计算机网络

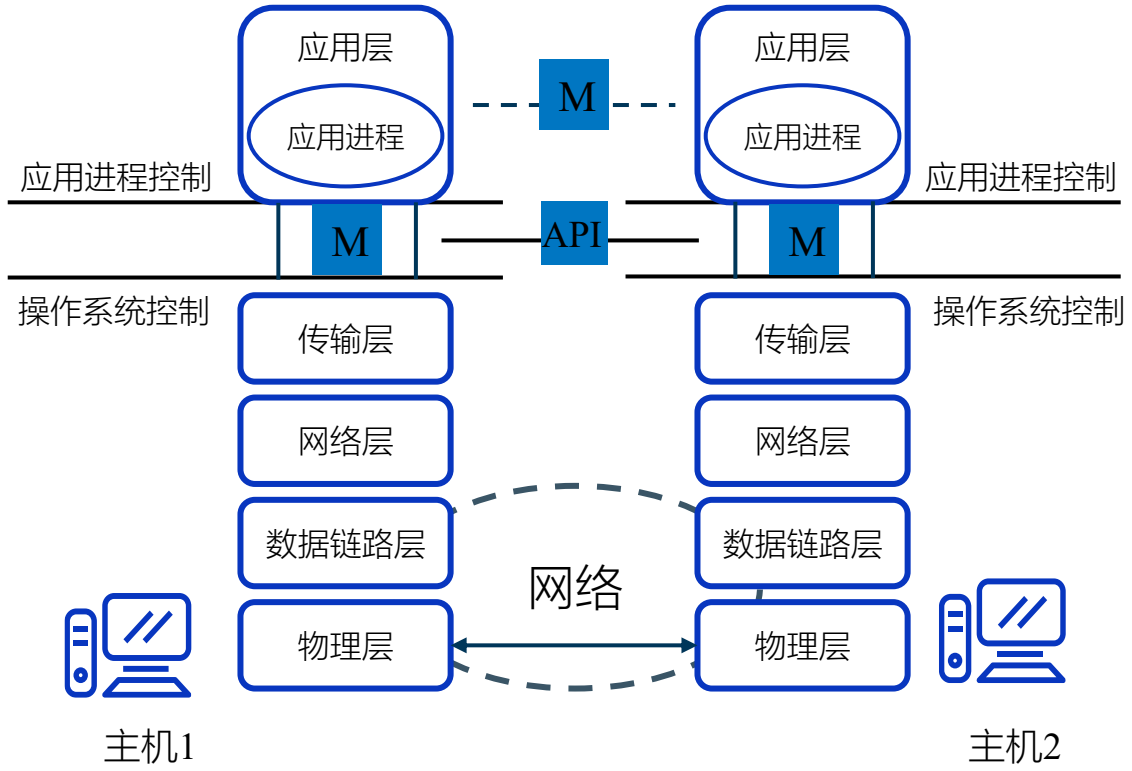
胡亮 等编著

第7章 应用层

7.1 应用层概述
7.2 DNS域名系统
7.3 电子邮件服务SMTP
7.4 FTP文件传输
7.5 Web 服务
7.6 网络流媒体应用
7.7 CDN内容分发网络及 P2P 应用
7.8 本章总结

7.1 应用层概述

- 网络应用程序都要通过位于不同主机上的应用进程之间的通信和协议共同来完成其业务功能
- 应用层就是为处于不同主机上的应用进程之间的通信提供相应的服务
  - 为解决某一类具体应用问题，规定应用进程在通信时所要遵循的协议或相关约定
  - 在TCP/IP的分层模型中，应用层在传输层之上，是最高层，再往上是某个具体应用

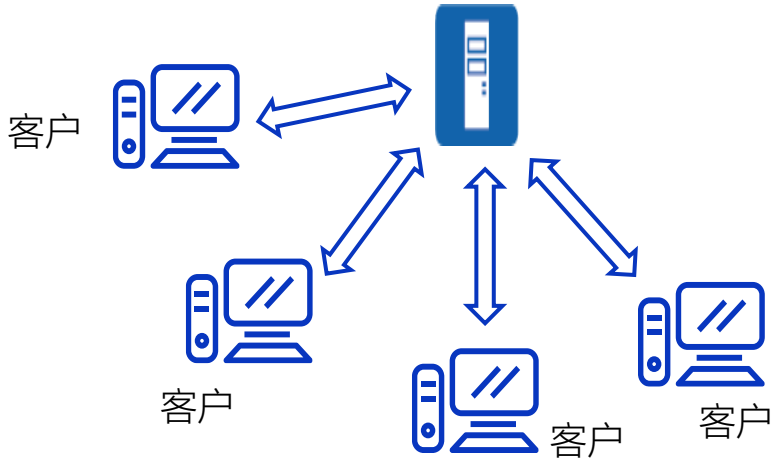


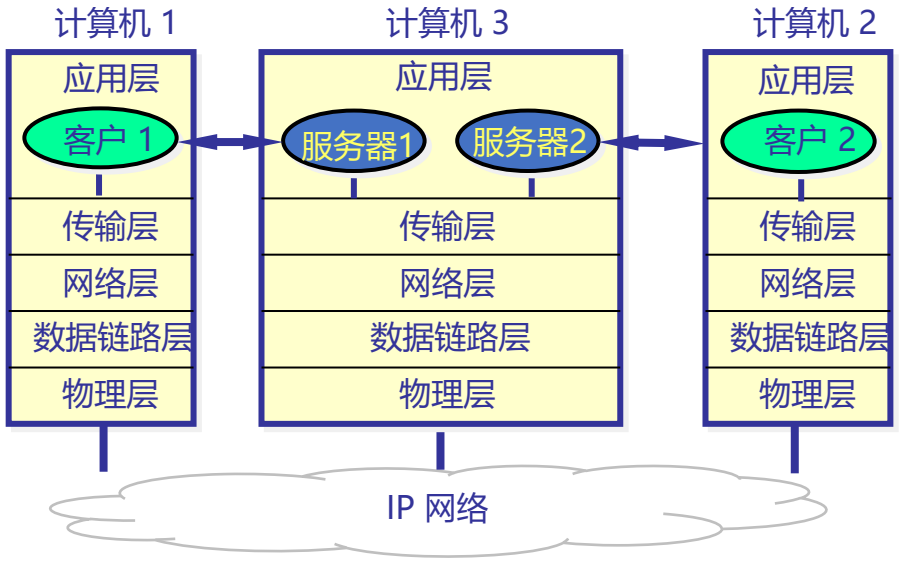
应用进程间通信方式分类

- 位于不同主机上的进程间的通信有三种通信方式：
  - C/S：Client/Server，客户/服务器方式
  - B/S：Browser/Server，浏览器/服务器方式
  - P2P：Peer to Peer，对等方式

应用进程间的通信方式：C/S 方式架构

- C/S方式，即客户/服务器（Client/Server）模式，是一种主从式架构
  - 服务器是主，客户端是从。服务器软件以多进程方式运行，可以同时为多个客户服务
  - 在移动互联网环境下，每个应用APP都是一个客户
  - C/S 是一个可缩放的架构
    - 服务器软件通常运行在性能强大的专用计算机上，承担复杂的业务逻辑处理、数据计算等
    - 客户端则一般运行于普通个人计算机上，为用户提供操作与显示等





应用进程间的通信方式：C/S 方式

- 应用层的许多协议是基于C/S方式，如传统的 Internet 的文件传输、电子邮件、远程登录等
  - 客户(client)和服务器(server)是指通信中所涉及的2个应用进程
  - 客户/服务器方式描述的是应用进程之间服务和被服务的关系
  - 客户是服务请求方（主动请求服务，被服务）
  - 服务器是服务提供方（被动接受服务请求，提供服务）
- C/S方式可以是面向连接的，也可以是无连接的
  - 面向连接时，C/S通信关系一旦建立，通信就是双向的，双方地位平等，都可发送和接收数据

应用进程间的通信方式：C/S 方式特点

- 客户端进程的特点
  - 在进行通信时临时成为客户，它也可在本地进行其它的计算
  - 用户计算机上运行，在打算通信时主动向远地服务器发起通信
  - 客户方必须知道服务器进程所在主机的IP地址才能发出服务请求
  - 需要时可以与多个服务器进行通信

应用进程间的通信方式：C/S 方式特点

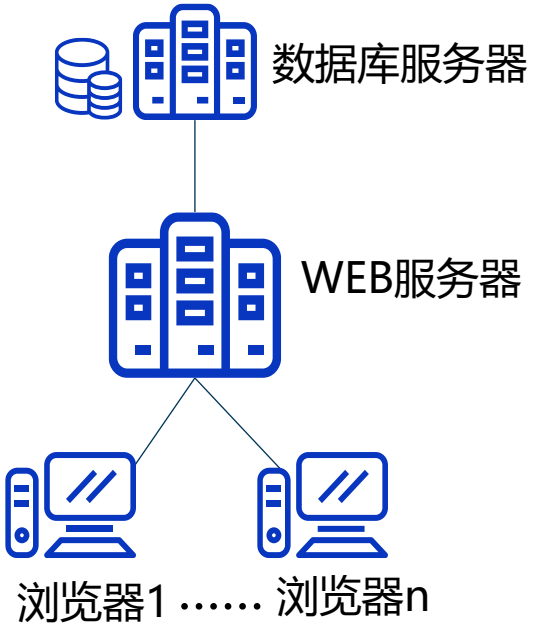
- 服务器进程的特点
  - 专门用来提供某种服务的程序，可“同时”处理多个远地或本地客户的请求
  - 必须始终处于运行状态才有可能提供服务
  - 通信开始之前服务器进程不需要知道客户进程所在主机的IP地址，无论客户请求来自哪里，服务器进程被动等待服务请求的到来即可
  - 当系统启动时即自动调用并一直运行着，也可以由用户或其它进程在通信前启动
  - 被动等待并接受来自多个客户的通信请求

应用进程间的通信方式：B/S方式

- B/S方式也就是浏览器/服务器（Browser/Server）模式，可看做是C/S方式的特例，即客户端软件改为浏览器罢了
- B/S方式采取浏览器请求、服务器响应的工作模式
- 在B/S方式下，用户界面通过Web浏览器实现，一部分简单的事务逻辑可以在客户端实现，但主要的、复杂的事务逻辑在服务器端实现

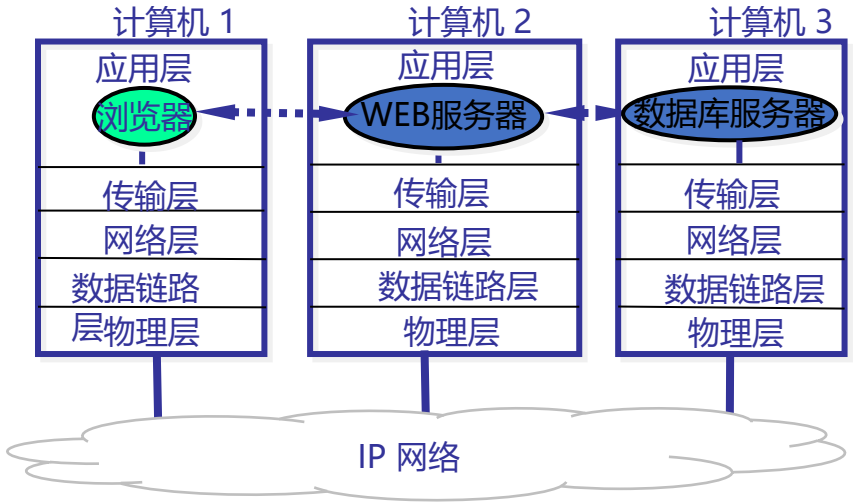
应用进程间的通信方式： B/S方式的架构

- B/S方式通常采取3层架构实现
  - 数据层：由数据库服务器承担数据处理逻辑，其任务是接受Web服务器对数据库服务器提出的数据操作请求，然后由数据库服务器进行数据处理并把处理结果返回给web服务器
  - 处理层：由Web服务器承担业务处理逻辑和页面存储管理，接受客户浏览器的任务请求，执行相应的事务处理
  - 展现层：浏览器仅承担网页信息的浏览功能， 以超文本格式实现信息的输入和浏览
- 实际部署时也可以把数据库服务器和web服务器部署在同一台设备上

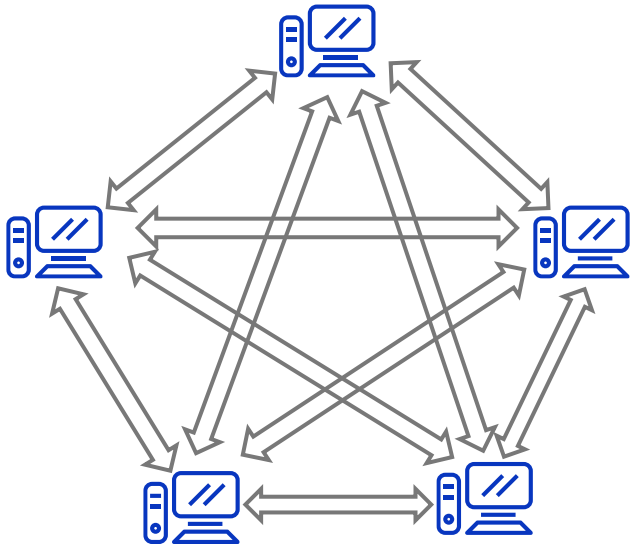


应用进程间的通信方式： B/S方式特点

- B/S方式的特点：
  - 界面统一，使用简单。客户端只需要安装浏览器软件
  - 易于维护。对应用系统升级时，只需更新服务器端的软件，方便了系统维护和升级
  - 可扩展性好。采用标准的TCP/IP和HTTP协议，具有良好的扩展性
  - 信息共享度高。HTML是数据格式的一个开放标准，目前大多数软件均支持HTML
  - 需要注意的是，在一种浏览器环境下开发的界面在另一种浏览器环境下可能有不完全适配的情况，这时需要安装对应的浏览器



应用进程间的通信方式：P2P方式



P2P 模型

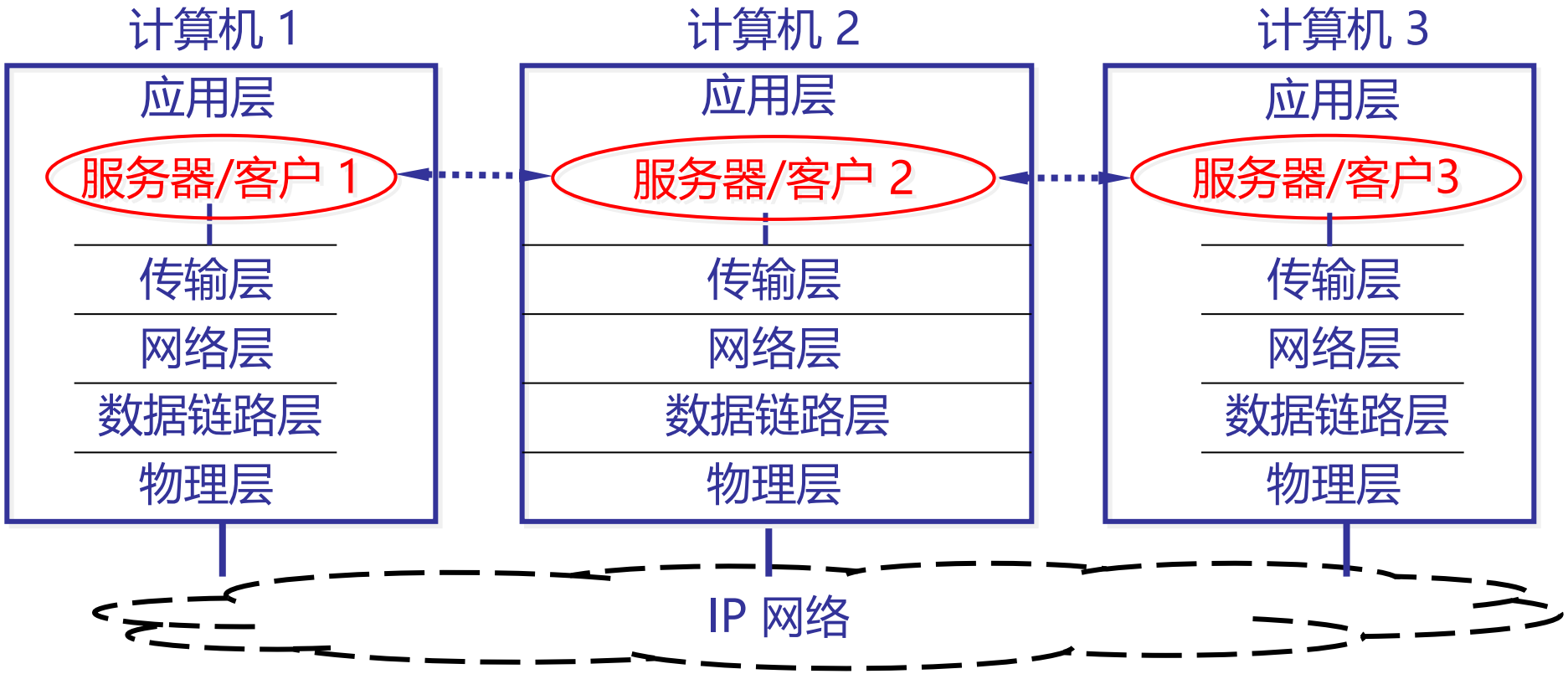
- P2P是Peer to Peer对等方式，属于分布式模型
- 每个主机既可以提供服务，也可以请求服务
- 任意端系统/结点之间可以直接通信
- 结点间歇性接入网络，不能保证结点永远在线
- 结点的 IP 地址可能会发生变化

应用进程间的通信方式：P2P方式

- 处于不同主机上的两个 P2P 进程在通信时不区分服务的请求方和服务的提供方
  - 只要两个主机都运行 P2P 软件，它们就可以进行平等、对等的通信
  - 双方都可以下载对方存储在硬盘中的共享文档，如果权限允许的话
- 音频/视频应用推动了 P2P 对等通信方式的发展
- 因特网上流量，音频/视频流量已占主要比例
  - 音视频采用 P2P对等通信方式，减轻了服务器负载，也影响了网络上的流量模型，从垂直方向流量改为很大一部分是水平方向流量

应用进程间的通信方式： P2P方式

- P2P方式从本质上看仍然是使用了C/S方式，但强调的是通信过程中的对等
  - 每一个P2P进程既是客户同时也是服务器



服务器进程工作方式分类

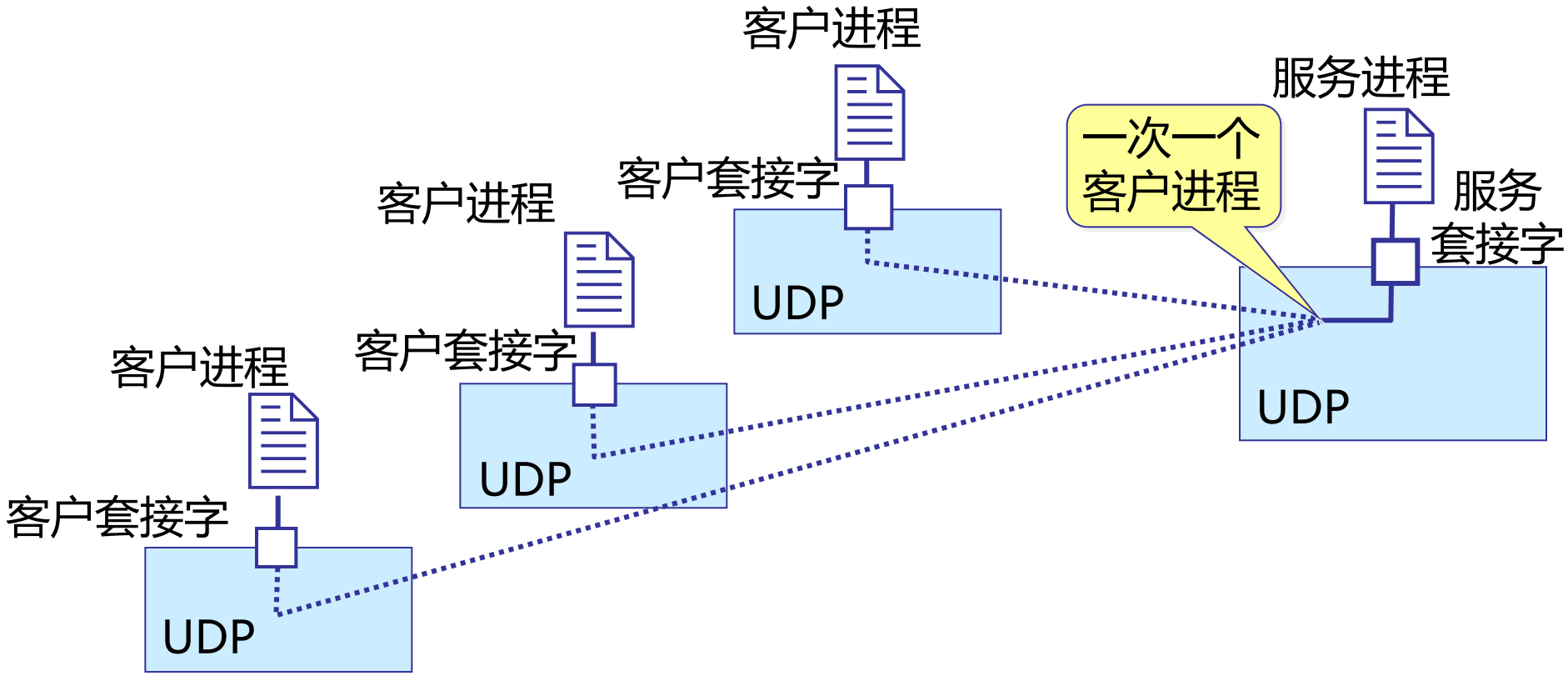
- 服务器上的进程有两种工作方式：
  - 循环方式(iterative mode)
    - 一次只运行一个服务进程
    - 当有多个客户进程请求服务时，服务进程就按请求的先后顺序依次做出响应 (阻塞方式)
  - 并发方式(concurrent mode)
    - 可以同时运行多个服务进程
    - 每一个服务进程都对某个特定的客户进程做出响应 (非阻塞方式)

服务器进程工作方式：无连接循环方式服务

- 使用无连接的UDP服务进程通常都工作在循环方式，即一个服务进程在同一时间只能向一个客户进程提供服务(顺序服务)
  - 服务进程收到客户进程的请求后，就发送UDP用户数据报响应该客户
  - 对其他客户进程发来的请求则暂时不予理睬，这些请求都在服务端的队列中排队等候服务进程的处理
  - 当服务进程处理完毕一个请求时，就从队列中读取来自下一个客户进程的请求，然后继续处理

服务器进程工作方式：无连接循环方式服务

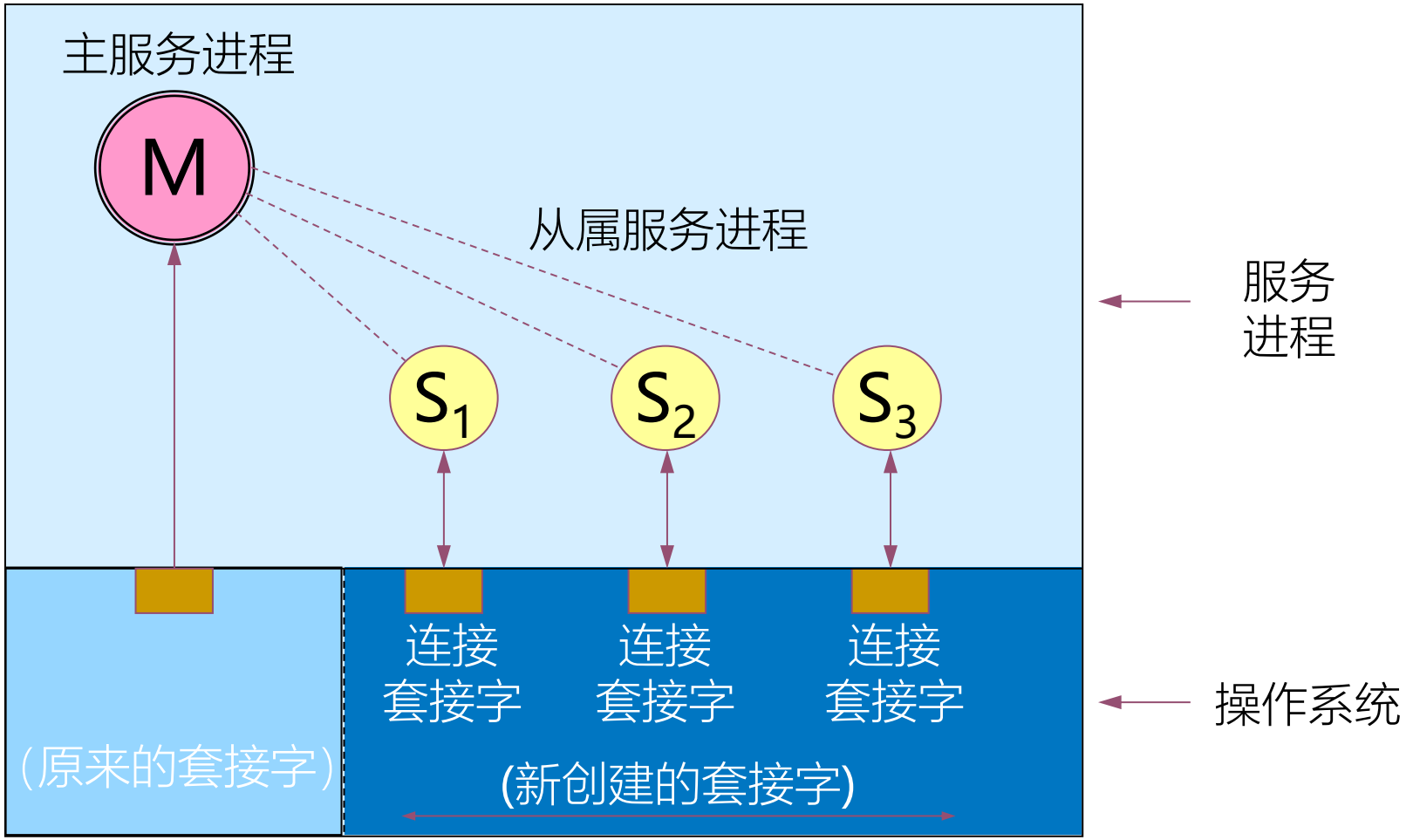
- 服务进程只使用一个服务套接字。每个客户使用自己设定端口号的客户套接字



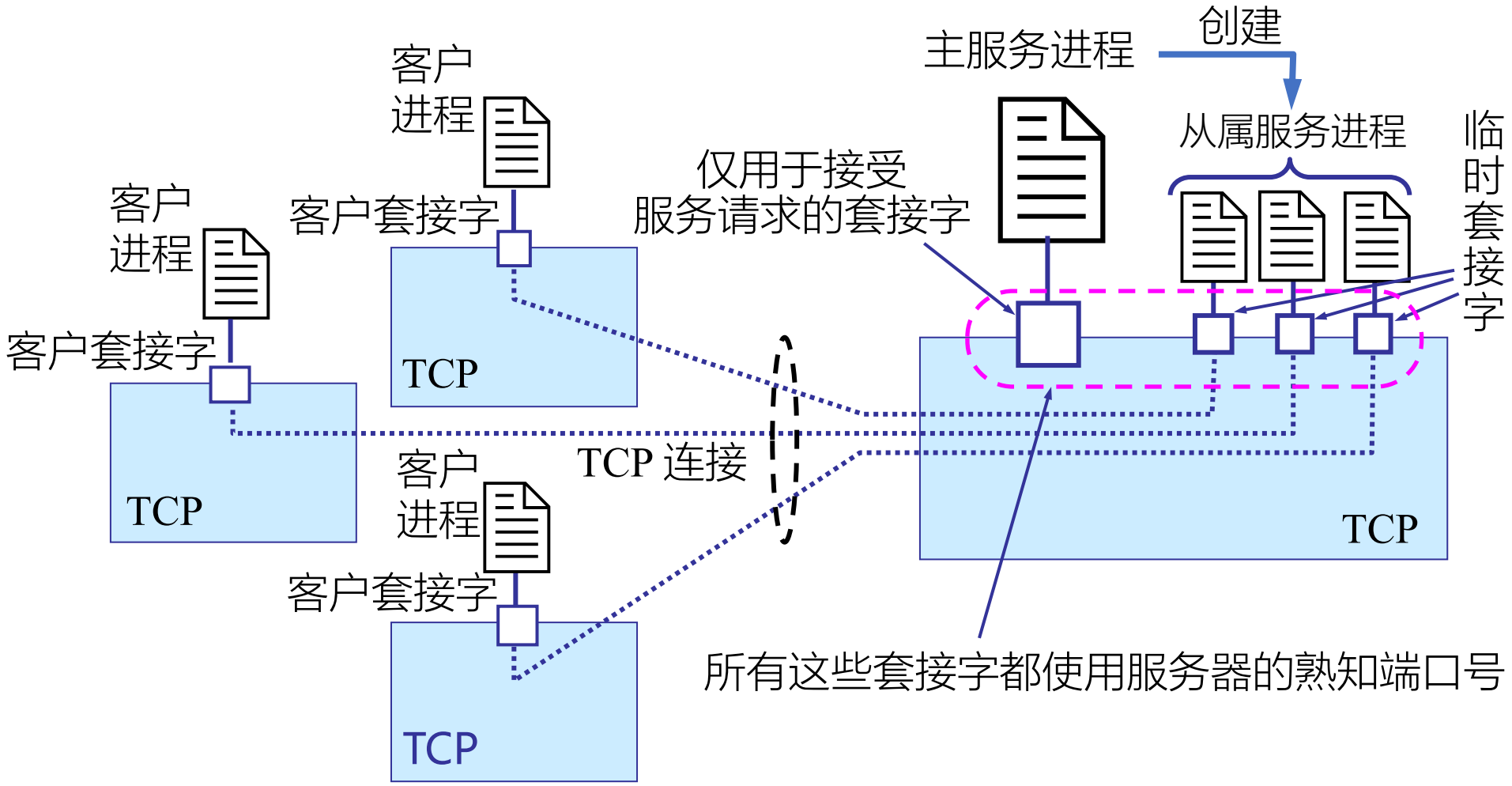
服务器进程工作方式：面向连接的并发方式服务

- 面向连接的TCP服务进程通常都工作在并发服务方式，服务进程在同一时间可同时向多个客户进程提供服务(并发服务)
- 在TCP服务进程与多个客户进程间需要建立多条TCP连接时，每条TCP连接在其数据传送完毕后释放
- 一个TCP连接对应一个（熟知）服务端口
- 主服务进程在熟知端口等待客户进程发出的请求。一旦收到客户的请求，就创建一个从属服务进程，并指明从属服务进程使用临时套接字与该客户建立TCP连接，然后主服务进程继续在原来的熟知端口等待向其他客户提供服务

服务器进程工作方式：面向连接的并发方式服务



服务器进程工作方式：面向连接的并发方式服务



主服务进程也称为父进程，从属服务进程又称为子服务进程

7.2 DNS域名系统

- 计算机之间通信是基于IP地址的，但IP地址很抽象，无语义，不易记忆
- 给计算机取一个有语义的名字，容易记忆。但通信时，仍然需要用IP地址
- DNS域名系统就是用来解决上述矛盾，即解决“将域名映射成IP地址”的问题

域名系统的历史

- ARPANET时期
  - Hosts.txt文件列出了所有计算机名称和它们的IP地址
  - 所有主机在网上从指定站点上获取Hosts.txt文件
  - Hosts.txt文件的使用在当时表现不错
- 互联网发展
  - Hosts.txt文件变得越来越大
  - 越来越难管理，需要集中管理来防止主机名冲突
- 1987年发布了域名系统的RFC文档（RFC1034、RFC1035）
  - 采用了一种层次的、基于域的命名模式，并使用分布式数据库系统实现
  - 主要用途是将主机名映射成IP地址

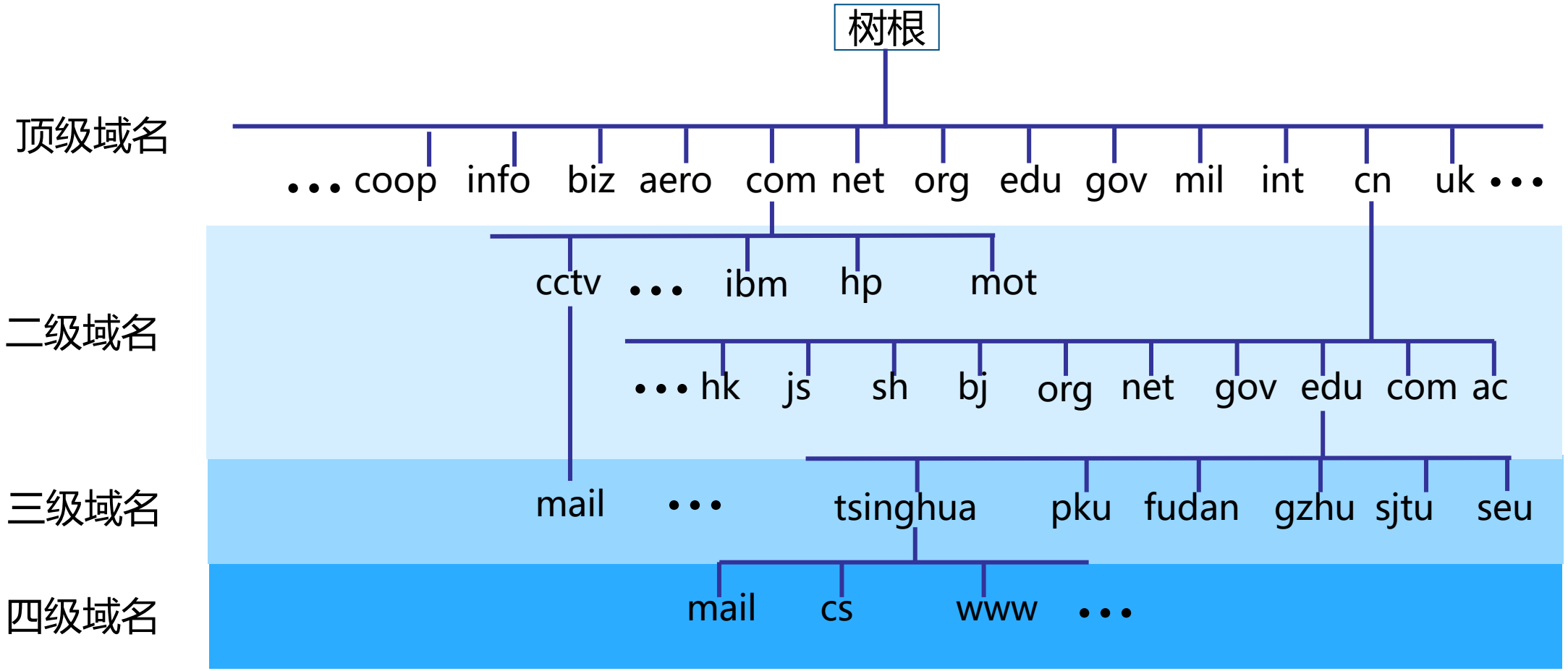
域名系统概述

- 域名系统（DNS，Domain Name System）是互联网重要的基础设施之一，向所有需要域名解析的应用提供服务，主要负责将可读性好的域名映射成IP地址

- Internet采用层次结构的命名树作为主机的名字，并使用分布式的域名系统DNS。Internet的DNS是一个联机分布式数据库系统

- 域名解析是由若干个域名服务器程序完成的。域名服务器程序在专设的结点上运行，相应的结点也称为名字服务器(Name Server)或域名服务器(Domain Name Server)

域名系统名字空间的层次化结构



- 域名系统中的名字空间采用层次化的树状结构，按级划分

域名系统名字空间的层次化结构

- Internet的域名结构采用了层次化的树状结构的命名方法：
  - 域名的结构由若干个分量组成，各分量之间用小数点(.)隔开，总长不超过255个字符
  - 各分量分别代表不同级别的域名(每个分量域名长度≤63字符)
  - 合法域名中，点“.”的个数至少为一个
  - 通常，点“.”对应的英文单词为dot，也可以读为point

… .三级域名.二级域名.顶级域名

顶级域名分类

- 顶级域名TLD（Top Level Domain）有三类：
  - 国家或地区顶级域nTLD，也记为ccTLD (cc: country code)
    - 例如.cn 表示中国，.us 表示美国，.uk 表示英国。目前有300多个
  - 基础设施域.arpa (Address and Routing Parameter Area)
    - 专用于Internet基础设施目的
    - 目前有二级域ip6.arpa；iris.arpa；in-addr.arpa；uri.arpa；urn.arpa；home.arpa；as112.arpa；in-addr-servers.arpa；ipv4only.arpa等
  - 通用顶级域gTLD
    - 早期规定了20个通用顶级域名，2011年批准新通用顶级域名(New gTLD)
    - 截至2020年，已注册有1200多个通用顶级域名

