



计 算 机 网 络

西北工业大学 软件与微电子学院

计算机网络

第2章 应用层

上网？做什么？

- ❑ 浏览新闻
- ❑ 聊天
- ❑ 听音乐
- ❑ 看电影
- ❑ 看电视
- ❑ 收发Email
- ❑ 下载软件

- ❑ 网上图书馆
- ❑ 网上商店
- ❑ 网上银行
- ❑ 网上医院
- ❑ 网上大学
- ❑ 电子商务

....

某些网络应用

- E-mail
- Web
- 即时讯息
- 远程注册
- P2P文件共享
- 多用户网络游戏
- 流式存储视频片段
- 因特网电话
- 实时视频会议
- 大规模并行计算

第2章 应用层

我们的目标:

- 网络应用协议的概念，实现方面
 - ◆ 运输层服务模型
 - ◆ 客户机/服务器模式
 - ◆ 对等范例

- 通过考察流行的应用级协议，学习协议
 - ◆ HTTP
 - ◆ FTP
 - ◆ SMTP / POP3 / IMAP
 - ◆ DNS
- 网络应用编程
 - ◆ 套接字 API

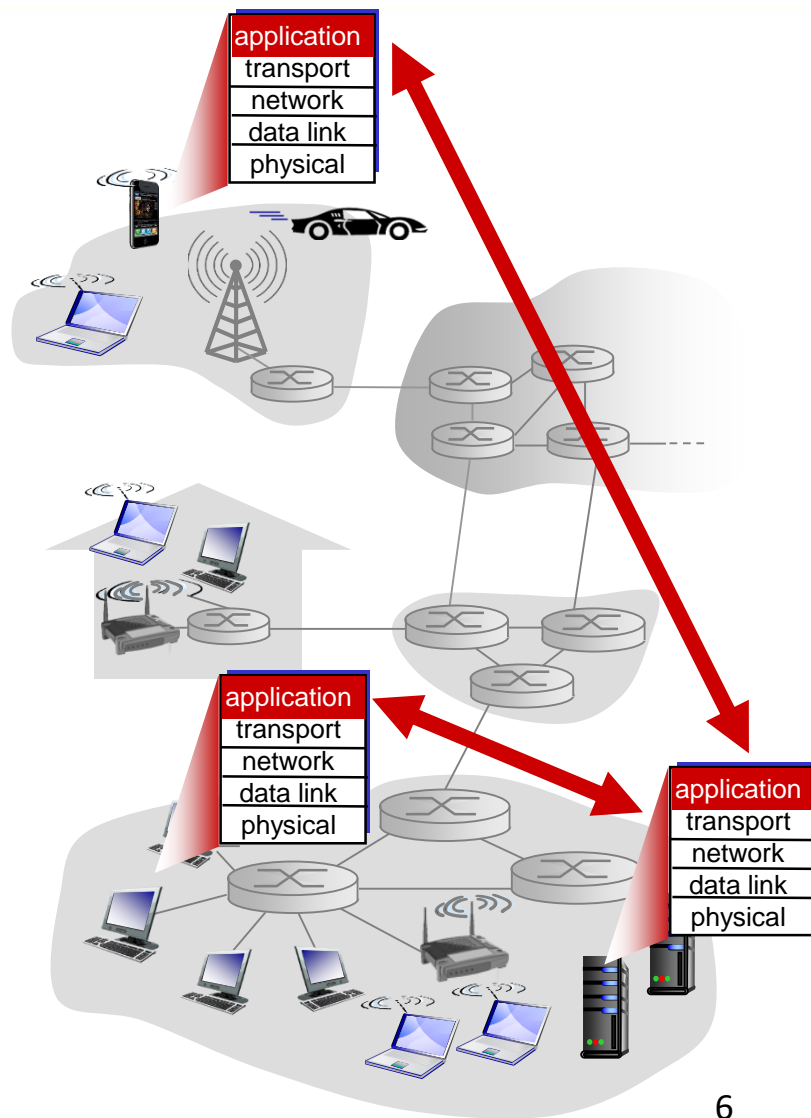
如何创建一个网络应用程序

编写程序能够

- ◆能够在不同端系统上运行
- ◆能够通过网络通信
- ◆如Web: Web服务器软件与浏览器软件通信

非网络核心设备的程序

- ◆三层协议软件(路由器)
- ◆二层协议软件(交换机、路由器)



➤ 2.1 应用层协议原理

➤ 2.2 Web应用和HTTP协议

➤ 2.3 文件传输协议：FTP

➤ 2.4 电子邮件

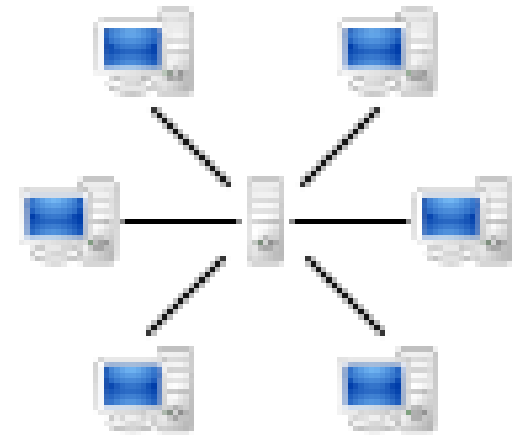
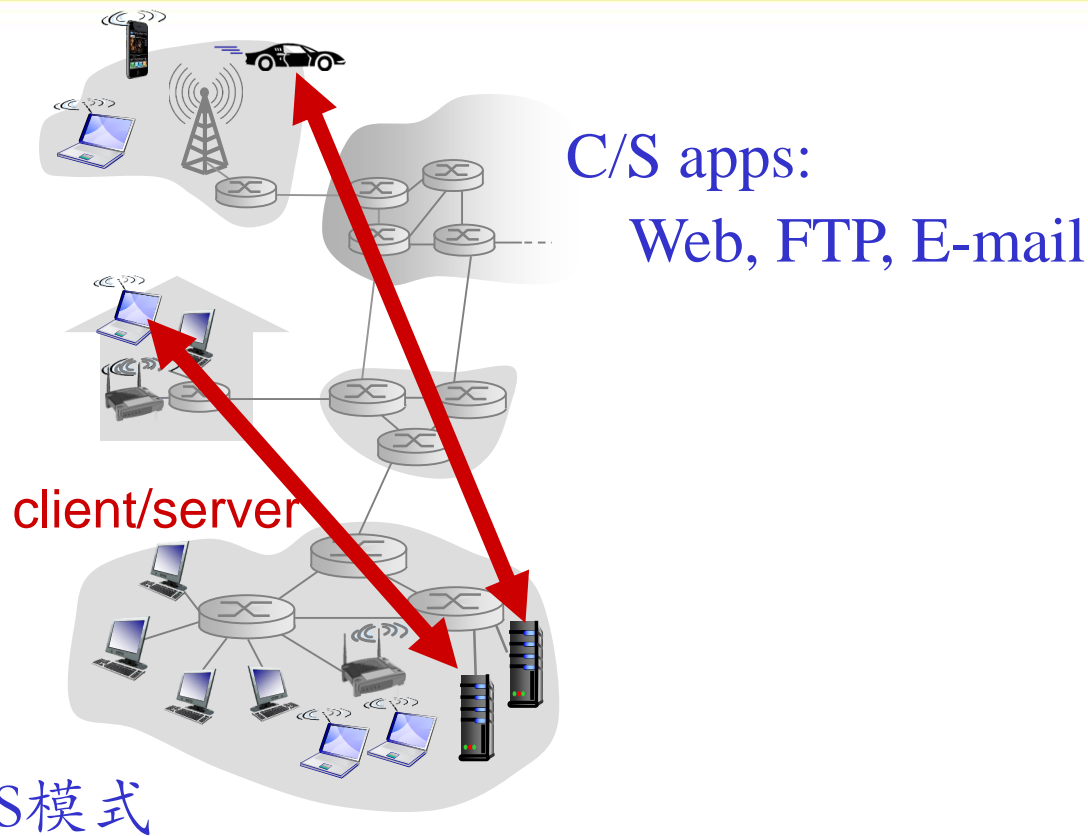
➤ 2.5 域名系统DNS

