器，收发双方的邮件服务器直接连接，无论相距多远

5) 端口号：25

6) 使用TCP持续连接
7) 有个massage queue，里面的邮件一般每半小时就尝试发送一次，一段时间还不能发出就告诉使用者说这邮件发不了

iii. POP3（Post Office Protocol——version 3，第三版的邮局协议）
1) 功能简单
a) 特许（authorization）
i) 用户代理以明文发送用户名和口令
b) 事务处理
i) 用户取回报文（list retr）
ii) 给想删除的邮件做一个标记（del）
iii) 取消删除标记
iv) 查看邮件的统计信息
c) 更新
i) 用户发出quit的命令，然后就结束了这个POP3会话
ii) 邮件服务器接着就删除那些被标记的报文
2) POP3服务器保留了状态信息，但是不会在会话过程中携带状态信息

iv. IMAP（Internet Mail Access Protocol，因特网邮件访问协议）
1) 比POP3复杂很多
2) 提供了文件夹用来分类这些邮件
3) 允许只读一部分的邮件（一般用于多媒体的大文件读取）

v. 基于Web的电子邮件
1) 90年代中期由Hotmail引入
2) 这时的用户代理就是普通浏览器
3) 邮件使用HTTP传输
4) 但是邮件服务器之间的传输还是通过SMTP进行的

h. DNS
i. DNS是
1) 一个由分层的DNS服务器实现的分布式数据库
2) 一种应用层协议，用来完成查询的
a) 和一般的Web应用不同，DNS通常是被其他应用层协议使用的

ii. 硬件信息
1) DNS服务器通常是运行BIND软件的UNIX机器
2) 使用UDP
3) 使用53号端口

iii. 在浏览器上输入URL时
1) 浏览器抽取主机的名字发给DNS应用的客户端
2) DNS的客户端向DNS服务器发送请求
3) 客户收到回答报文

iv. 时延
1) 有时使用域名系统带来的时延比较大
2) 但是想要的IP地址通常缓存在客户附件的DNS服务器上

v. 主机别名（host aliasing）
1) 一台主机可能有几个别名
2) 与之相对的是规范主机名（canonical hostname）

vi. 邮件服务器别名（mail server aliasing）
1) 电子邮件应用程序也可以使用DNS来解析主机别名，从而获得其规范名和IP地址
2) MX记录允许公司的邮件服务器和Web服务器使用相同的主机名，如都叫enterprise.com

vii. 负载分配（load distribution）

1) 有的站点需要比较多的服务器支持，这些服务器的内容差不多，所以这些有着不同IP地址的服务器就与同一个规范主机名对应了
2) 多个邮件服务器也可以具有相同的别名

viii. 工作机理

1) 分布式、层次数据库
   a) 三层DNS服务器
      i) 根DNS服务器：用来解析每一个顶级域名（TLD）是什么
      ii) TLD DNS服务器：每一个TLD都有自己的服务器
         One. Educause公司维护edu TLD的TLD服务器
         Two. Verisign公司维护 .com 的TLD服务器
      iii) 权威DNS服务器
         One. 每一个因特网上可以公共访问的主机都有自己的DNS记录
         Two. 可以自己维护DNS服务器或购买服务提供商的服务
   b) 本地DNS服务器
      i) 与主机一般间隔几个路由器内，可能在同一个LAN中
      ii) 每个ISP都有一台本地DNS服务器，即默认名字服务器
      iii) 当主机把DNS请求发往本地DNS服务器时，本地DNS服务器将请求转发给DNS服务器层次结构中
   c) 查询过程
      i) 方法1：请求主机把请求发给本地DNS服务器，然后本地DNS服务器分别发起三次查询：4*2个报文
      ii) 方法2：主机请求本地DNS服务器，本地请求根DNS，根DNS获得报文后请求TLD，TLD请求权威DNS服务器，然后再将报文转发回来：5*2个报文
2) DNS缓存
   a) 本地DNS服务器会缓存查询结果
   b) 由于主机名和IP的映射，服务器在一段时间后（一般是2天）会丢弃缓存的信息
   c) 所以大多数对于根服务器的查询都避免了，那些edu、com之类的根本不用查
3) DNS资源记录（resource record，RR）和报文
   a) 格式：（Name，Value，Type，TTL）
   b) 不同Type：
      i) A
         One. 例子：（relay.foo.com, 145.11.33.33, A, TTL ）
         Two. 是名字到IP的标准映射
      ii) NS（name server）
         One. 例子：(foo.com, dns.foo.com, NS, TTL ）
         Two. 获取域的权威DNS服务器
      iii) CNAME（alias to canonical name）
         One. 例子：(foo.com, relay1.bar.com, CNAME, TTL)
         Two. 查询主机名对应的规范主机名
      iv) MX
         One. 例子：(foo.com, mail.bar. foo. com, MX, TTL)
         Two. 查询别名为Name的邮件服务器的规范主机名
   c) 一台非权威DNS服务器会有一个NS记录，比如 dns.foo.com， 然后还会有这个权威服务器的A记录，记录这个 dns.foo.com的IP地址
   d) 在DNS数据库中插入记录
      i) 当你向ICANN（Internet corporation for assigned name and numbers，因特网名字和地址分配机构）注册登记后，你的域名和IP的关联、你的权威DNS服务器的名字和IP地址，就会被注册机构插入 到所有的TLD 服务器中