国家顶级域名

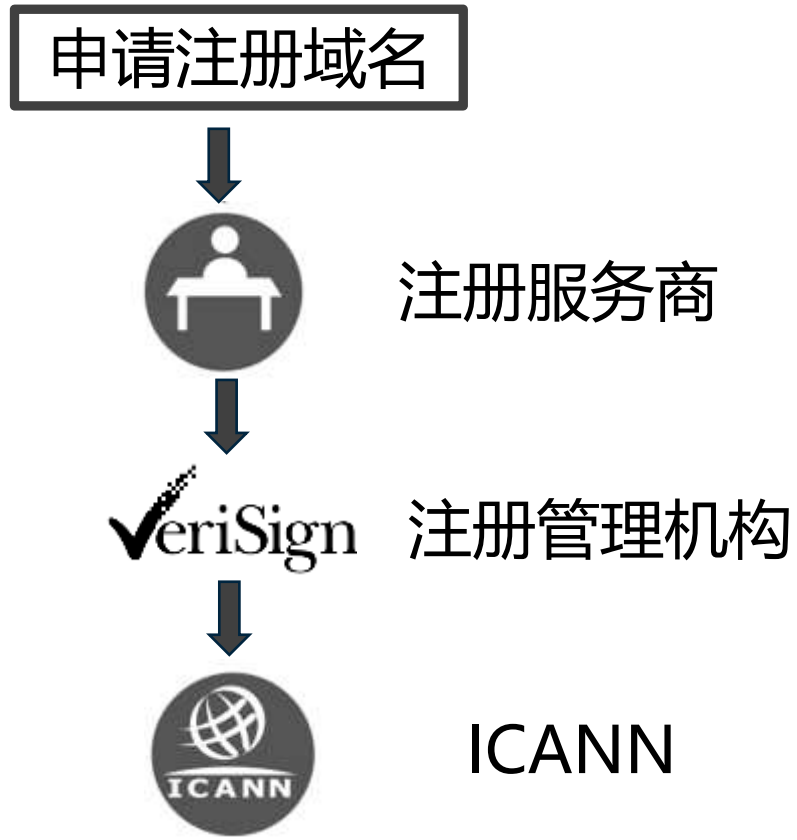
- 国家顶级域名 .cn下的二级域名分为三类：
  - 类别域名7个：
    - .edu.cn 教育
    - .gov.cn 政府
    - .org.cn 非营利组织
    - .net.cn 网络服务
    - .com.cn 工商金融等企业
    - .ac.cn 科研
    - .mil.cn 国防机构
  - 行政区域名34个：省、直辖市、自治区、特区等行政区域名，每个行政区域名为两个字母，例如：北京-bj、河北-he等
  - 无类别域名：例如 www.google.cn、www.tianya.cn等

通用顶级域名

- 早期的通用顶级域名：
  - .com 表示公司企业 (commerce)
  - .net 表示网络服务机构，例如NIC和NOC (network)
  - .org 表示非赢利性组织 (organization)
  - .edu 表示教育机构(美国专用) (education)
  - .gov 表示政府部门(美国专用) (government)
  - .mil 表示军事部门(美国专用) (military)
  - .int表示政府间国际合约建立的国际性组织 (international)
  - .mobi 用于提供移动产品和服务的用户和供应商 (mobile)
  - .aero 用于航空运输企业
  - .biz 用于商业 (business)
  - .cat 用于(西班牙)加泰罗尼亚语言和文化团体
  - .coop 用于合作团体
  - .info 适用于各种情况
  - .jobs 用于人力资源管理者
  - .museum 用于博物馆
  - .name 用于个人
  - .pro 用于有资质的专业人员及其实体
  - .tel 用于商业和个人公布其联系方式
  - .travel 用于旅游业实体
- 2012年开始，公司名可以作为新的顶级域名
- 也可以注册 .中国、.公司、.网络 等顶级域名(多语种域名国际标准RFC3454、RFC3490、RFC3491、RFC3492)

域名管理方法

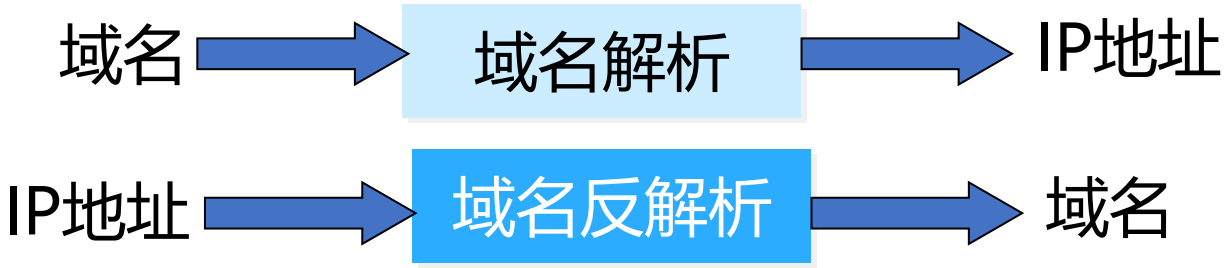
- 域名管理机构分级负责域名注册
- Internet的域名管理机构：ICANN (Internet Corporation for Assigned Names and Numbers) www.icann.org
- ccTLD下的二级域名该国自行确定
- 三级域名注册由其所属二级域名机构负责，以此类推
- .edu.cn下三级域名注册由CERNET负责
- 我国的其它二级域名注册由中国互联网络信息中心(CNNIC)负责



注册管理机构与注册服务商之间的关系

域名服务器：

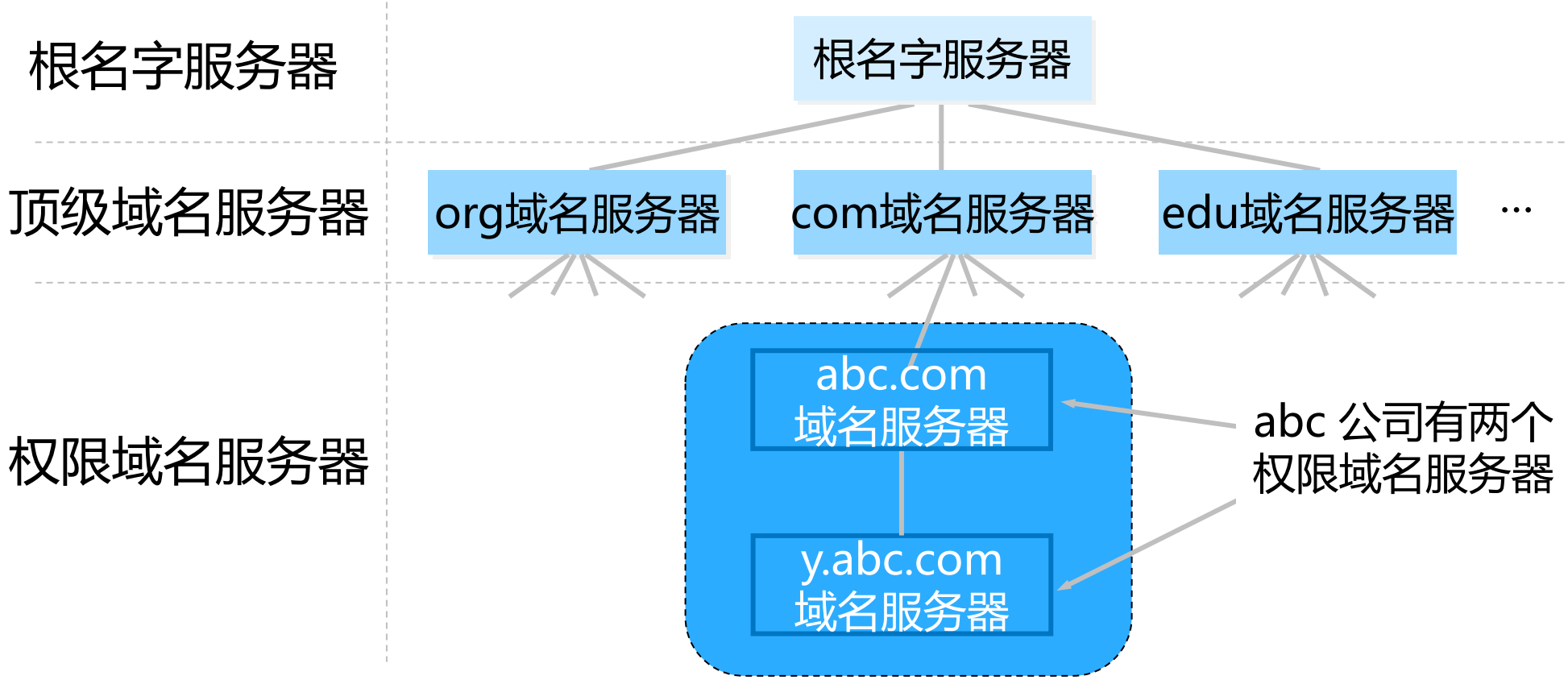
- 域名服务器用于域名解析：
  - 负责进行名字解析的服务器统称为名字服务器或域名服务器
  - 每个域名服务器必须具有连向其它有关域名服务器的信息，当自己不能完成域名与IP地址的转换时，能够知道或使其它域名服务器知道到哪里去找别的域名服务器，使域名解析过程能够完成
  - 域名与IP地址可以是一对一、一对多或者多对一的关系
  - 域名解析过程对用户透明



域名服务器：域名服务器分类

- Internet上域名服务器系统也按域名层次树状安排。每个名字服务器管辖一部分域。按管辖范围从大到小，域名系统的名字服务器可分为4类：
  - 根名字服务器(root name server) /根服务器(root server)
  - 顶级域名字服务器(TLD name server)
  - 权限域名字服务器(authoritative name server)
  - 本地域名字服务器(local name server)，也称为递归服务器/递归解析器(recursive resolver)

域名服务器：域名服务器的层次化树状结构的部署方式



域名服务器：根服务器

- 根域名服务器是最重要的域名服务器。所有的根域名服务器都知道所有的顶级域名服务器的域名和 IP 地址
- 不管是哪一个本地域名服务器，若要对因特网上任何一个域名进行解析，只要自己无法解析，就首先求助于根域名服务器
- 在因特网上共有13 个不同 IP 地址的根域名服务器（13套服务器系统）。这些根域名服务器相应的域名分别是：
  - a.rootservers.net
  - b.rootservers.net
  - …
  - m.rootservers.net

域名服务器：根服务器

- 为了方便用户，使世界上大部分 DNS 域名服务器都能就近找到一个根域名服务器，将根域名服务器在全球分布
  - 截止2023.9在全球已经有1732个根域名服务器节点了，由12个组织负责运营（注意：13套根服务器系统通过任播技术实现多节点分发，同一IP地址对应全球多个节点，提升响应速度，也增强冗余性和抗故障能力）
- 根域名服务器并不直接把域名直接转换成 IP 地址
  - 在使用迭代查询时，根域名服务器把下一步应当找的顶级域名服务器的 IP 地址告诉本地域名服务器

https://root-servers.org

域名服务器：顶级域名服务器

- 顶级域(TLD)名字服务器负责管理在该顶级域名服务器注册的所有二级域名
- 当收到DNS查询请求时就给出相应的回答（可能是最终结果，也可能是下一步应当找的域名服务器的 IP 地址）

域名服务器：权限域名服务器

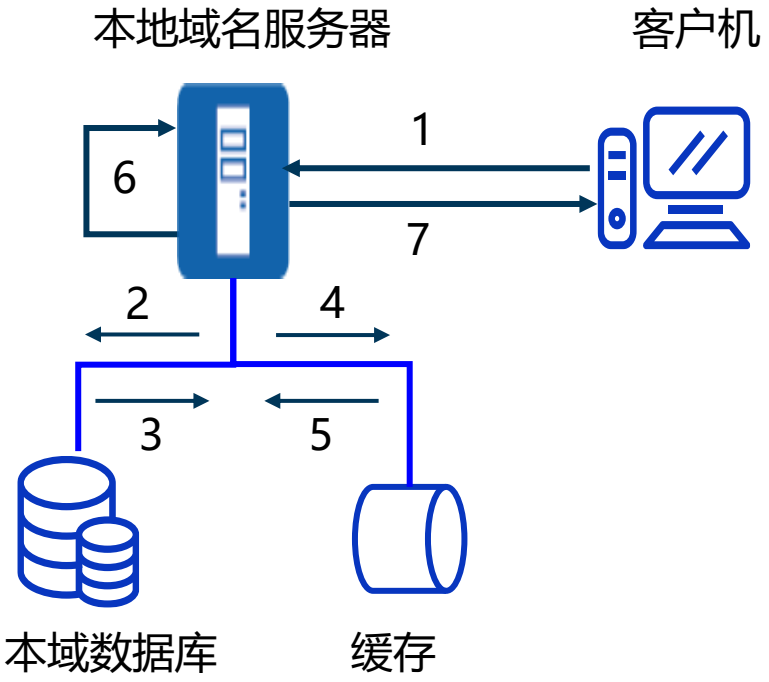
- 每个区设置相应的权限域名字服务器，用来保存该区中的所有主机的域名到IP地址的映射
- 权限域名字服务器一般只解析本辖域的域名
  - 每一个主机都必须在某个权限域名字服务器处注册登记。因此权限域名字服务器知道其管辖的主机名应当转换成什么IP地址
    - 各个单位根据自己的具体情况把本单位的域名划为若干个域名管辖区(zone)，也可简称为区
- 当一个权限域名服务器还不能给出最后的查询回答时，就会告诉发出查询请求的 DNS 客户，下一步应当找哪一个权限域名服务

域名服务器：本地域名服务器

- 每一个Internet服务提供者ISP(Internet Service Provider), 都至少有一个本地域名服务器，它也称为默认域名服务器
- 本地域名服务器离用户较近，一般不超过几个路由器的距离
- 当主机发出DNS查询报文时，这个查询报文就首先被送往该主机所在区域的本地域名字服务器
- 如果所要查询的主机也处在本地 ISP 的管辖范围，则本地域名字服务器就立即能进行域名解析，返回IP。否则就需要再以此去询问其他的域名服务器
- 实际部署中，解析请求路径上的本地域名服务器可能有多个

域名解析过程——本地解析

- 当某一应用进程需要进行域名解析时，该应用进程将域名放在DNS请求报文（UDP数据报,目的端口号为53）发给本地域名服务器（使用UDP是为了减少开销）
- 本地域名服务器查找域名后，将对应IP地址放在应答报文中返回给应用进程

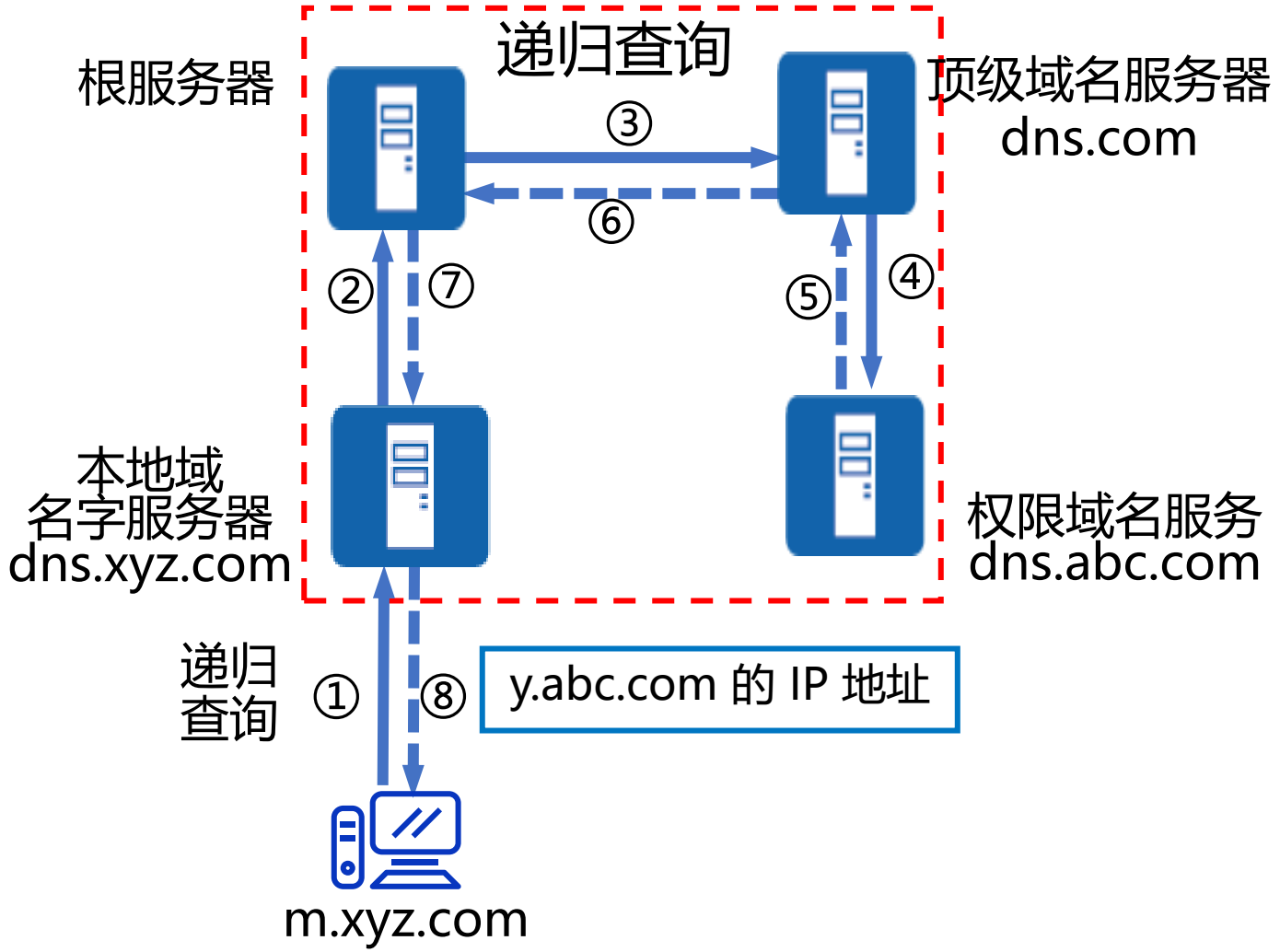


域名解析过程

- 域名查询有递归查询(recursive query)和迭代查询(或循环查询，iterative query)两种方式
  - 主机向本地域名字服务器的查询一般采用递归查询
  - 本地域名字服务器向根服务器可以采用递归查询，但一般优先采用迭代查询

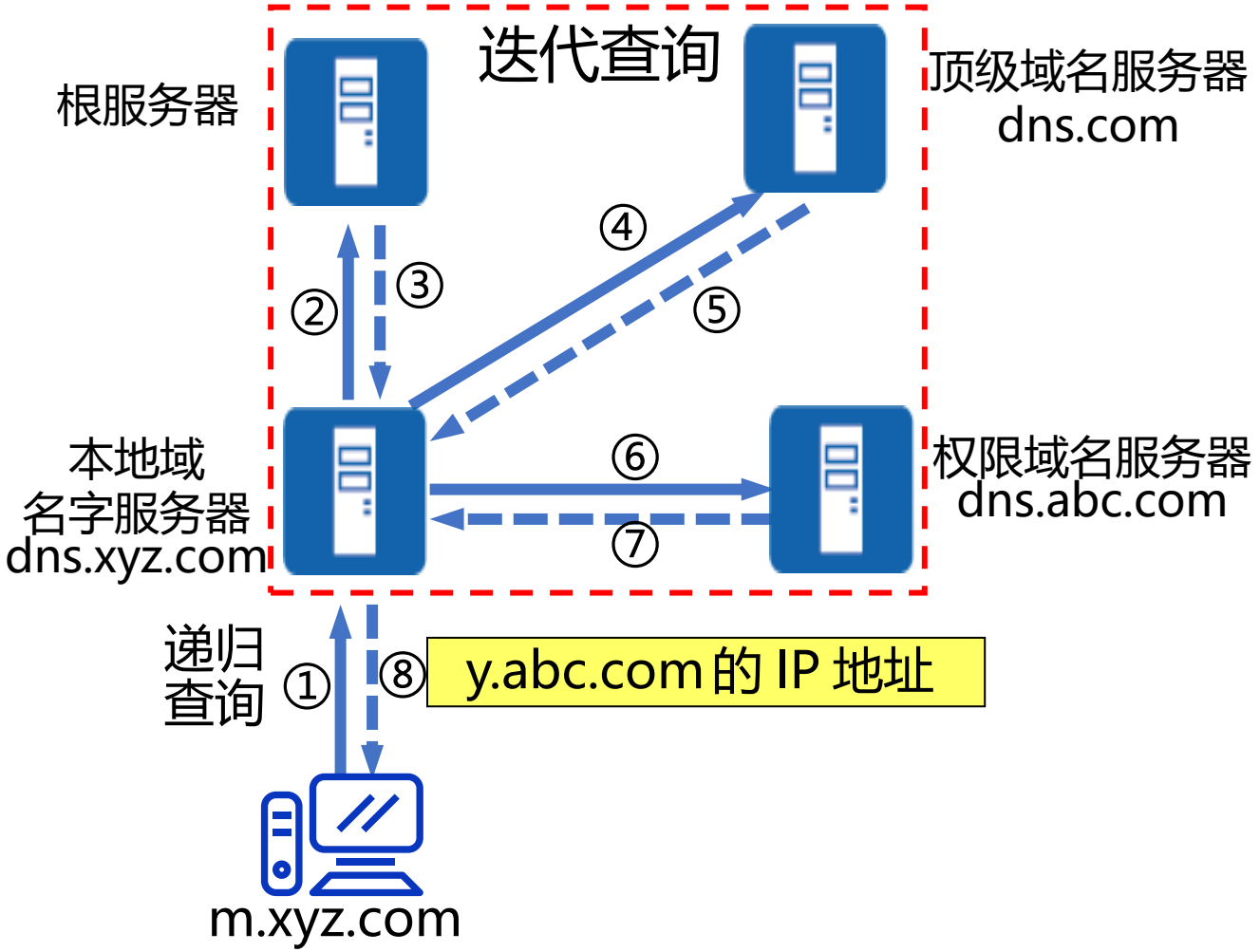
域名解析过程：递归查询方法

- 该域名服务器就以DNS客户的身份向下一步应查询的域名服务器发出查询请求，即替本地域名服务器继续查询
- 较少使用

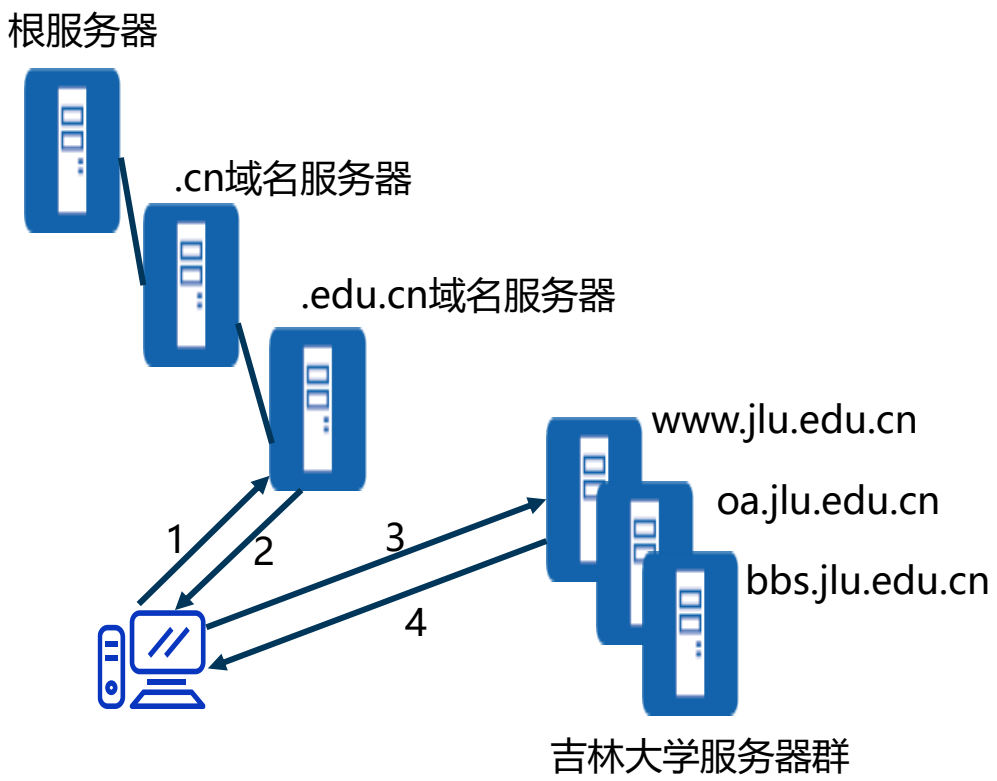


域名解析过程：迭代查询方法

- 被查询的域名服务器检索其本地数据（包括缓存中的数据）找寻所要求的数据
- 域名服务器未找到所需的数据，它将尽力返回可以帮助查询者继续解析过程的答案
- 通常使用

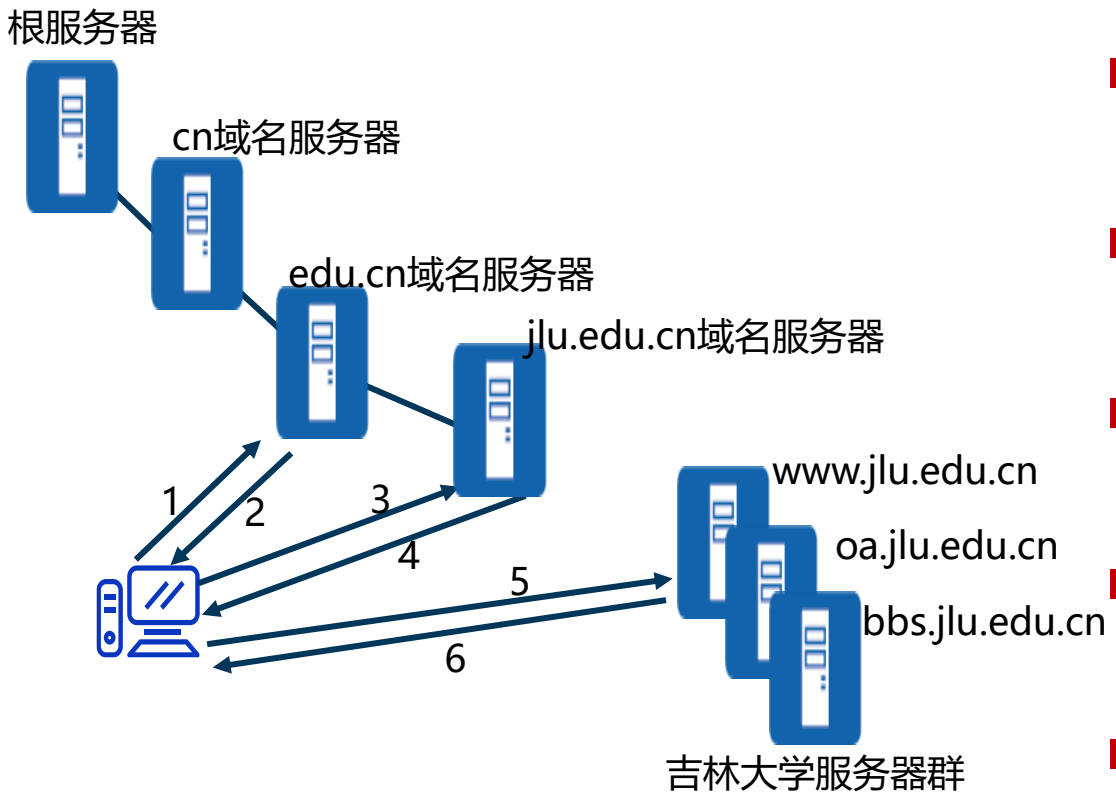


域名解析过程示例—无子域代理情形



- 无子域代理的名字解析过程：
  - 客户机向edu.cn域名服务器发出域名解析请求，以获得域名www.jlu.edu.cn的IP地址；
  - 服务器向客户机返回www.jlu.edu.cn的IP地址202.198.16.80；
  - 客户机用查询而得的IP地址向www.jlu.edu.cn发出http请求；
  - 吉林大学www服务器接受请求，并返回结果给客户机。

域名解析过程示例—有子域代理情形



- 客户机向edu.cn域名服务器发出域名解析请求，以获得域名www.jlu.edu.cn的IP地址；
- edu.cn域名服务器发觉所请求的地址已经代理出去，于是返回jlu域名服务器的IP的地址；
- 客户机向jlu.edu.cn域名服务器查询www.jlu.edu.cn的IP地址；
- jlu.edu.cn域名服务器返回www.jlu.edu.cn的IP地址202.198.16.80；
- 客户机用查询而得的IP地址向www.jlu.edu.cn发出http请求；
- 吉林大学www服务器接受请求，并返回结果给客户机。

DNS报文格式

- DNS报文分为三部分：基础结构(报文首部)、问题、资源记录(RR, Resource Record)
- 报文类型分为查询请求(query)和查询响应(reply)两类，请求和响应的报文结构基本相同

DNS报文格式

域名系统查询和响应：DNS报文格式

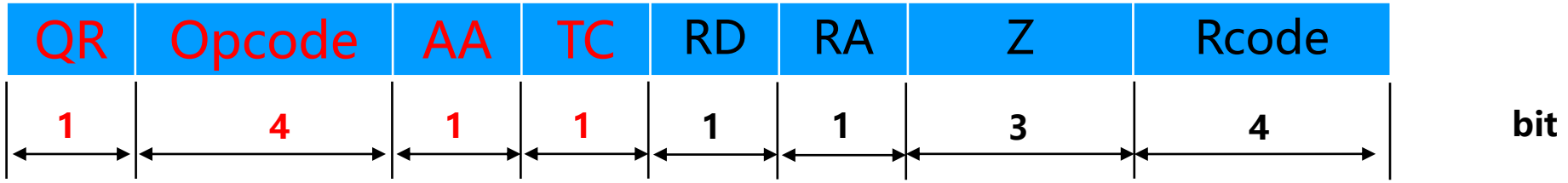
DNS报文格式的基础结构部分：

标识ID（Transaction ID）	标志（Flags）
问题数（Questions）	资源记录数（Answer RRs）
授权资源记录数（Authority RRs）	附加资源记录数（Additional RRs）

- 标识ID：DNS报文的ID。对于请求报文和其对应的响应报文，该字段的值是相同的。通过它可以区分 DNS 应答报文是对哪个请求进行响应的（2字节）
- 标志：DNS报文中的标志字段（2字节）
- 问题数：DNS查询请求的数目（2字节）
- 资源记录数：DNS响应的数目（2字节）
- 授权资源记录数：权限资源记录的数目（2字节）
- 附加资源记录数：额外的记录数目（权限名字服务器对应IP地址的数目）（2字节）

域名系统查询和响应：DNS报文格式

DNS报文格式的标志(Flags)字段：



- QR（Query/Response）：查询请求/响应标志信息。查询请求时值为0；响应时值为1
- Opcode：操作码。其中，0表示标准查询；1表示反向查询；2表示服务器状态请求
- AA（Authoritative）：授权应答，该字段在响应报文中有效。值为1时表示名称服务器是权限服务器；值为0时表示不是权限服务器
- TC（Truncated）：表示是否被截断。值为1时，表示响应已超过512字节并已被截断，只返回前512个字节

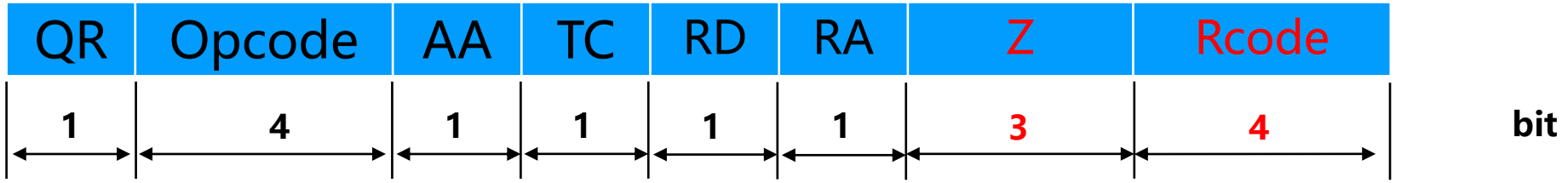
域名系统查询和响应：DNS报文格式

DNS报文格式的标志(Flags)字段：

- RD（Recursion Desired）：期望递归。该字段能在一个查询中设置，并在响应中返回。该标志告诉域名服务器必须处理这个查询，这种方式被称为一个递归查询。如果该位为0，且被请求的域名服务器没有一个授权回答，它将返回一个能解答该查询的其他域名服务器列表。这种方式被称为迭代查询
- RA（Recursion Available）：可用递归。该字段只出现在响应报文中。当值为1时，表示服务器支持递归查询

域名系统查询和响应：DNS报文格式

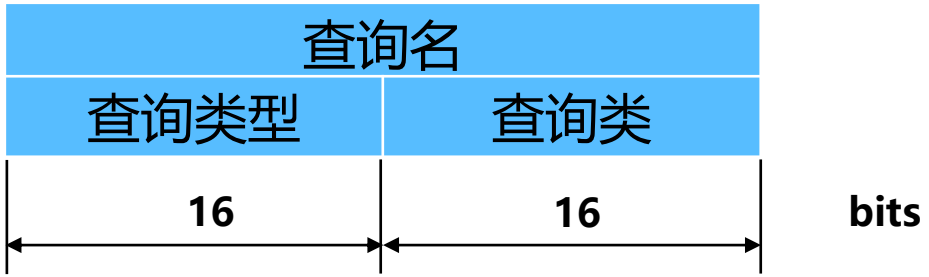
DNS报文格式的标志(Flags)字段：



- Z：保留字段，在所有的请求和响应报文中，它的值必须为0
- Rcode（Reply code）：返回码字段，表示响应的差错状态。常用Rcode有
  - 当值为0(NoError)时，表示没有错误
  - 当值为1(FormErr)时，表示报文格式错误，服务器不能理解请求的报文
  - 当值为2(ServFail)时，表示域名服务器失败，因为服务器的原因导致没办法处理这个请求
  - 当值为3(NXDomain)时，表示域名不存在，只有对授权域名解析服务器有意义，指出解析的域名不存在
  - 当值为4(NotImp)时，表示查询类型不支持，即域名服务器不支持查询类型
  - 当值为5(Refused)时，表示拒绝应答，一般是服务器由于设置的策略拒绝给出应答，如服务器不希望对某些请求者给出应答

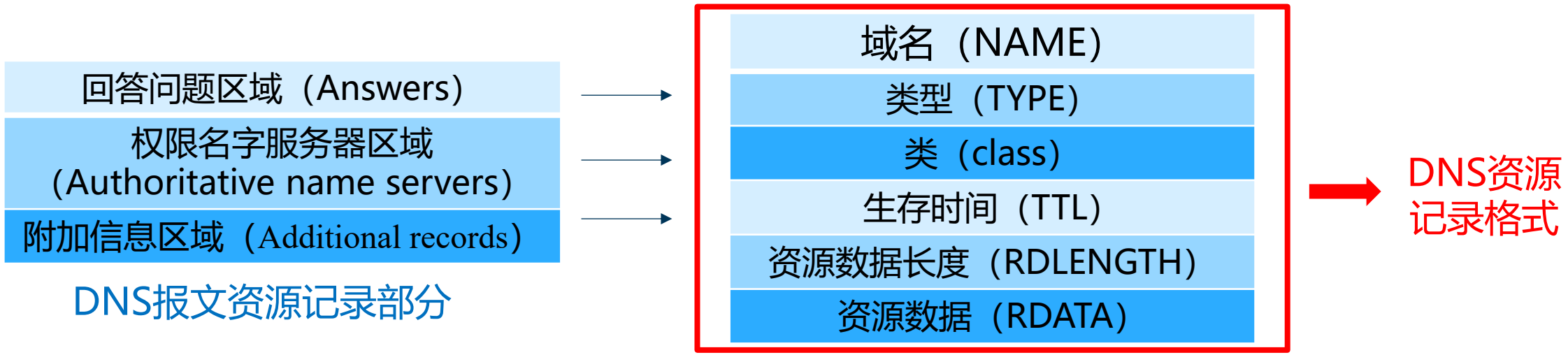
域名系统查询和响应：DNS报文格式中的“问题部分”

问题部分格式



- 用来显示DNS查询请求的问题，通常只有一个问题
- 正在进行的查询信息，包含查询名（被查询主机名字）、查询类型、查询类
  - 查询名(name)：一般为要查询的域名，有时是IP地址，用于反向查询
  - 查询类型(type)：DNS查询请求的资源类型。通常查询类型为A类型，表示由域名获取对应的IP地址
  - 查询类(class)：地址类型，通常为互联网地址，值为1(IN)

域名系统查询和响应：DNS报文格式中的“资源记录格式”



- 资源记录部分是DNS报文格式的最后3个字段，只有在DNS响应报文中才出现，包括回答问题区域字段、权限名字服务器区域字段、附加信息区域字段。这3个字段都采用资源记录的格式

域名系统查询和响应：DNS报文格式中的“资源记录格式”

- 域名：DNS请求的域名
- 类型：资源记录类型，与问题部分的查询类型值相同
- 类：地址类型，与问题部分的查询类值相同
- 生存时间：以秒为单位，表示资源记录的生命周期，一般用于当地址解析程序取出资源记录后决定保存及使用缓存数据的时间。它同时也可以表明该资源记录的稳定程度，稳定的信息会被分配一个很大的值
- 资源数据长度：资源数据的长度
- 资源数据：表示按查询段要求返回的相关资源记录数据

DNS资源记录格式

域名（NAME）
类型（TYPE）
类（class）
生存时间（TTL）
资源数据长度（RDLENGTH）
资源数据（RDATA）

DNS资源记录的不同类型

- DNS资源记录格式：(name, ttl, class, type, value)：
  - tpye=A
    - name是主机名
    - value是IPv4地址
  - tpye=NS
    - name是请求域名
    - value是管理请求域名的权限名称服务器的主机名
  - tpye=SOA
    - name是请求域名
    - value是请求域名的DNS区域的权限性信息
  - tpye=AAAA
    - name是请求域名
    - value是请求域名的IPv6地址
  - tpye=CNAME
    - name是某些"规范"名称的别名，例如www.cs.vu.nl.是主机名papac022.vu.nl.的别名
    - value是IP地址
  - tpye=MX
    - name是请求域名
    - value是与请求域名关联的SMTP邮件服务器的名称
  - tpye=TXT
    - name是请求域名
    - value是某个主机名或域名设置的说明
  - tpye=PTR
    - name是IP地址
    - value是与给定 IP 地址关联的域名