➤ 2.6 P2P 应用

网络应用程序体系结构

- 客户机/服务器
- 对等 (P2P)
- 客户机/服务器与P2P的混合

客户机/服务器体系结构



- ◆ 集中结构，一对多
- ◆ 服务器共享资源，客户机资源不共享
- ◆ 服务器可能负载过重
- ◆ 网络带宽限制

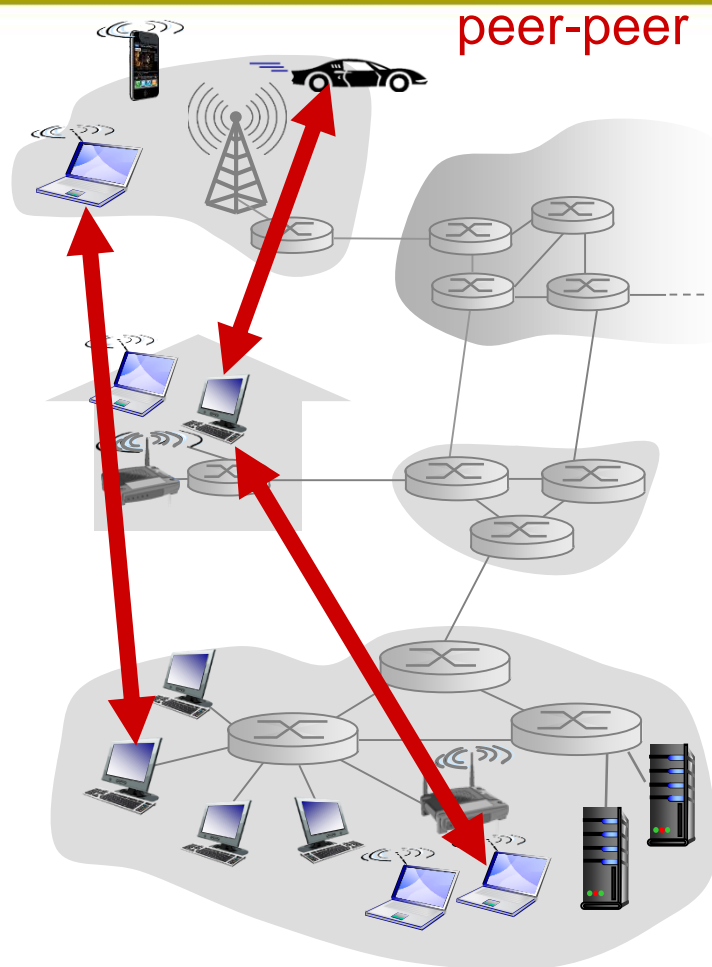
P2P体系结构

P2P模式

- ◆ 非集中结构，多对多
- ◆ 节点具备客户与服务器双重特性
- ◆ 充分利用终端资源
- ◆ 可扩展性好

类型

- ◆ 纯P2P: Gnutella
- ◆ 混合P2P: 迅雷



进程通信

进程：运行在端系统中的程序。

- 在同一台主机中：两个进程使用进程间通信 (由操作系统定义).
- 在不同的主机中：进程通过**消息交换机制**（应用层协议）通信

客户机进程：发起通信的进程

服务器进程：等待联系的进程

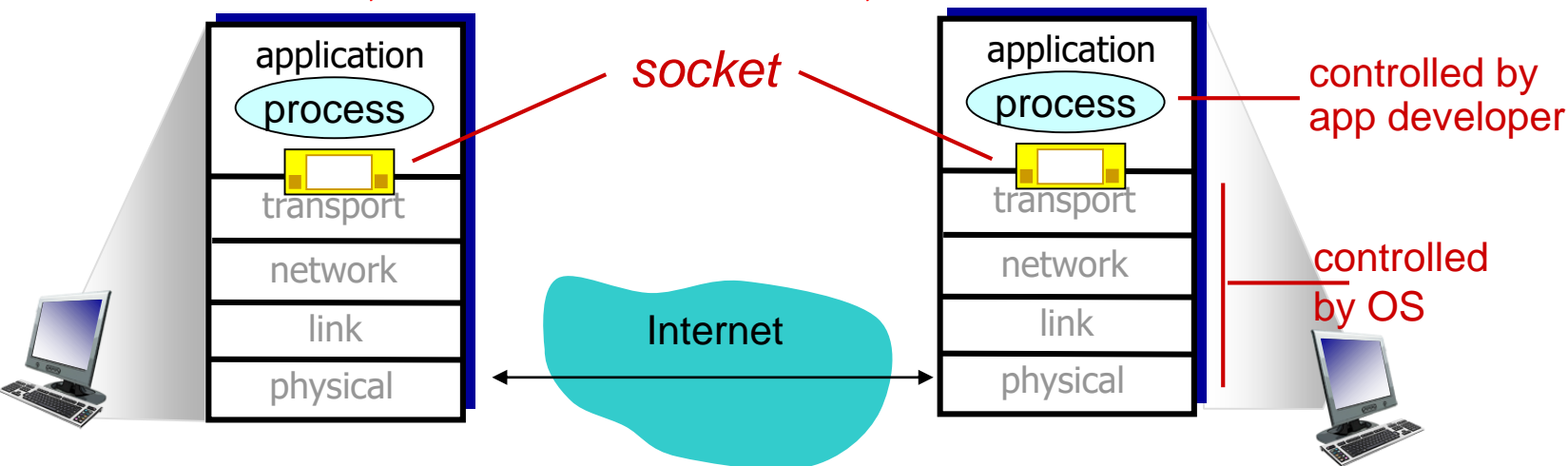
- 注意：具有P2P体系结构的应用程序**同时**具有客户进程和服务进程

一个主机中可能同时运行有多个进程，接收端从网络收到消息后要交给哪个进程？

- 对于接收报文的进程，必须具有一个标识
- 一台主机具有一个独特的32比特的IP地址
- **问题：**主机的IP地址足以标识该进程？
- **答案：**否，同一台主机上能够运行许多进程
- 标示符包括IP地址和与主机上该进程相关的端口号.
- 端口号例子：
 - ◆ HTTP 服务器: 80
 - ◆ 电子邮件服务器: 25

套接字(Socket)

套接字 (socket) 是通信的基石，是支持TCP/IP协议的网络通信的基本操作单元。它是网络通信过程中端点的抽象表示，包含进行网络通信必须的**五种信息**：**连接使用的协议，本地主机的IP地址，本地进程的协议端口，远地主机的IP地址，远地进程的协议端口。**



➤ 进程通过**套接字**在网络上发送/接收报文

➤ 套接字类似于门

上联应用进程，下联网络协议栈

➤ 套接字是应用程序与网络之间的**API**（应用程序编程接口）

应用程序需要什么样的运输服务?

可靠的数据传输

- 某些应用（如音频）能够容忍某些丢失
- 其他应用（如文件传输，Telnet）要求100%可靠数据传输

定时

- 某些应用（如因特网电话、交互式游戏）要求“有效的”低时延

吞吐量

- 某些应用（如多媒体）要求“有效的”最小量的带宽
- 其他应用（“弹性应用”）充分利用它们获得的所有带宽

安全

- 加密，数据完整性，...

普通应用的运输服务要求

应用程序	数据丢失	带宽	时间敏感
文件传输	不能丢失	弹性	不
电子邮件	不能丢失	弹性	不
Web 文档	不能丢失	弹性	不
实时音频/视频	容忍丢失	音频: 5kbps-1Mbps 视频: 10kbps-5Mbps	是, 100' s msec
存储音频/视频	容忍丢失	同上	是, 几秒
交互式游戏	容忍丢失	几kbps以上	是, 100 msec
即时讯息	不能丢失	弹性	

因特网运输协议服务

TCP服务（不能丢失）：

- **面向连接**：客户机和服务器之间所需的建立
- **可靠传输**：在发送和接收进程之间
- **流控制**：发送方不会淹没接收方
- **拥塞控制**：当网络过载时抑制发送方
- **并不提供**：定时，最小带宽保证

UDP服务（容忍丢失）：

- 在发送进程及接收进程之间的不可靠数据传输
- 不提供：建立连接建立，可靠性，流控，拥塞控制，定时或带宽保证

问题：为什么需要UDP？

因特网应用：应用协议与运输协议

应用	应用层协议	下面的传输协议
电子邮件	SMTP [RFC 2821]	TCP
远程终端访问	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
文件传输	FTP [RFC 959]	TCP
远程文件服务器	NFS [McKusik 1996]	UDP或TCP
流媒体	通常专用 (YouTube)	UDP或TCP
因特网电话	通常专用 (Skype)	典型用UDP

- 2.1 应用层协议原理
- 2.2 Web应用和HTTP协议
- 2.3 文件传输协议：FTP
- 2.4 电子邮件
- 2.5 域名系统DNS
- 2.6 P2P 应用

Web和HTTP






Web和HTTP

- Web: World wide Web, 万维网, 也称Web, 是一种互联网应用。
- web: 网页, 网站
- 发展
 - ◆ Web1.0 : HTML网页
 - ◆ Web2.0: 社交网站、博客、Wiki

Web和HTTP

- Web的应用层协议是HTTP，超文本传输协议
- Web页面由对象组成
 - ◆ 一个基本HTML文件
 - ◆ 多个引用对象：图片，java小程序，视频文件
- 每个对象可由URL寻址
- URL(Uniform Resource Locator 统一资源定位符)
的例子：

`http://www.someschool.edu/someDept/pic.gif`

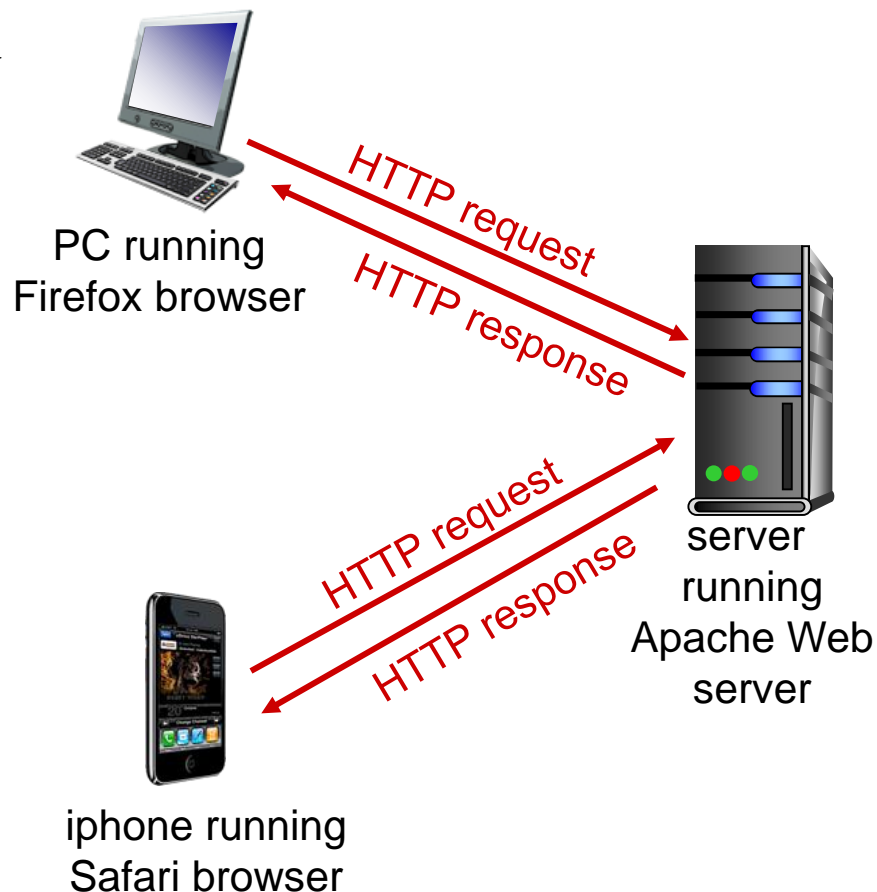
		
协议	主机名	路径名

HTTP概况

➤ HTTP协议运行在客户与服务器的应用程序中

◆ **客户机**: 请求、接收, 展示 Web对象

◆ **服务器**: Web服务器响应请求而发送对象



HTTP概述 (续)

使用TCP:

- 客户机向服务器发起TCP连接(产生套接字)，端口80
- 服务器从客户机接受TCP连接
- 在浏览器(HTTP客户机)和Web服务器(HTTP服务器)之间交换HTTP报文(应用层协议报文)
- 关闭TCP 连接