4) DNS的安全性
    a) 对根服务器的DDoS（分布式拒绝服务）攻击：几乎无效，因为本地DNS服务器缓存了顶级域名服务器的IP地址，而且服务器配置的分组过滤器过滤了相当一部分的ICMP ping报文
    b) 对TLD 服务器的攻击：比较难以绕过，但是DNS缓存还是能缓解一部分的攻击
    c) 中间人攻击：攻击者截获来自主机的请求并且返回伪造的回答。这也很困难，因为这要求截获分组或遏制住服务器
    d) 目前，还没有任何攻击成功妨碍了DNS服务

i. P2P文件分发
    i. P2P结构 是scalable
    ii. churn：peer may come and go
    iii. BitTorrent
        1) torrent（洪流）指参与某个文件分发的所有对等方集合
        2) 一个大文件被分成很多文件块（chunk），一般256KB
        3) 一个torrent中有一个追踪器（tracker），当一个peer加入这个torrent中时，它就到tracker那里注册一下，然后周期性地告诉tracker自己还在洪流中
        4) 过程：
            a) 假设Alice加入一个torrent，tracker发给Alice一个有50个peer的list，Alice从中选择几个作为她的邻近对等方
            b) 通过TCP连接，Alice周期性地咨询她的邻近对等方有哪些chunk
            c) rarest first（最稀缺优先）策略
                i) 对于自己没有的chunk，Alice每次都优先请求邻居中最少人有的那个
            d) tit-for-tat（一报还一报）
                i) 作用：选择要响应哪个请求
                ii) 过程：
                    One. 持续监视流量流入速度，找到4个给她传输比特速度最快的邻居，作为疏通（unchoked）
                    Two. 每过30秒，都随机选择另一个邻居，向其发送块
                    Three. 如果Alice发送块的速度足够高，那么她可能成为BOB的4个上载者之一
                    Four. 如果双方都满足成为对方疏通的条件，那么就将对方加入前4名的列表中
                    Five. Alice不会响应除了这5个人之外的其他人的需求
    iv. DHT（分布式散列表）也是一种P2P应用
j. HTTP流和DASH（dynamic adaptive streaming over HTTP，经HTTP的动态适应性流）
    i. 普通的HTTP视频流不能根据客户的网络状况做调整，但是DASH可以
    ii. DASH：
        1) 客户先请求一个manifest file（告示文件），得知每一个版本的比特率及其URL
        2) 然后客户根据自身情况选择一个版本并且每发一次GET报文就获得一个几秒钟的视频文件
k. CDN（content distribution network，内容分发网）
    i. 在不同的地理位置上存储自己的内容拷贝，每一次用户请求文件时都重定向到相应的位置上
    ii. 专用的（private）CDN：YouTube
    iii. 第三方（third-party）CDN：如Akamai：把这些分布在世界各地的服务器租给这些内容提供商以建立他们的CDN
    iv. CDN服务器安置原则
        1) 深入（enter-deep）
            a) 高度分布式，在很多接入ISP中部署
            b) 好处：离用户近，快
            c) 问题：难以维护管理
        2) 邀请做客
            a) 有几个比较大的服务器集群，放在了IXP这里