域名系统查询和响应：DNS的安全性

DNS 的安全性DNSSEC：

- DNS最初并没有设计强大的安全机制来保证数据的完整性和真实性。多年来，发现了许多威胁DNS系统可靠性和可信性的漏洞
- DNSSEC依靠数字签名保证DNS应答报文的真实性和完整性，包括通过身份验证机制
- 在DNSSEC的使用中，域名服务器用自己的私有密钥对资源记录(Resource Record, RR)进行签名，解析服务器用域名服务器的公开密钥对收到的应答信息进行验证。如果验证失败，表明这一报文可能是假冒的，或者在传输过程、缓存过程中被篡改了

DNS安全性控制流程


支持DNSSEC
的
解析器
5.请求test.net的DS记录
6.返回test.net的DS记录和该DS的RRSIG
管理net.的域名服务器
net. DNSKEY (public key of net.)
RRSIG DNSKEY (signature by net.)
test.net. DS (hash of dnskey of test.net)
RRSIG DS net.(signature)
1.请求www.test.net
2.返回www.test.net的A记录和RRSIG
3.请求test.net的DNSKEY记录
4.返回test.net的DNSKEY记录和该DNSKEY的RRSIG记录
管理test.net.的域名服务器
test.net. DNSKEY (public key of test.net.)
RRSIG DNSKEY (signature by test.net.)
www.test.net. A 192.168.1.100
www.test.net. RRSIG A test.net.(signature)

DNS安全性控制流程

- DNSSEC工作流程：
  - 一个支持DNSSEC的解析器向支持DNSSEC并管理test.net域名的权限域名服务器请求域名www.test.net.
  - 解析器除了得到一个标准的A记录(IP 地址)以外，还收到一个同名的RRSIG记录，其中包含test.net这个授权域的数字签名，它是用test.net.的私有密钥来签名的
  - 为了验证这一签名的正确性，解析服务器可以再次向test.net.的域名服务器查询响应的公开密钥，即DNSKEY资源记录
  - 解析服务器就可以用其中的公钥验证上述记录(关于www.test.net.的RRSIG记录)的真实性与完整性

域名系统高速缓存

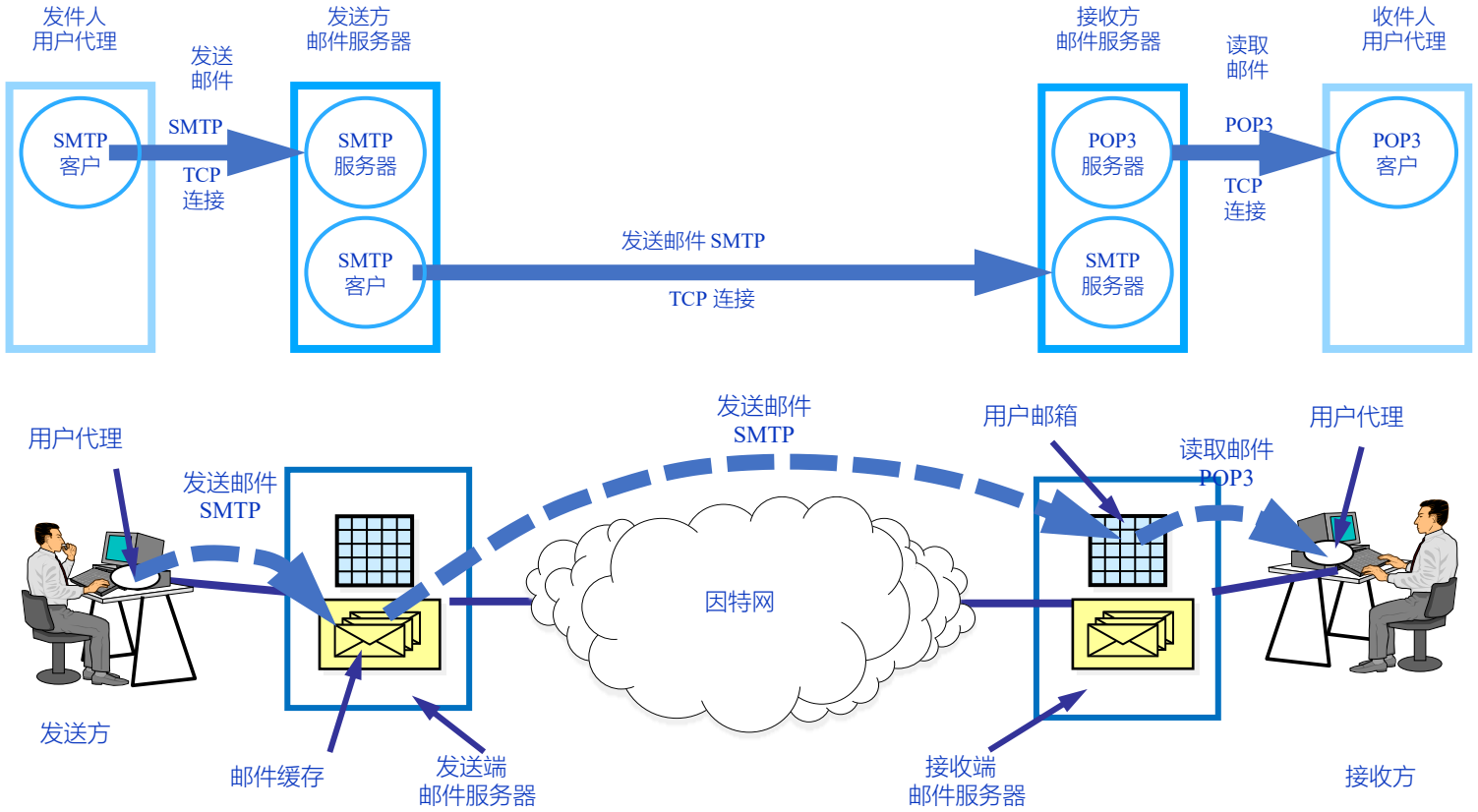
- 每个域名服务器都维护一个高速缓存，存放最近用过的名字以及从何处获得名字映射信息的记录
- 可大大减轻根域名服务器负荷，使互联网上 DNS 查询请求和回答报文的数量大为减少。而且也能够使Internet上的DNS查询请求和回答报文的数量大为减少
- 为保持高速缓存中的内容正确，域名服务器应为每项内容设置计时器，并处理超过合理时间的项（例如，每个项目只存放两天）
- 当权限域名服务器回答一个查询请求时，在响应中都指明绑定有效存在的时间值。增加此时间值可减少网络开销，而减少此时间值可提高域名转换的准确性

7.3 电子邮件概述

- 电子邮件（E-mail）是 Internet 中最广泛的应用
- 电子邮件是一种异步通信媒介
- 优越性
  - 使用方便
  - 传递迅速
  - 费用低廉
  - 可以传送多种类型的信息（包括：文字信息，声音和图像等）

电子邮件系统体系结构

- 电子邮件系统采用客户/服务器工作模式
- 邮件系统包括：用户代理、邮件服务器、电子邮件传输协议等三个主要构件

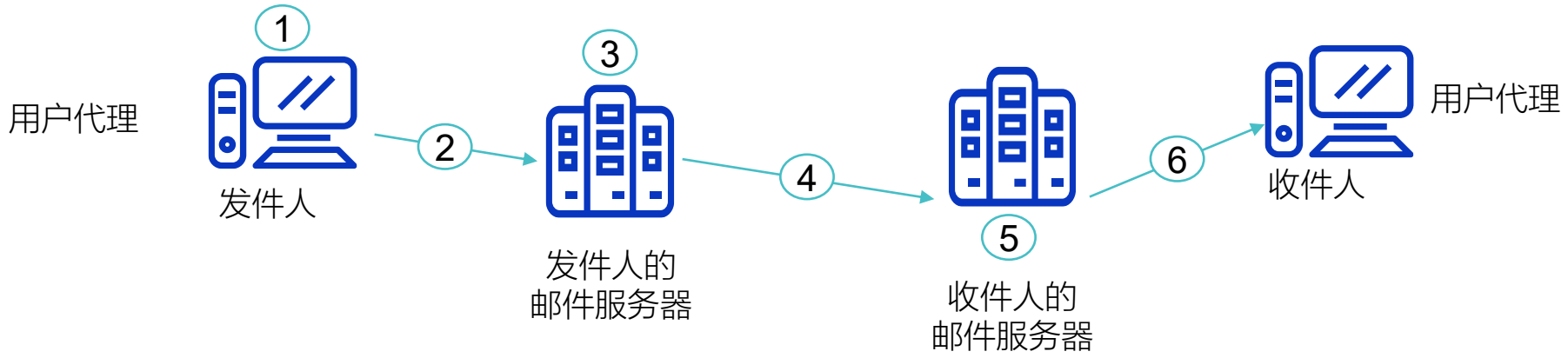


电子邮件系统中的主要构件

- 用户代理：邮件客户端软件，是用户编辑、发送、接收、阅读邮件的入口
  - 编辑和发送邮件
  - 接收、读取和管理邮件
  - 管理地址簿
  - 无统一标准
- 邮件服务器：保存用户邮件，将用户邮件转发到邮件接收方的邮件服务器上
  - 邮箱：保存用户收到的邮件　　送的邮件
  - 邮件输出队列：存储等待发
  - 运行电子邮件协议
- SMTP邮件发送协议：发送邮件到服务器以及服务器转发邮件到接收方服务器所用的协议
  - smtp客户: 发送邮件端
  - smtp服务器: 接收邮件端
- PoP3邮件读取协议：邮件接收方从邮件服务器接收邮件、阅读邮件时所用的协议

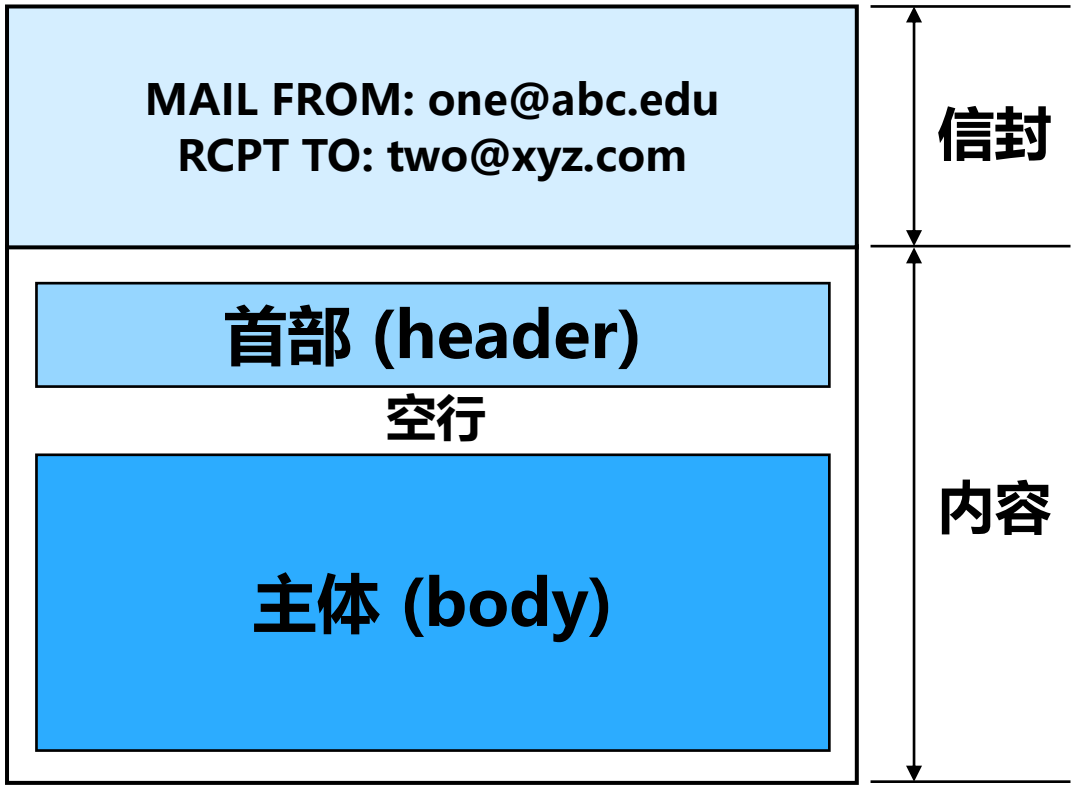
发送和接收电子邮件的几个重要步骤

- 发件人调用 PC 中的用户代理撰写和编辑要发送的邮件
- 发件人的用户代理把邮件用 SMTP 协议发给发送方邮件服务器
- SMTP 服务器把邮件临时存放在邮件缓存队列中，等待发送
- 发送方邮件服务器的 SMTP 客户与接收方邮件服务器的 SMTP 服务器建立 TCP 连接，然后就把邮件缓存队列中的邮件依次发送出去
- 运行在接收方邮件服务器中的SMTP服务器进程收到邮件后，把邮件放入收件人的用户邮箱中，等待收件人进行读取
- 收件人在收信时运行PC机中的用户代理，使用POP3（或 IMAP）协议读取发送给自己的邮件



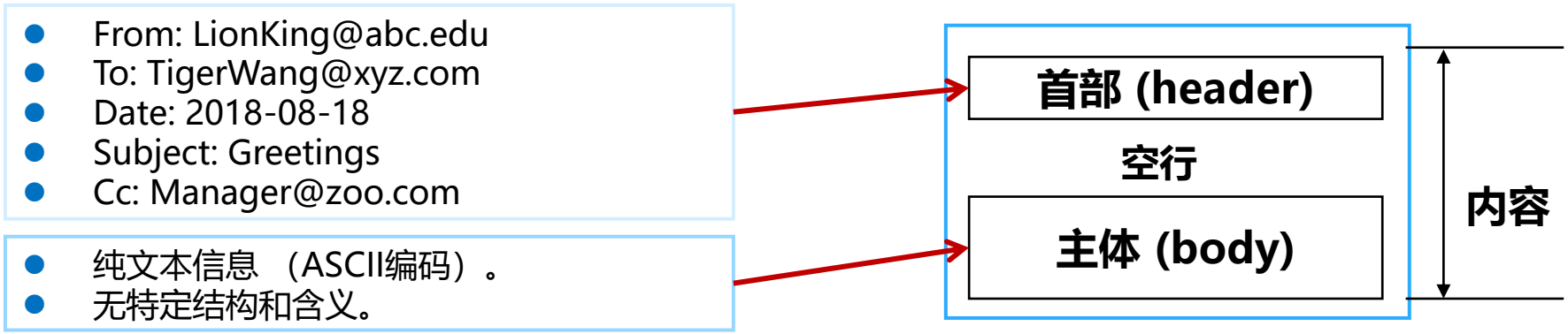
电子邮件的组成：信封 + 内容

- 电子邮件由信封 (envelope) 和内容 (content) 两部分组成
- 在邮件的信封上，最重要的就是收件人的地址。电子邮件在传输过程中，根据邮件信封上的信息来传送邮件的
- 用户在从自己的邮箱中读取邮件时才能见到邮件的内容



电子邮件内容的格式

- 首部（header）必须含有一个From:首部行和一个To:首部行，还可以包含 Subject: 等其他可选的首部行，消息体（body）指邮件正文
  - “To:” 后面填入一个或多个收件人的电子邮件地址。用户只需打开地址簿，点击收件人名字，收件人的电子邮件地址就会自动地填入到合适的位置上
  - “Subject:” 是邮件的主题。它反映了邮件的主要内容，便于用户查找邮件
  - “Cc:” 表示应给某某人发送一个邮件副本
  - “From” 和 “Date” 表示发信人的电子邮件地址和发信日期
  - “Reply-To” 是对方回信所用的地址



RFC5322

- RFC 5322 是关于Internet邮件格式的规定，是RFC 822的修订版，用于基本ASCII电子邮件
- RFC5322 只规定了邮件内容中的首部 (header) 格式，而对邮件的主体 (body) 部分则让用户自由撰写

RFC 5322与邮件传输相关的头字段

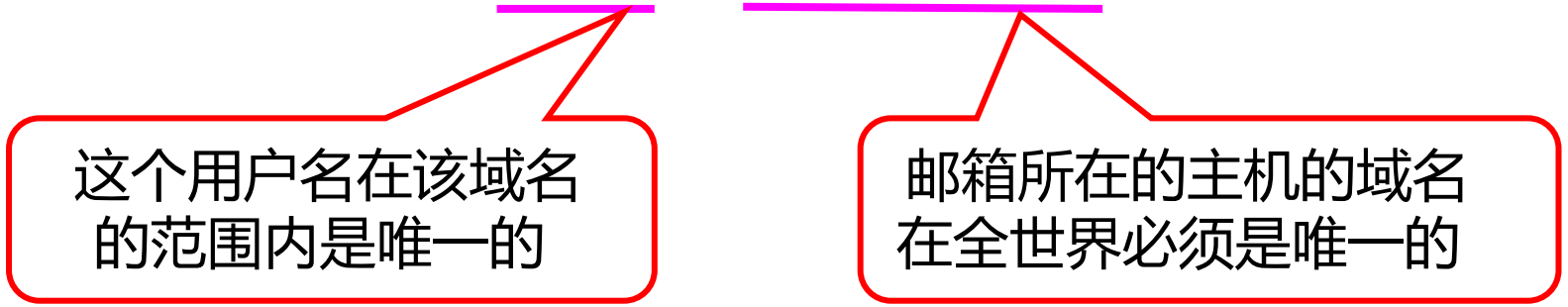
Header	Meaning
To:	Email address(es) of primary recipient(s)
Cc:	Email address(es) of secondary recipient(s)
Bcc:	Email address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	Email address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

RFC 5322邮件头使用的某些字段

Header	Meaning
Date:	The date and time the message was sent
Reply-To:	Email address to which replies should be sent
Message-Id:	Unique number for referencing this message later
In-Reply-To:	Message-Id of the message to which this is a reply
References:	Other relevant Message-Ids
Keywords:	User-chosen keywords
Subject:	Short summary of the message for the one-line display

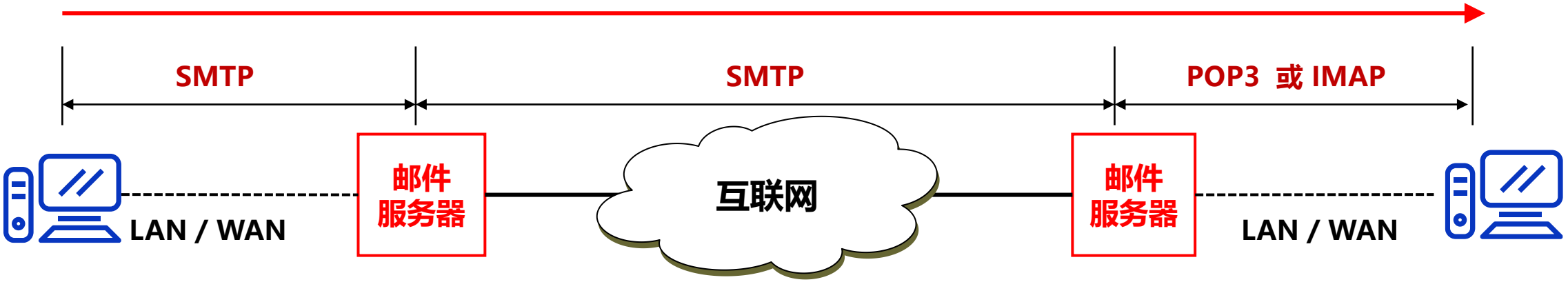
电子邮件的地址

- 电子邮件地址是一个字符串，格式规定为：
  - 收件人邮箱名@邮箱所在主机的域名
  - 符号“@”读作“at”，表示“在”的意思
- 例如电子邮件地址　jcst @ jlu.edu.cn



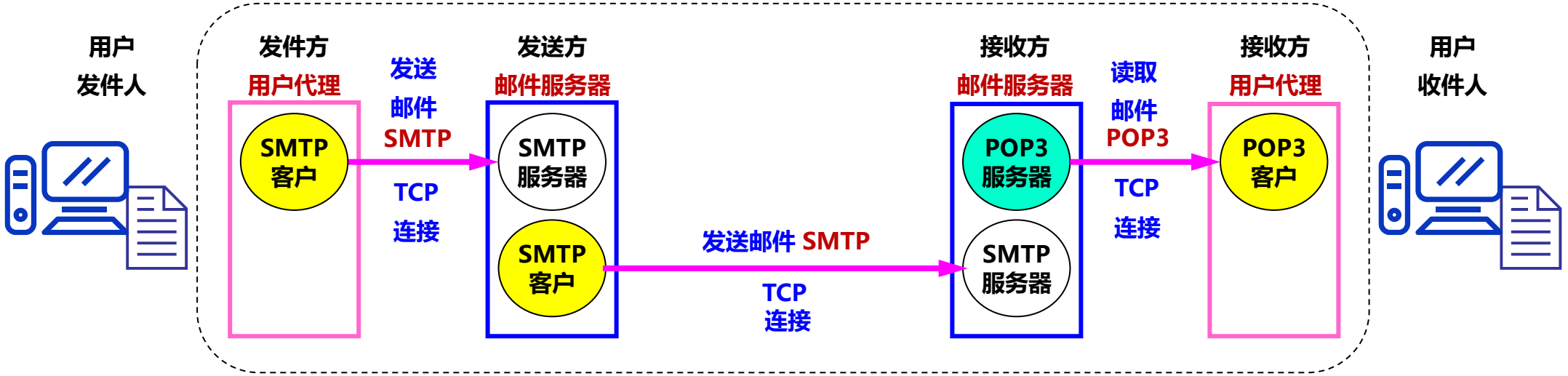
邮件发送和邮件读取协议：SMTP 和 POP/IMAP 协议

- 邮件读取协议POP或IMAP与邮件传送协议SMTP完全不同
- 发信人的用户代理向源邮件服务器发送邮件，以及源邮件服务器向目的邮件服务器发送邮件，都是使用 SMTP 协议
- POP协议或IMAP协议则是用户从目的邮件服务器上读取邮件所使用的协议



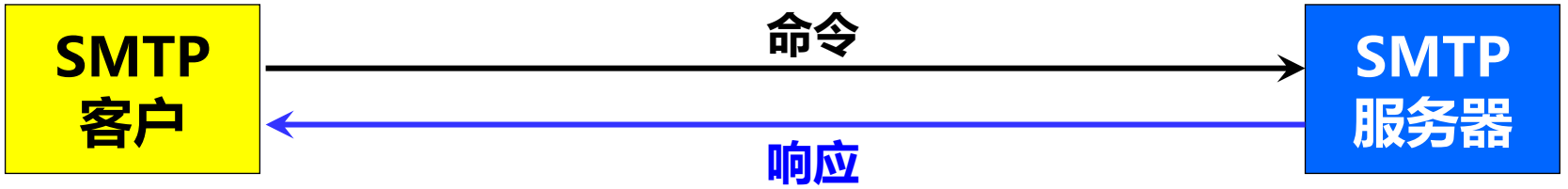
邮件发送协议：简单邮件传送协议 SMTP

- SMTP （Simple Mail Transfer Protocol） 是发送邮件时用的协议。发送方用户代理将邮件发送到自己邮箱所对应的邮件服务器，发送方邮件服务器将邮件转发到邮件接收方的邮件服务器上，都使用SMTP协议
- SMTP 使用客户–服务器方式，规定了在两个相互通信的 SMTP 进程之间交换信息的方法



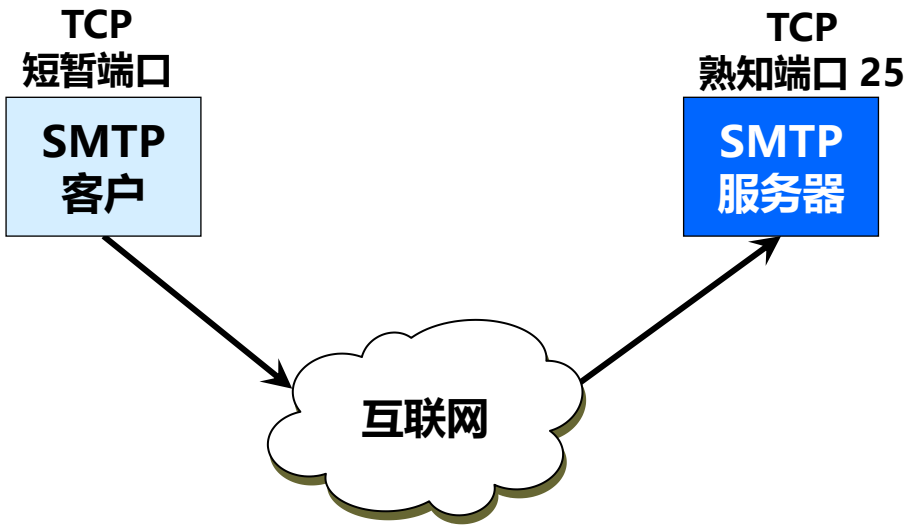
SMTP协议：基于文本方式的命令和响应

- SMTP是一个基于文本的 （即 ASCII 码） 的协议，邮件必须为7位ASCII
- SMTP 客户与服务器之间采用命令—响应方式进行交互
- 一个邮件服务器既可以作为客户，也可以作为服务器。例如：
  - 当邮件服务器 A 向另一个邮件服务器 B 发送邮件时， A 就作为 SMTP 客户，而 B 是 SMTP 服务器
  - 当邮件服务器 A 从另一个邮件服务器 B 接收邮件时， A 就作为 SMTP 服务器，而 B 是 SMTP 客户



SMTP 协议：基于 TCP 通信的三个阶段

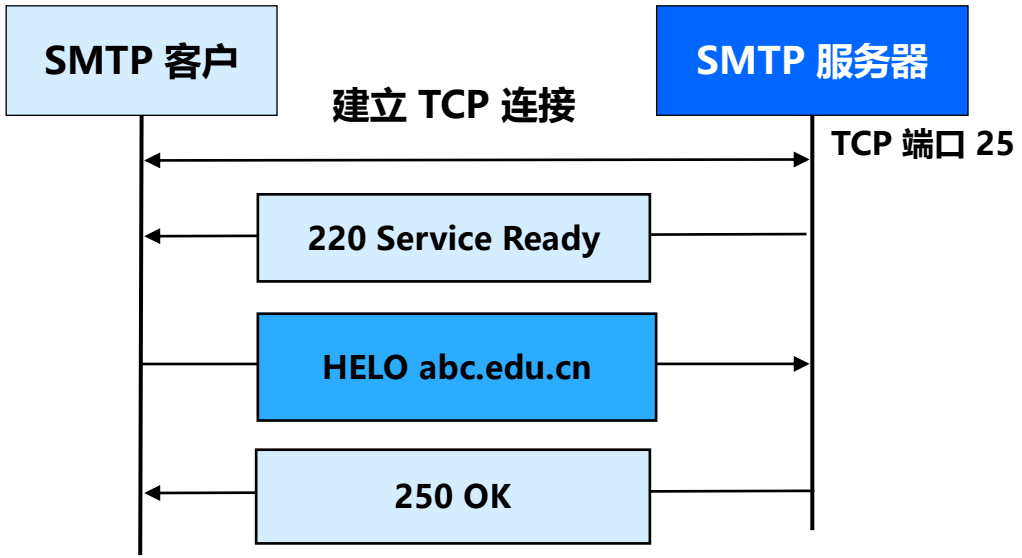
- SMTP的3个阶段：
  - 连接建立：
    - 连接是在发送主机的 SMTP 客户和接收主机的 SMTP 服务器之间建立的。SMTP不使用中间的邮件服务器
  - 邮件传送
  - 连接释放
    - 邮件发送完毕后，SMTP 应释放 TCP 连接



SMTP 基于 TCP 实现客户与服务器之间的通信，使用的端口号25

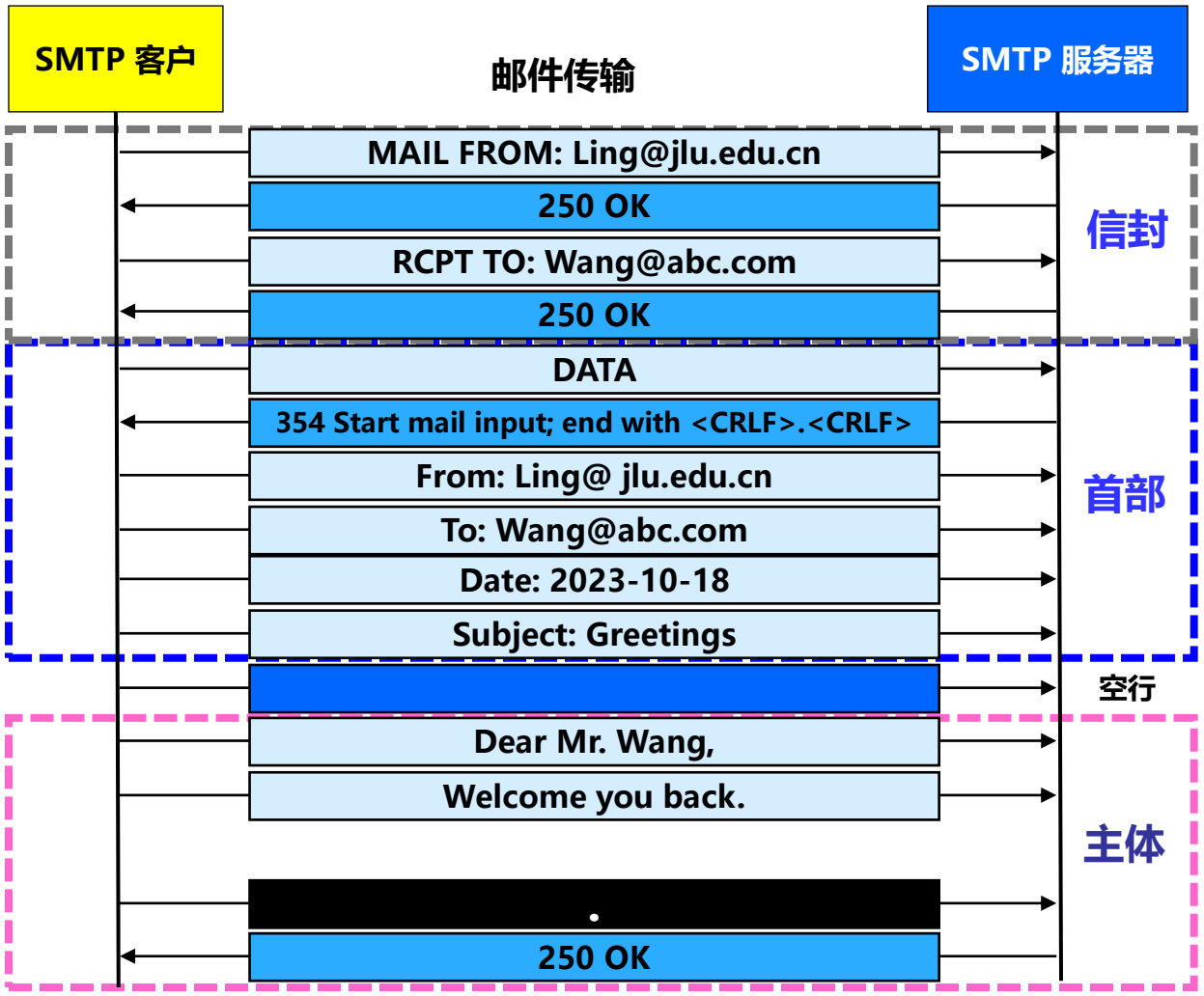
SMTP 通信的三个阶段——建立阶段

- 使用SMTP的默认端口（25）与目的主机的SMTP服务器建立TCP连接
- SMTP服务器发出“220 SMTP ready”
- SMTP客户向SMTP服务器发送HELO命令，附上发送方的主机名
- SMTP服务器若有能力接收邮件，则回答：“250 OK”，表示已准备好接收



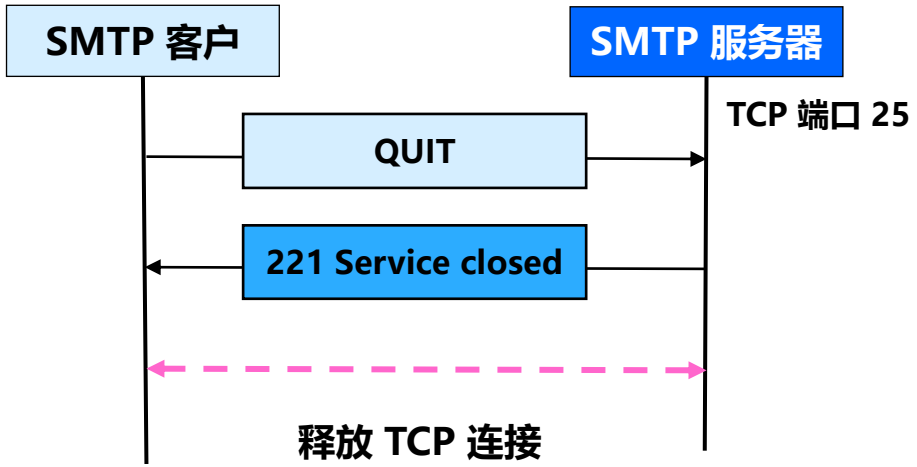
SMTP 通信的三个阶段——邮件传输

- 邮件传输首先从MAIL命令开始
- 根据收件人数量，发送一个或多个RCPT命令
  - 先确认接收端系统是否已做好接收邮件的准备，然后才能发送邮件
  - 每发送一个命令，都应当有相应的信息从SMTP服务器返回
- 接下来是DATA命令，表示要开始传输邮件的内容了



SMTP 通信的三个阶段——释放连接

- 邮件发送完毕后，SMTP客户应发送QUIT命令
- SMTP服务器返回的信息是“221 close connection”，表示SMTP同意释放TCP连接



SMTP 通信的三个阶段：邮件传送过程中的应答消息类别

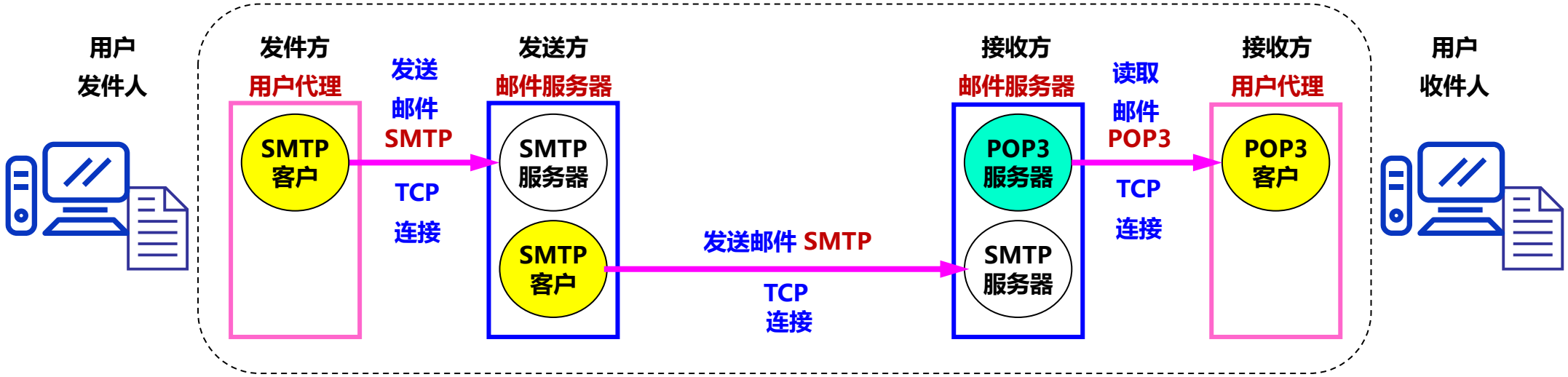
应答代码	含义	应答代码	含义
500	格式错误，命令不可识别（此错误也包括命令行过长）	**501**	参数格式错误
502	命令不可实现	**503**	错误的命令序列
504	命令参数不可实现	**211**	系统状态或系统帮助响应
214	帮助信息	**220**	服务就绪
221	服务关闭传输信道	**421**	服务未就绪，关闭传输信道（当必须关闭时，此应答可以作为对任何命令的响应）
250	要求的邮件操作完成	**251**	用户非本地
450	要求的邮件操作未完成，邮箱不可用（例如，邮箱忙）	**550**	要求的邮件操作未完成，邮箱不可用（例如，邮箱未找到，或不可访问）
451	放弃要求的操作；处理过程中出错	**551**	用户非本地，请尝试
452	系统存储不足，要求的操作未执行	**552**	过量的存储分配，要求的操作未执行
553	邮箱名不可用，要求的操作未执行（例如邮箱格式错误）	**354**	开始邮件输入，以“.”结束
554	操作失败		

邮件读取协议：POP3和IMAP

- 两个常用的邮件读取协议：
  - POP3：邮局协议 (Post Office Protocol) 第3个版本
  - IMAP：网际报文存取协议 (Internet Message Access Protocol)

POP3协议

- POP3由RFC1939定义，是一个非常简单的邮件读取协议
- POP3使用客户/服务器工作方式，在接收邮件的用户PC机中必须运行POP客户程序，而在用户所连接的ISP的邮件服务器中则运行POP服务器程序
- POP3协议基于 TCP 连接的，端口号 110



POP3协议的三个阶段

- POP3的三个阶段：
  - 认证(Authorization)：处理用户登录的过程
  - 事务处理(Trnsactions)：用户收取电子邮件，并将邮件标记为删除
  - 更新(Update)：将标为删除的电子邮件删除