HTTP是”无状态的“

- 服务器不保留有关客户机过去请求的任何信息

思考：无状态的HTTP有什么的缺陷

非持续连接与持续连接

非持续连接HTTP

- 每个HTTP请求/响应对经过一个单独TCP连接发送
- HTTP/1.0使用非持久HTTP

持续连接HTTP

- 多个对象能够经过客户机和服务器之间的单个TCP连接发送
- HTTP/1.1以默认模式使用持久连接

非持续HTTP

假定输入URL `www.someSchool.edu/someDepartment/home.index`

1a. HTTP客户机向HTTP服务器(进程)
的80端口的`www.someSchool.edu`
发起TCP连接

(包括文本和对10个
jpeg图片的引用
images)

1b. 在主机`www.someSchool.edu`
的HTTP服务器在80端口等待
TCP连接“接受”连接，通知
客户机

2. HTTP客户机发送 HTTP *请求报
文* (包含URL)进TCP 连接套接字.
报文指示客户机要对象
`someDepartment/home.index`

3. HTTP服务器接收请求报文，形成
*响应报文*包含请求对象，并向套
接字其发送报文

time



非持久HTTP(续)

5. HTTP客户机接收包含html文件的响应报文，显示html. 解析html文件，发现10个引用的jpeg对象

4. HTTP服务器关闭TCP 连接

time



6. 对10个jpeg对象重复步骤1-5

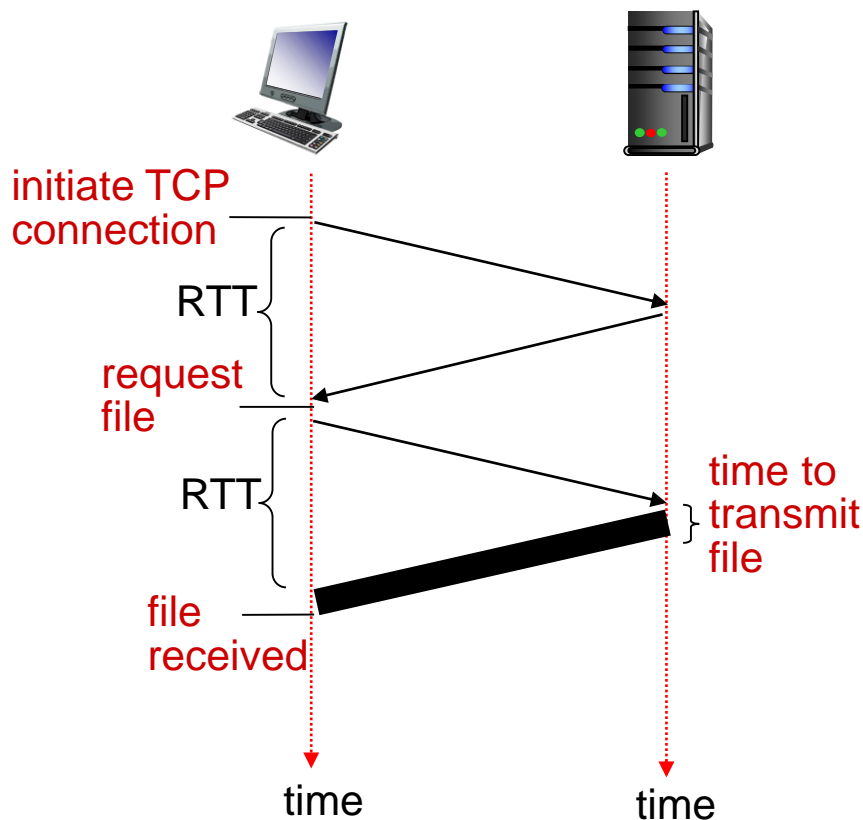
响应时间估算

往返时间RRT的定义: 从客户机到服务器发送一个小分组并返回所历经的时间.

响应时间(1个HTML文件):

- 建立TCP连接: 约1个RTT
- 对HTTP请求和响应返回的前几个字节: 约1个RTT
- 文件传输时间

获取单个对象的总时延=
 $2RTT + \text{传输时间}$



持续HTTP

非持续HTTP缺点:

- 串行访问时间长
- 并行访问资源占用多

持续HTTP

- 在发送响应后, 保持TCP连接持续打开
- 后继HTTP报文通过该连接持续发送

无流水线的持续:

- 仅当前面的响应已经收到, 客户机发出新的请求
- 对每个引用对象花费一个RTT

有流水线的持续:

- 在HTTP/1.1 为默认
- 服务器发送完响应后保存TCP连接
- 所有引用对象花费一个RTT (略多) 时间

HTTP请求报文

➤ 两类HTTP报文：请求报文，响应报文

➤ HTTP请求报文：

◆ 例子，ASCII (人可读的格式)

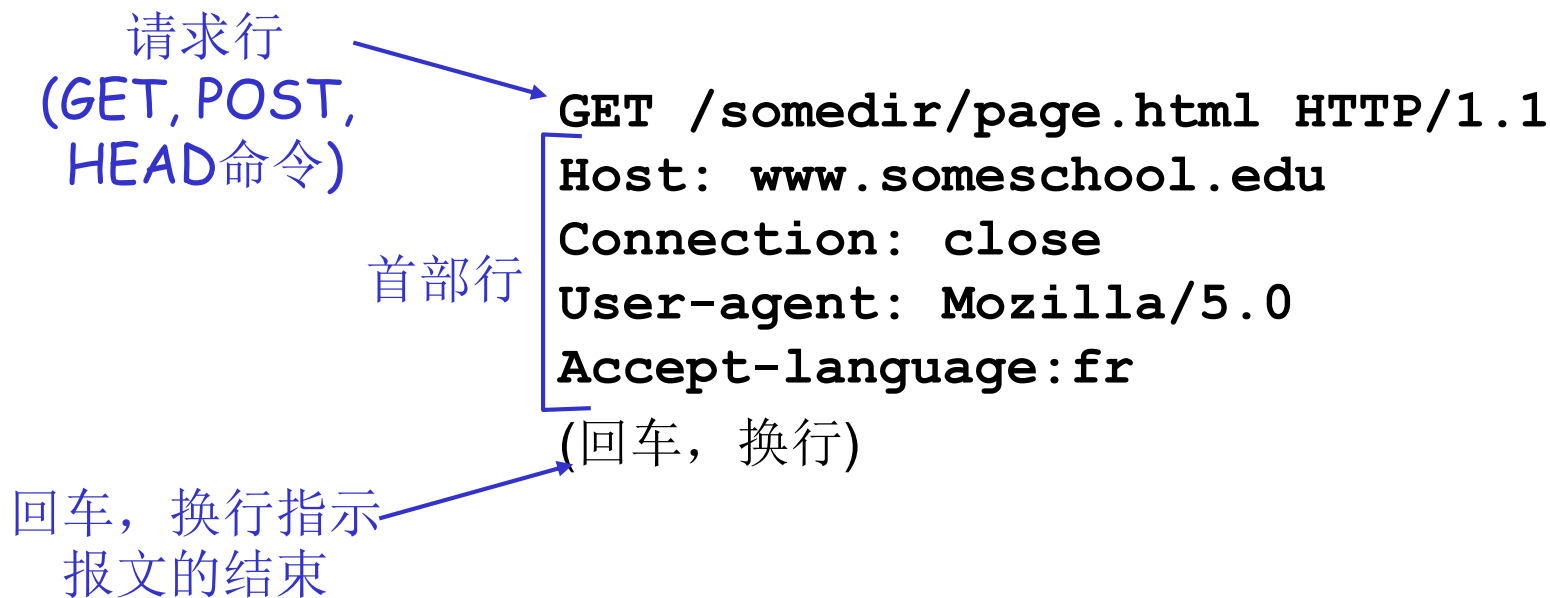
请求行
(GET, POST,
HEAD命令)

首部行

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

(回车, 换行)