b) 慢，但是管理维护费用低

v. CDN操作

1) 用户请求一个网址，LDNS（local DNS）然后就从authoritative dns那里得到了CDN的主机名（而非IP地址）
2) LDNS然后从CDN的authoritative DNS那里得到了CDN的IP地址，然后发给了用户
3) 用户使用这个IP地址请求文件（可能是DASH文件）

vi. 集群选择策略（cluster selection strategy）

1) 找地理上最近的（geographically closest）
   a) CDN收到了LDNS的请求后，找一个和LDNS地理上最近的集群，把这个集群的主机名发给LDNS
   b) 性能：一般不错，但是有时
      i) LDNS和用户可能相距甚远
      ii) 地理最近的集群和用户间的网络状况其实很糟糕
2) 改进：CDN对集群和LDNS之间的时延、丢包情况进行周期性地实时测量（real-time measurement）
   a) 问题：很多LDNS被配置为不会响应这些探测的报文

3. Transport lawyer（运输层）

a. 概述

i. RFC也将TCP的运输层分组称为报文段，而将UDP分组称为数据报（datagram）
ii. IP的服务模型是 尽力而为交付服务（best-effort delivery service）

b. 多路复用和多路

i. multiplexing：将sender的需求发给IP
ii. demultiplexing：将包发给合适socket
iii. 所有的（四个层面上的）包都叫packet
iv. 端口号有16比特，0~1023是周知端口号（well-known port number），用于周知应用程序
v. 可以用nmap扫描因特网中任何地方的主机
vi. network lawyer是host层次上的逻辑交流，而运输层是进程间的

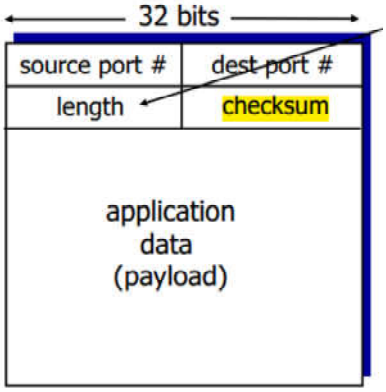
c. UDP

i. 概述

1) TCP中一个socket（即一个connection）由（source IP，source port，dest IP，dest port）决定，所以一个端口可以有多个socket，因为它们的source port可以不同
2) 除了复用/分解以及少量的差错检测以外，UDP几乎没有对IP增加别的东西，如果程序员选择UDP，几乎就是在和IP打交道
3) 优点
   a) 选择UDP的应用可以把更多的决定权留给自己，更好地确定发送时间和发送内容
   b) 不用建立连接，没有多余的内容和延时
   c) 无连接状态，所以不用跟踪各种参数，也不用维护连接状态，从而运行在UDP上的程序一般能支持更多的活跃用户

ii. 格式

1)



UDP segment format

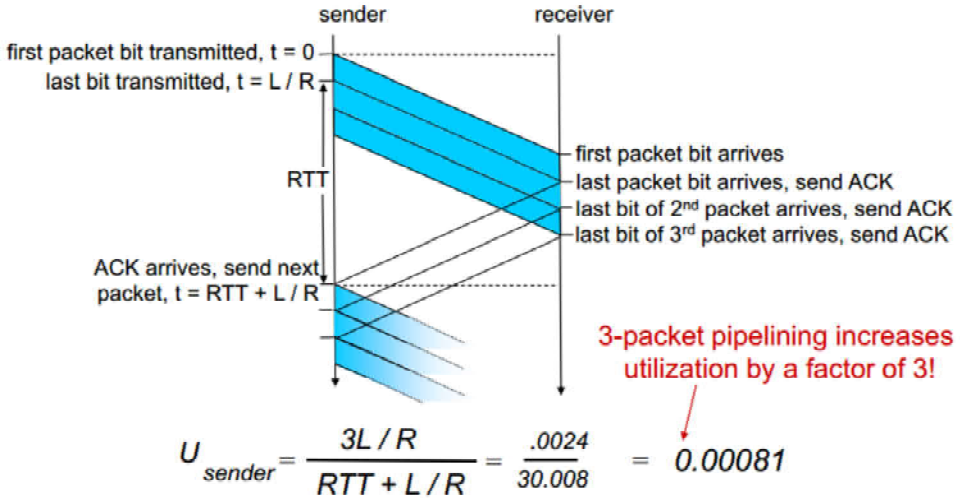
2) 其中，length以bytes作为单位，并且包括了header
3) checksum
   a) 求和、反向进位、取反