POP3的三个阶段：认证阶段

- 认证阶段：
  - 客户命令：
    - user：用户名
    - pass：密码
  - 服务器响应
    - +OK
    - -ERR

S: +OK POP3 server ready

C: user bob

S: +OK

C: pass hungry

S: +OK user successfully logged on

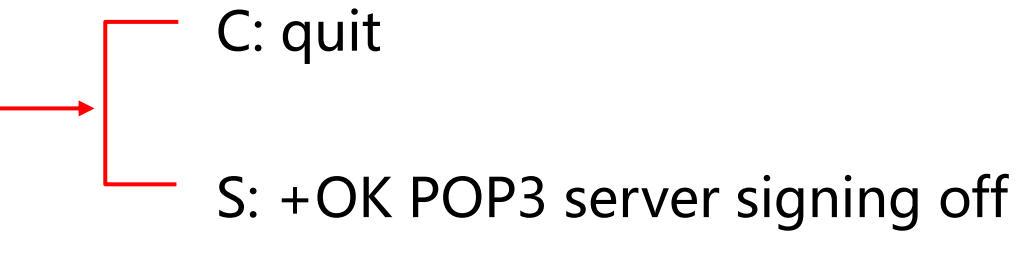
POP3的三个阶段：事务处理阶段

- 事务处理阶段：
  - list: 邮件列表
  - retr: 通过邮件号获取邮件
  - dele: 删除

C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2

POP3的三个阶段：事务处理阶段

- 更新阶段：
  - quit



POP3 的命令与响应

- POP3最小命令集
  - USER name
  - PASS string
  - QUIT
  - STAT
  - LIST [msg]
  - RETR msg
  - DELE msg
  - NOOP
  - RSET
  - QUIT
- 可选的POP3命令
  - APOP name digest
  - TOP msg n
  - UIDL [msg]
- POP3命令响应
  - +OK
  - -ERR

IMAP协议

- IMAP 使用客户–服务器方式。IMAP 基于TCP实现客户与服务器的通信
- IMAP是一个联机协议。要想查阅邮件，必须先联网
- 连接后只下载邮件首部（部分下载）
- 用户直接在IMAP服务器上创建和管理文件夹
- 用户可以搜索邮件内容
- 用户可以在不同的地方使用不同的计算机随时上网阅读和处理自己的邮件
- 允许收信人只读取邮件中的某一个部分

IMAP 协议

- IMAP 是Internet邮件访问协议，RFC 2060协议标准：
  - 用于最终交付的主要协议
  - IMAP是较早使用的最终交付协议—POP3(邮局协议，版本3)的改进版
  - 邮件服务器运行侦听端口143的IMAP服务器
  - 用户代理运行一个IMAP客户端
  - 客户端连接到服务器并开始发出命令

IMAP特性

- IMAP允许用户在不同地方使用不同的计算机随时上网阅读和处理自己的邮件
- IMAP服务器把每个邮件与一个文件夹联系起来，当邮件第一次到达服务器时，与收件人的INBOX文件夹相关联。收件人把邮件移到一个新创建的文件夹中，阅读邮件，删除邮件等
- IMAP还为用户提供了在远程文件夹中查询邮件的命令，按指定条件去查询匹配的邮件
- 与POP3不同，IMAP服务器维护了IMAP会话的用户状态信息，例如，文件夹的名字以及哪些邮件与哪些文件夹相关联
- IMAP具有允许用户代理获取邮件某些部分的命令。例如，一个用户代理可以只读取一个邮件的首部，或只是一个多部分MIME邮件的一部分
- IMAP的缺点是如果用户没有将邮件复制到自己的PC上，则邮件一直是存放在IMAP服务器上。用户需要经常与IMAP服务器建立连接

IMAPv4 协议命令

Command	Description
CAPABILITY	List server capabilities
STARTTLS	Start secure transport (TLS; see Chap. 8)
LOGIN	Log on to server
AUTHENTICATE	Log on with other method
SELECT	Select a folder
EXAMINE	Select a read-only folder
CREATE	Create a folder
DELETE	Delete a folder
RENAME	Rename a folder
SUBSCRIBE	Add folder to active set
UNSUBSCRIBE	Remove folder from active set
LIST	List the available folders
LSUB	List the active folders
STATUS	Get the status of a folder
APPEND	Add a message to a folder
CHECK	Get a checkpoint of a folder
FETCH	Get messages from a folder
SEARCH	Find messages in a folder
STORE	Alter message flags
COPY	Make a copy of a message in a folder
EXPUNGE	Remove messages flagged for deletion
UID	Issue commands using unique identifiers
NOOP	Do nothing
CLOSE	Remove flagged messages and close folder
LOGOUT	Log out and close connection

POP3 和 IMAPv4 协议比较

Feature	POP3	IMAP
Where is protocol defined	RFC 1939	RFC 2060
TCP port used	110	143
Where is e-mail stored	User's PC	Server
Where is e-mail read	Off-line	On-line
Connect time required	Little	Much
Use of server resources	Minimal	Extensive
Multiple mailboxes	No	Yes
Who backs up mailboxes	User	ISP
Good for mobile users	No	Yes
User control over downloading	Little	Great
Partial message downloads	No	Yes
Are disk quotas a problem	No	Could be in time
Simple to implement	Yes	No
Widespread support	Yes	Growing

通用互联网邮件扩充：MIME

- MIME(Multipurpose Internet Mail Extensions)—多用途Internet邮件扩展，基本格式的多媒体扩展，可传输
  - 多媒体消息
  - 二进制文件
- MIME沿用了RFC 822格式，但是加入了消息体结构，定义了非ASCII消息编码规则
- MIME类型的消息使用现有邮件程序和协议进行传送，但需要用户代理能够解释MIME格式

Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Type and format of the content

MIME增加的消息头部

内容传输编码——Quoted-Printable编码

- Quoted-Printable编码方法可用于包含了非英文文本（例如汉字）的邮件中
  - 对于可打印的ASCII码（除等号外）都不改变
  - 对等号以及编号超过127的ASCII码的编码方法是：
    - 先将每个字节的二进制代码用两个十六进制数字表示，然后在前面再加上一个等号
    - 示例：汉字的“系统”的二进制编码是：10111111 10100101 10111100 10100011，其十六进制数字表示为：CFB5CDB3，用Quoted-Printable编码表示为：=CF=B5=CD=B3
    - 等号的二进制代码为00111101，即十六进制的3D，因此等号的Quoted-Printable编码为“=3D”

内容传输编码——Base64编码

- 这种编码方法是先将24位的代码划分为4个6位组
- 6位组的二进制代码共有64种不同的值，从0到63
  - 用A表示0，用B表示1
  - 26个大写字母排列完毕后，接下去再排26个小写字母，再后面是10个数字
  - 最后用“+”号表示62，而用“/”号表示63。再用两个连在一起的等号“= =”和一个等号“=”分别表示最后一组的代码只有8或16位

➢示例：二进制代码，共24位：01001001 00110001 01111001，先将其划分为4个6位组，即010010 010011 000101 111001。对应的base64编码为：STE5

内容类型

- MIME标准规定Content-Type说明必须含内容类型（type）和子类型（subtype），中间用“/”分开，参数（parameters）则是可选的
- MIME用于传输文本、图像、声音等多种媒体数据
- 媒体分为两大类：
  - 独立媒体类型（discrete）
  - 复合媒体类型（composite）

内容类型

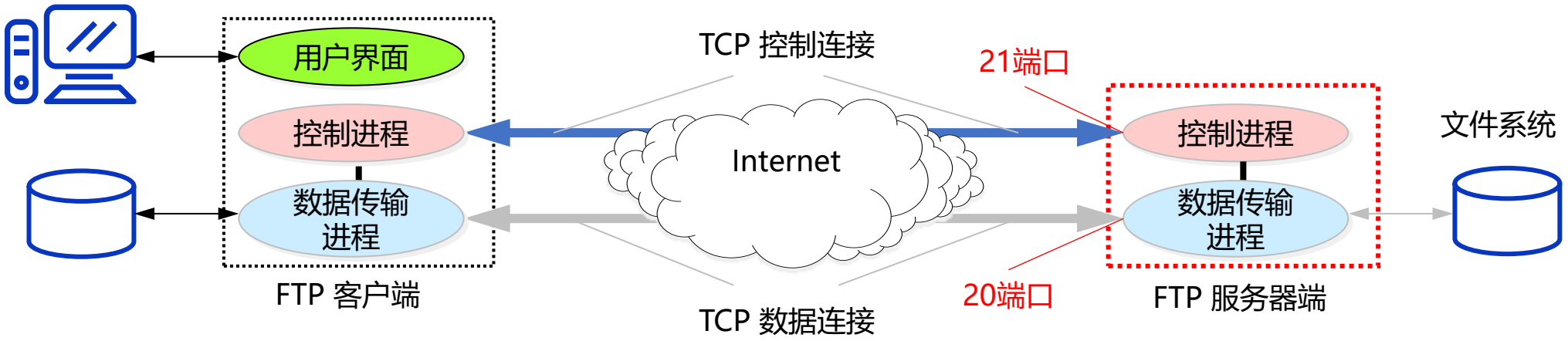
- MIME标准的独立媒体类型的五个基本媒体类型：
  - 文本（text）
  - 图像（image）
  - 音频（audio）
  - 视频（video）
  - 应用（application）
- 复合媒体类型包含两个基本媒体类型：多重（multipart）类型和消息（message）类型
- MIME标准中定义了多重类型的四种子类型：
  - 混合（mixed）子类型
  - 选择（alternative）子类型
  - 并行（parallel）子类型
  - 摘要（digest）子类型

7.4 FTP服务

- 网络环境下复制文件的复杂性
  - 计算机存储数据的格式不同
  - 文件目录结构和文件命名规则不同
  - 对于相同的文件存取功能，操作系统使用的命令不同
  - 访问控制方法不同
- 文件传输协议FTP
  - FTP(File Transfer Protocol)是Internet上使用最广泛的应用层协议之一
  - FTP提供交互式的访问，允许用户指明文件的类型与格式，并允许文件具有存取权限
  - FTP屏蔽了各计算机系统的细节，适用于在异构网络中任意计算机之间传送文件
  - RFC 959早在1985年就已经成为Internet的正式标准

FTP 工作过程

- FTP的两个端口与两个连接：
  - 控制连接在整个会话期间一直保持，客户进程发出的文件传输请求通过控制连接发送给服务器控制进程（工作在TCP21端口），但控制连接不用来传输文件
  - 服务器控制进程在接收到客户进程发送来的文件传输请求后就创建数据传输进程（工作在TCP20端口，也可以协商一个端口）和数据连接
  - 数据连接用来连接客户进程和服务器数据传输进程，实际完成文件的传输。服务器数据传输进程在文件传输完毕后关闭数据连接并结束运行

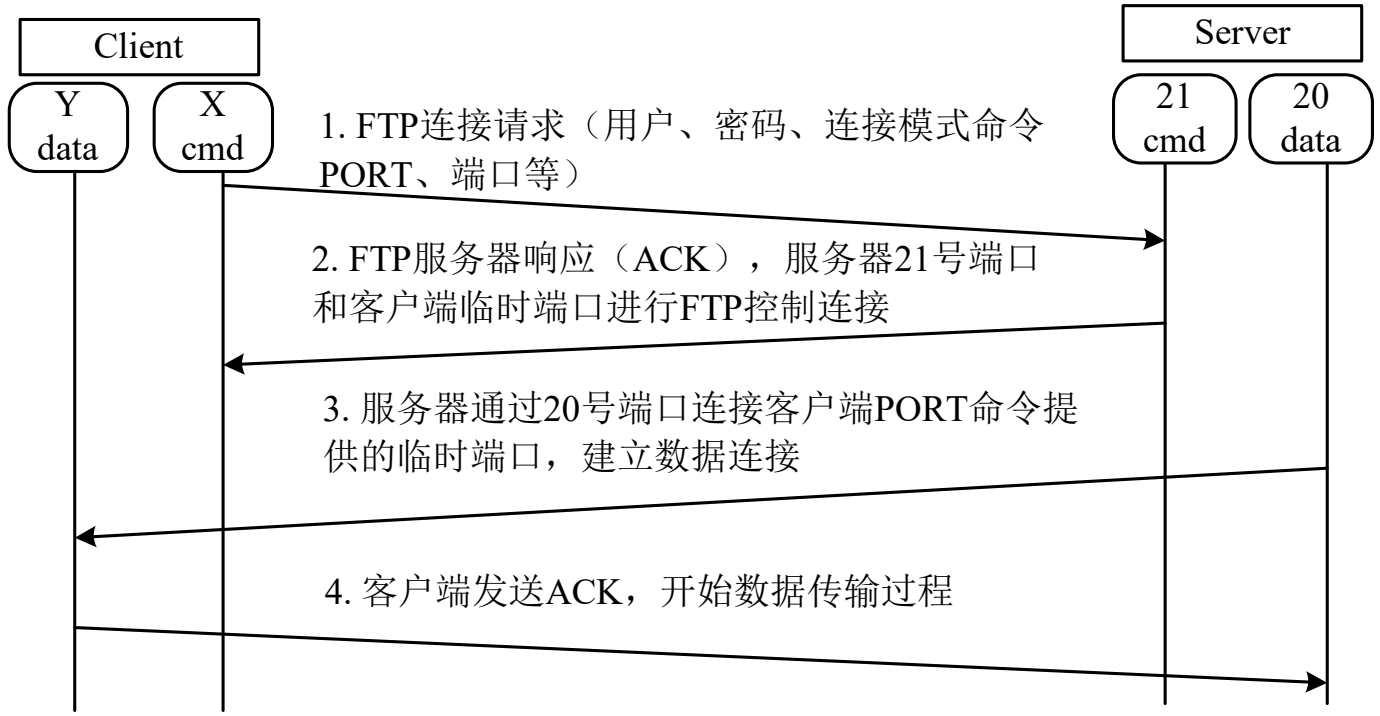


FTP 工作过程

- FTP使用C/S方式实现
- FTP工作过程
  - 服务器主进程打开TCP21端口，等待客户进程发出的连接请求
  - 客户可以用分配的任意一个本地端口号与服务器进程的TCP21端口连接
  - 客户请求到来时，服务器主进程启动从属进程来处理客户进程发来的请求
  - 服务器从属进程对客户进程的请求处理完毕后即终止，但从属进程在运行期间根据需要还可能创建其他一些子进程
  - 服务器主进程返回，继续等待接收其他客户进程发来的连接请求，服务器主进程与从属进程并行工作

FTP 的两种工作模式：主动模式（PORT模式）

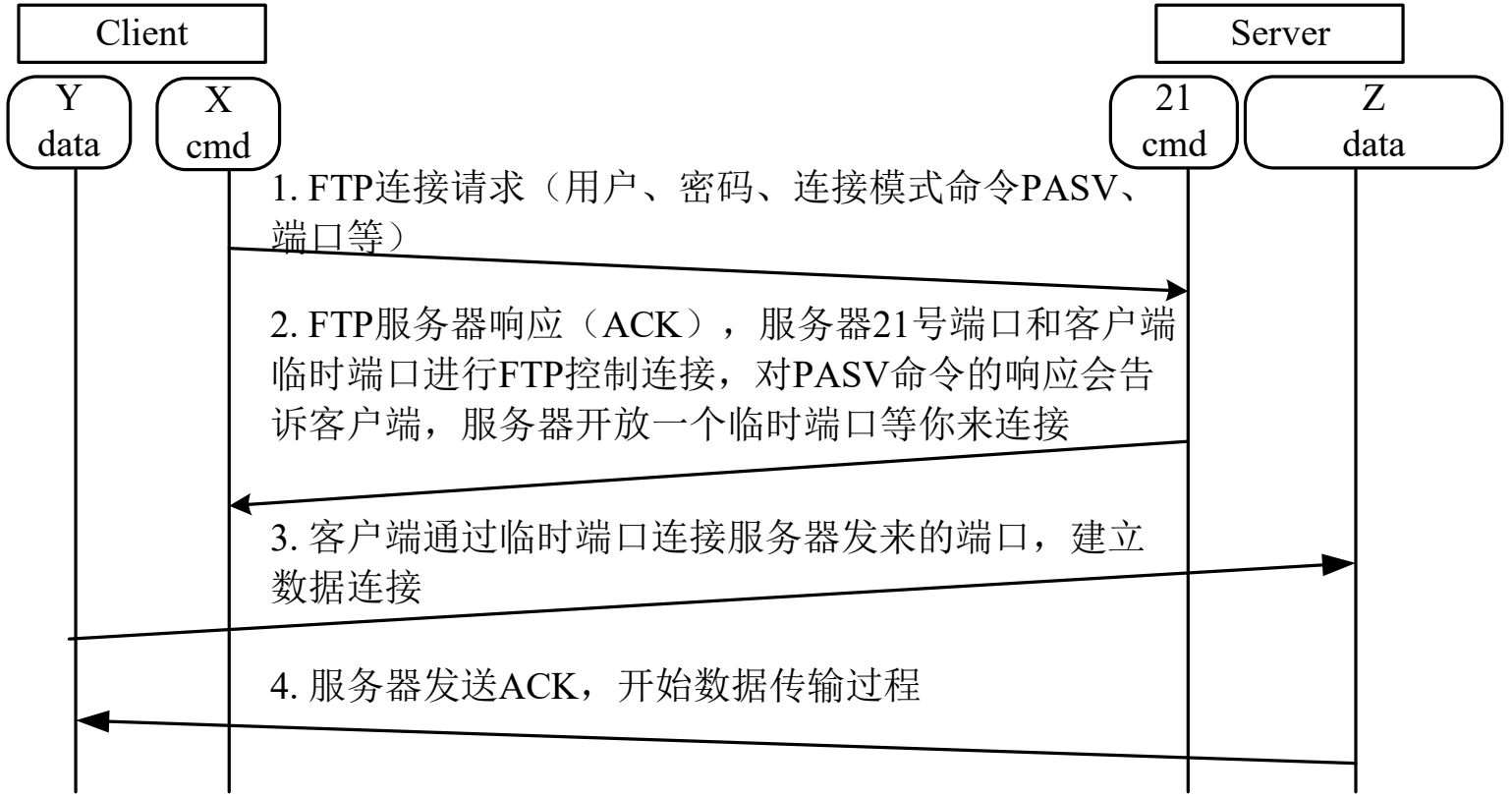
- 数据通道连接发起方为服务器端
- 服务器使用20号端口连接客户端的Y端口建立数据连接



(a)主动连接

FTP 的两种工作模式：被动模式（PASV模式）

- 数据通道连接发起方为客户端
- 客户端使用端口号Y去连接服务器的数据连接端口Z



（b）被动连接

FTP 客户端常用命令

ABOR	放弃当前的FTP命令和数据传输
LIST filelist	显示当前目录中的文件和目录（文件列表由数据端口返回）
PASS password	向服务器发送用户口令
PORT n1,n2,n3,n4,n5,n6	客户端IP地址（n1,n2,n3,n4）和端口（n5*256+n6）
QUIT	注销
RETR filename	从远程主机的当前目录中取回（get）文件
STOR filename	将文件发送（put）至远程主机的当前目录
SYST	服务器返回系统类型
TYPE type	说明文件类型：A表示ASCII码，I表示图像
USER username	向服务器发送用户名

FTP 命令应答码及含义

- 1yz　肯定预备应答。它仅仅是在发送一个命令前，期待另一个应答时启动
- 2yz　肯定完成应答。一个新命令可以发送
- 3yz　肯定中介应答。该命令已被接受，但另一个命令必须发送
- 4yz　暂时否定完成应答。请求的动作没有发生，但差错状态是暂时的，命令可以过后再发
- 5yz　永久性否定完成应答。命令不被接受，并且不再重发
- x0z　语法错误
- x1z　信息
- x2z　连接。用于数据和控制连接
- x3z　鉴别和记账。用于注册或记账命令
- x4z　未定义
- x5z　文件系统状态

FTP 特点

- 文件传送协议 FTP 只提供文件传送的一些基本的服务，它使用 TCP 可靠的传输服务
- FTP 的主要功能是减少或消除在不同操作系统下处理文件的不兼容性
- FTP 使用客户服务器方式。一个 FTP 服务器进程可同时为多个客户进程提供服务。FTP 的服务器进程由两大部分组成：一个主进程，负责接受新的请求；另外有若干个从属进程，负责处理单个请求

TFTP：简单文件传输协议

- TFTP(Trivial File Transfer Protocol) 是一个很小且易于实现的文件传输协议
- 使用C/S方式和UDP协议实现。因此 TFTP 需要有自己的差错改正措施
- 只支持文件传输而不支持交互
- 没有庞大的命令集，没有列目录的功能，也不能对用户进行身份鉴别
- 支持ASCII 码或二进制传送
- 支持对文件进行读或写
- 使用很简单的首部

TFTP工作过程

- TFTP的工作过程类似停止-等待协议。每发送完一个文件块后就等待对方的确认，确认时应指明所确认的块编号
- 发完数据后在规定时间内收不到确认就要重发该数据PDU
- 发送确认PDU的一方若在规定时间内收不到下一个文件块，也要重发确认PDU。这样就能保证文件的传输不会因某一个数据PDU的丢失而失败

TFTP 的数据传输过程

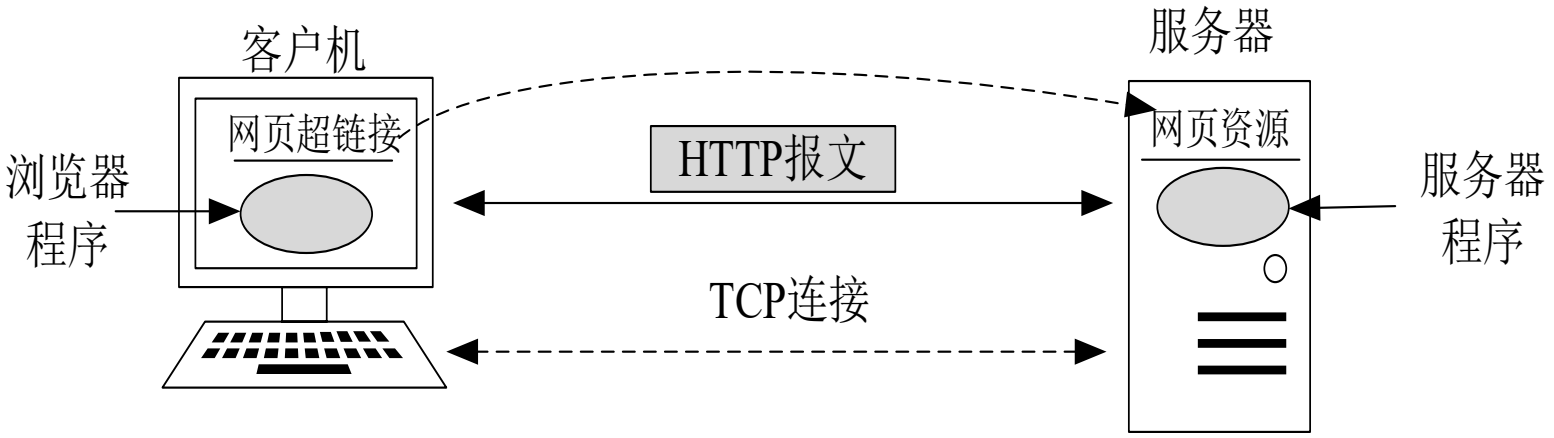
- 开始工作时，TFTP客户进程发送一个读请求PDU或写请求PDU给TFTP服务器进程，其UDP端口号为69
- TFTP服务器进程要选择一个新的端口和TFTP客户进程进行通信
- 数据 PDU 也称为文件块 (block)，每个块按序从1开始编号
- 每次传送的数据PDU中有512字节的数据，但最后一次可不足512字节
- 若文件长度恰好为512字节的整数倍，则在文件传输完毕后，还必须在最后发送一个只含首部而无数据的数据 PDU
- 若文件长度不是512字节的整数倍，则最后传送数据PDU的数据字段一定不满512字节，这正好可作为文件结束的标志

TFTP 的主要特点

- 每次传送的数据 PDU 中有 512 字节的数据，但最后一次可不足 512 字节
- 数据 PDU 也称为文件块 (block)，每个块按序编号，从 1 开始
- 支持 ASCII 码或二进制传送
- 可对文件进行读或写
- 使用很简单的首部

7.5 Web服务

- Web应用结构包括三个部分：
  - Web服务器：所有的资源存储在Web服务器上
  - 浏览器：运行在客户机上，是用户请求页面，对页面内容进行解释，并以恰当格式显示
  - 超文本传输协议HTTP：客户浏览网页资源通过超文本传输协议实现

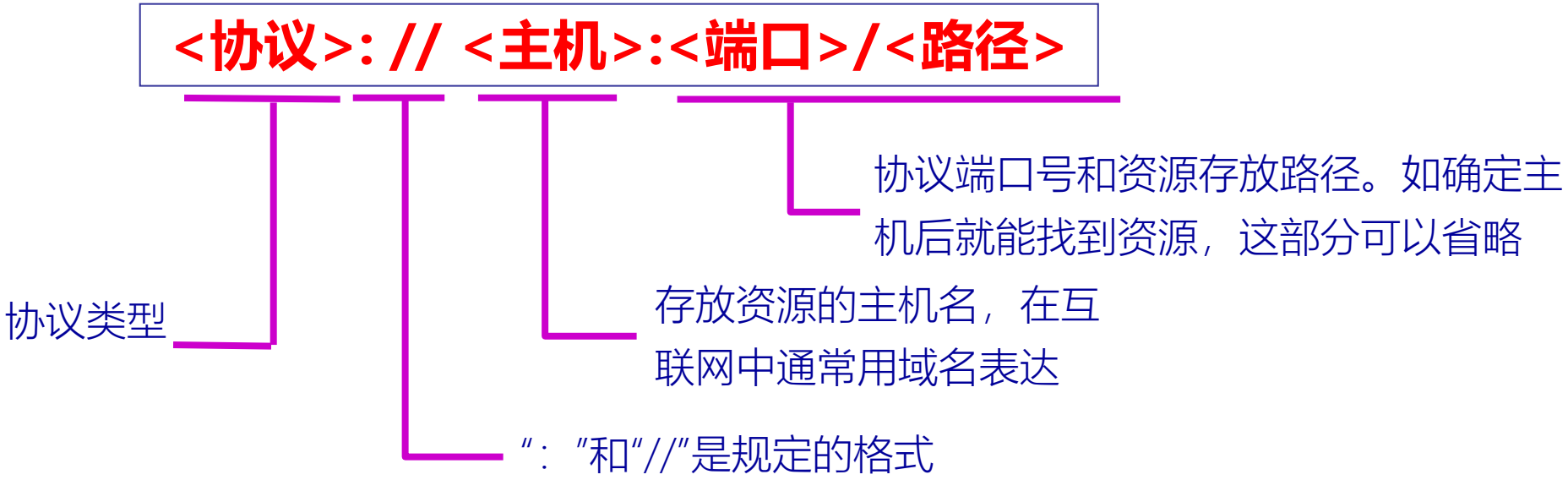


客户浏览网页的基本过程

- 客户机向Web服务器发起TCP连接
- 客户机上的浏览器程序根据要访问页面的网址，发出HTTP请求报文
- HTTP报文中包含统一资源定位器（Uniform Resource Locator，URL），即通常所说的网址。互联网上的所有资源都有一个唯一的URL。网页超链接中包含URL的信息，当点击网页上的一个超链接时，HTTP的请求报文会把URL携带在报文当中，发送给服务器
- 服务器根据收到URL找到相应的网页资源，网页资源使用HTML语言编写，网页资源作为HTTP报文的响应信息返回给浏览器
- 浏览器收到HTML语言的网页资源后，进行解释，以一定的格式显示在浏览器当中，呈现给客户

URL的格式

- URL由两大部分组成，以“//”冒号隔开，且对字符大、小写没有要求。一般形式：



- 现在有些浏览器为了方便用户，在输入 URL 时，可以把最前面的“http://”甚至把主机名最前面的“www”省略，然后浏览器替用户把省略的字符添上。如：用户只要键入XXXX.com，浏览器就自动把未键入的字符补齐，变成http://www.XXXX.com

URL 的一般形式—举例

例如： http://www.XXXXX.com:8000/comics.php

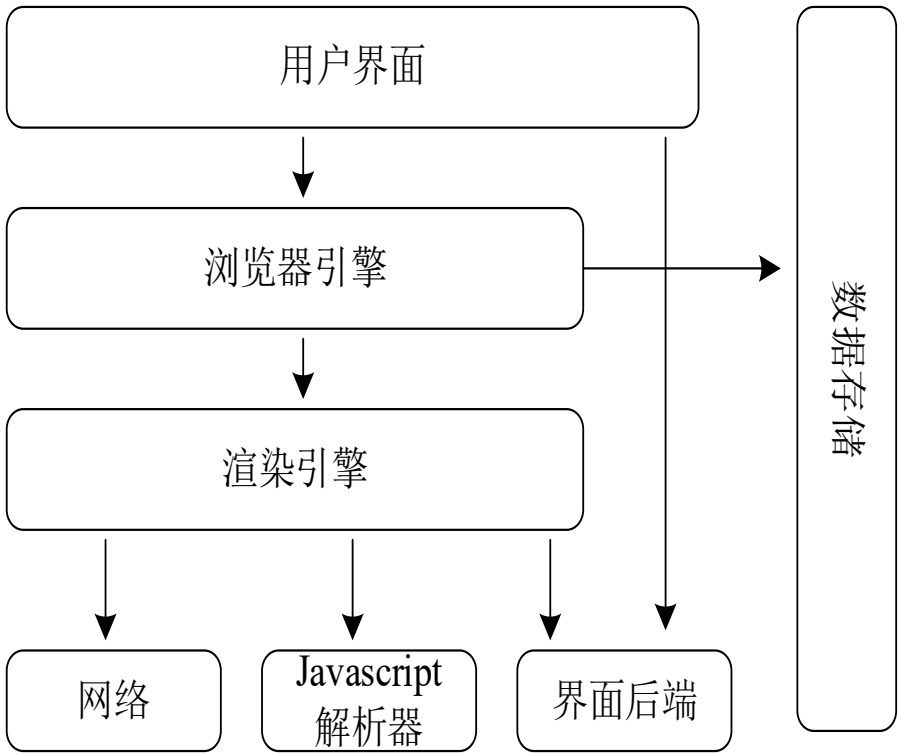
协议类型　　主机名即服务器　　端口　　路径和文件名

名字	用途	实例
http	超文本HTML	http://www.XXXX.edu.cn/xxgk/xxjj.htm
https	安全超文本	https://www.XXXX.com/
ftp	FTP	ftp://ftp.XXXX.edu.cn
file	本地文件	file:///usr/XXXX/prog.c
mailto	发送邮件	mailto:xautmail@XXXX.edu.cn
rtsp	流媒体	rtsp://XXXXX.com/montypython.mpg
sip	多媒体呼叫	sip:eve@advesary.com

用户浏览页面的两种方法

- 浏览页面的两种方法：
  - 在浏览器的地址窗口中键入所要找的页面的 URL
  - 在某一个页面中用鼠标点击一个可选部分，这时浏览器会自动在互联网上找到所要链接的页面

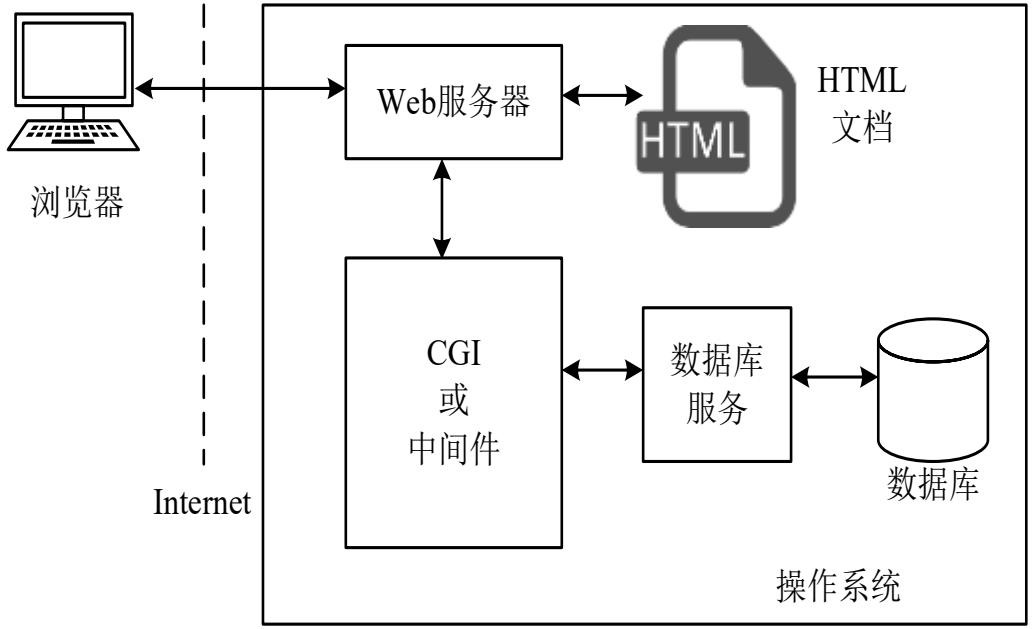
Web浏览器的软件架构



Web浏览器的软件架构

- 用户界面：包括地址栏、前进/后退按钮、书签菜单等
- 浏览器引擎：加载URL资源和一些其它高级操作方法
- 渲染引擎：对网页语法解析并渲染显示到用户界面上
- 网络：用于网络调用，例如HTTP请求
- Javascript解析器：用于解析和执行JavaScript代码
- 用户界面后端：绘制基本的窗口小部件
- 数据存储：浏览器在硬盘上保存各种数据，如Cookie

WEB网站的软件架构



Web服务器结构

- 通过 HTTP 协议携带 HTML 超文本，将要显示的内容和显示格式推送给浏览器
- CGI 是 HTML 网页与各种数据库系统之间的桥梁，也是动态网页的有效支持机制。CGI 程序的主要功能有：
  - 从 Web 服务器得到用户的访问请求
  - 根据用户的请求对数据库进行访问
  - 将结果数据封装在一个 HTML 文档中并通过标准输出返回给 Web 服务器
  - 再由 Web 服务器将结果返回用户浏览器

Web网页上的对象分类

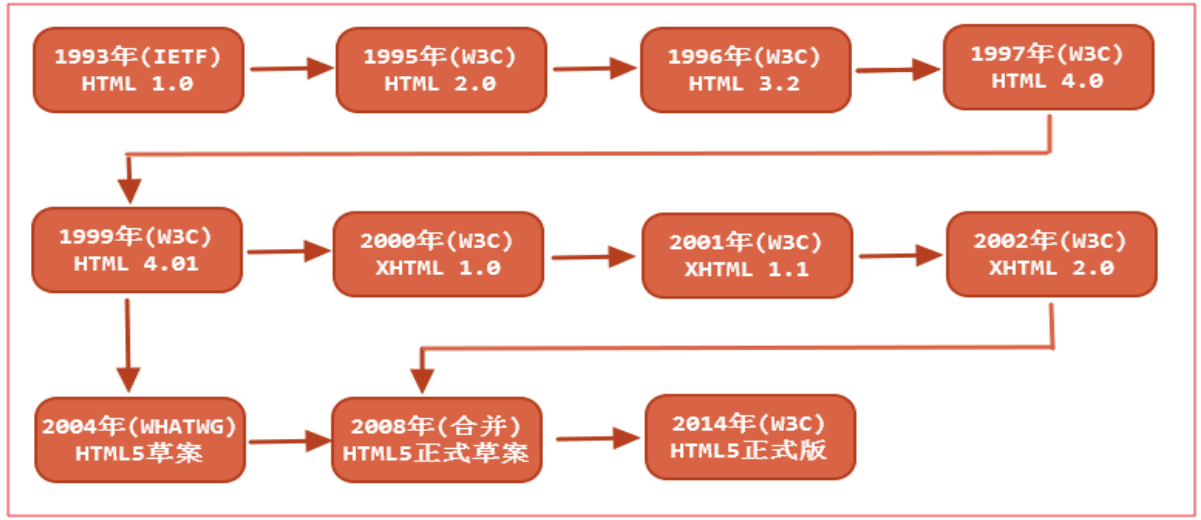
- Web 网页上的对象分为以下三类：
  - 静态对象与静态网页
    - 确定的文本、媒体信息
    - 如文本，表格，图片，图像和视频等多媒体类型的信息
    - 实现语言：标记语言如：HTML，XML，PHP等
    - 表达展现信息
    - 字体、颜色和布局等风格类型的信息
    - 实现语言：层叠样式表CSS
  - 动态对象与动态网页
    - 交互信息
    - 比如，用户注册信息、登录信息等
    - 实现语言：PHP/JSP等语言+MySQL等数据库
  - 链接
    - 超链接（HyperLinks）

超文本标记语言 HTML

- 超文本标记语言HTML中的Markup的意思就是“设置标记”。HTML定义了许多用于排版的命令（即标签）
- HTML把各种标签嵌入到万维网的页面中，就构成了所谓的HTML文档。HTML文档是一种可以用任何文本编辑器创建的ASCII码文件
- 仅当HTML文档是以.html或.htm为后缀时，浏览器才对此文档的各种标签进行解释
- 当浏览器从服务器读取HTML文档后，就按照HTML文档中的各种标签，根据浏览器所使用的显示器的尺寸和分辨率大小，重新进行排版并恢复出所读取的页面
- HTML规定了两种不同的链接设置方法
  - 远程链接：超链的终点是其他网点上的页面
  - 本地链接：超链指向本计算机中的某个文件

HTML 和 XML 对比

比较内容	HTML	XML
扩展性	不具有可扩展性，标记固定	元标记语言，可定义新的标记语言，标记可由用户定义
侧重点	侧重信息的表现形式	侧重结构化的信息描述
语法	不严格（嵌套、配对）	严格要求嵌套、配对严格按照DTD的要求
可读性和可维护性	难于阅读，难于维护	结构清晰，便于阅读和维护
数据本身与显示	数据与显示通常合在一起	数据与显示分离
重用性	低	高