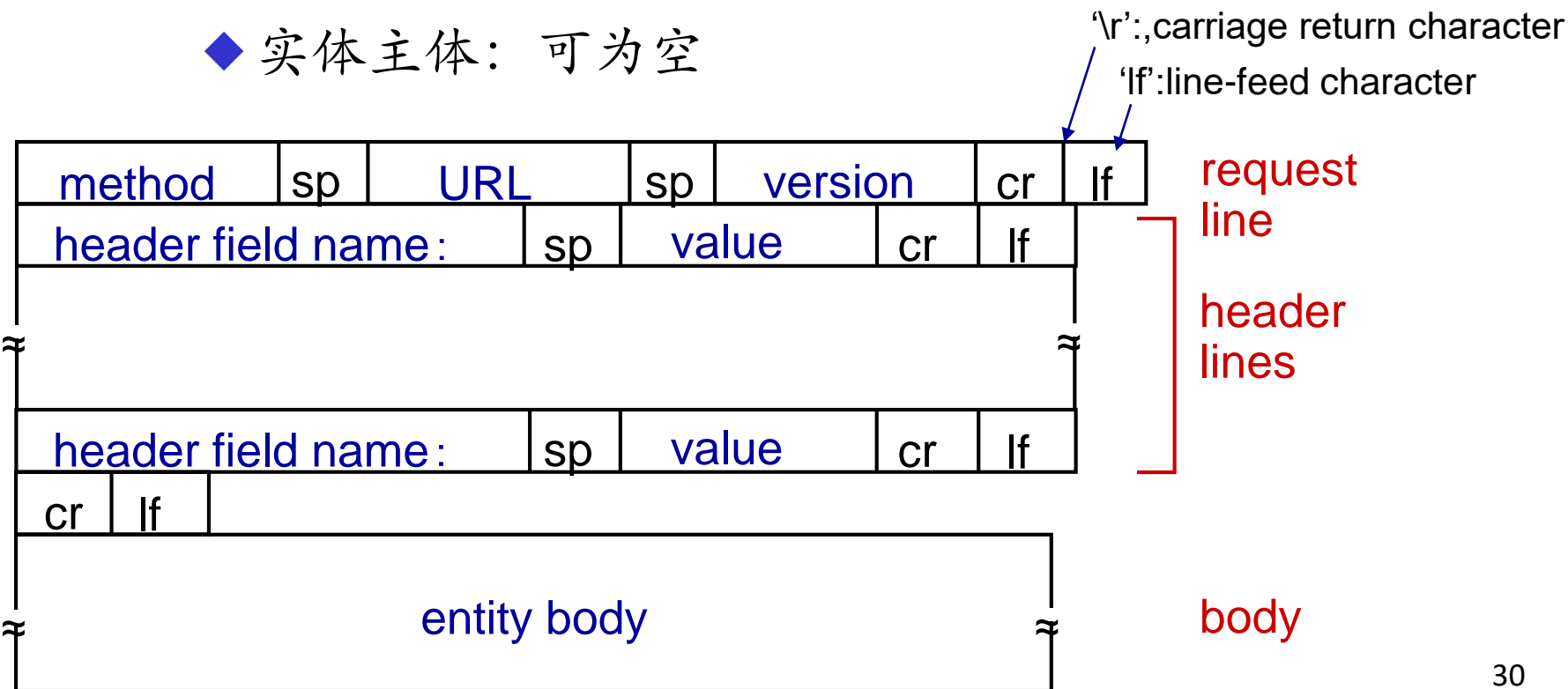
回车, 换行指示
报文的结束



HTTP请求报文: 通用格式

► 通用格式

- ◆ 请求行: 第1行, 方法字段, URL字段, 版本字段
- ◆ 首部行: 一般多行
- ◆ 实体主体: 可为空



方法类型

- GET: 最常用, 请求访问网页, 实体主体为空
- POST: 常用, 提交表单同时请求访问网页, 如使用搜索引擎, 实体主体中为表单输入值
- HEAD: 少用, 测试用, 与GET区别在于响应中去掉请求的对象
- PUT: 很少用, 向URL字段中定义的路径, 上载在实体主体中文件
- DELETE, 很少用, 删除在URL字段中定义的文件

HTTP响应报文

观察HTTP响应报文例子

状态行
(版本,状态码,状态短语)

HTTP/1.1 200 OK

首部行

Connection close

Date: Thu, 06 Aug 2011 15:44:04 GMT

Server: Apache/2.2.3 (Unix)

Last-Modified: Thu,09 Aug 2011

Content-Length: 6821

Content-Type: text/html

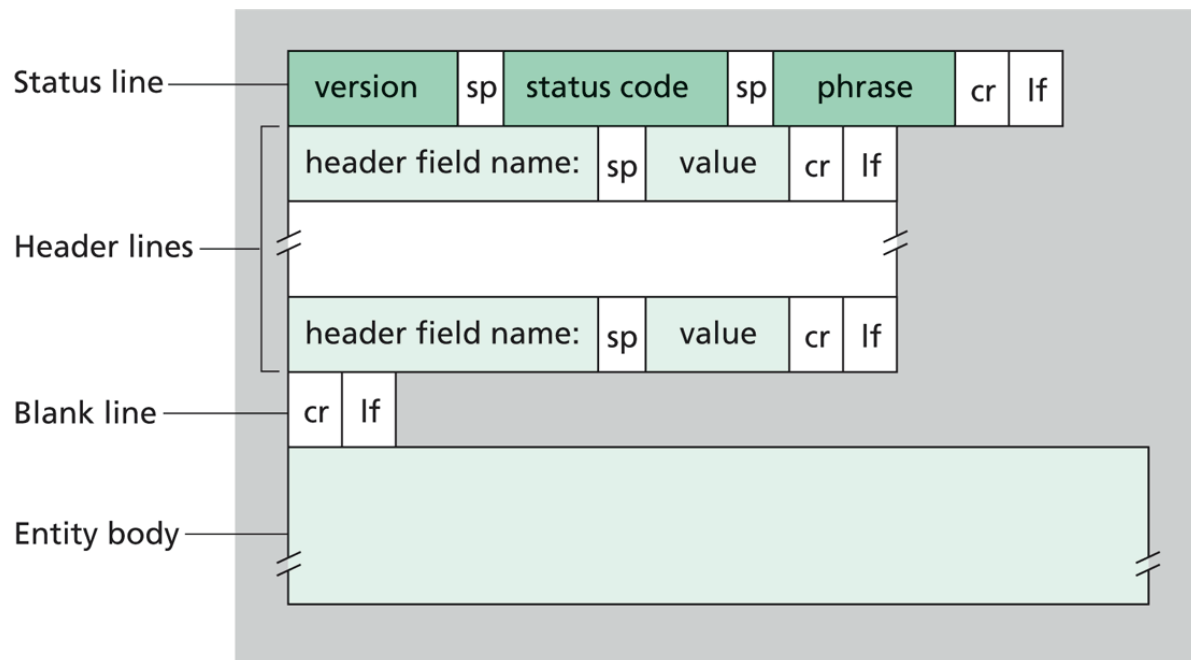
数据, 如请求的
HTML文件

data data data data data ...

HTTP响应报文: 通用格式

► 通用格式

- ◆ 状态行: 第1行, 版本字段, 状态码字段, 短语字段
- ◆ 首部行: 一般多行
- ◆ 实体主体: 报文的主体



HTTP响应状态码

在服务器到客户机响应报文中的首行.

一些编码的例子:

200 OK

- ◆ 请求成功, 请求的对象在这个报文后面

301 Moved Permanently

- ◆ 请求的对象已转移, 新的URL在响应报文的Location:首部行中指定

400 Bad Request

- ◆ 请求报文不为服务器理解

404 Not Found

- ◆ 请求的文档没有在该服务器上发现

505 HTTP Version Not Supported

用户与服务器的交互：cookie

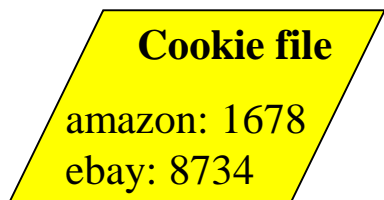
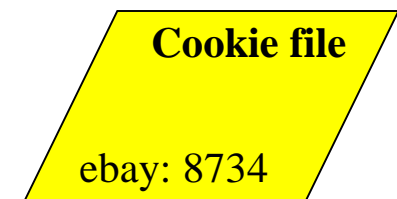
- HTTP是一种无状态协议，不能保存客户信息
- Cookie是一种在客户端保持HTTP状态信息的技术（好比：商场VIP卡）
 - ◆ 客户端访问网站时，Web服务器会查看、创建、修改Cookie资料
 - ◆ 帮助Web站点保存访问者信息：浏览历史，购物车
- cookie技术组成
 - ◆ cookie识别码（http请求报文，http响应报文）
 - ◆ 客户端保留cookie文件
 - ◆ 服务器提供后端数据库

Cookie工作过程（例）

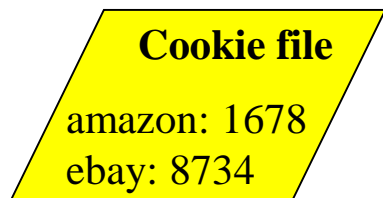
Susan访问Amazon网站

客户机

服务器



一个星期以后:



服务器为用户生成
ID 1678

后端数据库中的表项



特定cookie
动作



特定cookie
动作

访问

访问

Cookie的应用场合

Cookie的用途:

- 授权、认证
- 购物车
- 用户推荐
- 用户 session 状态

如何保持状态:

- 基于协议: 在多个事物的发送/接收报文时维护状态信息
- cookie: 通过http报文携带状态信息

Cookie: 便利与隐私:

- 为用户提供便利
- 用户的个人信息泄露

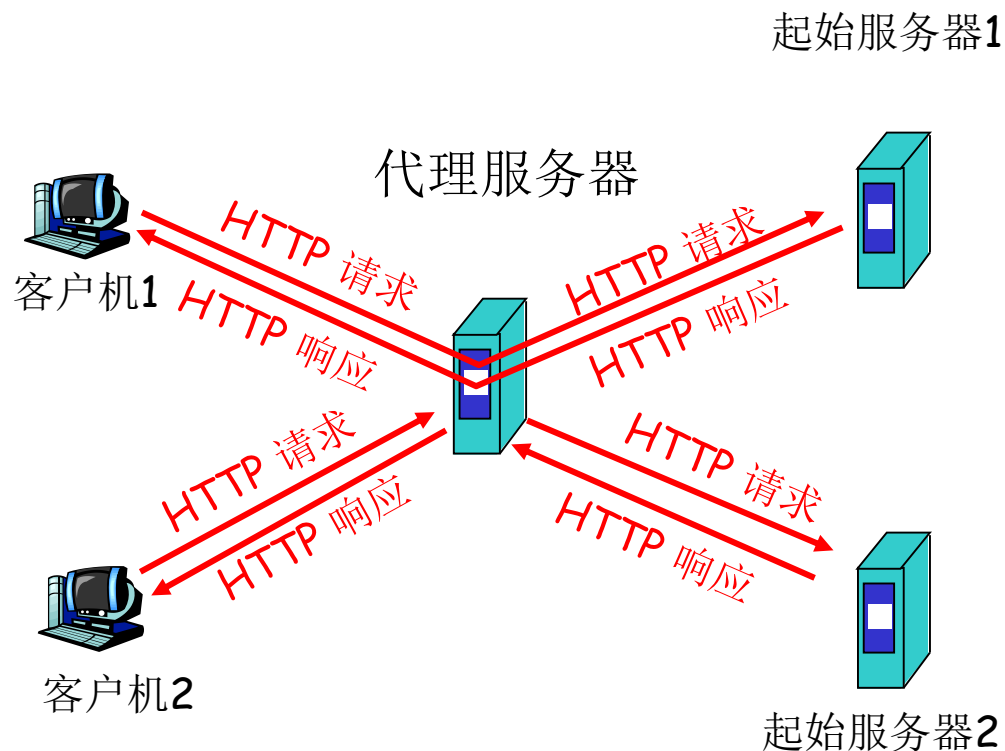
Web缓存(代理服务器)