b)

Note: when adding numbers, a carryout from the most significant bit needs to be added to the result

   c) 端到端原则（end-end principle）：如果某种功能必须基于端到端实现，则应该尽量在高级层面上实现

d. 可靠数据传输（rdt）原理
   i. 概论
      1) 名词
         a) sequence number：序号
         b) unidirectional data transfer :单向数据传输
         c) rdt是通过增强利用不可靠的传输实现的
         d) acknowledgements (ACKs)
         e) negative acknowledgements (NAKs)
         f) in-flight pkt：还在传输的包
   ii. pipeline
      1) 基本结构



2) utilization=忙的时间/总时间
    a) 一般没有流水线的情况下是：
    b) 有流水线的话就直接翻n倍了

iii. 差错恢复的解决方法

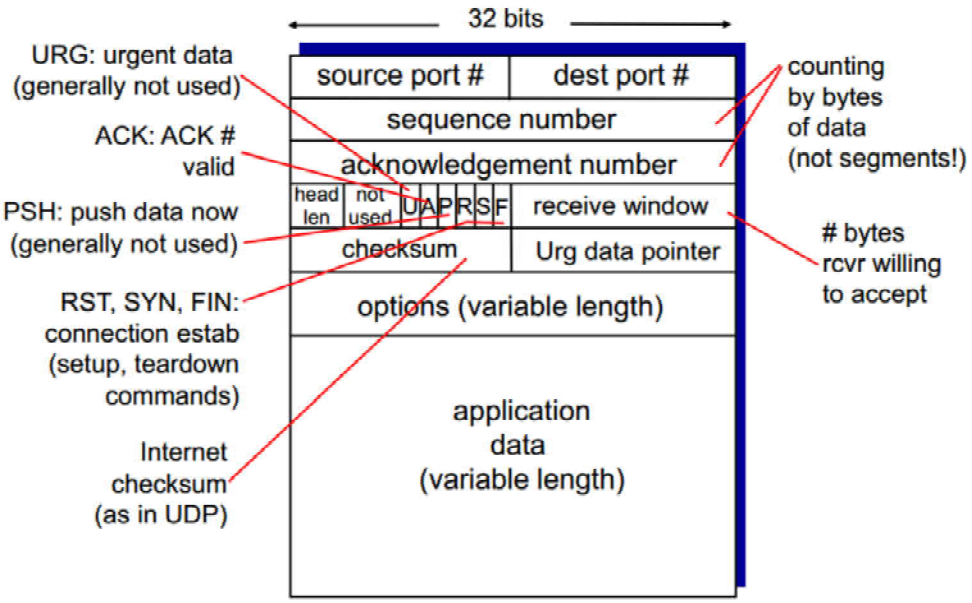
1) go-Back-N (GBN)
    a) No buffer
    b) cumulative ACK：如果发送ACK1，表示1以及1之前的包都已经收到了
    c) receiver对于传过来错序的包丢弃并且发最高有序的序列号
    d) sender对最长等待时间的包设置时间，一旦达到就重传所有unack的包
2) selective repeat (SR)
    a) have buffer
    b) individual ACK：仅仅表示收到了当前这个包
        i. 如果收到之前收到的包，还是要发一个ACK，因为可能sender没收到之前的这个ACK
    c) sender对每一个包都设置timer
    d) 要小心sender发的pkt0被收下时，ack0丢了，然后sender重发pkt0，receiver以为是新一轮的pkt0
        i) sender的窗口编号要大一些，至少得是窗口大小的两倍
3) 如果ACK的数字是错的，但是整个pkt是对的，我们不会处理这种情况