CSS层叠样式的文档布局：为 HTML 文档定义页面布局

- CSS (Cascading Style Sheets) 是层叠样式表，它是一种样式表语言，用于为 HTML 文档定义页面布局。CSS 与 HTML 的区别是：
  - HTML 用于结构化内容
  - 而 CSS 则用于格式化结构化的内容

Web文档分类

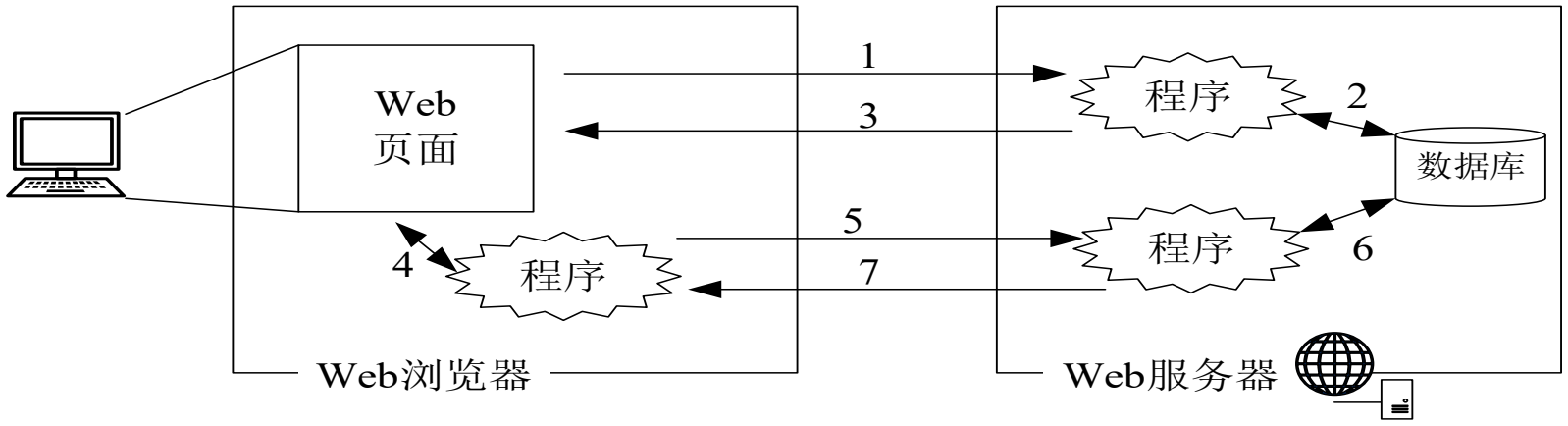
- 静态文档
  - 文档创作完毕后存放在万维网服务器中，在被用户浏览的过程中，内容不会改变
  - 客户利用浏览器访问该文档时,其副本被传送到客户，客户用浏览程序解释、显示这个文档
- 动态文档
  - 动态文档是能够提供一种连续更新屏幕内容的技术,这种技术把创建文档的工作移到浏览器端进行
  - 当浏览器请求一个活动文档时，服务器就返回一段程序副本在浏览器端运行。活动文档程序可与用户直接交互，并可连续地更新屏幕的显示内容

动态Web页面的典型技术：AJAX

- AJAX：Asynchronous JavaScript and XML
  - 超文本标记语言HTML+CSS: 用于Web网页内容的显示
  - 文档对象模型DOM（Document Object Model）：文档对象模型 (DOM) 是HTML和XML文档的编程接口，其本质上是页面的API，允许程序读取和操作页面的内容、结构和样式；采用树形结构组织
  - 扩展标记语言XML（eXtensible Markup Language）：用于程序与服务器交换应用数据
  - 异步式工作方式：用于发送和检索XML数据
  - JavaScript：用于将以上所有功能进行组合，并协同工作

客户浏览器端活动文档的生成过程

- 浏览器给定一个位置请求
- Web服务器上运行一个程序，该程序查询一个数据库，以便生成相应的页面
- 该页面返回给Web浏览器显示
- 浏览器中运行的程序允许用户发现一条路线，它可以更新页面，跟随用户的指示缩小或放大
- 浏览器中运行的程序将给Web服务器发送一个请求
- Web服务器中运行的程序从数据库中检索出更多的信息
- Web服务器中运行的程序返回一个响应。然后，该程序将继续更新页面（第4步）

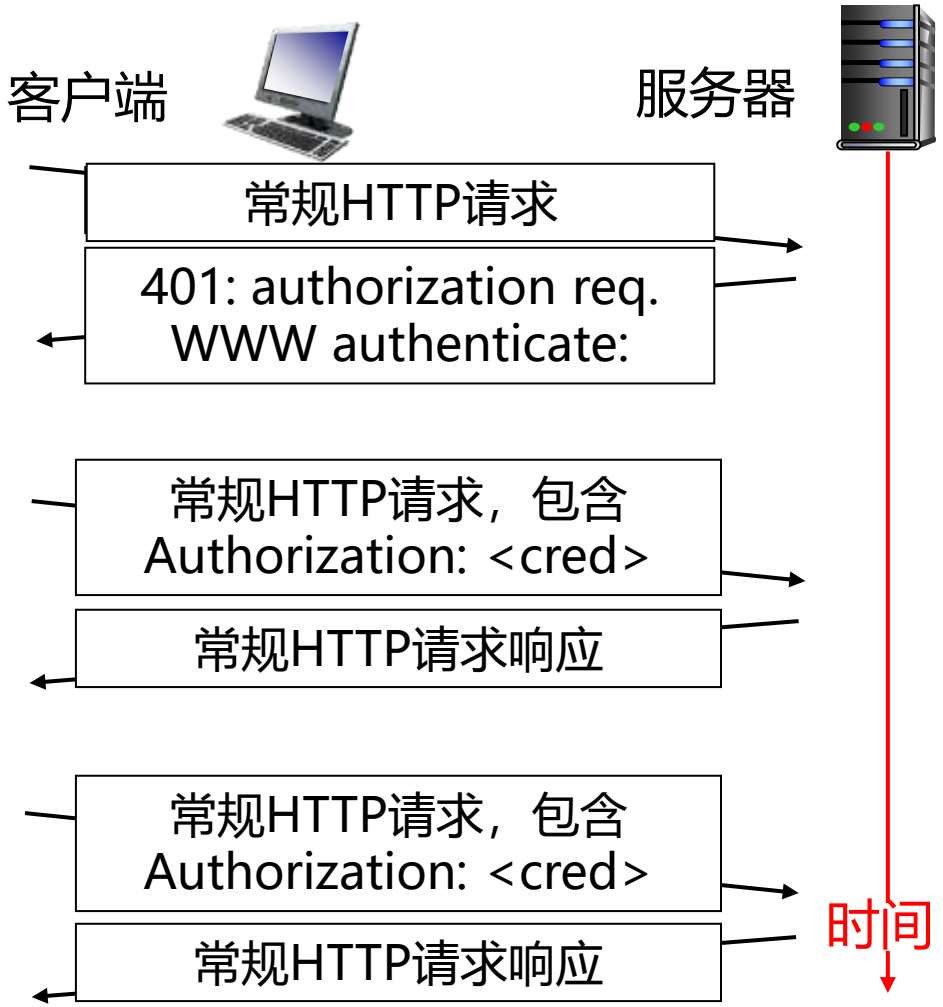


Web安全与隐私：Web访问安全

- 并非所有Web页都会向公众开放
- Web服务器可以限定客户端访问的IP地址空间，比如限制只向公司内部员工开放
- Apache服务器将设置限制访问规则的文件.htaccess放置在被限制访问的页面所在的目录，客户端访问时进行规则匹配
- 认证方法：在浏览器客户端的HTTP请求中给出“用户名-密码”，服务器进行HTTP验证

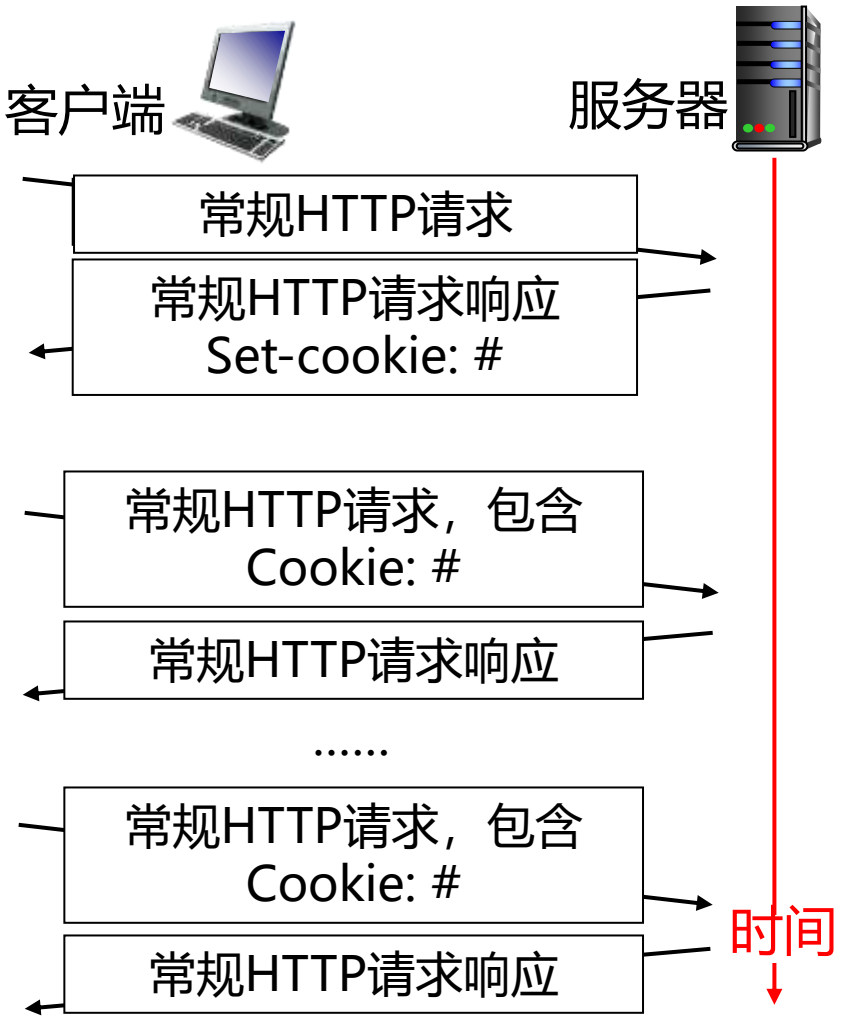
Web安全与隐私：Web访问安全

- 无状态：客户端需要在每个请求中携带认证信息
- 认证方法：通常在HTTP请求中使用“用户名-密码”
- 每个请求头中包含关键字authorization:
- 如果请求头中无authorization:，则服务器拒绝访问，并在响应头中包含 WWW authenticate:

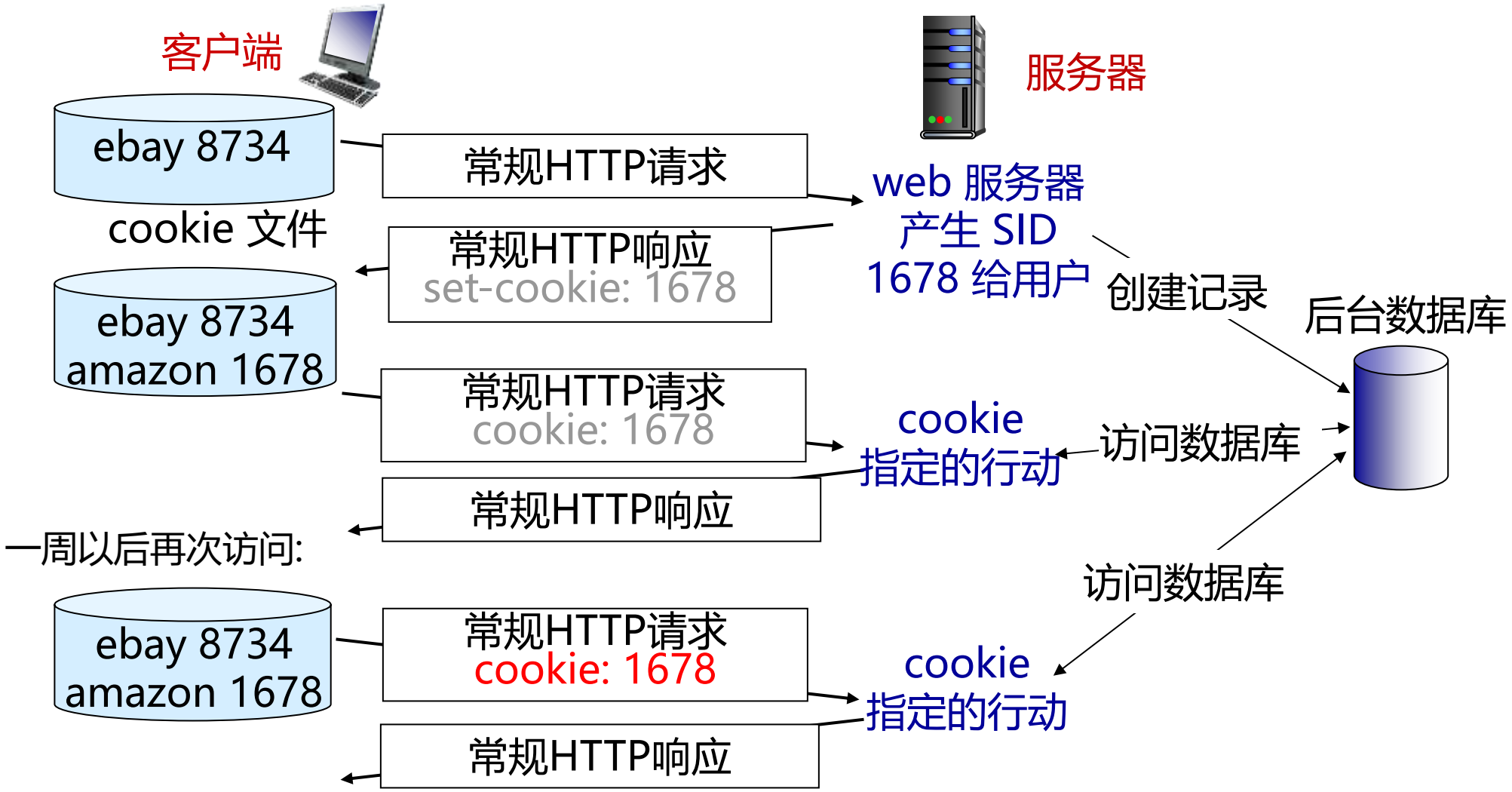


Web安全与隐私：Cookie

- HTTP无状态协议，服务器用cookies保持用户状态
  - HTTP在响应的首部行里使用一个关键字头set-cookie：选择的cookie号具有唯一性
  - 后继的HTTP请求中使用服务器响应分配的cookie：
  - Cookie文件保存在用户主机中，内容是服务器返回的一些附加信息，由用户主机中的浏览器管理
  - Web服务器建立后端数据库，记录用户信息，cookie作为关键字，例如：
    - Set-Cookie: SID=31d4d96e407aad42; Path=/
    - Domain=example.com
    - Cookie: SID=31d4d96e407aad42



Web安全与隐私：Cookie



Web安全与隐私：Cookie

- Cookies一般包含5个字段
  - 域指明Cookie来自何方，每个域为每个客户分配Cookie有数量限制
  - 路径标明服务器的文件树中哪些部分可以使用该Cookie：
  - 内容采用“名字=值”的形式，是Cookie存放内容的地方，可以达到4K容量，内容只是字符串，不是可执行程序
  - 安全指示浏览器只向使用安全传输连接的服务器返回Cookie

域	路径	内容	过期	安全
Toms-casino.com	/	CustomerID=297793521	15-10-1017:00	是
Jills-store.com	/	Cart=1-00501;1-07031;2-13721	11-1-1114:22	不
Aportal.com	/	Prefs=Stk:CSCO+ORCL;Spt:Jets	31-12-2023:59	不
Sneaky.com	/	UserID=4627239101	31-12-1923:59	不

Web安全与隐私：Cookie

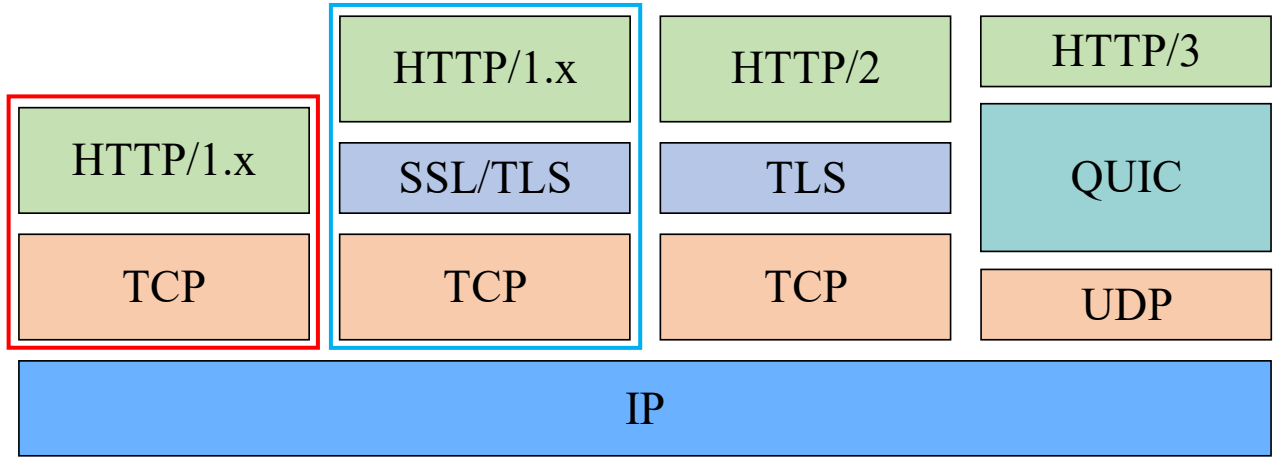
- Cookie技术是把双刃剑，能分析用户喜好，向用户进行个性化推荐
  - 用Cookie在某网站标识用户信息，查找用户以前浏览网站记录
  - 用Cookie记录用户购物清单
  - 用Cookie可以保存4K内容，跟踪用户浏览网站的喜好
  - 用Cookie跨站点跟踪用户点击广告
- Cookie技术是把双刃剑，也能跟踪用户网络浏览痕迹，泄露用户隐私
  - Cookie跟踪用户以前浏览过哪些网站，跟踪用户频繁浏览哪类网站
  - Cookie收集用户信息，用户网络交互时关注的关键词
- Cookie容易嵌入间谍程序，这是个误区，Cookie文件保存的只是文本串，没有可执行程序
  - 用户可以设置浏览器限制使用Cookie

HTTP 协议

- 为了使超文本的链接能够高效率地完成，需要用超文本传输协议HTTP（ HyperText Transfer Protocol）来传送一切必须的信息。
- HTTP在传输层使用TCP协议，缺省使用TCP的80端口。
- HTTP为无状态协议，服务器端不保留之前请求的状态信息。
- 从层次的角度看，HTTP 是面向事务的 (transaction-oriented) 应用层协议，它是万维网上能够可靠地交换文件（包括文本、声音、图像等各种多媒体文件）的重要基础。
- HTTP标准
  - HTTP/1.0: RFC 1945（1996年）
  - HTTP/1.1: RFC 2616（1999年）
  - HTTP/2: RFC 7540（2015年）、RFC 8740（2020年）

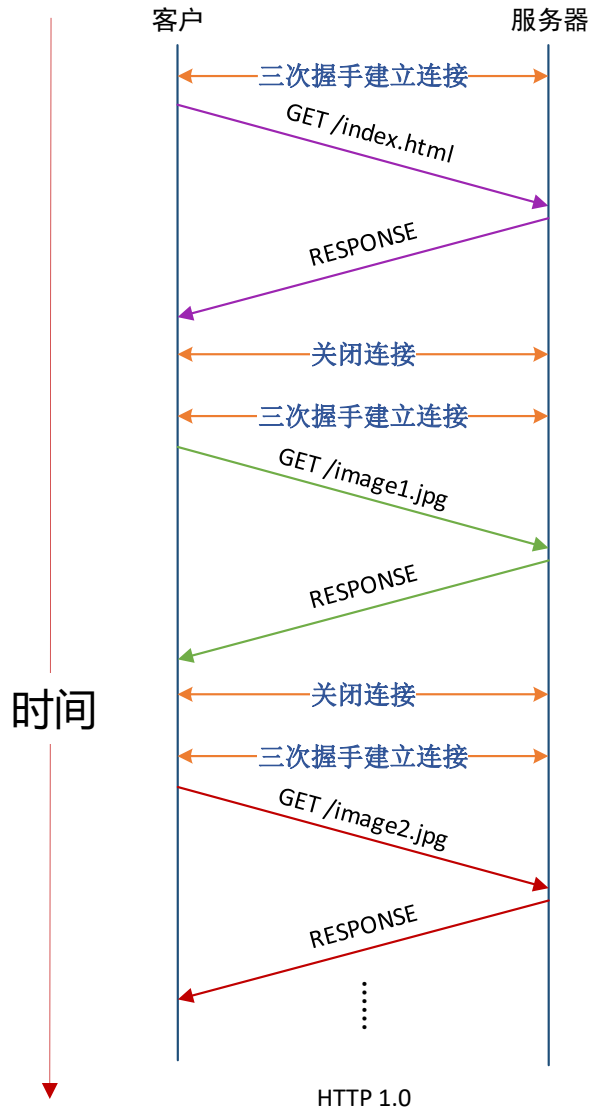
HTTP 发展现状

- HTTP/1.0（1996）
  - 无状态，非持久连接
- 与HTTP/1.1（1999）
  - 支持长连接和流水线机制
  - 缓存策略优化、部分资源请求及断点续传
- HTTPS：HTTP+TLS（2008）
  - 增加SSL/TLS（TLS 1.2）层，在TCP之上提供安全机制
- HTTP/2.0（2015、2020）
  - 目标：提高带宽利用率、降低延迟
  - 增加二进制格式、TCP多路复用、头压缩、服务端推送等功能



HTTP1.0 协议执行过程

- 假设用户输入URL：
  - http://www.nankai.edu.cn/computer/index.html
  - 如该页面包含2幅jpg图像
    - 需要**执行三次完整的连接过程**（即：连接的建立，数据传输，连接终止），**包含三次TCP过程（含三次HTTP过程）**
    - 三次握手连接建立
    - Request （Get /index.html）
    - Respone
    - 关闭连接
    - ……
  - 如若含100副图像呢?



HTTP 协议执行过程

- HTTP 报文通常都使用 TCP 连接传送
- 每个万维网网点都有一个服务器进程，它不断地监听 TCP 的端口 80，以便发现是否有浏览器向它发出连接建立请求
- 一旦监听到连接建立请求并建立了 TCP 连接之后，浏览器就向万维网服务器发出浏览某个页面的请求，服务器接着就返回所请求的页面作为响应
- 在浏览器和服务器之间的请求和响应的交互，必须按照规定的格式和遵循一定的规则。这些格式和规则就是超文本传送协议 HTTP
- HTTP 规定在 HTTP 客户与 服务器之间的每次交互，都由一个 ASCII 码串构成的请求和一个类似的通用互联网扩充，即“类 MIME (MIME-like)”的响应组成
- 最后，TCP 连接就被释放了

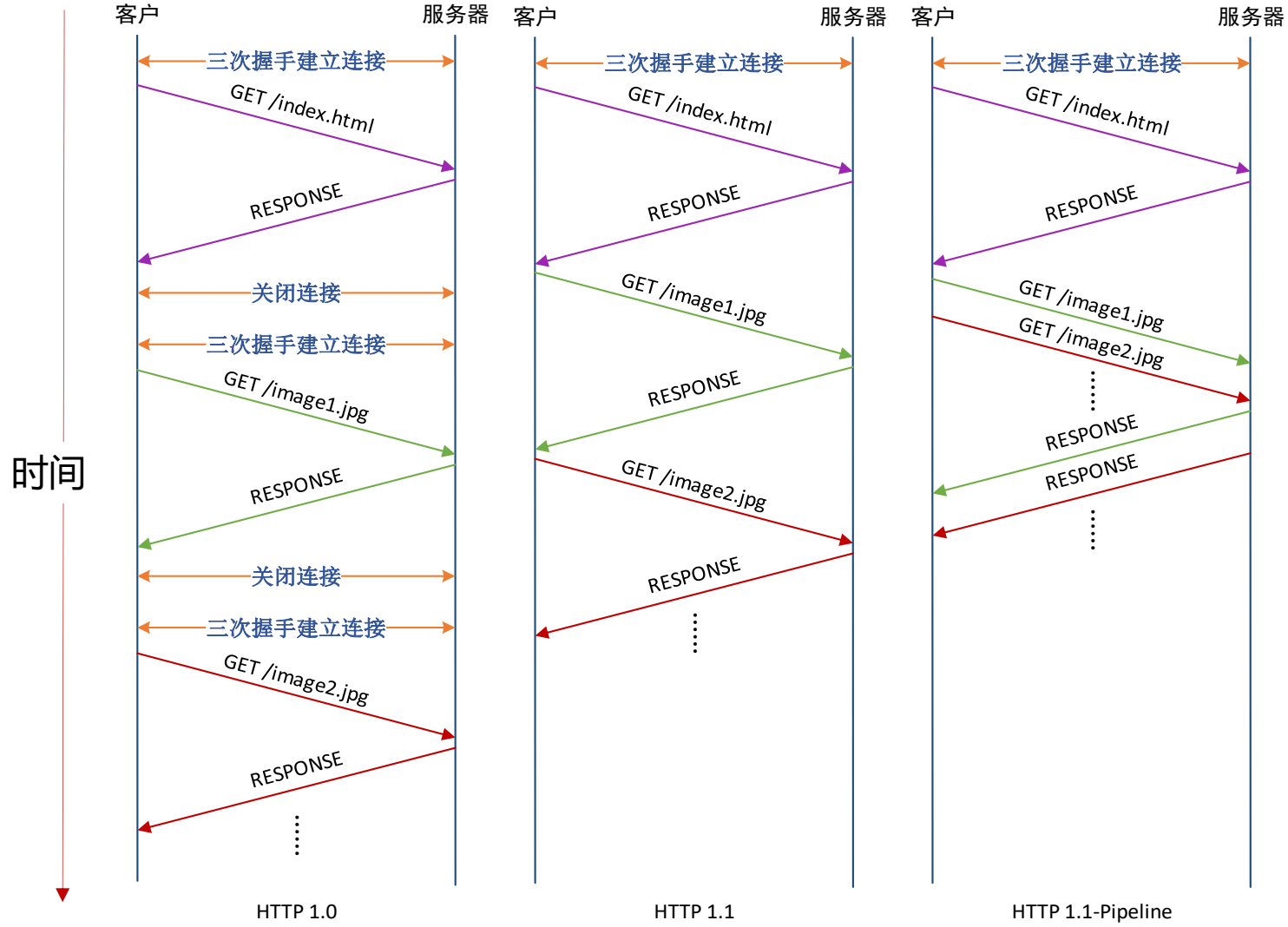
HTTP1.x 协议的持久连接和非持久连接

- 非持久连接
  - HTTP/1.0缺省为非持久连接
    - 服务器接收请求、响应、关闭TCP连接
  - 获取每个对象需要两阶段
    - 建立TCP连接
    - 对象请求和传输
  - 每次连接需要经历TCP慢启动阶段

HTTP1.x 协议的持久连接和非持久连接

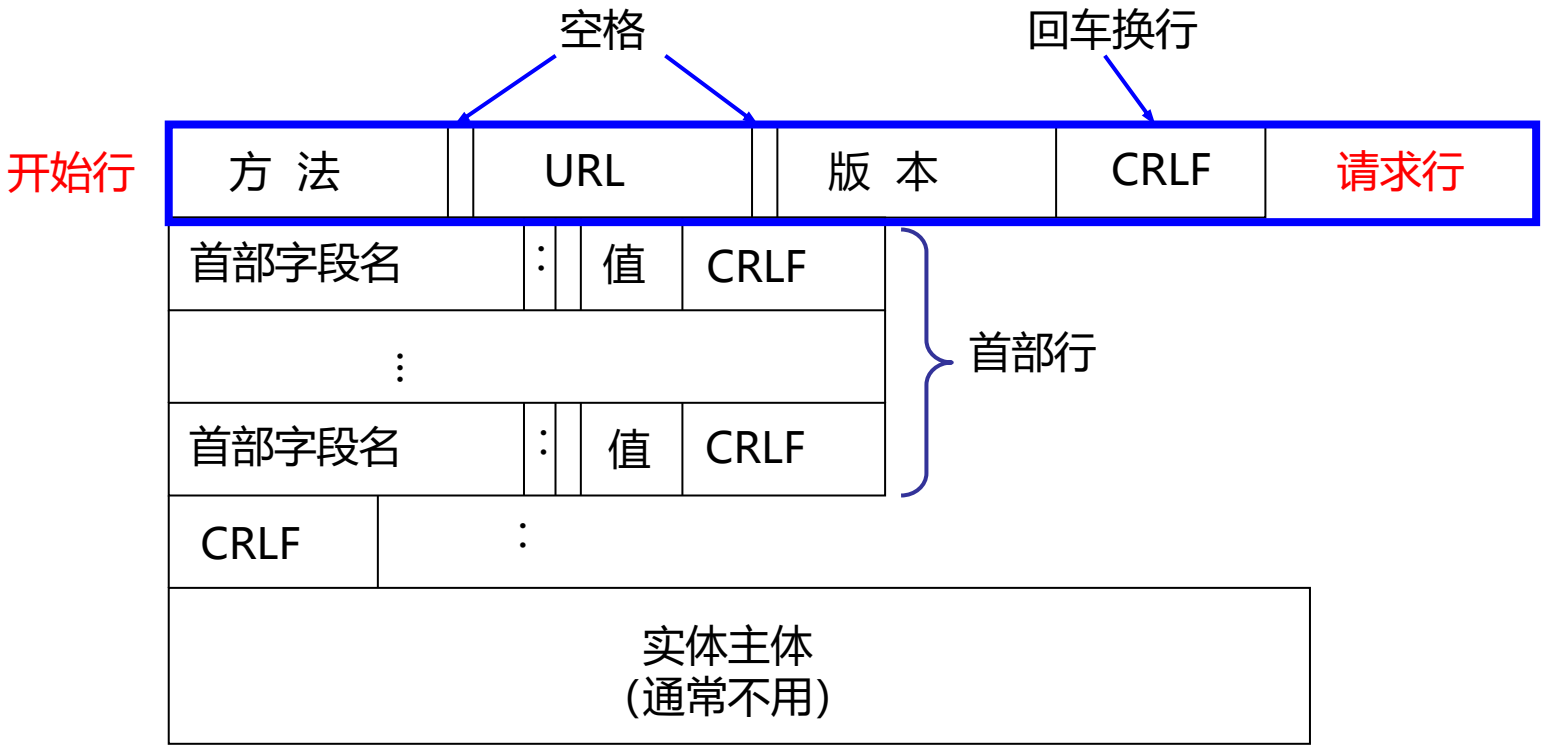
- 持久连接
  - HTTP/1.1缺省为持久连接
    - 在相同的TCP连接上，服务器接收请求、响应；再接收请求、响应；响应后保持连接
  - HTTP/1.1支持流水线机制
    - 需要按序响应
  - 经历较少的慢启动过程，减少往返时间
    - 降低响应时间

HTTP1.x 协议的比较



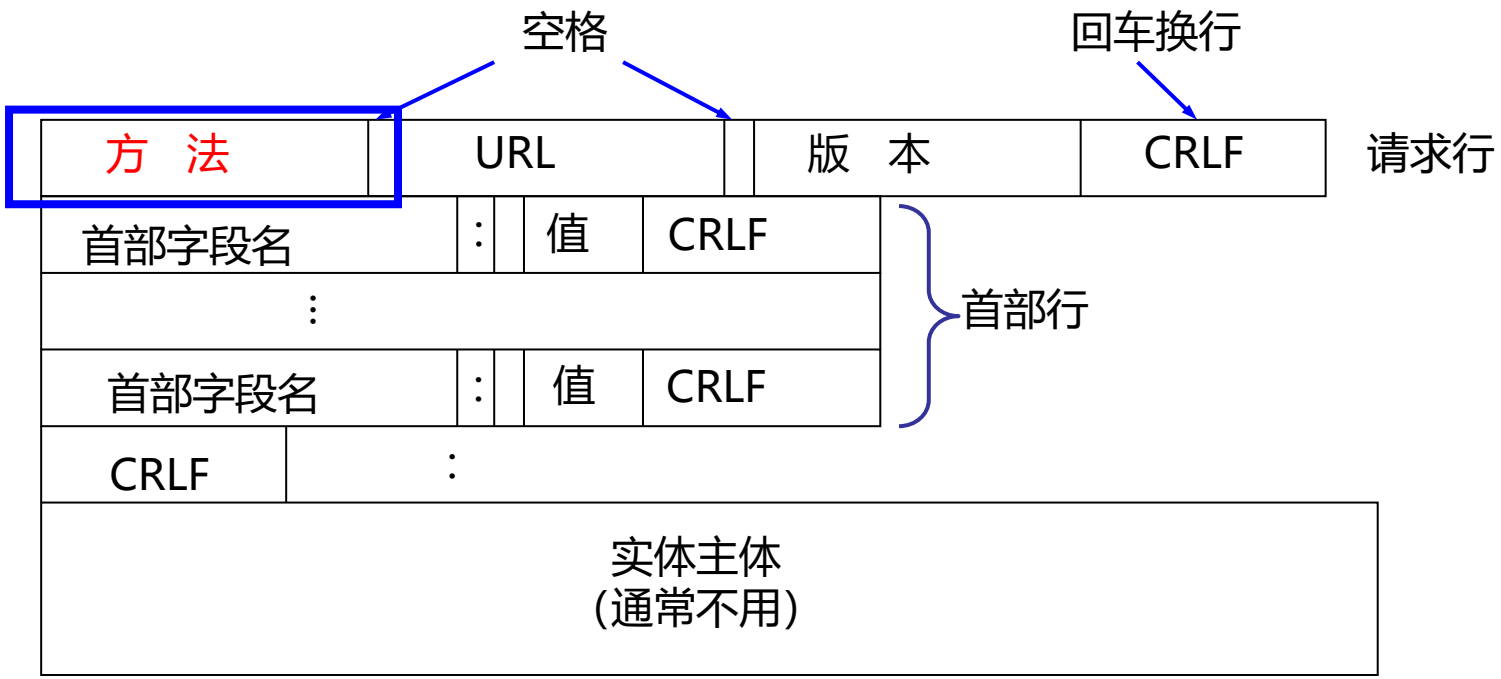
HTTP报文结构：请求报文

- 报文由三个部分组成，即开始行、首部行和实体主体
- 在请求报文中，开始行就是请求行



HTTP报文结构：方法

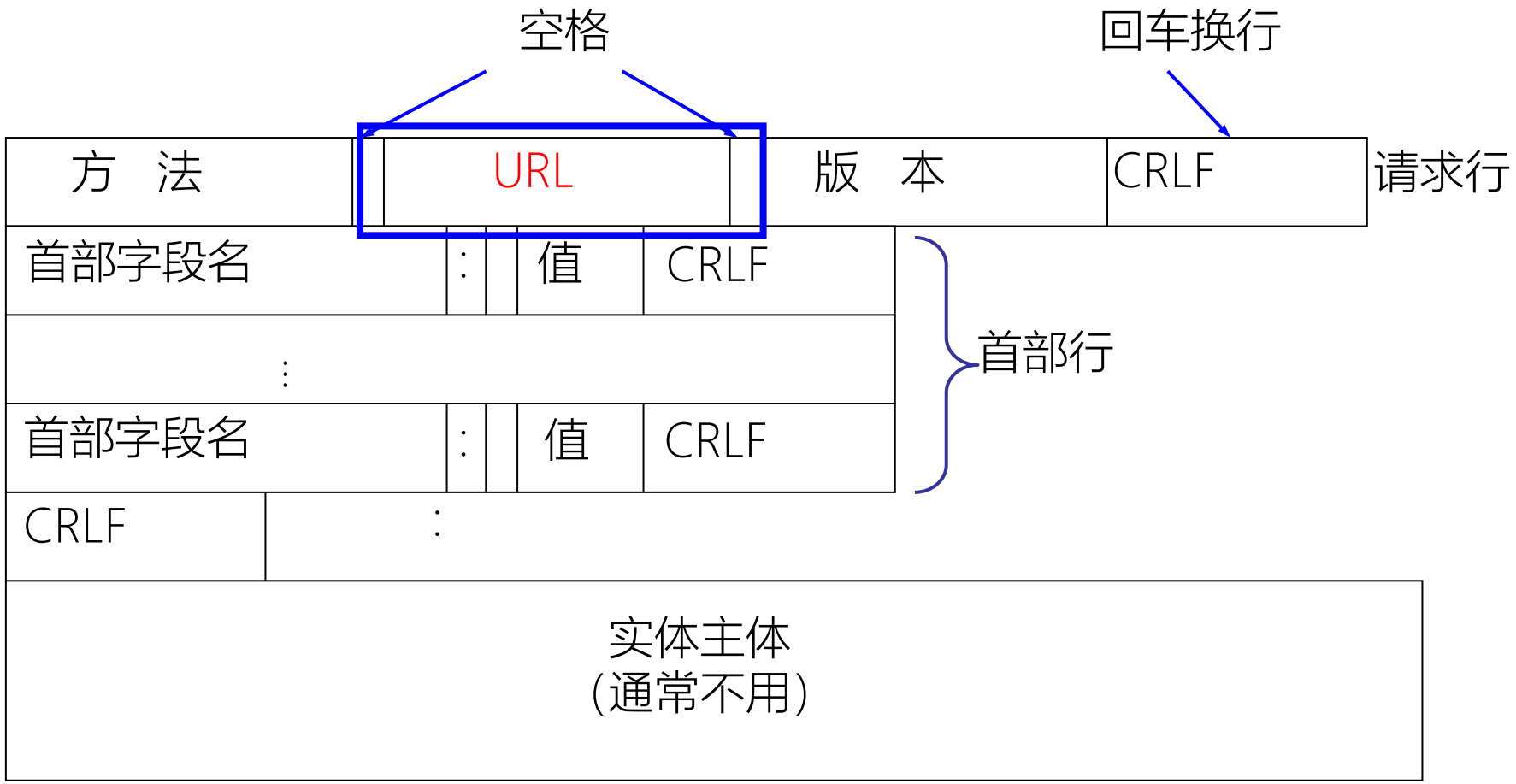
- 方法是对所请求的对象进行的操作，实际上也就是一些命令
- 请求报文的类型是由它所采用的方法决定的