➤ Web缓存器：保存最近请求过的web对象

➤ 浏览器向缓存发送所有HTTP请求

◆ 对象在缓存中：缓存返回对象

◆ 否则缓存向起始服务器请求对象，然后向客户机返回对象



好处：1) 减小客户机请求的响应时间；2) 减少机构内部网与因特网接入链路的通信量

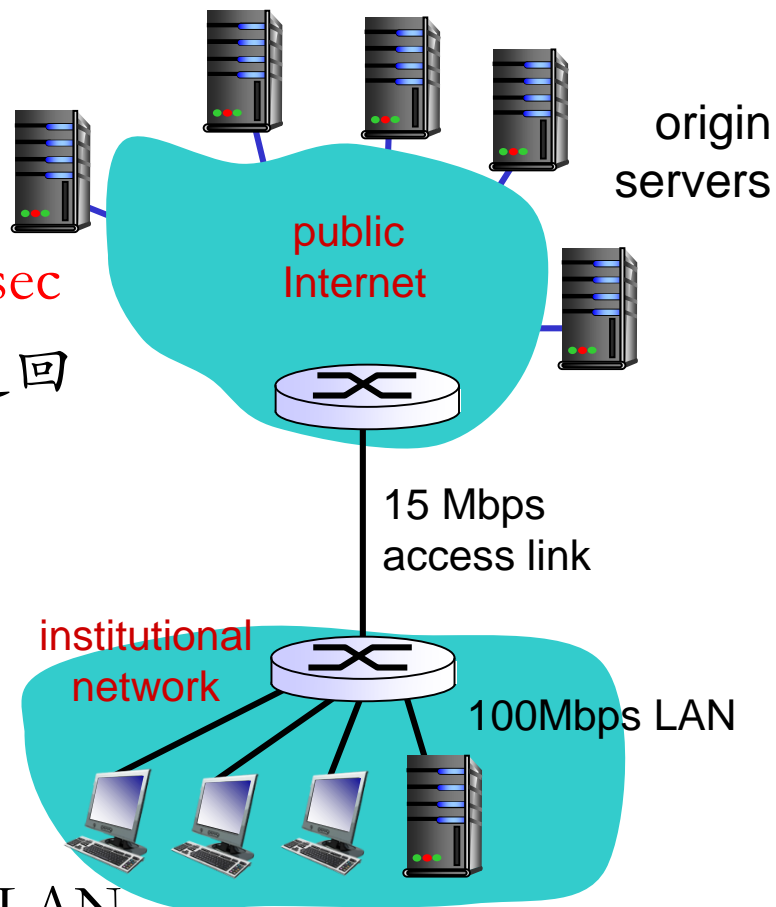
缓存例子

假定

- 平均对象长度 = 1Mb
- 来自机构的浏览器的平均请求 = 15/sec
- 从机构路由器到任何初始服务器并返回的时延 = 2 sec
- 接入网速率: 15Mbps

结果

- 局域网的使用率: 15%
- 接入链路使用率: 100%
- 总时延 = 因特网时延 + 访问时延 + LAN
时延 = 2 sec + 分钟 + 毫秒



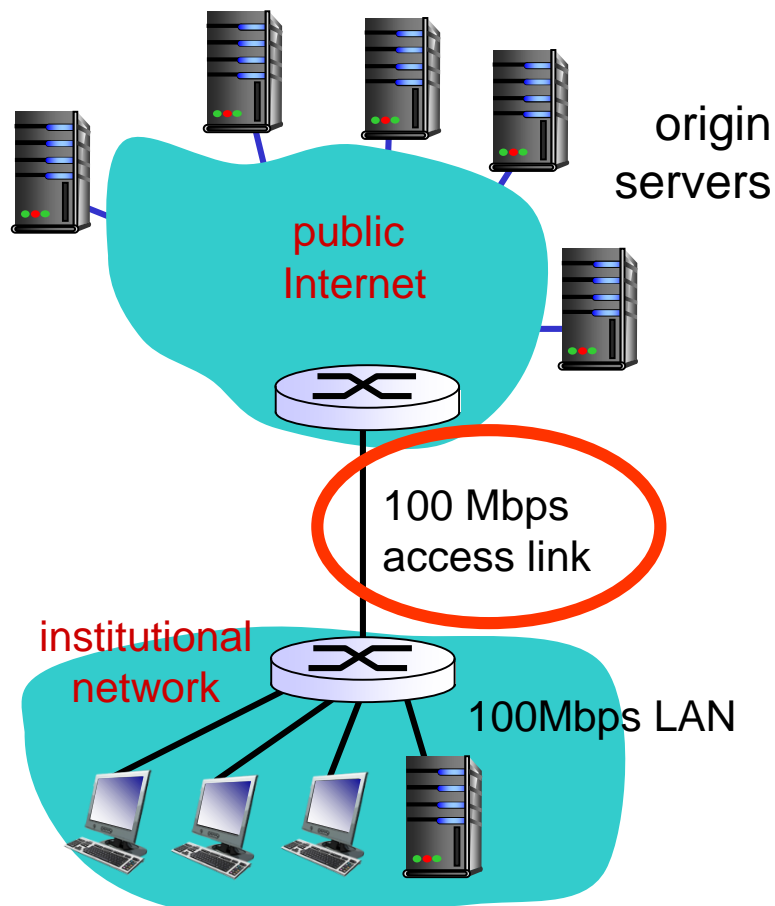
缓存例子(续)

可能的解决方案

- 将访问链路的带宽提高到如 100 Mbps

结果

- LAN利用率 = 15%
- 访问链路利用率 = 15%
- 总时延 = 因特网时延 + 访问时延 + LAN时延
= 2 sec + 毫秒 + 毫秒
- 通常升级费用可观



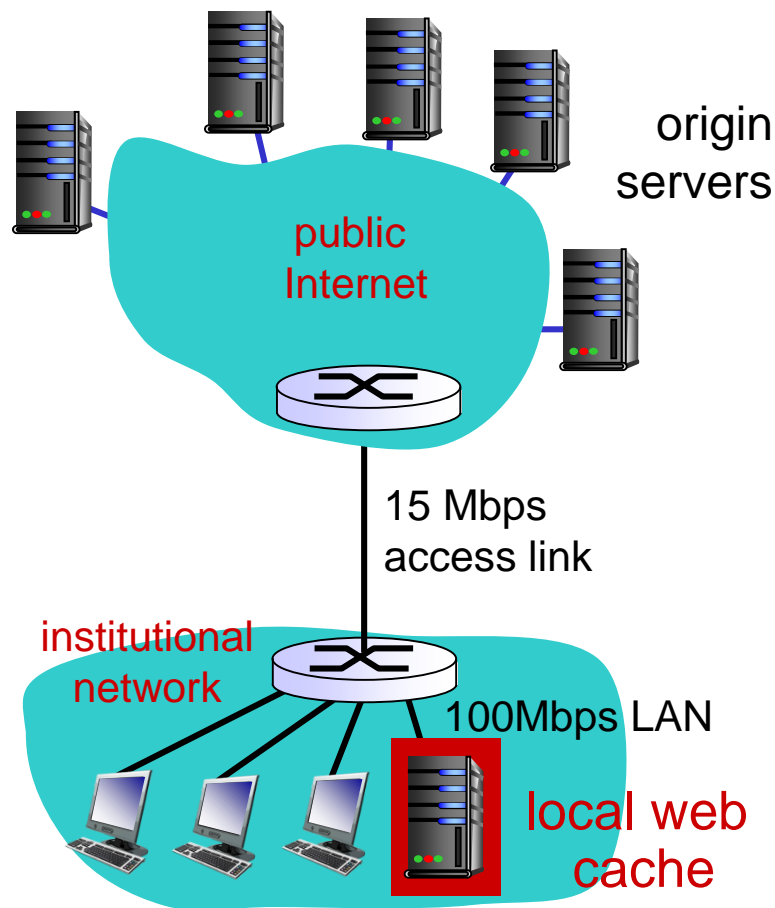
缓存例子(续)

安装缓存器

- 假定命中率是0.4

结果

- 40%请求几乎立即得到满足
- 60%请求由初始服务器满足
- 接入链路的流量强度减少到60%, 产生可忽略不计的时延 (如10 msec)
- 总平均时延 = 因特网时延 + 访问时延 + LAN时延
$$= 0.6 * 2.01 \text{ secs} + 0.4 * 10 \text{ msec}$$
$$\approx 1.2 \text{ secs}$$



条件GET方法

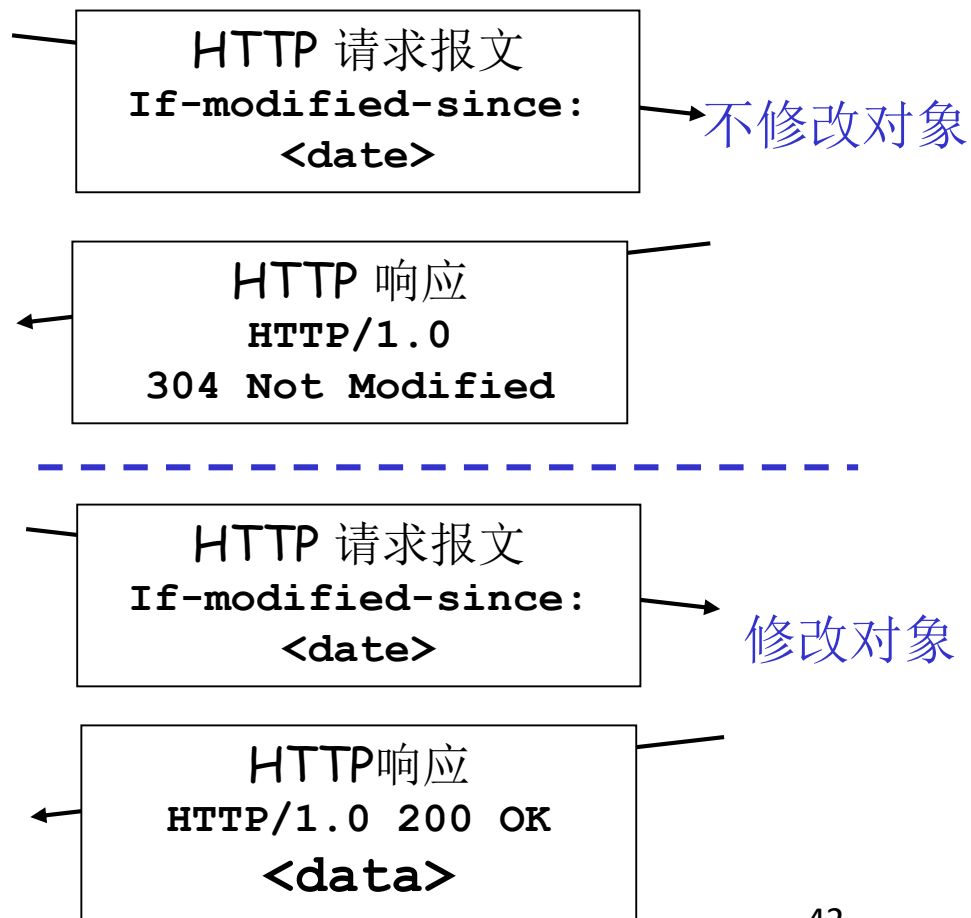
缓存中的对象可能不是最新?

