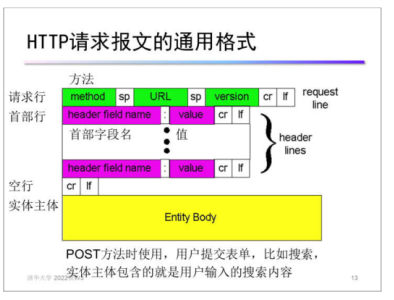
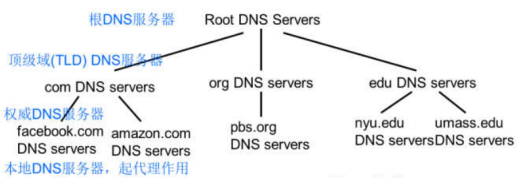
应用层

- 应用程序体系结构
  - client-server
  - p2p
- 进程通信 — 套接字 — IP+端口寻找进程
- 应用层协议
  - HTTP
    - URL，80端口，TCP
    - 带内传输：控制和数据用同一个连接
    - 无状态协议
    - 非持续连接/持续连接 — 响应时间计算 =
    - 报文格式



    - 条件GET：cache中有对象的最新版本就不从服务器发对象，而从缓存里找。该方法存在于缓存器和初始服务器之间。
  - SMTP
    - 邮件报文协议，面向邮件服务器之间
    - 端口25，TCP
    - 流程：握手、报文传送、断开连接
    - 和HTTP对比：HTTP拉，SMTP推；HTTP可以非ASCII，SMTP必须ASCII；
    - 邮件访问协议：从邮件服务器拉邮件：POP3\IMAP\HTTP
  - FTP
    - server：控制端口21，数据连接端口20
    - 带外传输：控制和传输使用不同的TCP
- DNS系统
  - DNS服务：主机名到ip的翻译



  - 访问流程
    - 递归查询
    - 迭代查询
    - 实际上本地的是递归的（保姆型的），而向外的是迭代的。
  - 存储RR
    - 格式：name, value, type, TTL

DNS: 储存资源记录(RR)的分布式数据库

资源记录格式: (Name, Value, Type, TTL)

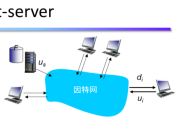
- Type=A
  - Name 为主机名
  - Value 为IP 地址
- Type=NS
  - Name 为域 (e.g. foo.com)
  - Value 为该域的权威DNS服务器的主机名
- Type=CNAME
  - Name 为某些“规范”名字的别名 www.sohu.com 实际上是 fcer.a.sohu.com
  - Value 为规范名字
- Type=MX
  - Name 为别名
  - Value 是别名为name 的邮件服务器的规范主机名

可以与Web服务器具有相同别名

- P2P
  - 文件分发时间

文件分发时间: client-server

- 服务器传输: 必须顺序地发送 (上传) N 份文件拷贝:
  - 发送一份拷贝的时间: F/u_s
  - 发送N 份拷贝的时间: NF/u_s
- 客户机: 每台客户机都必须下载文件拷贝
  - d_{min} = 最慢客户机的下载速率
  - 最慢客户机的下载时间: F/d_{min}

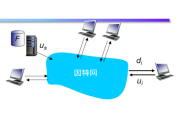


使用CS结构的分发时间 $D_{c-s} \geq \max\{NF/u_s, F/d_{min}\}$

随N线性增加

文件分发时间: P2P

- 服务器传输: 必须上传至少一份拷贝
  - 发送一份拷贝的时间: F/u_s
- 客户机: 每台客户机必须下载一份文件拷贝
  - 最慢客户机的下载时间: F/d_{min}
- 客户机: 总体上必须下载 NF bits
  - 最大上传速率 (限制最大下载速率) 为 $u_s + \Sigma u_i$



使用P2P结构的分发时间 $D_{P2P} \geq \max\left\{F/u_s, F/d_{min}, NF/(u_s + \sum_{i=1}^{N} u_i)\right\}$

随N线性增加…

… 但最大上传速率也随N线性增加，因为每个对等方也带来服务能力

  - 集中式目录 — 版权问题、单点故障、性能瓶颈
  - 洪泛查询 — 完全分布式、可供共享的文件
- DASH：经HTTP的动态适应性流 — 将视频切分、不同比特率的存储、告示文件
- 内容分发网
  - 单一大规模数据中心
  - 分布式站点存储CDN