HTTP报文结构：具体方法

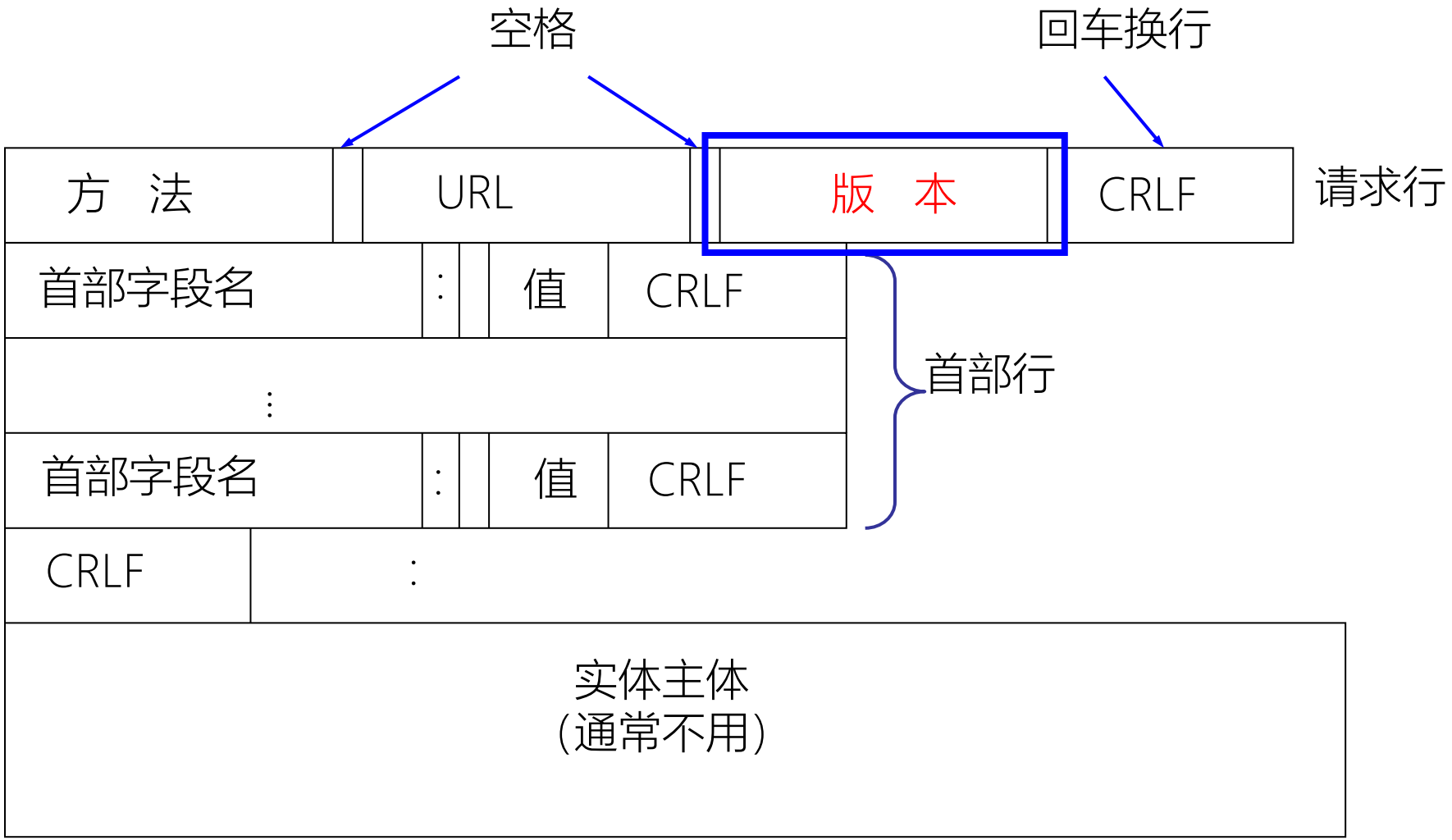
方法或操作	含义
OPTION	请求一些选项的信息
GET	请求读取由 URL所标志的信息
HEAD	请求读取由 URL所标志的信息的首部
POST	给添加信息（例如，注释）
PUT	在指明的 URL下存储一个文档
DELETE	删除指明的 URL所标志的资源
TRACE	用来进行环回测试的请求报文
CONNECT	用于代理服务器

HTTP报文结构：URL

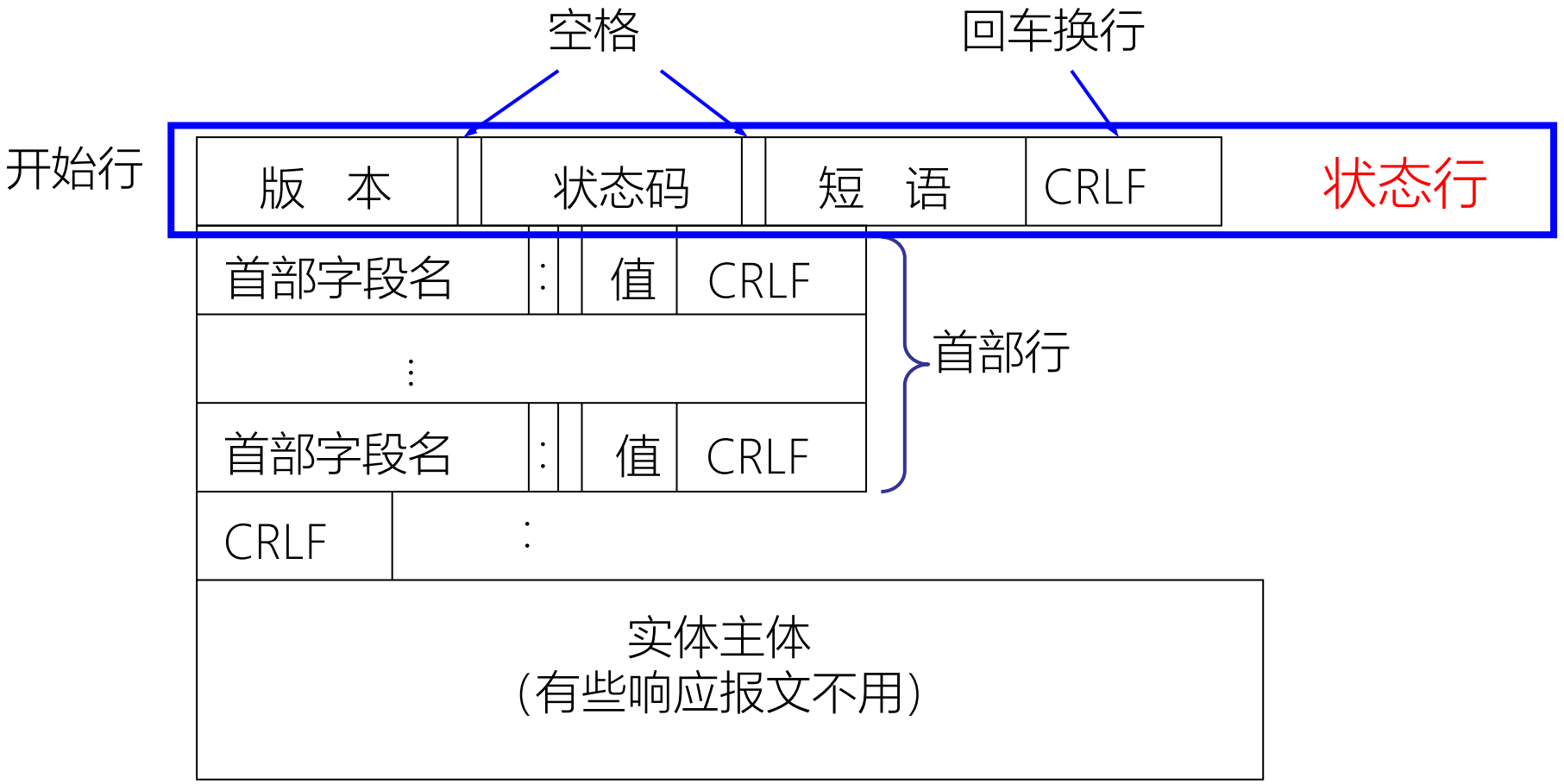


- “URL”是所请求资源的URL

HTTP报文结构：版本


空格
回车换行
方　法
URL
版　本
CRLF
请求行
首部字段名
:
值
CRLF
首部行
首部字段名
:
值
CRLF
CRLF
:
实体主体
（通常不用）

HTTP报文结构：响应报文

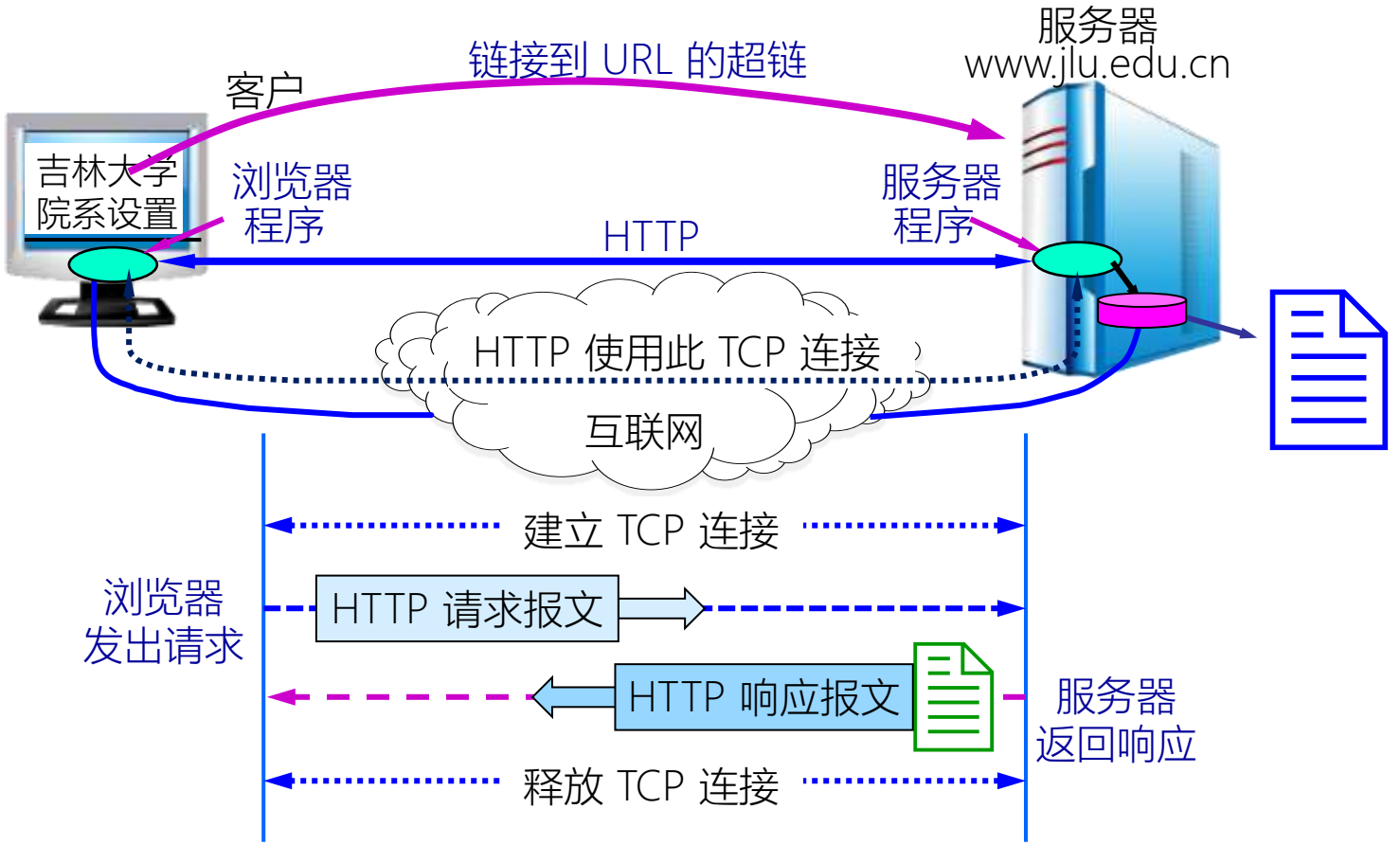


HTTP响应报文：状态码

- 状态码都是三位数字
  - 1xx 表示通知信息，如请求收到了或正在进行处理。
  - 2xx 表示成功，如接受或知道了。
  - 3xx 表示重定向，表示要完成请求还必须采取进一步的行动。
  - 4xx 表示客户的差错，如请求中有错误的语法或不能完成。
  - 5xx 表示服务器的差错，如服务器失效无法完成请求。

- 典型的状态码
  - 200 OK
    - 请求成功，被请求的对象包含在该响应的数据部分
  - 301 Moved Permanently
    - 请求的对象被移走，新的位置在响应中通过Location: 给出
  - 400 Bad Request
    - 服务器不能解释请求报文
  - 404 Not Found
    - 服务器中找不到请求的文档
  - 505 HTTP Version Not Supported
    - 服务器不支持相应的HTTP版本

HTTP 协议工作过程：协议流程示例



HTTP 协议工作过程：协议流程示例

- 用户在某处链接处点击鼠标后所发生的事件：

(1) 浏览器分析超链接指向页面的 URL

(2) 浏览器向 DNS 请求解析 www.jlu.edu.cn 的 IP 地址

(3) 域名系统 DNS 解析出清华大学服务器的 IP 地址

(4) 浏览器与服务器建立 TCP 连接

(5) 浏览器发出取文件命令：

GET /jgsz/yxsz.htm

(6) 服务器给出响应，把文件 yxsz.htm发给浏览器

(7) TCP 连接释放

(8) 浏览器显示“吉林大学院系设置”文件 yxsz.htm中的所有文本

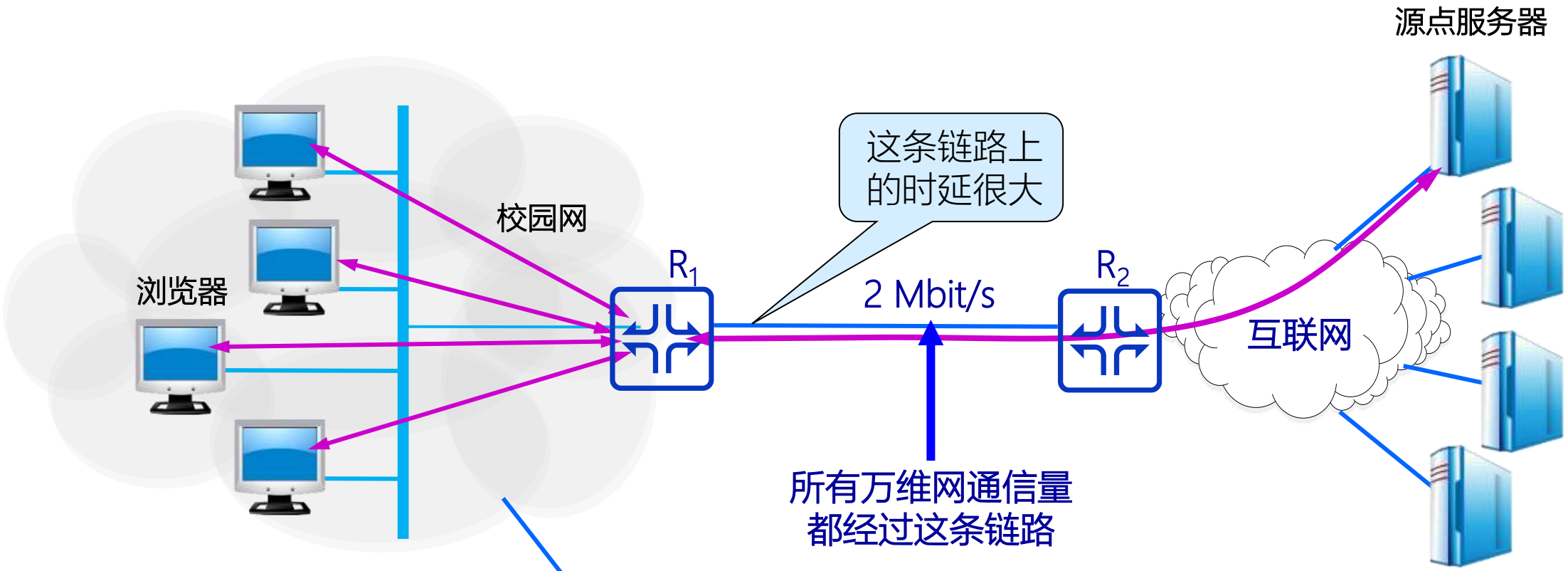
HTTP 协议的主要特点

- HTTP 使用了面向连接的 TCP 作为运输层协议，保证了数据的可靠传输
- HTTP 协议本身也是无连接的，虽然它使用了面向连接的 TCP 向上提供的服务
- HTTP 是面向事务的客户服务器协议
- HTTP 1.0 协议是无状态的 (stateless)

代理服务器

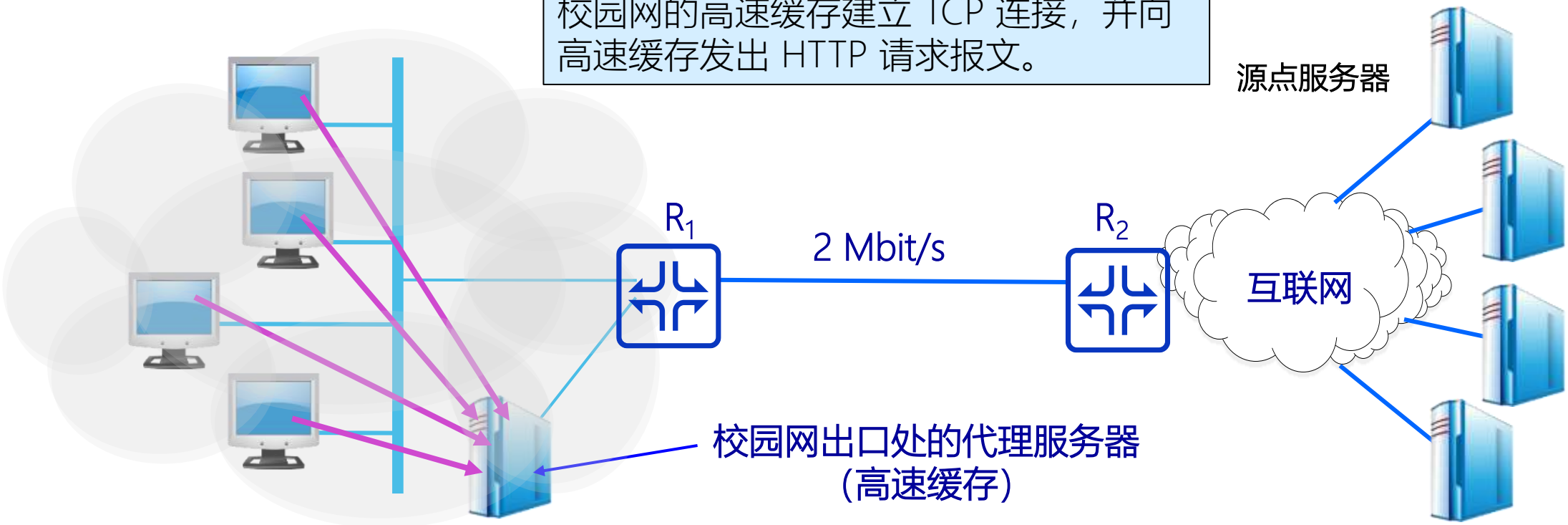
- 代理服务器 (proxy server) 又称为万维网高速缓存 (Web cache)，它代理浏览器向URL服务器发出 HTTP 请求
- 万维网高速缓存把最近的一些请求和响应暂存在本地磁盘中
- 当与暂时存放的请求相同的新请求到达时，万维网高速缓存就把暂存的响应发送出去，而不需要按 URL 的地址再去互联网访问该资源

代理服务器工作原理


源点服务器
这条链路上
的时延很大
校园网
浏览器
R_1
R_2
2 Mbit/s
互联网
所有万维网通信量
都经过这条链路
校园网出口处没有使用代理服务器的情况

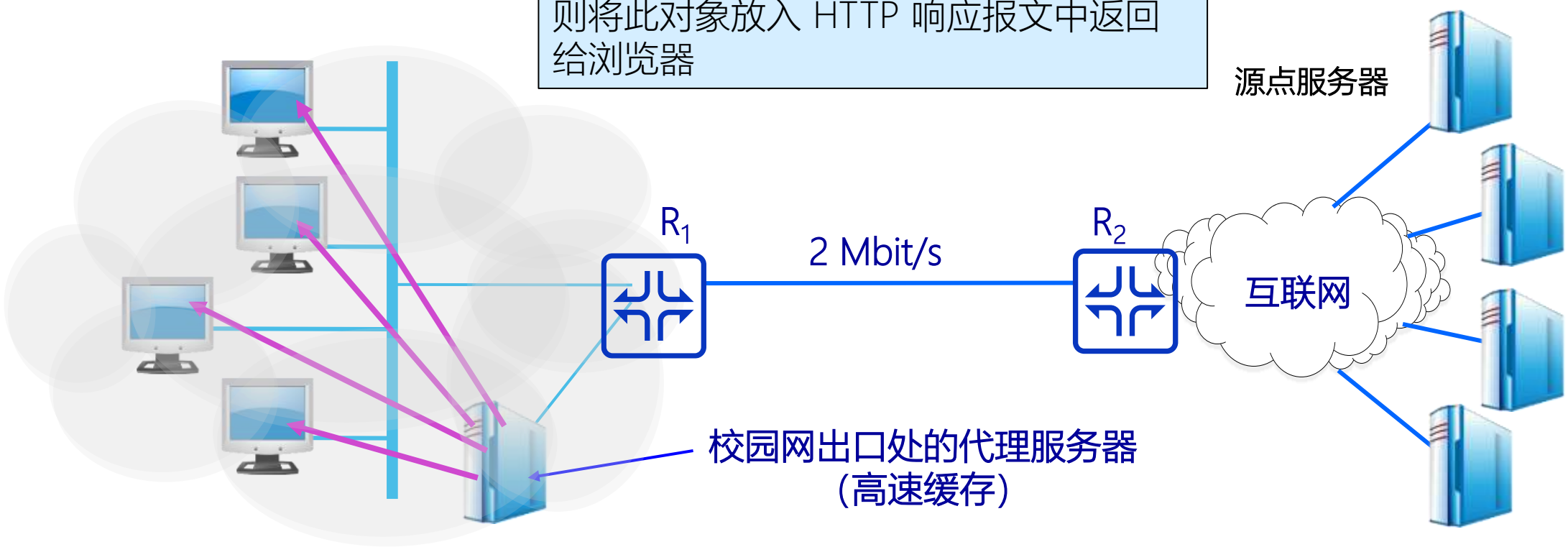
代理服务器工作原理

浏览器访问互联网的服务器时，要先与校园网的高速缓存建立 TCP 连接，并向高速缓存发出 HTTP 请求报文。



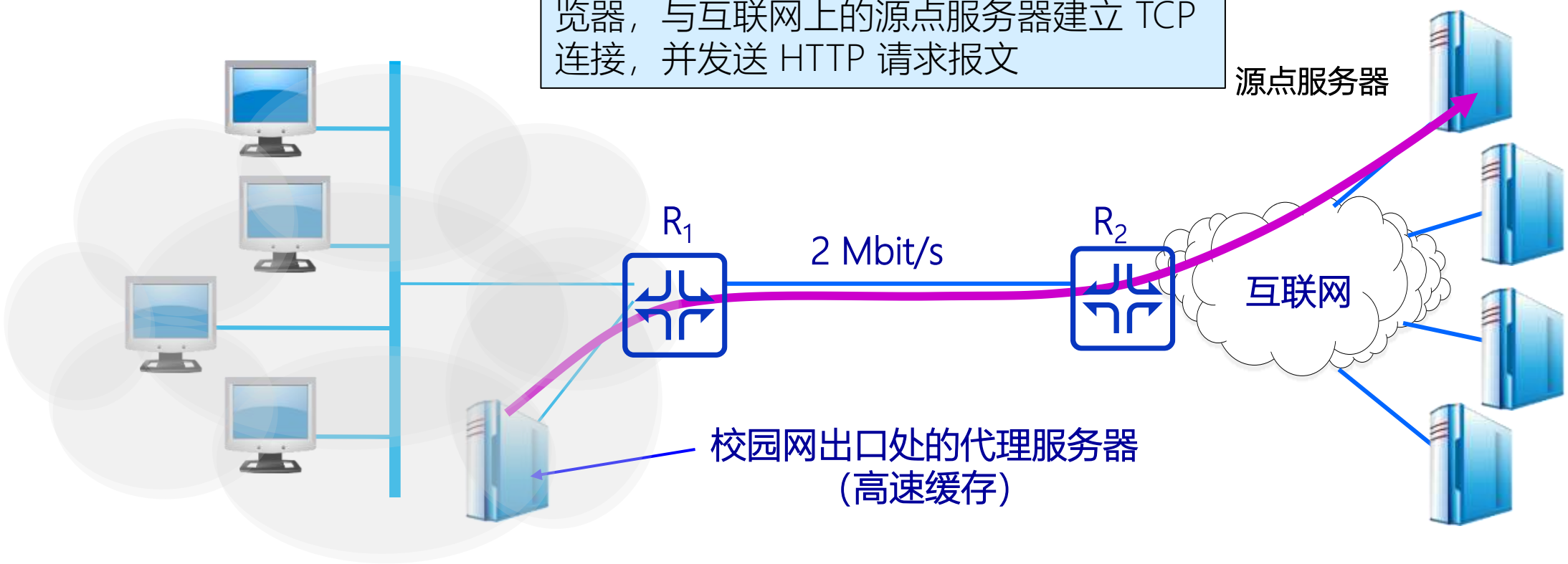
代理服务器工作原理

若高速缓存已经存放了所请求的对象，则将此对象放入 HTTP 响应报文中返回给浏览器



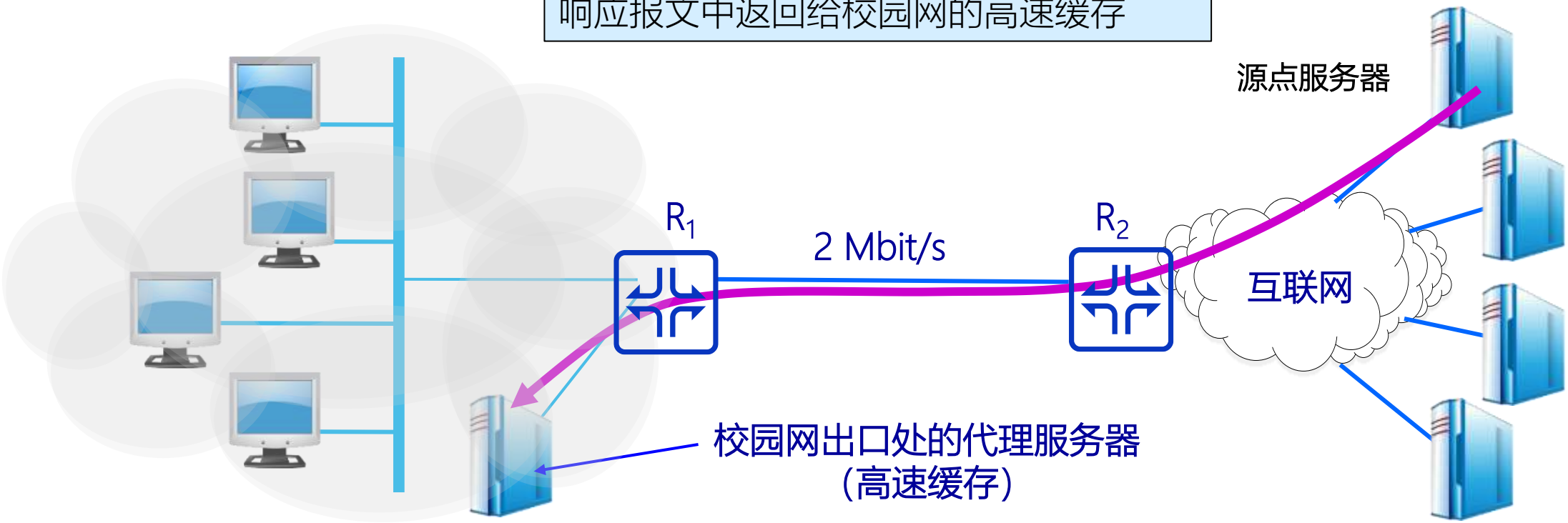
代理服务器工作原理

否则，高速缓存就代表发出请求的用户浏览器，与互联网上的源点服务器建立 TCP 连接，并发送 HTTP 请求报文



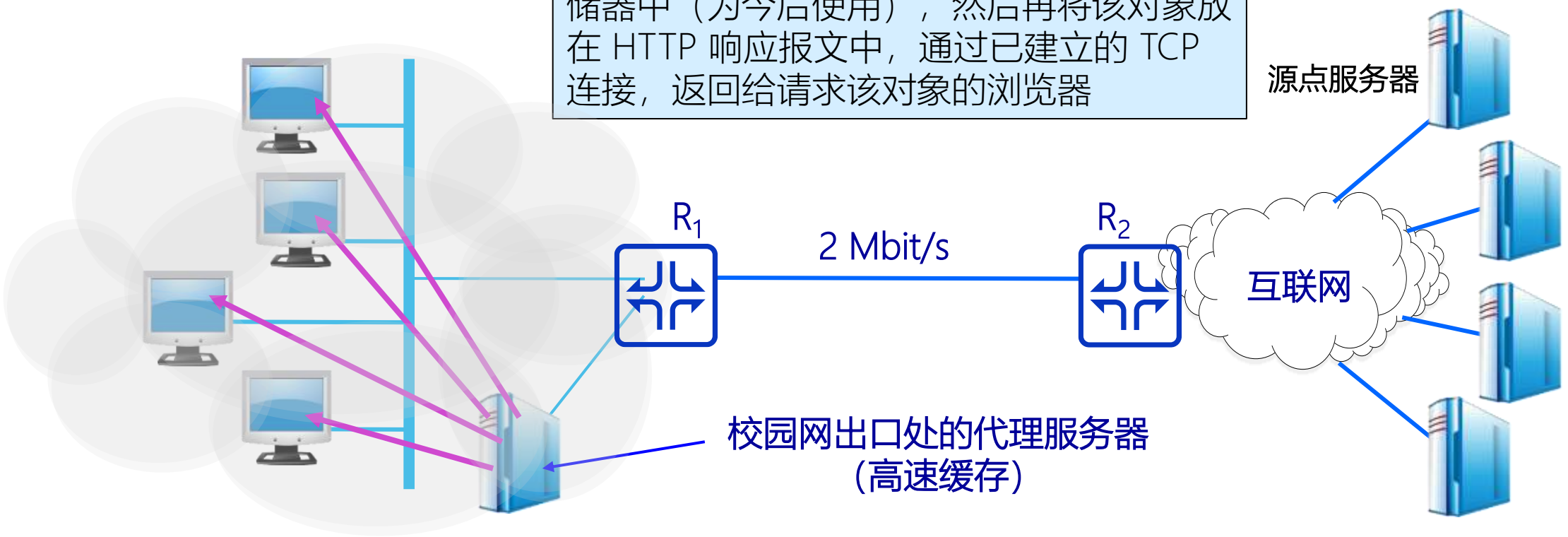
代理服务器工作原理

源点服务器将所请求的对象放在 HTTP 响应报文中返回给校园网的高速缓存



代理服务器工作原理

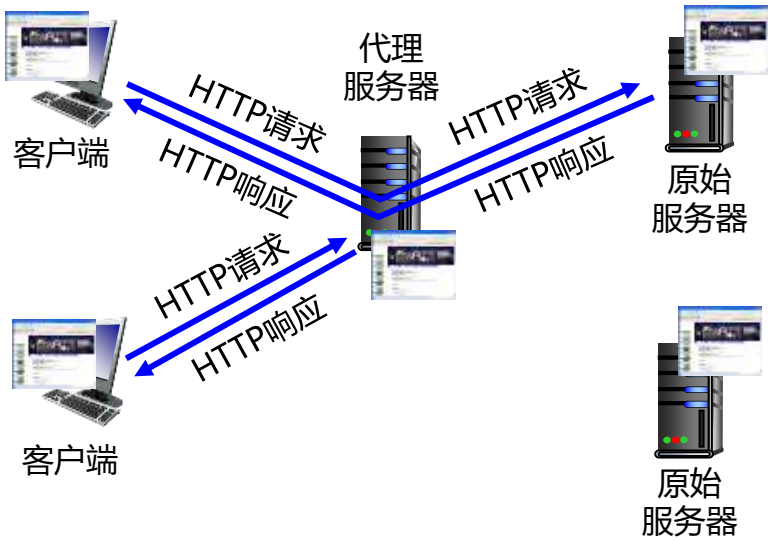
高速缓存收到此对象后，先复制在其本地存储器中（为今后使用），然后再将该对象放在 HTTP 响应报文中，通过已建立的 TCP 连接，返回给请求该对象的浏览器



- 使用代理服务器的高速缓存，可：
  - 减少访问互联网服务器的时延
  - 节省网络带宽

代理服务器的主要技术：Web缓存

- Web缓存的目标：代理服务器缓存已访问过的Web页副本，满足用户浏览器从代理服务器提取Web页，尽量减少原始服务器参与
  - 设置用户浏览器，通过代理服务器进行Web访问
  - 浏览器将所有HTTP请求发送到代理服务器
    - 如果缓存中有被请求的对象，则直接返回对象
    - 否则，代理服务器向原始服务器请求对象，再将对象返回给客户端



降低时延、减少网络流量

7.6 流媒体应用

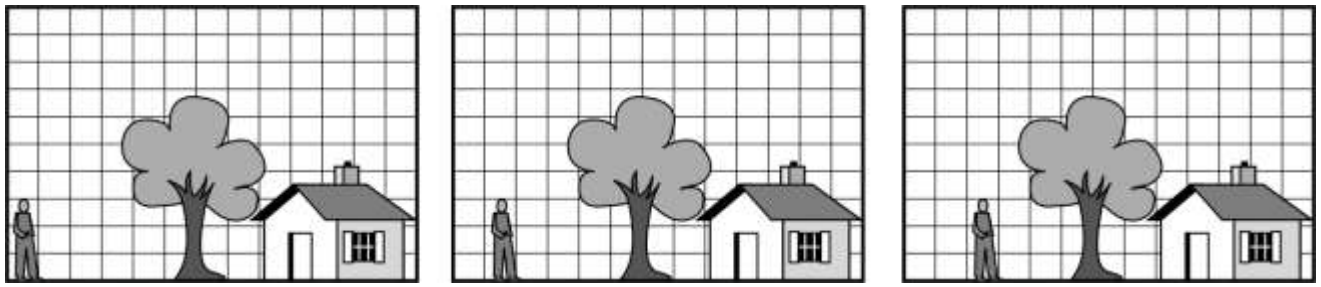
- 常见的流媒体服务：
  - 点播：提前录制好，边下载边播放（起始时延<10s；类VCR操作（例如拖动进度条）<1~2s)
  - 直播：边录制边上传，边下载边播放（大规模直播往往有数秒的时延）
  - 实时交互 ： 双方或多方实时交互式通信（时延<400ms可接受，VR则需要<25ms)

流媒体概述

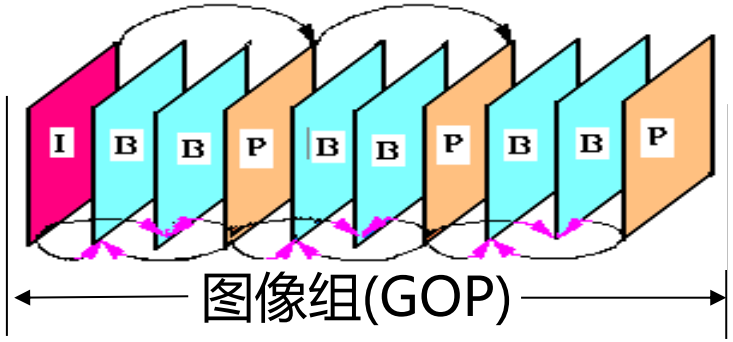
- 流媒体概念
  - 连续媒体（音视频）经压缩编码、数据打包后，经过网络发送给接收方
  - 接收方对数据进行重组、解码和播放
- 流媒体的特性
  - 端到端时延约束
  - 时序性约束：流媒体数据必须按照一定的顺序连续播放
  - 具有一定程度的容错性：丢失部分数据包也可完成基本功能
- 流媒体面临的挑战
  - 约束条件：网络特性（带宽有限、动态变化、延迟与抖动、丢失、异构性）
  - 目标：流媒体服务质量要素（画质、启动延迟、平滑、交互性）
  - 如何在“尽力服务”的网络传输条件下获得良好的视频质量？期望在点播、直播、会议、连麦、云游戏、VR、……场景下，高清、低延迟、不卡顿