解决: 条件GET

- 目的: 如果缓存中有最新缓存版本, 就不发送该对象
- 缓存器: 在HTTP请求**If-modified-since:** **<date>**中, 指定缓存版本的日期
- 服务器: 如果缓存的拷贝是最新, 响应不包含对象:
HTTP/1.0 304 Not Modified

缓存器

服务器



If-Match

如果远端资源的实体标签与在 **ETag** 这个首部中列出的值相同的话，表示条件匹配成功。默认地，除非实体标签带有 'W/' 前缀，它将会执行强验证。

If-None-Match

如果远端资源的实体标签与在 **ETag** 这个首部中列出的值都不相同的话，表示条件匹配成功。默认地，除非实体标签带有 'W/' 前缀，它将会执行强验证。

If-Modified-Since

如果远端资源的 Last-Modified 首部标识的日期比在该首部中列出的值要更晚，表示条件匹配成功。

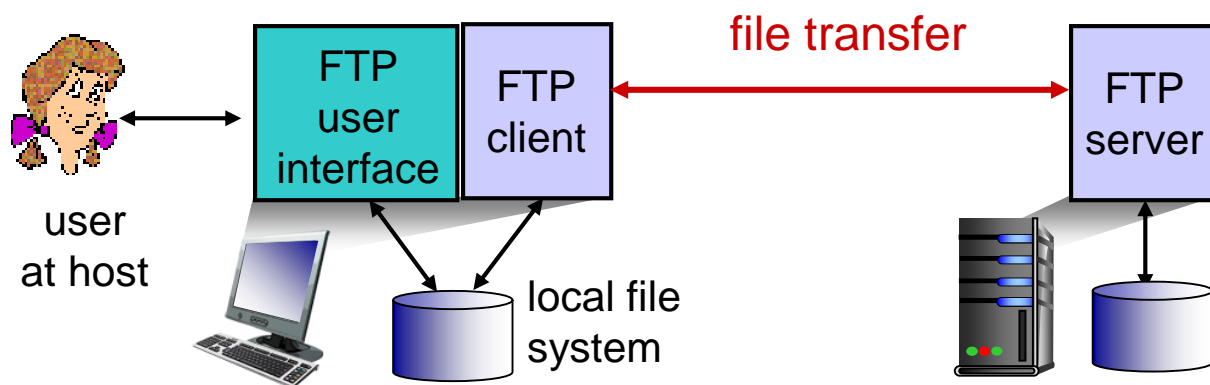
If-Unmodified-Since

如果远端资源的 HTTPHeader("Last-Modified")) 首部标识的日期比在该首部中列出的值要更早或相同，表示条件匹配成功。

- 2.1 应用层协议原理
- 2.2 Web应用和HTTP协议
- **2.3 文件传输协议：FTP**
- 2.4 电子邮件
- 2.5 域名系统DNS
- 2.6 P2P 应用

FTP: 文件传输协议

- 提供网络中的文件共享与传输
- FTP早于HTTP协议约10年，互联网还相对封闭



- 传输文件到/来自远程主机
- 客户机/服务器模型
 - ◆ **客户机**: 发起传输的一侧(到/来自远程之一)
 - ◆ **服务器**: 远程主机

FTP协议特点

- 双TCP连接
 - ◆ 控制连接，端口21（带外控制）
 - ◆ 数据连接，端口20
- 通过操作命令实现远程交互式访问
 - ◆ 登录，用户名，密码
 - ◆ 浏览文件
 - ◆ 选择文件，下载



FTP命令, 响应

命令示例:

- 经控制信道以ASCII 文本发送
- **USER *username***
- **PASS *password***
- **LIST** 返回当前目录中的文件列表
- **RETR *filename*** 获取(get) 文件
- **STOR *filename*** 存储 (puts) 文件到远程主机

返回码示例:

- 状态码和短语(如在HTTP中的那样)
- **331 Username OK, password required**
- **125 data connection already open; transfer starting**
- **425 Can't open data connection**
- **452 Error writing file**

- 2.1 应用层协议原理
- 2.2 Web应用和HTTP协议
- 2.3 文件传输协议：FTP
- **2.4 电子邮件**
- 2.5 域名系统DNS
- 2.6 P2P 应用

电子邮件系统

电子邮件系统组成:

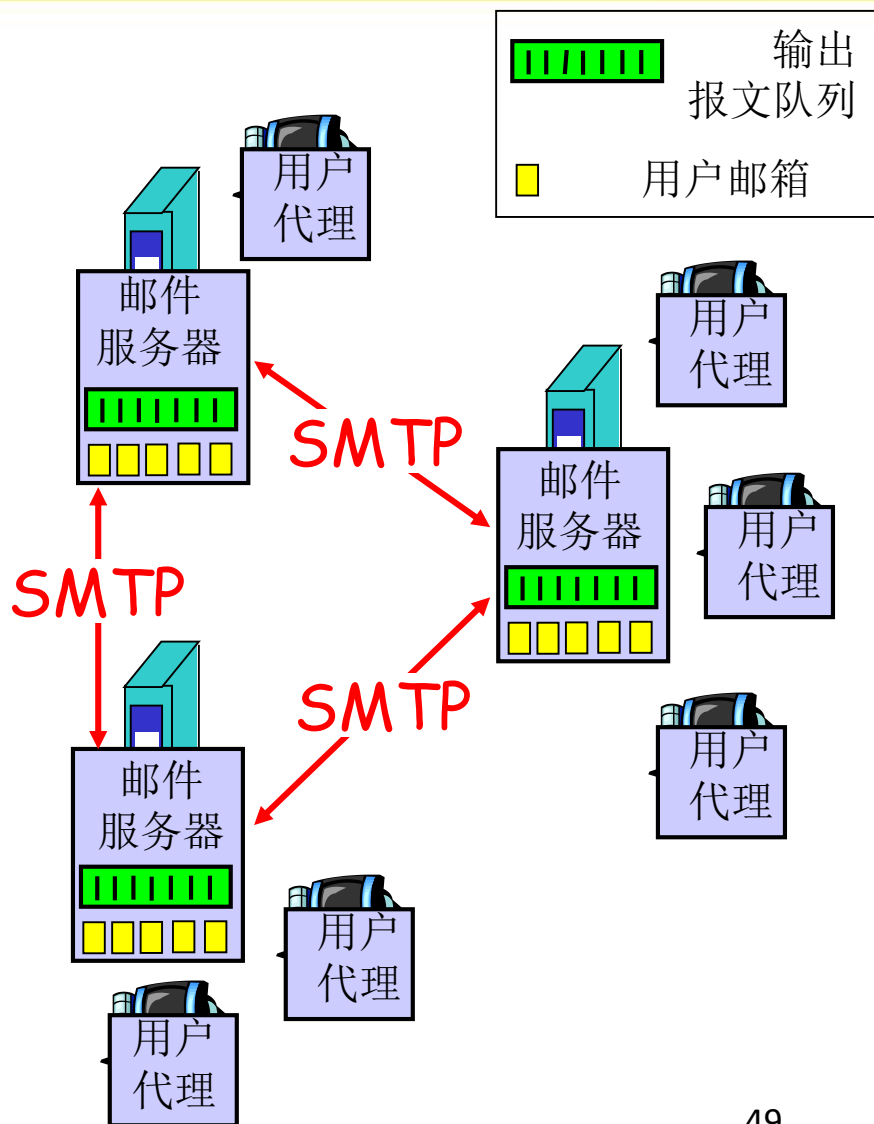
- 用户代理:客户端
- 邮件服务器
- 简单邮件传输协议: SMTP

用户代理

- 亦称为“邮件阅读器”
- 例如Outlook, Foxmail

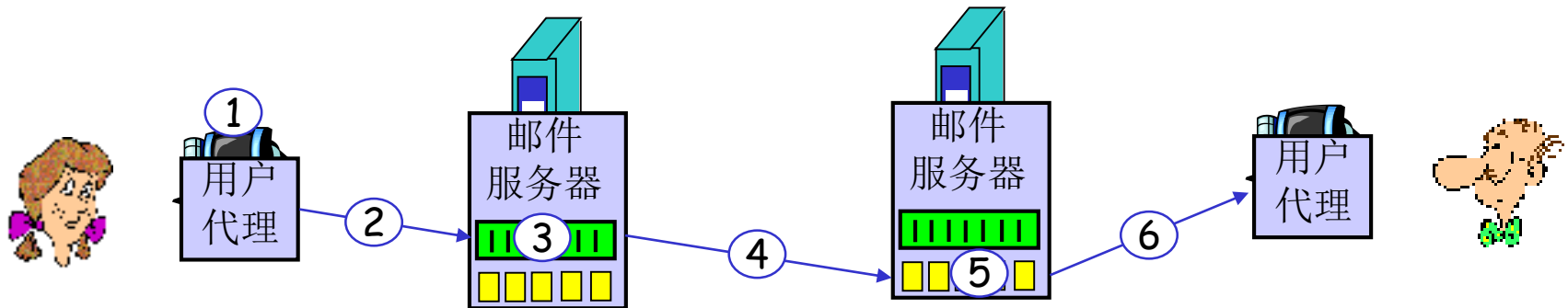
邮件服务器

- 邮件队列
- 存储转发, 非实时通信



场景: Alice 向 Bob 发送报文

- 1) Alice使用UA写作报文并向 `bob@someschool.edu` 发送
- 2) Alice的UA向其邮件服务器发送报文；报文放置在报文队列中
- 3) SMTP的客户机侧打开与Bob的邮件服务器的TCP连接
- 4) SMTP通过TCP连接发送Alice的报文
- 5) Bob'的邮件服务器将该报文放入Bob的邮箱
- 6) Bob调用其用户代理来读报文