e. TCP初步

i. 特点

1) 流水线
2) cumulative acks
3) 只有一个timer
4) seq # 希望报文中第一个报文的序号
5) ACK#是希望对方发的报文的第一个序号
6) 在一个连接中双向传输信息
7) MSS: maximum segment size
    a) 包括首部的报文段长度 叫做maximum transmission unit（最大传输单元，MTU），通常是1500bytes
    b) M 指报文中的payload，一般TCP的headers大小是40bytes，所以MSS一般是1460bytes
8) TCP没有规定收到乱序的segment该怎么办，取决于实现（比如receiver有没有buffer）

ii. TCP报文格式



iii. RTT(round trip time)

1) SampleRTT: 计算上一个 ACK 花了多久被接收
   a) 忽略重传
2) 使用最近几次RTT的均值
   a) 估计RTT
      i) EstimatedRTT = (1- α)*EstimatedRTT + α*SampleRTT
      ii) typical value: α = 0.125
   b) deviation（偏离）
      i) DevRTT = (1-β)*DevRTT + β*|SampleRTT-EstimatedRTT|
      ii) 通常β为0.25
   c) 总和
      i) TimeoutInterval = EstimatedRTT + 4*DevRTT

iv. 产生TCP ACK的建议

1) 如果收到segment并且之前没有未ACK的seq，则延迟ACK，并且更新expected seq
2) 如果再次收到expected seq，立即发出这次的cumulative ACK，并且更新expected seq
3) 如果收到乱序的segment，立即发出duplicate ACK，告知expected seq

v. TCP fast retransmit

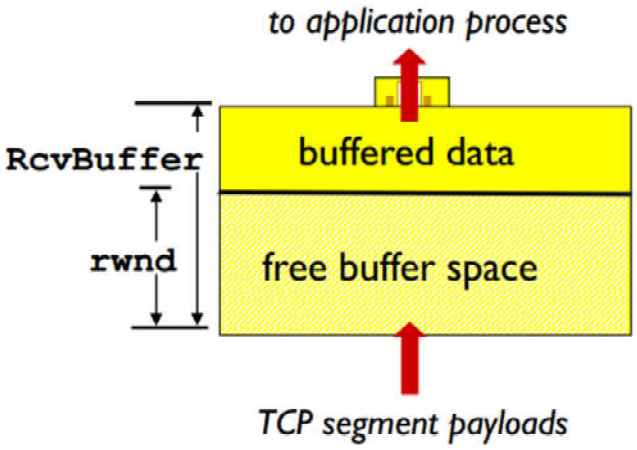
1) if sender receives 3 ACKs for same data, resend unacked segment with smallest seq #
2) likely that unacked segment lost, so don’ t wait for timeout