数字音视频与编码

- MPEG视频压缩
  - 摄像机固定不变，演员慢慢走来走去：从前一帧中减去当前帧，对两帧之差运行JPEG算法
  - 摄像机在推拉移动：需要某种方法来补偿这种运动，这正是MPEG擅长的
  - MPEG的输出包括3类帧
  - 帧内编码帧（I帧）：包含了压缩的静止图片（帧内编码，用JPEG来压缩静止图像）
  - 预测帧（P帧）：是与前一帧的逐块差值（帧间编码，消除跨帧的冗余度）
  - 双向帧（B帧）：是与前一帧和后一帧的逐块差值（帧间编码）



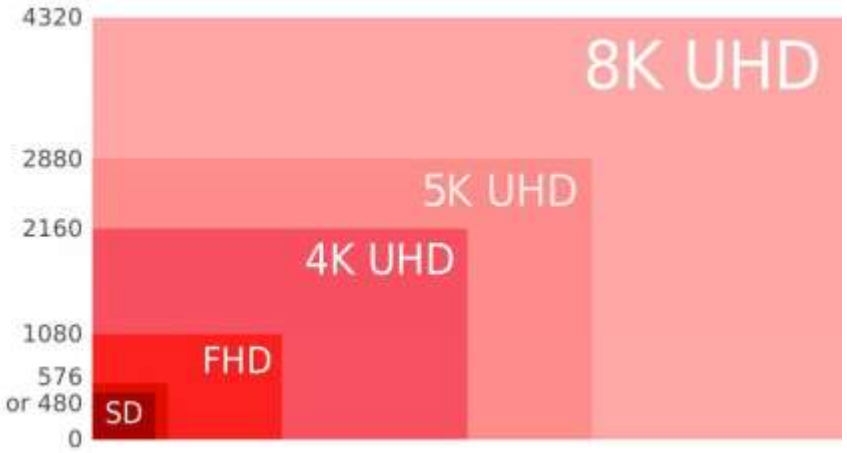
视频连续帧之间的差异往往较小，可将差值进行编码



一组连续的IPB画面
没有I帧，P帧和B帧就无法解码

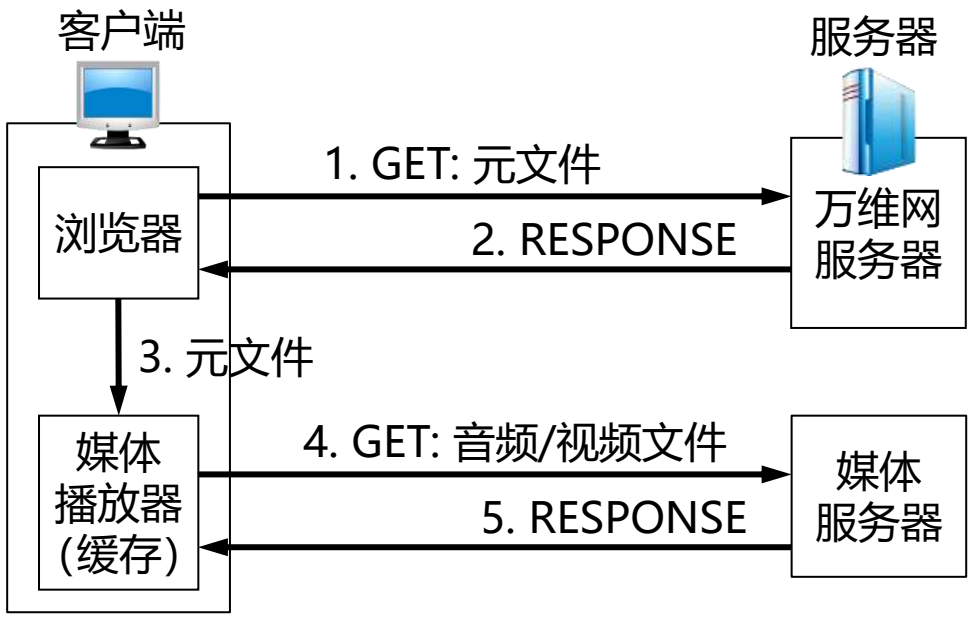
数字音视频与编码

- I帧必须周期性地出现在媒体流中
  - 流媒体直播中，后加入的观众需要收到I帧才能成功解码
  - 如果任意一帧发生了接收错误，则后续非I帧无法解码（由于B/P帧依赖损坏的帧）
  - 快进或者回退到某位置时，解码器需要从该位置前面的I帧开始计算
  - 常见帧速率：24帧/秒、30帧/秒、60帧/秒（VR视频）
  - 常见分辨率：标清SD 480p/576p、高清HD 720p、全高清FHD 1080p、4K超高清 2160p、8K超高清 4320p



流式存储媒体

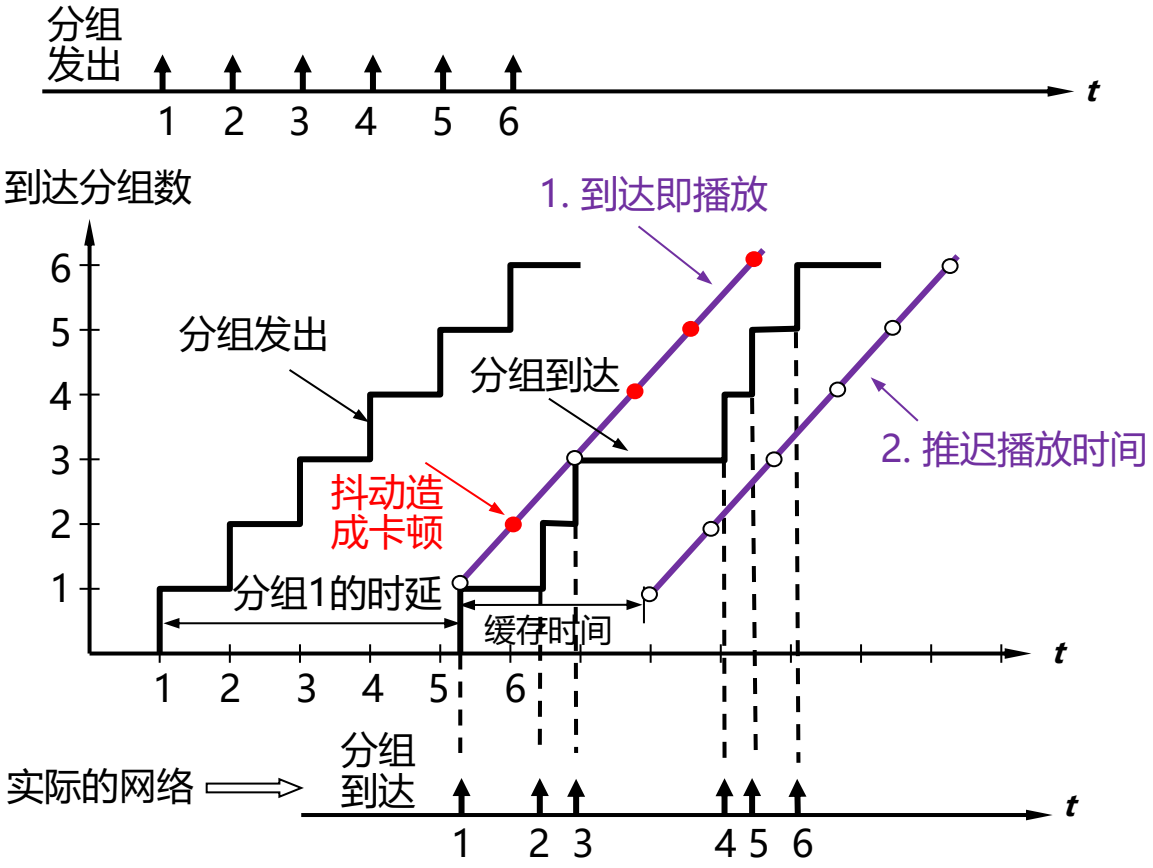
- 浏览器从服务器下载流媒体文件：
  - 浏览器用户使用 HTTP 的 GET 报文接入到万维网服务器；这个超链接指向一个元文件（有音/视频文件的统一资源定位符 URL）
  - 万维网服务器把该元文件装入 HTTP 响应报文的主体，发回给浏览器
  - 浏览器调用媒体播放器，把提取出的元文件传输给媒体播放器
  - 媒体播放器使用元文件中的 URL，向媒体服务器发送 HTTP 请求报文，要求下载音/视频文件（如对应的某个GOP）
  - 媒体服务器发送 HTTP 响应报文，把音/视频文件发送给媒体播放器；媒体播放器边下载边解压缩边播放（通过时间戳同步音频流和视频流）



流式存储媒体的典型下载过程

流式存储媒体

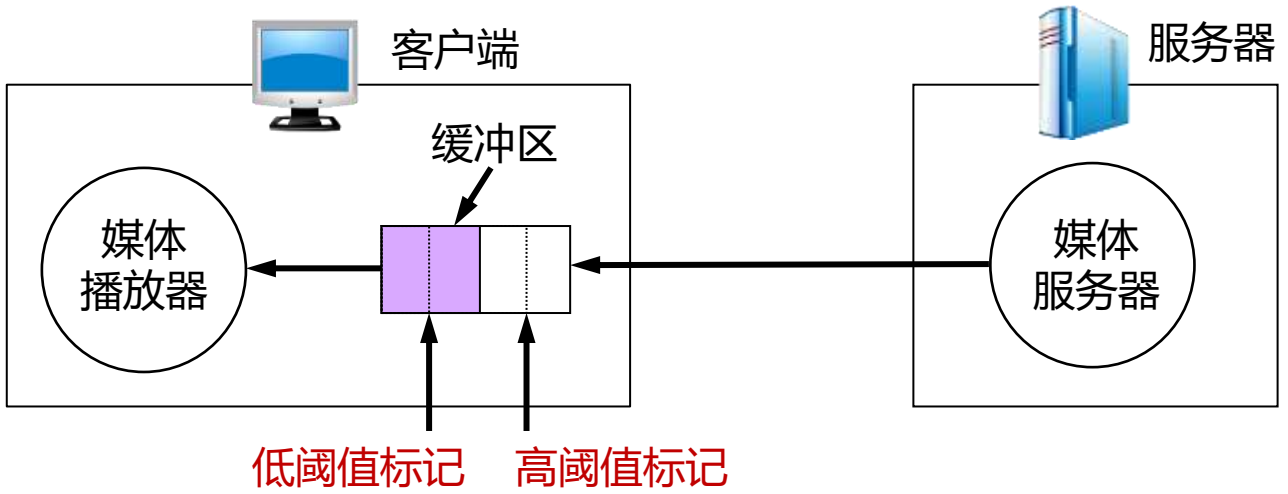
- 发送端以恒定速率产生数据分组
- 网络传输后的结果
  - 由于网络传输的抖动特性，分组到达接收端时变成了非恒定速率
  - 此时如果到达时就随即播放，则会出现卡顿
  - （分组1、3到达接收端可播放，分组2、4、5、6未到达，出现卡顿）
- 如何应对网络传输的抖动特性
  - 在接收端经过缓存后，再以恒定速率播放（推迟播放时间）
  - 能够在一定程度上消除了时延的抖动
  - 但付出的代价是增加了时延



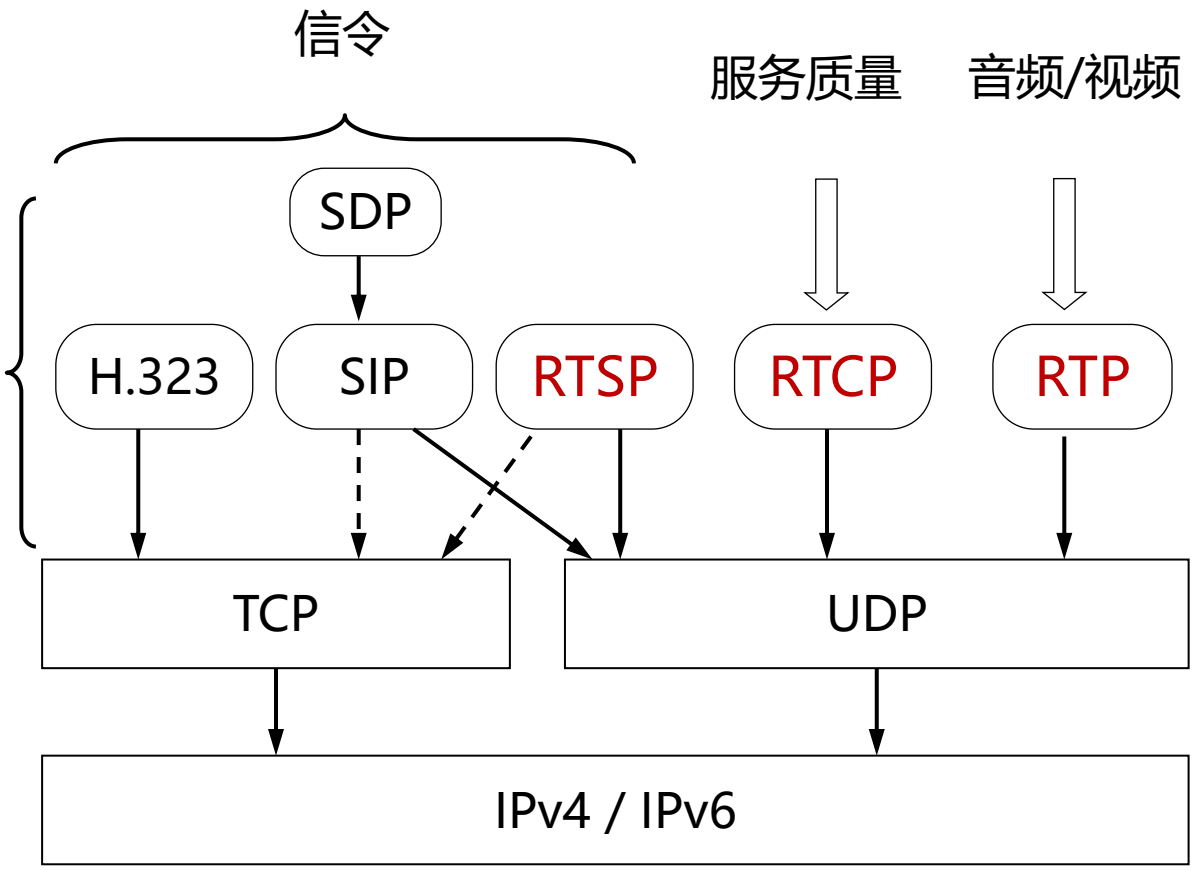
流式存储媒体

客户端缓冲区

- 客户端播放的是本地缓冲区的内容，而不是立即播放来自网络的实时内容
- 缓冲区内容小于低阈值标记：数据即将播完，容易出现卡顿；需要加速传输
- 缓冲区内容大于高阈值标记：增大播放时延，占用存储空间；可以减慢传输
- 需要的决策：需要多大缓存，服务器以多快速率发送，才能在不稳定的网络中，尽量满足用户期望：高清、低延迟、不卡顿
- 上述决策需要特定网络协议支持



流媒体服务协议栈

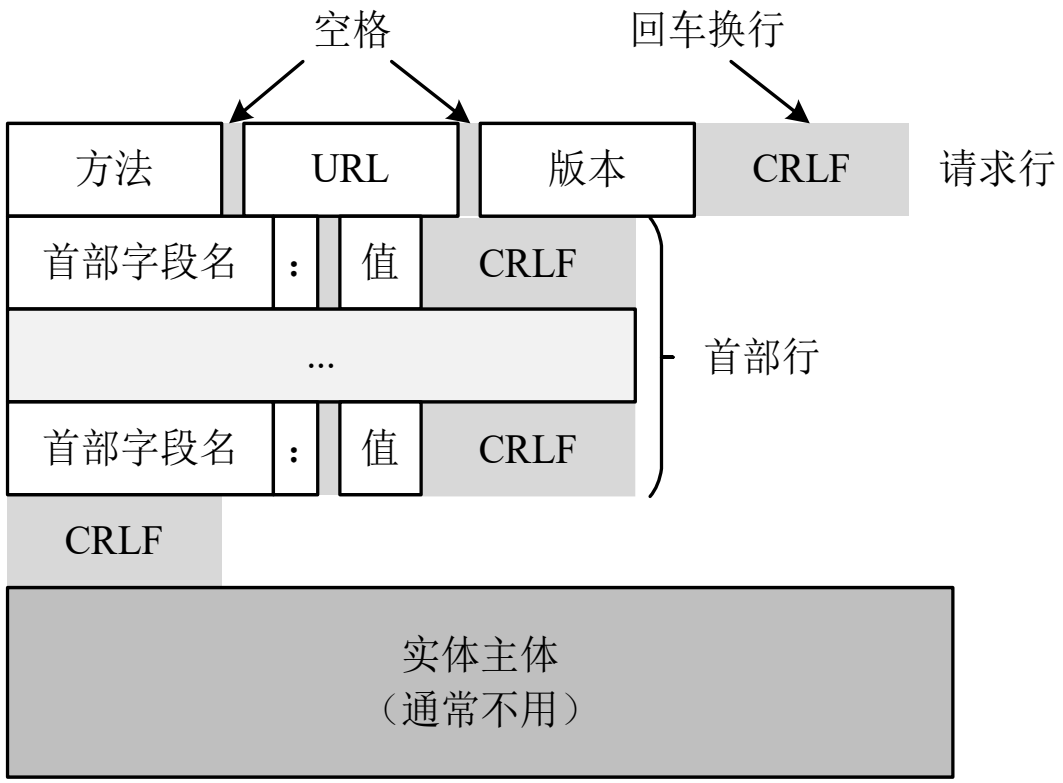


流媒体服务协议栈

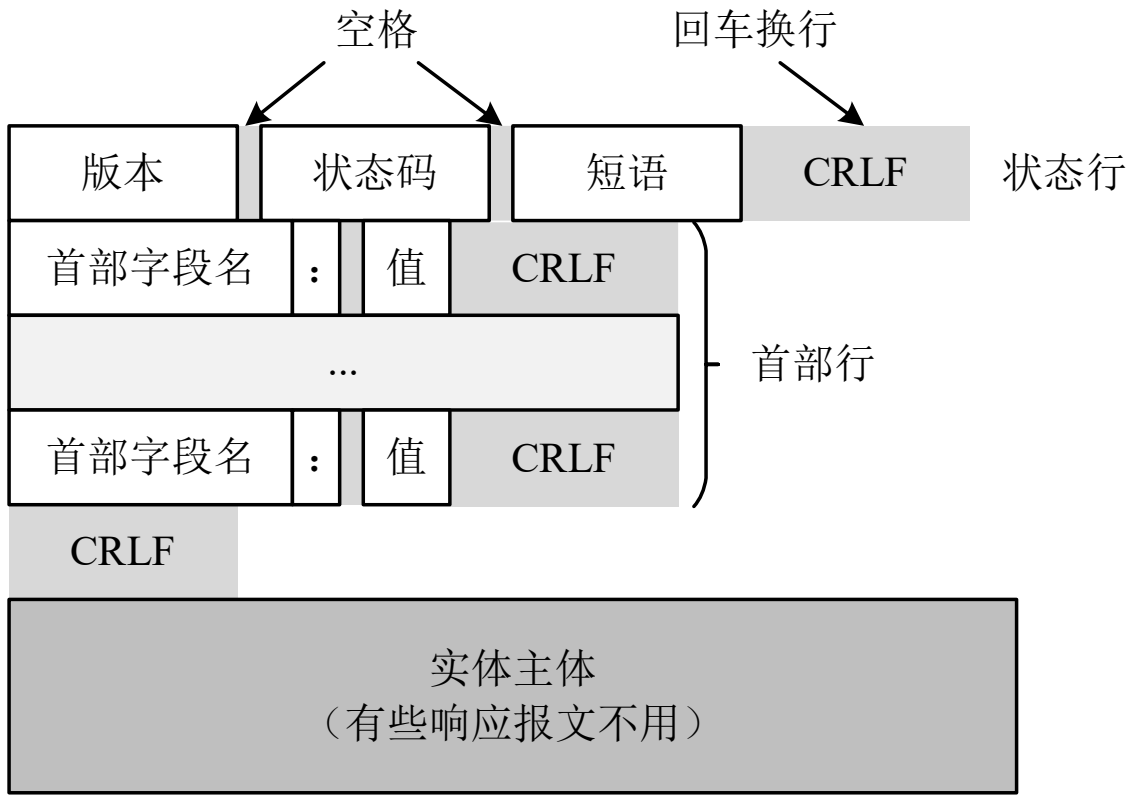
直播与实时音视频的协议

- 实时音频/视频所需要的几种应用协议：
  - 一种是信令协议，对建立的连接起控制作用，如RTSP
  - 一种是数据分组传送协议，使音/视频能够以时延敏感属性传送，如RTP/RTCP
  - 使用TCP，还是UDP？
  - UDP不可靠但效率高，更适合实时类应用
  - UDP需要自行实现流控算法，增加了成本和复杂性
  - UDP传输音视频可能会被路由器丢弃或防火墙阻拦，而TCP可以畅通无阻
  - 实际流媒体系统往往先尝试UDP，如果失败则转为TCP

音视频连接控制协议：RTSP协议



RTSP请求消息



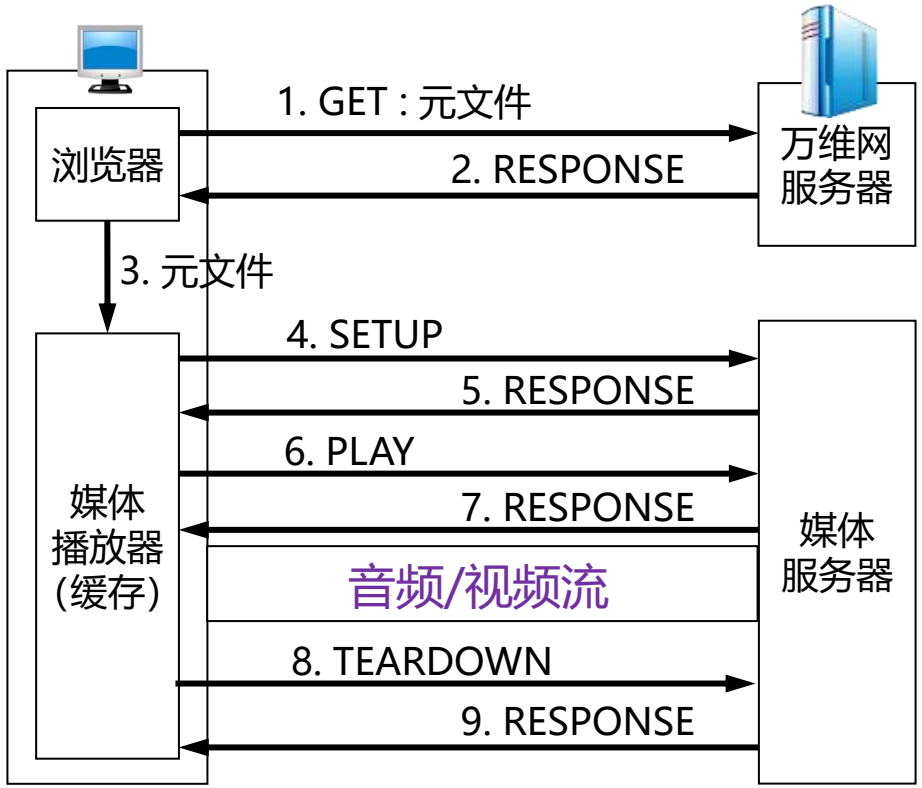
RTSP应答消息

音视频连接控制协议：RTSP协议

- 实时流式协议RTSP (Real-Time Streaming Protocol)：
  - RTSP本身并不传送数据，是一个多媒体播放控制协议
  - RTSP对用户下载的实时数据的播放情况进行控制，如：暂停/继续、后退、前进等。 又称为“互联网录像机遥控协议”
  - RTSP 是有状态的协议，它记录用户所处于的状态（初始化状态、播放状态或暂停状态）
  - RTSP 控制分组既可在 TCP 上传送，也可在 UDP 上传送
  - RTSP 没有定义音频/视频的压缩方案，也没规定音频/视频在网络中传送时应如何封装在分组中
  - RTSP 协议本身没有规定音频/视频流在媒体播放器中应如何缓存，由协议的具体实现负责

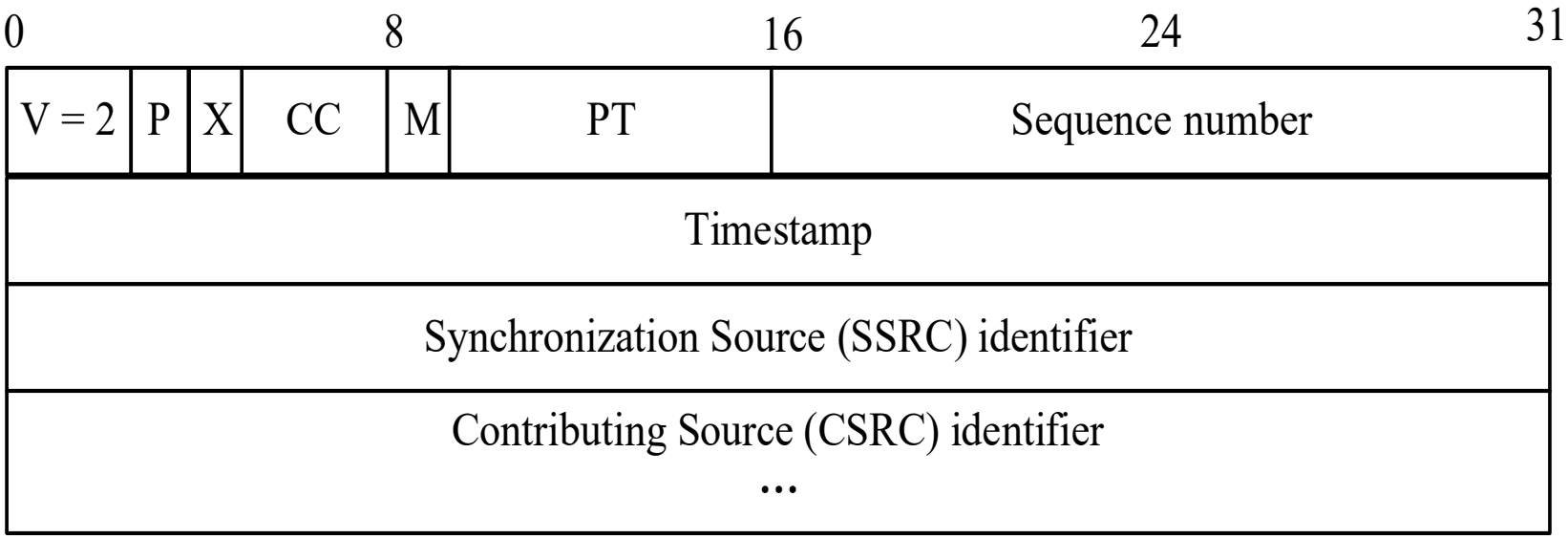
音视频连接控制协议：RTSP协议

- 使用 RTSP 的媒体服务器的工作过程：
  - 浏览器向万维网服务器请求音/视频文件
  - 万维网服务器从浏览器发送携带有元文件的响应
  - 浏览器把收到的元文件传输给媒体播放器
  - RTSP 客户与媒体服务器的 RTSP 服务器建立连接
  - RTSP 服务器发送响应 RESPONSE 报文
  - RTSP 客户发送 PLAY 报文，开始下载音/视频文件的特定位置
  - RTSP 服务器发送响应 RESPONSE 报文
  - 开始传输音视频数据（使用RTP协议）
  - RTSP 客户发送 TEARDOWN 报文断开连接
  - RTSP 服务器发送响应 RESPONSE 报文



RTSP 协议的工作过程

直播与实时音视频传输协议：RTP协议



- 实时传输协议 RTP (Real-time Transport Protocol)：
  - RTP 为实时应用提供端到端的数据传输，但不提供任何服务质量的保证
  - RTP 是一个协议框架，只包含了实时应用的一些共同的功能
  - RTP 不对多媒体数据块做任何处理，而只是向应用层提供一些附加的信息，让应用层知道应当如何进行处理

直播与实时音视频传输控制协议：RTCP

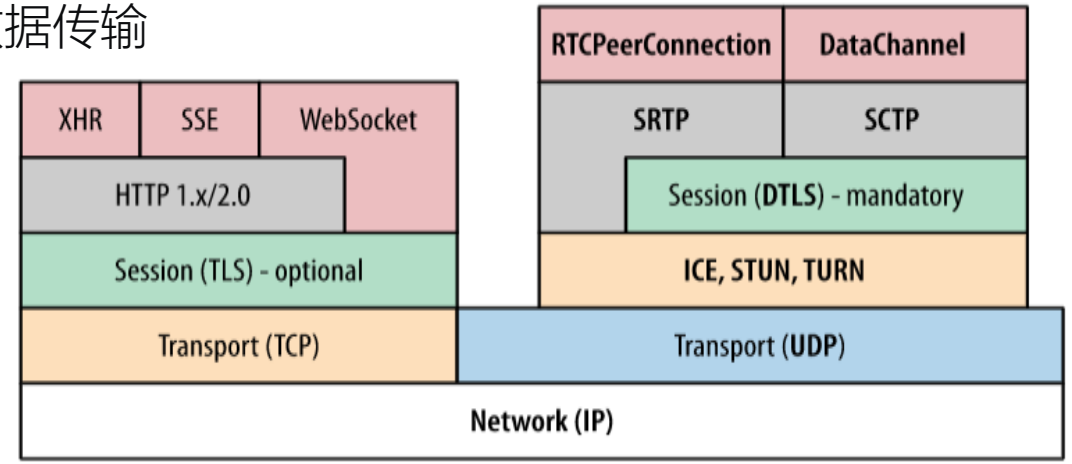
- 实时传输控制协议 RTCP (RTP Control Protocol)：
  - RTCP 是与 RTP 配合使用的控制协议
  - RTCP 的主要功能：服务质量的监视与反馈、媒体间的同步、播组中成员的标识
  - RTCP 分组也使用 UDP 传送，但 RTCP 并不对声音或视像分组进行封装
  - 可将多个 RTCP 分组封装在一个 UDP 用户数据报中
  - RTCP 分组周期性地在网上传送，它带有发送端和接收端对服务质量的统计信息报告

直播与实时音视频的网页实时通信协议：WebRTC

- 网页实时通信 WebRTC（Web Real-Time Communication）
  - 由Google发起的实时音视频通信开源项目
  - 建立浏览器之间点对点的连接，实现音/视频流的传输

开源项目与IETF国际标准同步推进

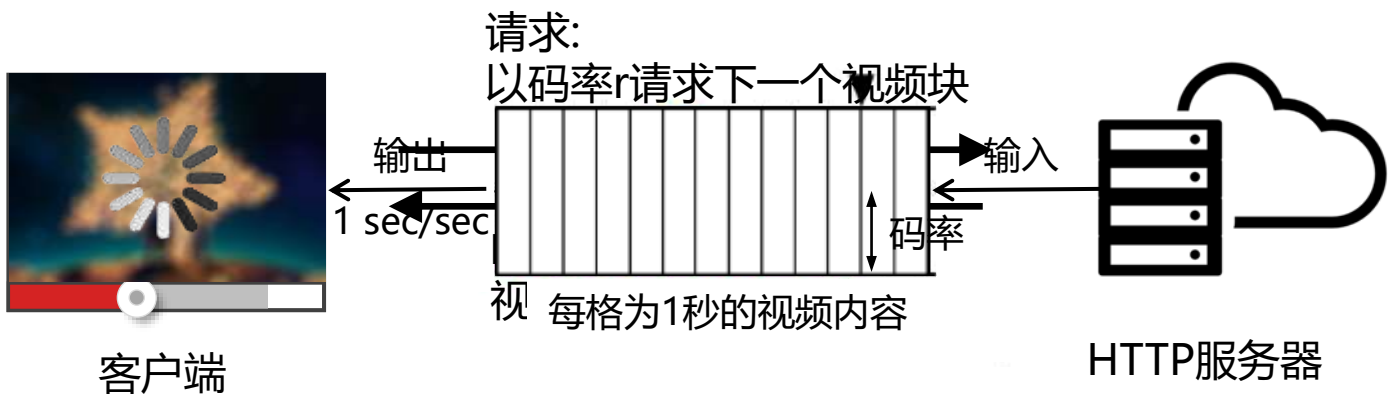
- WebRTC协议栈
  - 为了满足实时性需求，其核心协议是在右侧基于 UDP 基础上搭建起来的
  - Secure RTP（SRTP）与 Secure RTCP（SRTCP） 是对媒体数据的封装与传输控制协议
  - RTCPeerConnection 用来建立和维护端到端连接，提供高效的音视频流传输
  - RTCDataChannel 用来支持端到端的任意二进制数据传输
  - 流控制传输协议SCTP，提供类似 TCP 的特性
  - DTLS 对传输内容进行加密，是 UDP 版 TLS
  - ICE、STUN、TURN 用于内网穿透，应对NAT等私有地址转换的问题

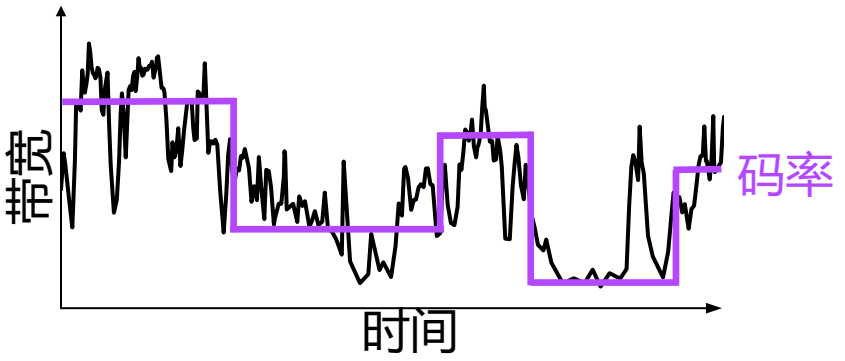


流媒体动态自适应传输

- 视频传输优化的挑战
  - 客户端基于当前网络状况，向服务器请求视频块
  - 若视频块的码率>可用带宽：视频块难以及时抵达客户端，出现卡顿
  - 若视频块的码率<可用带宽：视频质量较低，没有充分利用带宽资源
  - 如何为视频块选择贴近当前可用带宽的码率？
- 进一步挑战：网络带宽随时间不断变化

视频块的码率
VS
网络可用带宽



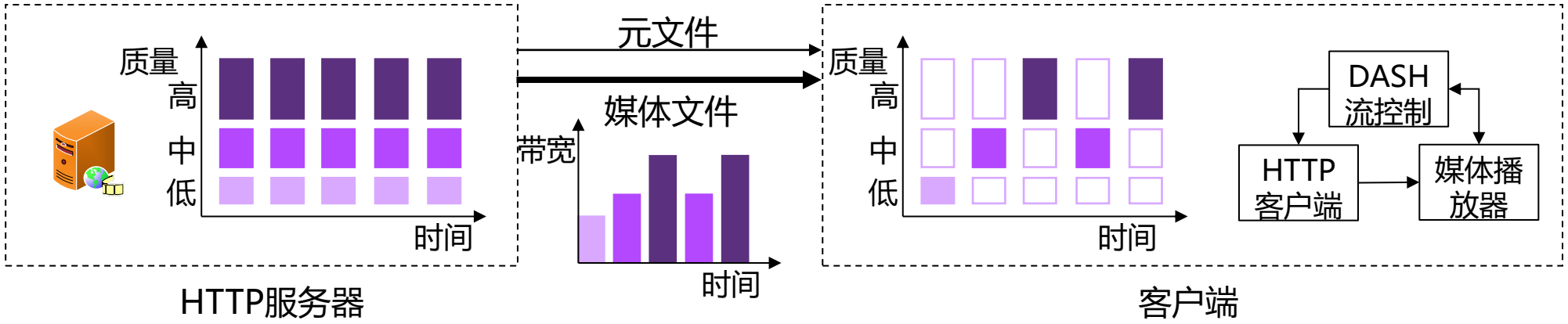


网络可用带宽不断变化时，
如何及时选取合适的码率？

流媒体动态自适应传输

- DASH (Dynamic Adaptive Streaming over HTTP)
- 动态自适应流媒体传输协议DASH，由MPEG组织制定的标准
- 类似协议：苹果HTTP Live Streaming（HLS）； Adobe的HTTP Dynamic Streaming（HDS）；微软的Microsoft Smooth Streaming
- DASH 基本思想
- 完整视频被拆分为固定时长 (2s-10s) 的视频片段(segment)， 每段提供不同码率
- 视频片段与其对应的元文件（URL）一同存放于DASH服务器
- 客户端基于网络条件、缓冲大小等，对每个视频片段，自适应选择合适的视频码率来下载