电子邮件协议

- SMTP（简单邮件传输协议）：最常用的电子邮件传输协议
- POP3（邮局协议）：最常用的电子邮件接收协议
- IMAP4（因特网邮件访问协议）：POP3的替代协议，提供邮件处理新功能
- 另：HTTP协议也用于电子邮件，网页访问邮箱

- SMTP是一个相对简单的基于文本的协议
 - ◆报文必须以7比特ASCII格式
 - ◆二进制文件可通过MIME编码后再传
- SMTP服务器端使用端口号25
- 采用命令/响应交互
 - ◆命令：HELO, MAIL FROM, RCPT TO, DATA, QUIT
 - ◆响应：状态码及短语
- 传输的三个阶段
 - ◆握手, 传输, 关闭

SMTP交互的示例

```
S: 220 hamburger.edu (220 Service ready)
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr> (250:Requested mail
S: 250 alice@crepes.fr... Sender okaction okay, completed
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles? (221:Service closing
C: . transmission channel)
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

邮件报文格式

➤ 首部行,如

◆To:

◆From:

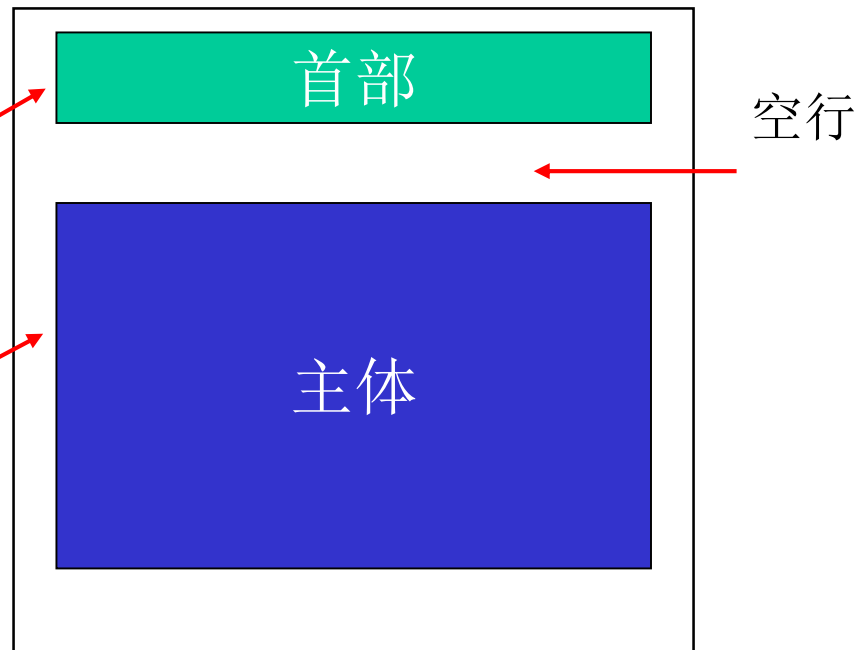
◆Subject:

不同于SMTP命令!

➤ 主体

◆“报文”, 均为ASCII 字符

➤ CRLF.CRLF结束

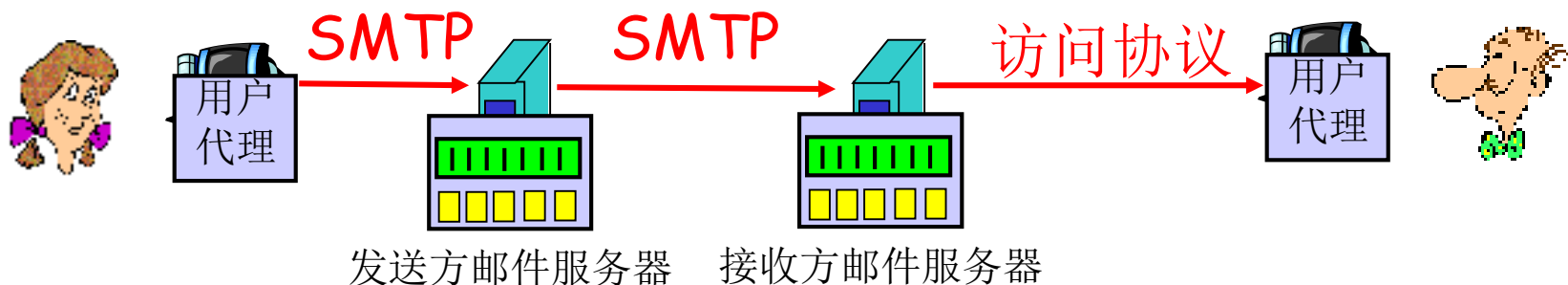


报文格式: 多媒体扩展

- MIME: **M**ultipurpose **I**nternet **M**ail **E**xtension (多用途因特网邮件扩展)
- 在报文首部的附加行声明MIME内容类型



邮件访问协议(POP3&IMAP4)



➤ 邮件访问协议：从服务器获取邮件

◆ POP3: 邮局协议 (Post Office Protocol)

◆ 授权 (代理 <--> 服务器) 并下载

◆ IMAP4: 因特网邮件访问协议

◆ 更多功能

◆ 操作存储在服务器上的报文

基于Web的电子邮件

- 今天越来越多的用户使用基于Web的电子邮件
 - ◆ Hotmail, 163邮箱等等
- 基于Web的电子邮件
 - ◆ UA是 浏览器
 - ◆ 使用 HTTP而不是SMTP发送到源邮件服务器
 - ◆ 使用SMTP 发送邮件到目的邮件服务器

- 2.1 应用层协议原理
- 2.2 Web应用和HTTP协议
- 2.3 文件传输协议：FTP
- 2.4 电子邮件
- 2.5 域名系统DNS
- 2.6 P2P 应用

DNS: 域名系统

➤ DNS: Domain Name System 域名系统

- ◆ 同时: Domain Name Server 域名服务器, DNS 协议
- ◆ 分布式数据库, 提供名字服务

人: 许多标识符

- ◆ 名字, 身份证号, 学号, 护照号

因特网主机、路由器:

- ◆ 主机名: www.nwpu.edu.cn
- ◆ IP地址(32 bit) : 113.107.238.154