4. transport lawyer 3

a. 接收者的rwnd要和发送者的差不多大，这个rwnd信息会发送到报文header中

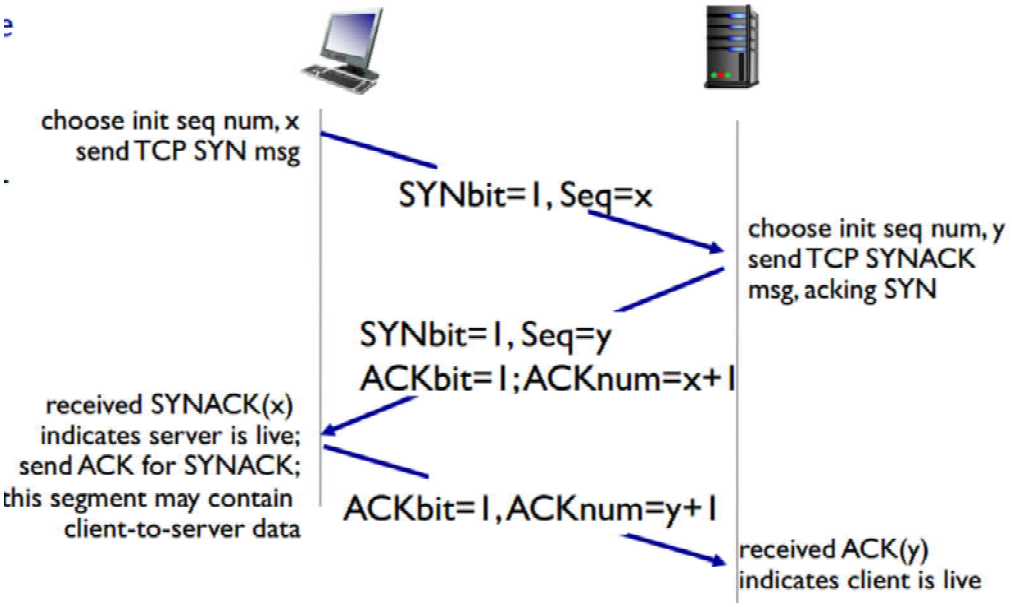
b. RcvBuffer一般4096 bytes大小

i.



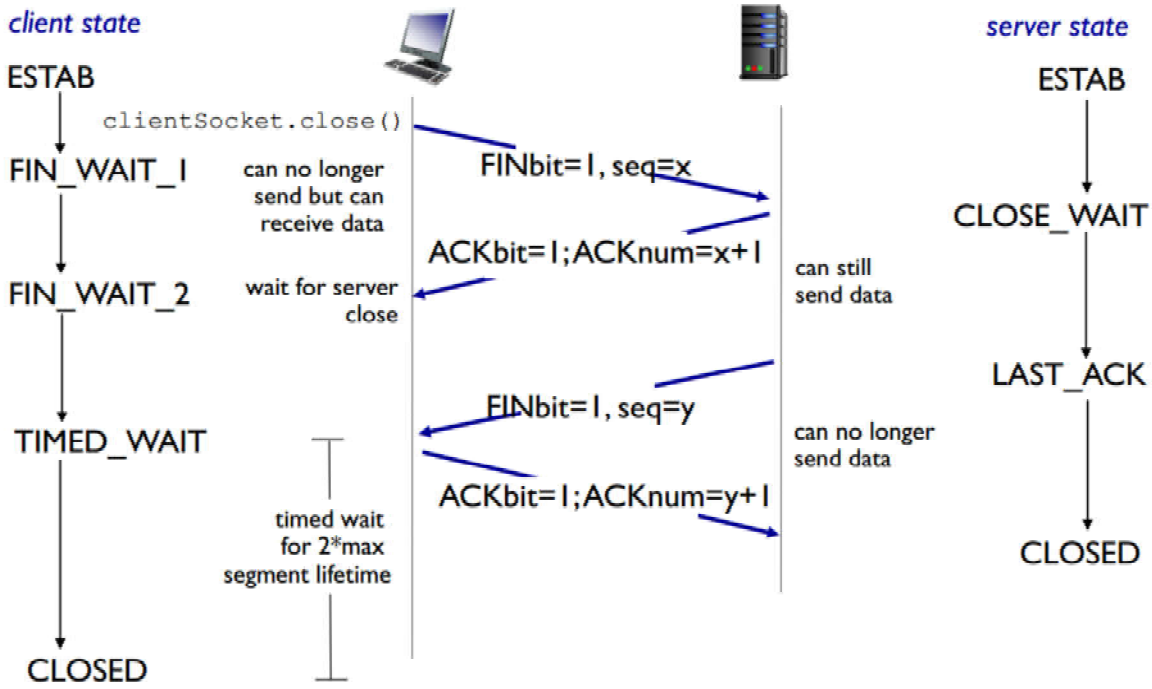
c. 握手的过程

i.



d. 分手的过程

i.



ii. 在收到FIN时，自己在发送的ACK时也可以加入FIN，表示自己也结束了

e. congestion control

i. 和flow control 不同

ii. 指网络无法承载这么多的data