DASH
选取合适码率的协议



流媒体动态自适应传输

■DASH中普遍使用的自适应码率ABR（Adaptive bitrate）

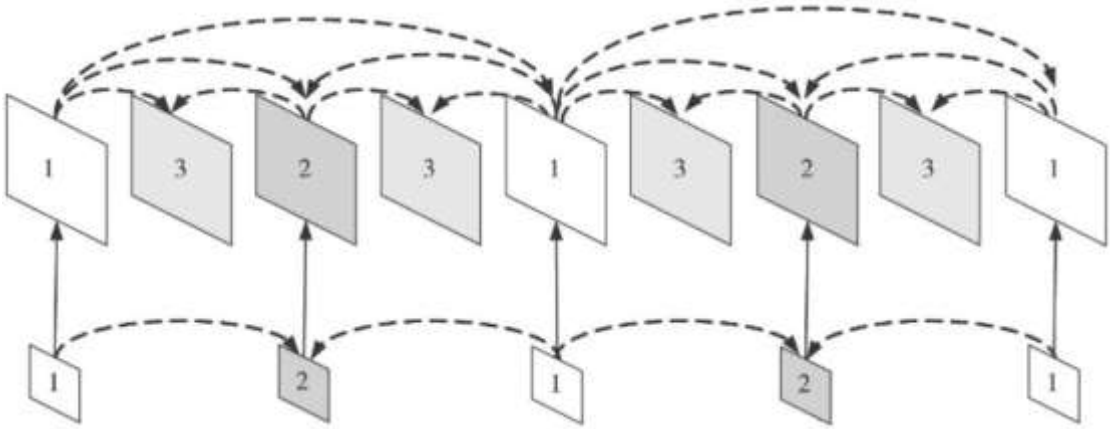
ABR选取合适码率的**算法**

- 2个主要码率切换规则
  - 基于吞吐量的算法：例如：使用滑动窗口平均估计未来吞吐量，选择不高于估计值的最大码率视频
  - 基于缓冲的算法：例如：使用缓冲区满的级别来决定下一个视频块的请求码率
- 放弃请求规则(Abandon Requests Rule)：在下载视频块的过程中，如果算法检测到下载该视频块的过程有卡顿，则终止当前请求，转而重新下载相应的低码率视频块

- 自适应码率ABR广泛应用于DASH和直播、实时通信等各类流媒体传输
- 由于IBP帧互相不独立，ABR调节往往以GOP为单位，难以实时调节
- 由于网络可用带宽难于预测，因此寻找与可用带宽匹配的最佳码率，很有挑战
- 选高清还是选低清帧的难题：（1）选高清可能卡顿，（2）选低清可能浪费带宽？（3）低清、高清一起传，则收到高清时需要丢弃低清帧，也浪费带宽，如何解决？

流媒体动态自适应传输

- 可扩展视频编码 SVC（Scalable Video Coding）
  - SVC 是以H.264为基础，支持多层分级特性：立足基础层，采用锦上添花的增强层
  - 编码器产生的码流包含多个可以单独解码的子码流：不同的码率，帧率和空间分辨率
  - 当带宽不足时，只对基础层的码流进行传输和解码，这时解码的视频质量不高
  - 当带宽充足时，可以传输和解码增强层的码流来提高视频的解码质量
  - 优点：确保传输基础层来避免卡顿，使用富裕的带宽传输增强层，充分利用带宽
  - 缺点：编解码复杂度增加，有多个增强层时开销过大



SVC编码示意图（1、2提升分辨率，3提升帧率）

→ 增强层（高帧率、分辨率）

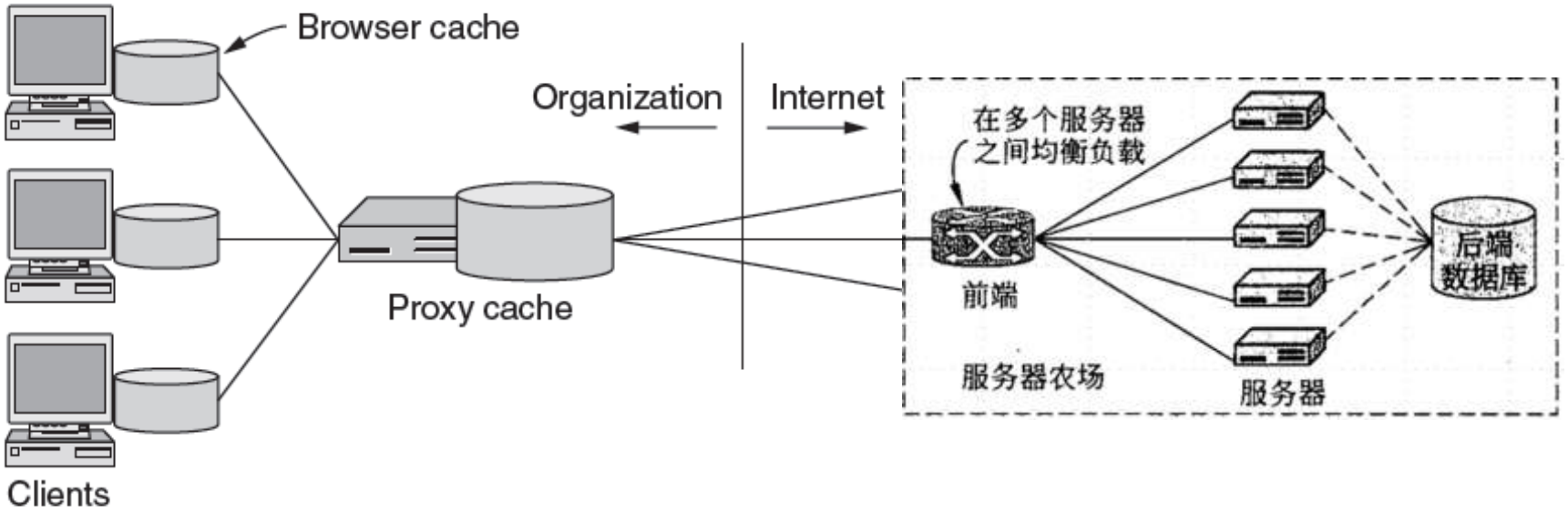
→ 基础层（低帧率、分辨率）

7.7 CDN与 P2P

- 震撼般地转变 (email→FTP→Web→P2P→video)
- 大量站点有很少的流量

- Internet IP流量发展趋势？大部分是IP Video流量，到2022年将占82%

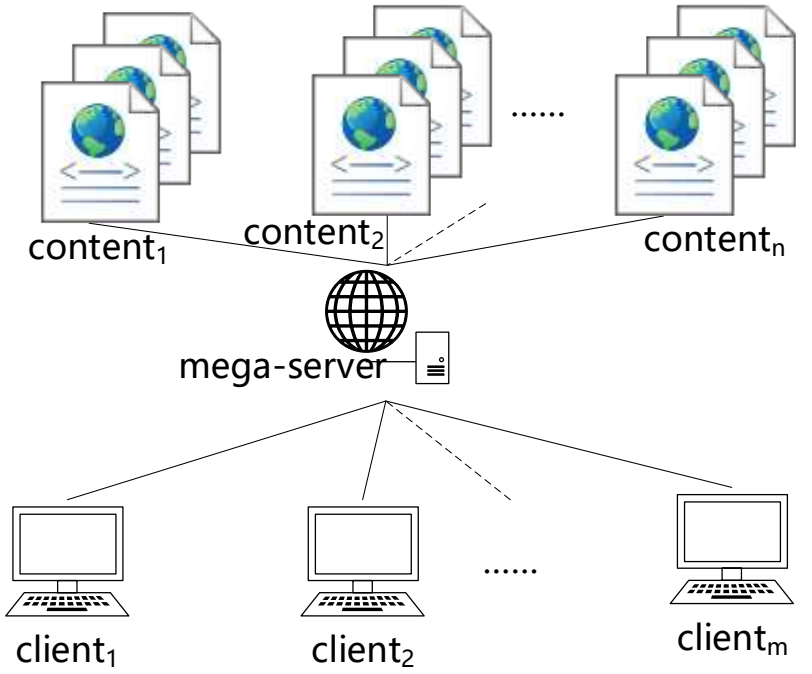
服务器群和Web代理



- 代理缓存（Web代理）帮助组织扩展Web，提高Web访问的性能
  - 将服务器内容缓存到客户上以提高性能，降低时延，减少网络带宽需求
  - 实现组织的策略(如访问策略)

内容分发网络CDN

- 怎样将同一内容（如从百万的视频中选定的内容）分发给同时发起访问的数百万用户?
- 由单个、大型、高性能的"服务器"统一提供? 存在如下问题:
  - 单点故障
  - 网络拥塞
  - 远程用户的长路径



内容分发网络CDN

内容分发网络CDN

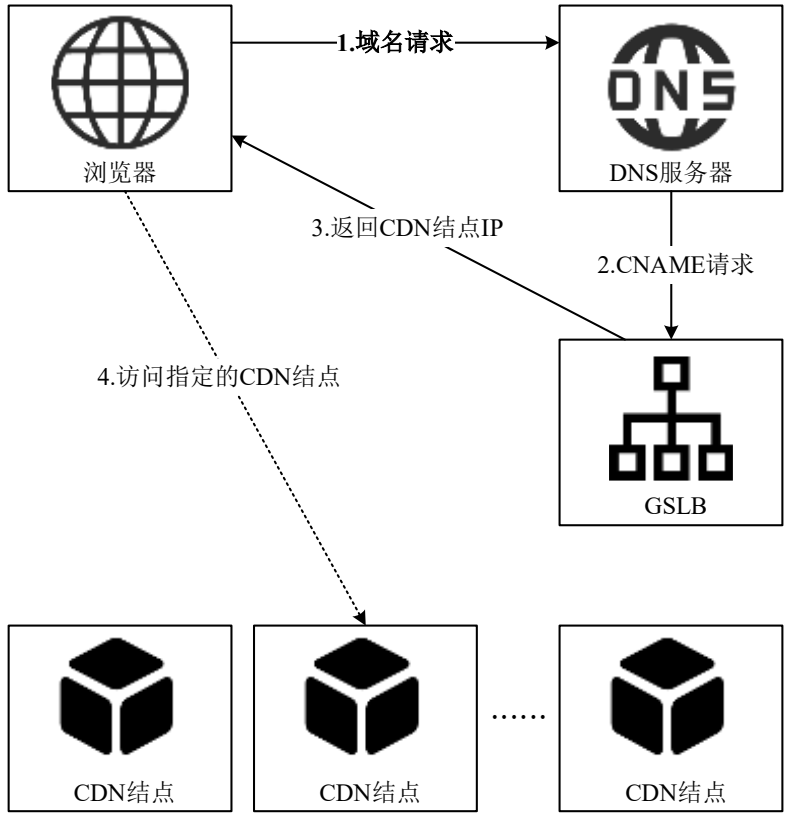
- Content Delivery Network，or Content Distribution Network
- 依靠部署在各地的边缘服务器，通过中心平台的负载均衡、内容分发、调度等功能模块，使用户就近获取所需内容

主要优点

- 降低响应时延，避免网络拥塞
- 避免原始服务器过载及防止DDoS攻击
- 分布式架构，具有良好的可扩展性
- 对用户透明，无需用户感知

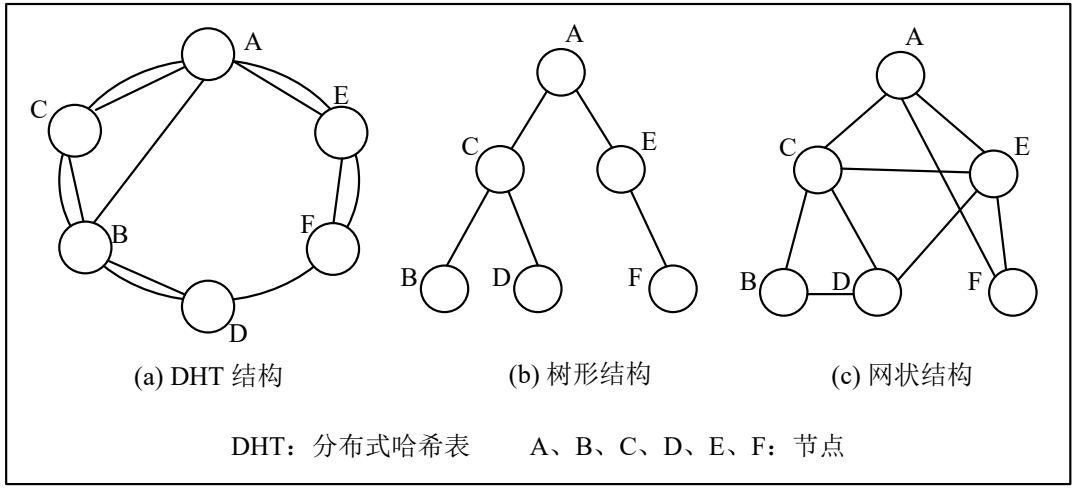
内容分发网络CDN要解决的问题

- 静态资源是如何被缓存到CDN结点中
- 如何匹配至最合适的CDN结点
  - 浏览器向DNS服务器发送域名请求
  - DNS服务器根据CNAME的别名记录向GSLB发送请求
  - GSLB返回性能最好（通常距离请求地址最近）的CDN结点（边缘服务器，真正缓存内容的地方）的地址给浏览器
  - 浏览器直接访问指定的CDN结点
  - 防止资源被盗



P2P应用系统体系结构

- 对等网络（peer-to-peer，P2P）是一种文件共享网络
- 基本思路是将多台计算机聚在一起并且把它们的资源组成一个池，从而形成一个内容分发系统
- P2P技术属于覆盖层网络（Overlay Network）的范畴，是一种对等网络结构
- P2P网络有3种比较流行的组织结构，被应用在不同的P2P应用中



(a) DHT 结构　(b) 树形结构　(c) 网状结构

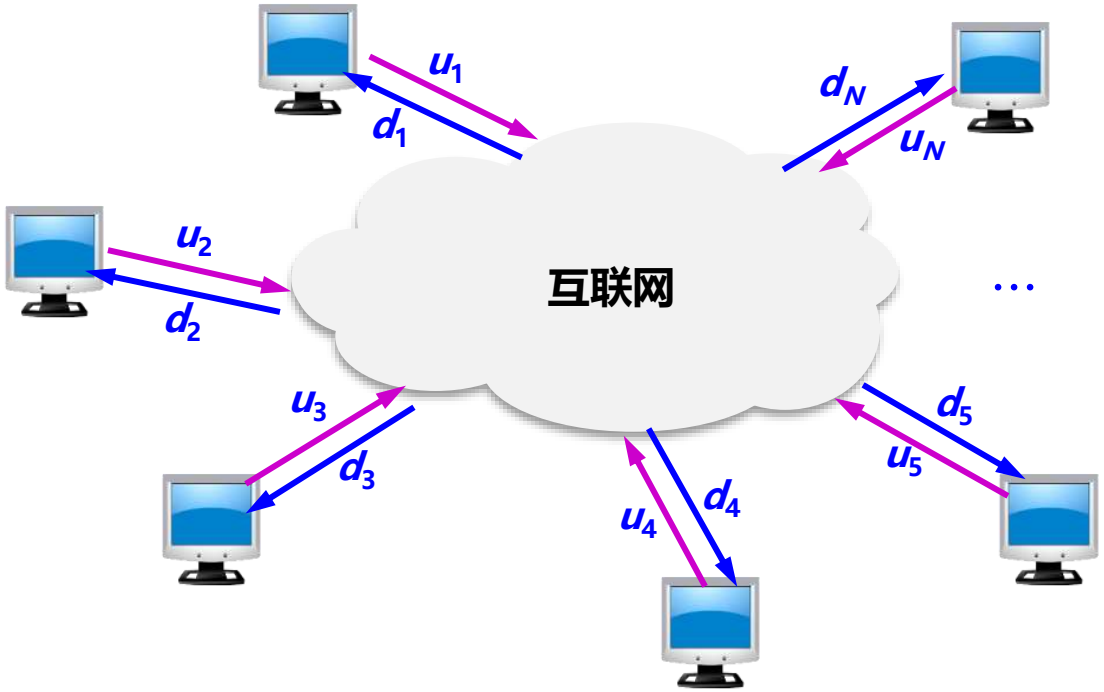
DHT：分布式哈希表　A、B、C、D、E、F：节点

P2P典型的应用领域包括

- 分布式科学计算
- 文件共享
- 流媒体直播和点播
- IP层语音通信
- 网络游戏平台
- 区块链

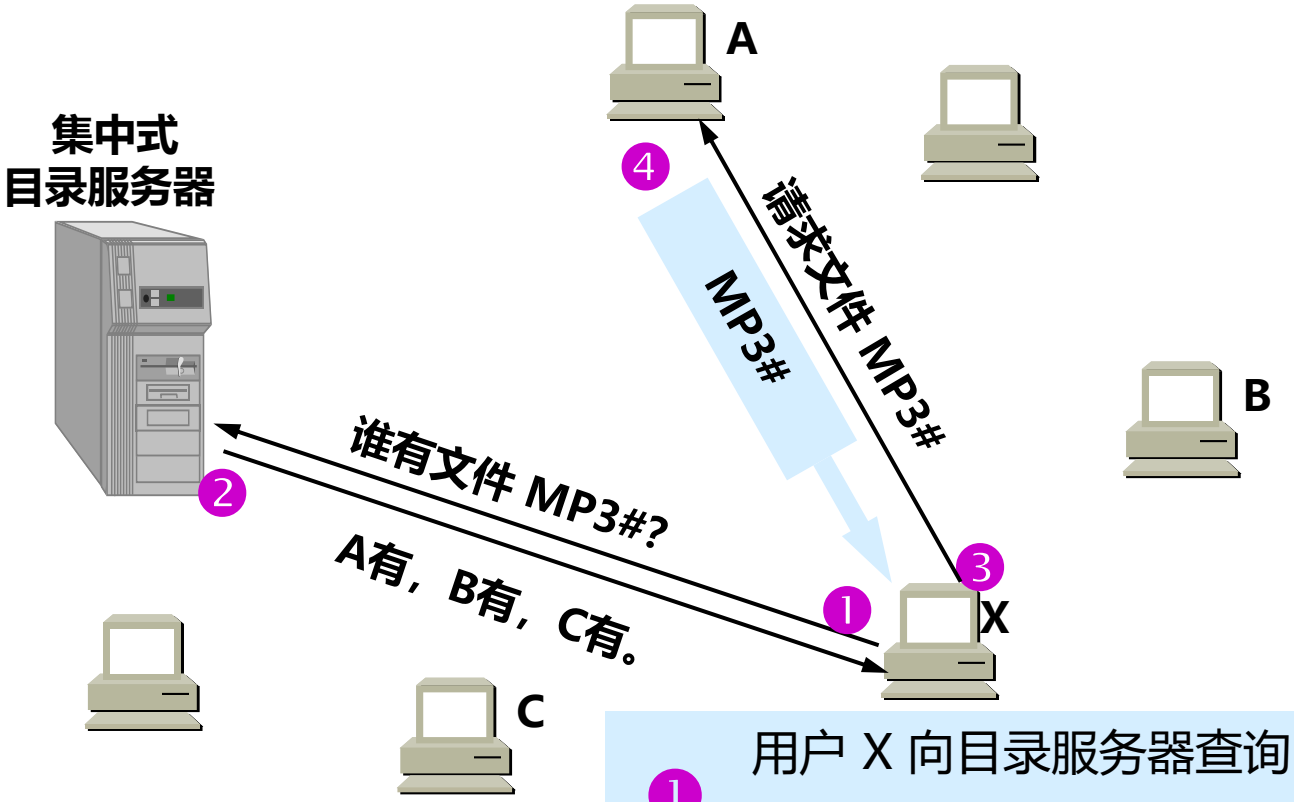
基于P2P 下载媒体流示例：下载MP3

- P2P （Peer to Peer）是一个点到点的对等模型，每台主机既是服务器，又是客户端
- 在 P2P 工作方式下，所有的音频/视频文件都是在普通的互联网用户之间传输，解决了集中式媒体服务器可能出现的瓶颈问题。



基于P2P 下载媒体流示例：下载MP3

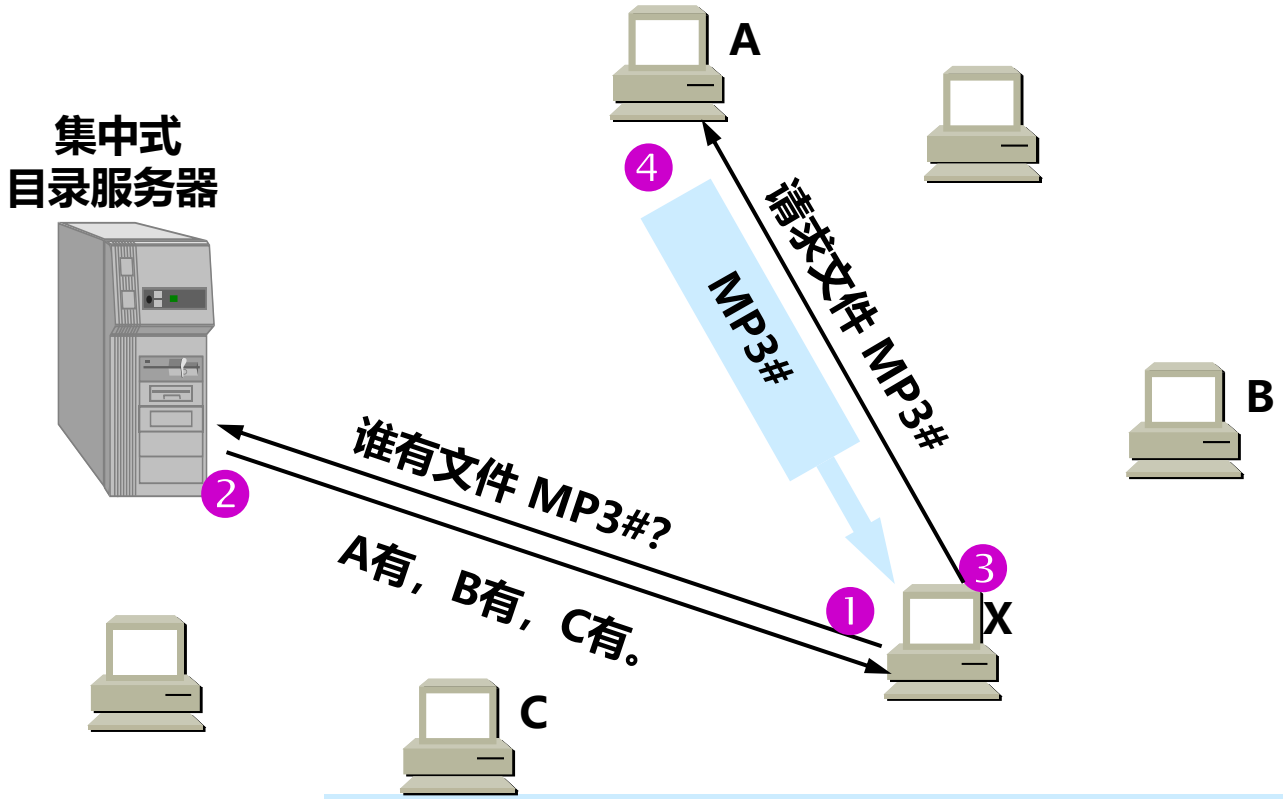
- 将所有音乐文件的索引信息都集中存放在目录服务器中
- 使用者只要查找目录服务器，就可知道应从何处下载所要的MP3文件
- 用户要及时向目录服务器报告自己存有的音乐文件



1 用户 X 向目录服务器查询（客户–服务器方式）谁有音乐文件 MP3#

基于P2P 下载媒体流示例： 下载MP3

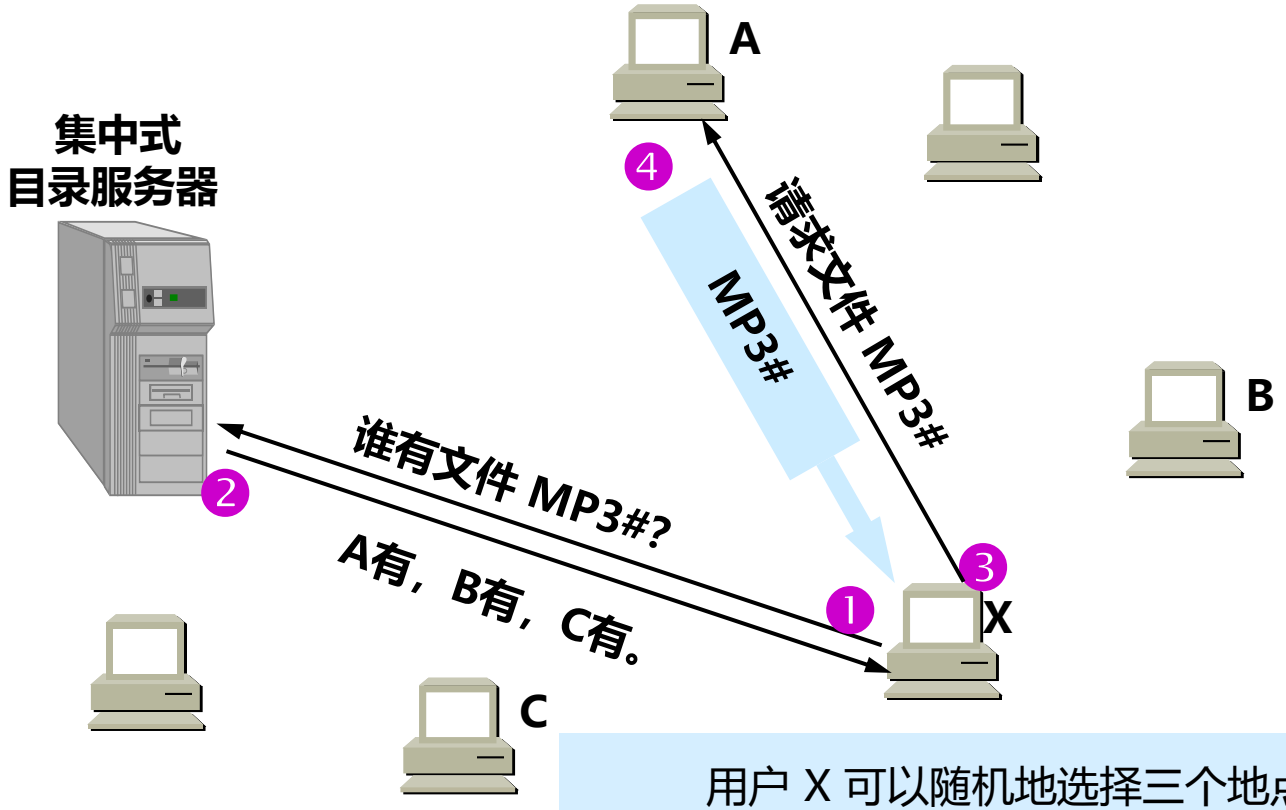
- 将所有音乐文件的索引信息都集中存放在目录服务器中
- 使用者只要查找目录服务器，就可知道应从何处下载所要的MP3文件
- 用户要及时向目录服务器报告自己存有的音乐文件



2 目录服务器回答 X：有三个地点有文件 MP3#，即 A, B 和 C（给出了这三个地点的IP地址）。于是用户 X 得知所需的文件 MP3# 的三个下载地点

基于P2P 下载媒体流示例： 下载MP3

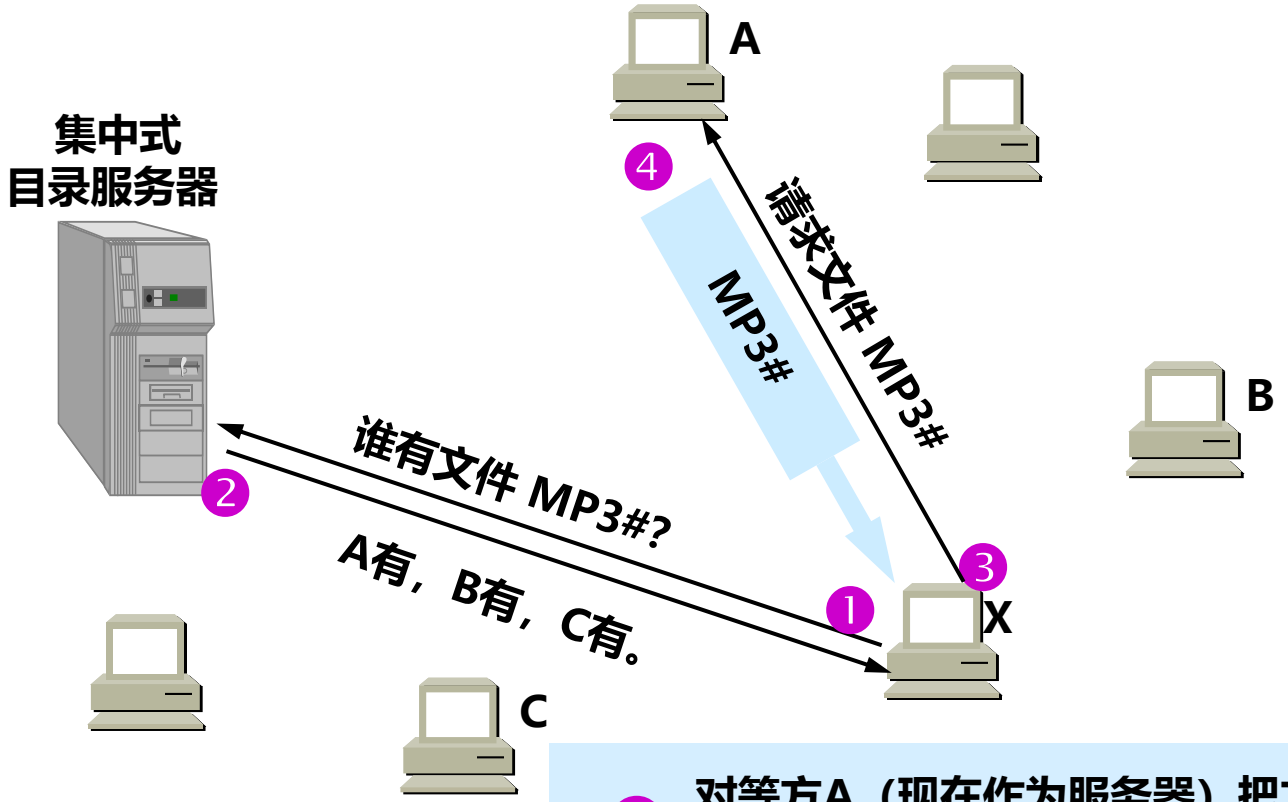
- 将所有音乐文件的索引信息都集中存放在目录服务器中
- 使用者只要查找目录服务器，就可知道应从何处下载所要的MP3文件
- 用户要及时向目录服务器报告自己存有的音乐文件



3 用户 X 可以随机地选择三个地点中的任一个。假定 X 向 A 发送下载文件 MP3#的请求报文。双方都使用 P2P 方式通信

基于P2P 下载媒体流示例： 下载MP3

- 将所有音乐文件的索引信息都集中存放在目录服务器中
- 使用者只要查找目录服务器，就可知道应从何处下载所要的MP3文件
- 用户要及时向目录服务器报告自己存有的音乐文件



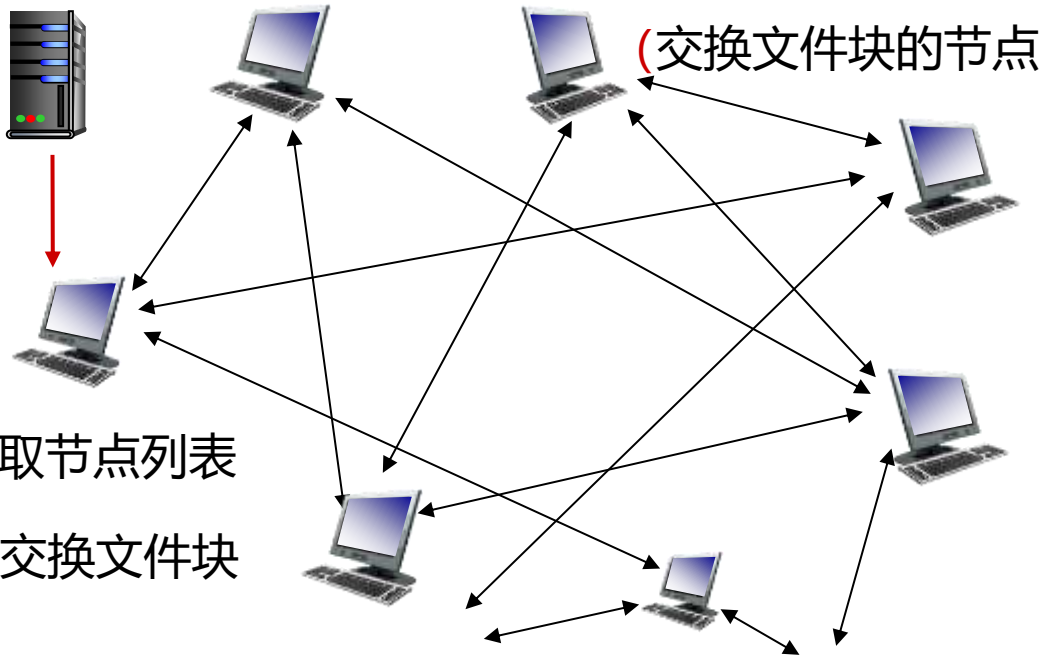
P2P网络实现内容分发

■跟踪器tracker

(负责帮助节点获取其他节点的信息)

■种子torrent

(交换文件块的节点)



... 从跟踪器获取节点列表

... 同种子节点交换文件块

■P2P文件分发协议：BitTorrent

- 文件被划分为256Kb大小的块
- 具有种子(torrents)的节点发送或接收文件

P2P网络实现内容分发

- P2P文件分发协议：BitTorrent
  - BitTorrent 使得对等方可以下载种子（torrents）
  - 对等方可以通过Tracker（跟踪器）找到对方，跟踪器是一个服务器，它维护着一个正在主动上传和下载该内容的所有其他对等用户列表
  - 对等节点彼此之间交换各自拥有的块清单，并选择罕见的很难找到的块下载
  - 趋向于匹配那些相互之间具有可比性上传和下载速率的对等节点
  - 一个对等节点为其他对等节点作出的贡献越多，它预期的回报就越大
  - 下载同时向其他节点上传文件块
  - 节点动态加入和退出
  - 获取整个文件后，存在自私离开可能

P2P网络实现内容分发

- 分布式哈希表（Distributed Hash Tables，DHTs) 是一个完全分布式索引，可扩展到多客户端（或实体）
  - 需要为N个实体，跟踪o(logN)个路径
  - 可以在没有服务器的情况下，使用跟踪器找到对等方
  - 查找DHT中的种子（torrent）来找到对等方的IP地址
  - Kademlia用于BitTorrent，使用了分布式哈希表

P2P网络实现内容分发

- 工作过程
  - 使用一个哈希函数hash把任何IP地址转换成一个160位的数字，该数字称为节点标识符（node identifier），并排好顺序，successor函数表示后继节点
  - 有些标识符的节点才对应于实际节点，其余的节点都不存在实际节点
  - 关键字（key）也可以由哈希函数产生
  - 节点必须请求successor (hash (torrent))来存储my-IP-address

7.8 本章总结

- OSI七层模型中的会话层、表示层和应用层统称为OSI高层
- TCP/IP中，统称为应用层或用户层
- 互联网提供的网络服务都属于应用层，服务模式包括两类：客户/服务器模型、P2P模型
- TCP/IP网络架构以IP地址来识别不同的主机，需要DNS将域名翻译成IP地址
- 域名的解析请求由主机解析器向本地服务器发起，如果本地服务器查询到该域名是在本网络子域中，则从数据库中找出域名并返回数据信息
- 如果该域名非本网络子域域名，则解析方法有两种：重复解析和递归解析

本章总结

- 电子邮件（Email）是互联网上广泛使用的一种应用
- 电子邮件系统应具有三个主要组成部件：用户代理、邮件服务器、电子邮件传输协议（发送/传递协议，读取协议）
- FTP应用程序通过本地的客户端程序与远程FTP服务器进行交互，实现访问远程文件系统；
- FTP服务端需要开放两个独立的TCP端口：
  - 用于控制连接，缺省端口号是21；
  - 用于数据连接，缺省端口号是20；
- FTP有两种工作模式：主动模式（PORT模式）、被动模式 （PASV模式）
- TCP/IP协议簇中还有一个简单文件传输协议（TFTP）
  - 使用UDP，只支持文件传输，它不支持交互

本章总结

- Web服务是目前最广泛的互联网应用,由大量分布在互联网上、以超文本页面形式存在的内容组成
- Web应用结构包括三个部分： Web服务器、浏览器、超文本传输协议HTTP
- 互联网上的所有资源都有一个唯一的URL，有3部分： 协议、页面所在主机的域名（或者IP地址）、唯一指示特定页面的路径
- 流式传输有顺序流式传输和实时流式传输两种方式
  - 实时流式传输是实时传送，“实时”的概念是指在一个应用中数据的交付必须与数据的产生保持精确的时间关系，如RTSP、RTP 、RTCP
  - 顺序流式传输是顺序下载，在传输期间根据用户连接的速度进行调整，如RTMP、HTTP、MMS和HLS等

本章总结

- 流式传输有顺序流式传输（Progressive Streaming）和实时流式传输（Realtime Streaming）两种方式
- 实时流式传输（Realtime Streaming）是实时传送，“实时”的概念是指在一个应用中数据的交付必须与数据的产生保持精确的时间关系，如RTSP、RTP 、RTCP
- 顺序流式传输（Progressive Streaming）是顺序下载，在传输期间根据用户连接的速度进行调整，如RTMP、HTTP、MMS和HLS等

本章总结

- 内容分发网络（CDN）是构建在现有网络基础之上的智能虚拟网络，依靠部署在各地的边缘服务器
- CDN通过中心平台的负载均衡、内容分发、调度等功能模块，使用户就近获取所需内容，降低网络拥塞，提高用户访问响应速度和命中率
- P2P是将多台计算机聚在一起并且把它们的资源组成一个池，从而形成一个内容分发系统，每个客户都参与到分发内容的任务中，而且通常没有中心控制点
- P2P网络模型主要分为非结构化和结构化两类。非结构化P2P网络模型按结点的集中化程度分为集中式、非集中式和混合型