问题: IP地址和名字之间的映射?

DNS完成主机名
到IP地址的解析

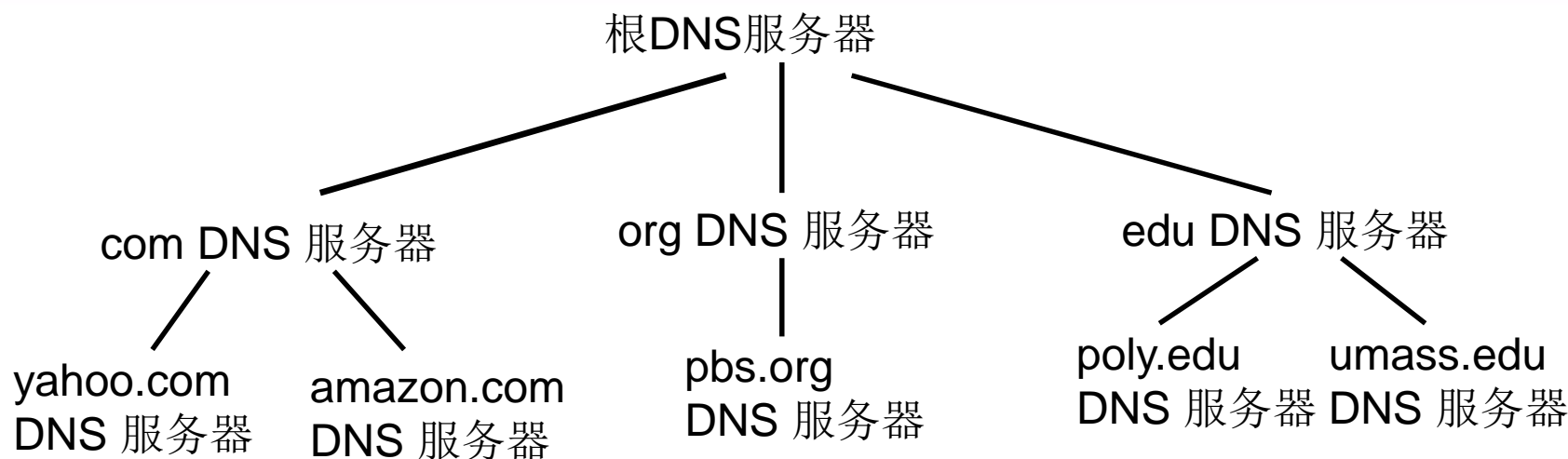
DNS服务

- 主机名到IP地址的转换
- 主机别名：
n 主机名 → 1 IP
- 负载分配
1 主机名 →
- 邮件服务器别名
Foxmail/qq.com

为何不用集中式DNS?

- 单点故障
- 通信量
- 远距离集中式数据库
- 维护

分布式、等级制数据库



➤ 三类DNS服务器

- ◆ 根DNS服务器
- ◆ 顶级域(TLD)服务器
- ◆ 权威DNS服务器

➤ 本地DNS服务器：代理

➤ 域名结构

- ◆ 根域：(.)
- ◆ 顶级域：.com, .cn
- ◆ 二级域：edu.cn
- ◆ 三级域：nwpu.edu.cn
- ◆ 更多级别：<http://rjwdz.nwpu.edu.cn/>

DNS服务器

- **根DNS服务器:** (逻辑) 全球13个, 美国10个, 欧洲1个, 亚洲1个
- **顶级域(TLD)服务器:** 负责com, org, net, edu等, 以及所有顶级国家域cn, uk, fr, ca, jp.
- **权威DNS服务器:** 组织的DNS 服务器为组织的服务器(如Web和电子邮件)提供对IP映射的权威主机名: 国内10万余台

查询策略: 逐级查询 (递归或迭代)

➤ 递归查询

在该模式下DNS 服务器接收到客户机请求，必须使用一个准确的查询结果回复客户机。如果DNS 服务器本地没有存储查询DNS 信息，那么该服务器会询问其他服务器，并将返回的查询结果提交给客户机。

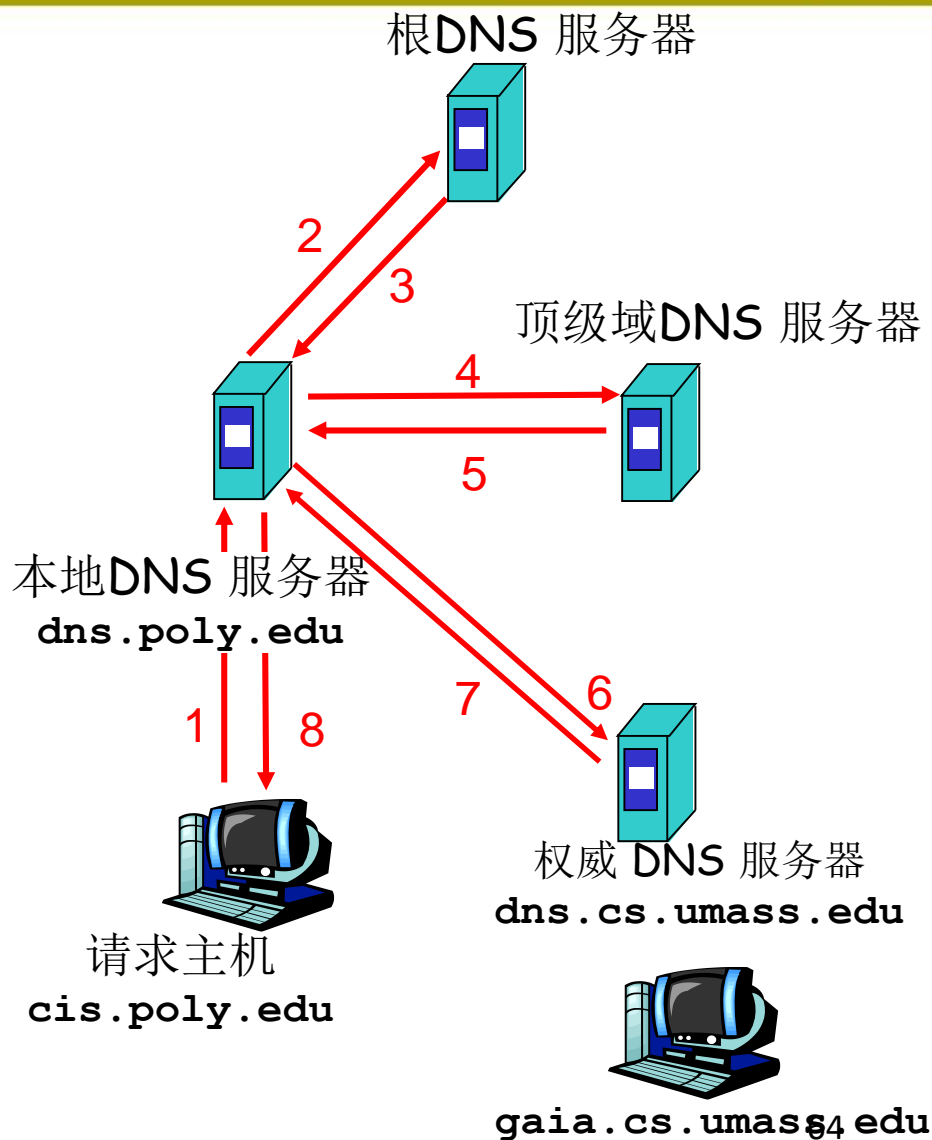
➤ 迭代查询

DNS 服务器会向客户机提供其他能够解析查询请求的DNS 服务器地址，当客户机发送查询请求时，DNS 服务器并不直接回复查询结果，而是告诉客户机另一台DNS 服务器地址，客户机再向这台DNS 服务器提交请求，依次循环直到返回查询的结果为止。

例子：迭代查询

➤ 位于cis.poly.edu的主机为
gaia.cs.umass.edu 要求IP地址

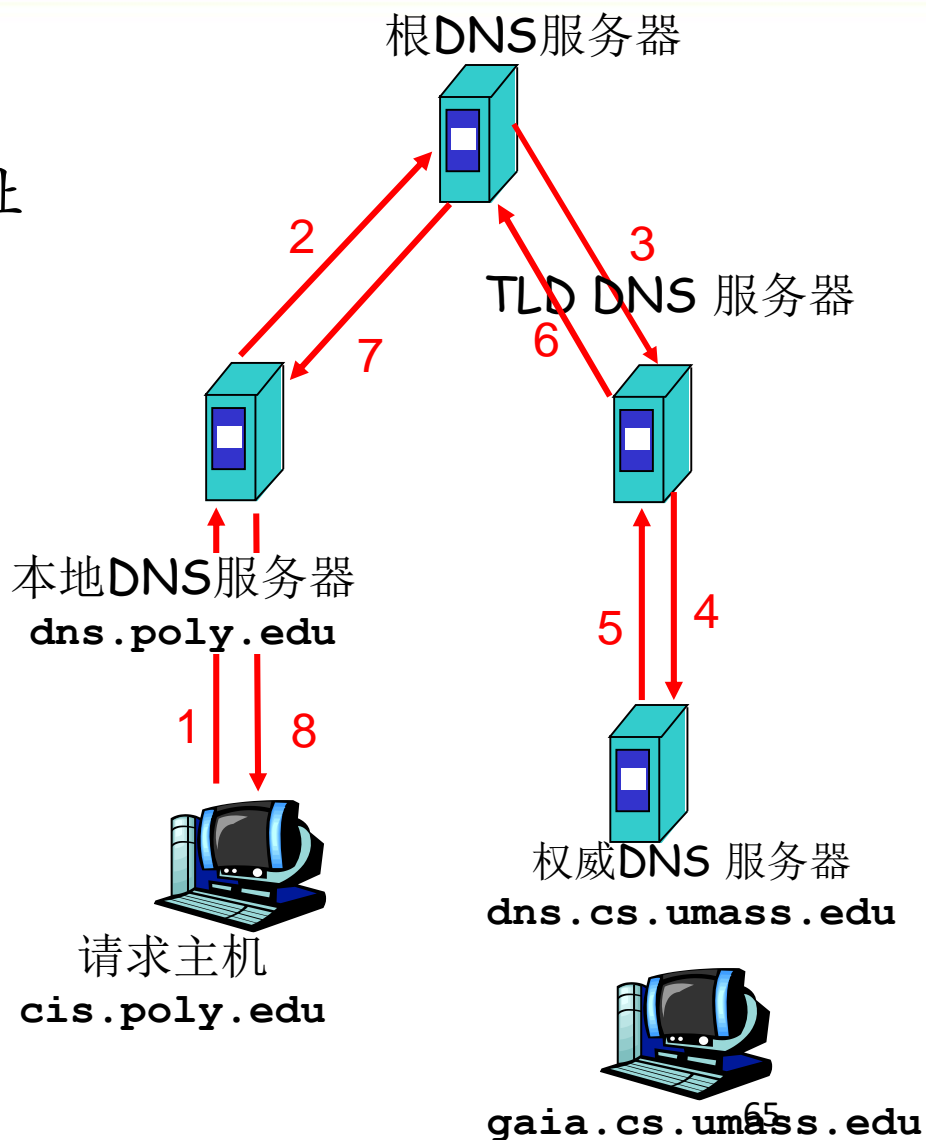
1. 客户端向本地DNS服务器发起查询:请求gaia.cs.umass.edu的IP
2. 本地DNS向根DNS查询: .edu
顶级域DNS服务器地址
3. 根DNS回应.edu顶级域DNS服务器地址
4. 本地DNS向.edu顶级域DNS查询.umass.edu的权威DNS服务器地址
5. ~8 ...



例子：递归查询

➤ 位于cis.poly.edu的主机为
gaia.cs.umass.edu 要求IP地址

1. 客户端向本地DNS服务器发起查询:请求gaia.cs.umass.edu的IP
2. 本地DNS向根DNS查询: 请求gaia.cs.umass.edu的IP
3. 根DNS向.edu顶级域DNS服务器查询: 请求gaia.cs.umass.edu的IP
4. .edu顶级域DNS向.umass.edu的权威DNS查询:
5. ~8 ...



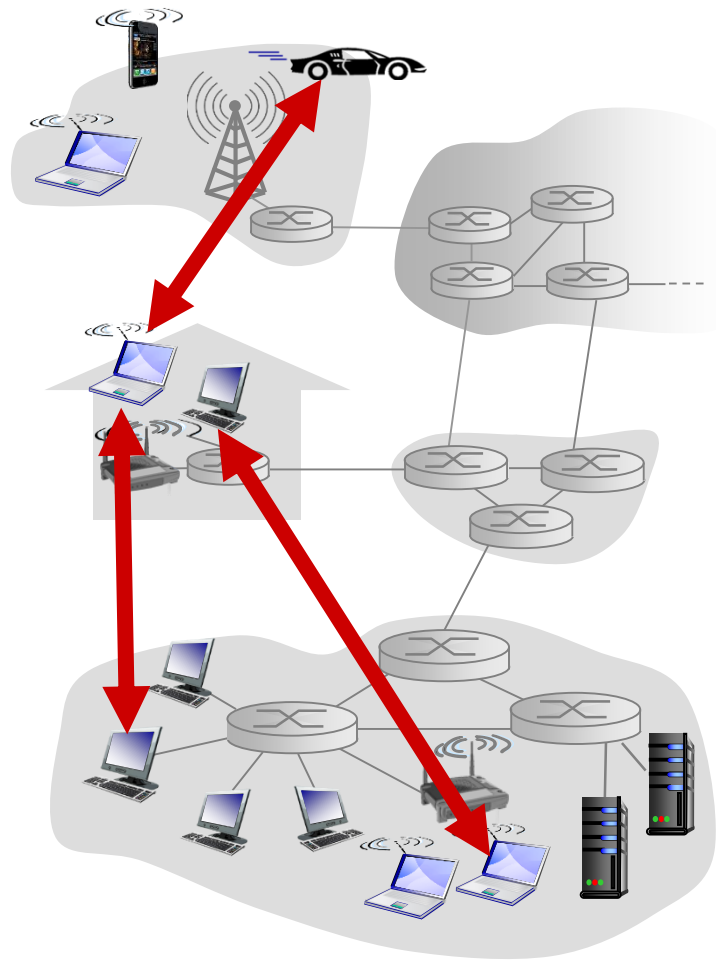
纯P2P

➤ P2P体系结构

- 没有始终运行着的服务器
- 任意的端系统（称之为对等方）之间直接通信
- 对等方之间间歇性的互联，其IP地址也可能改变

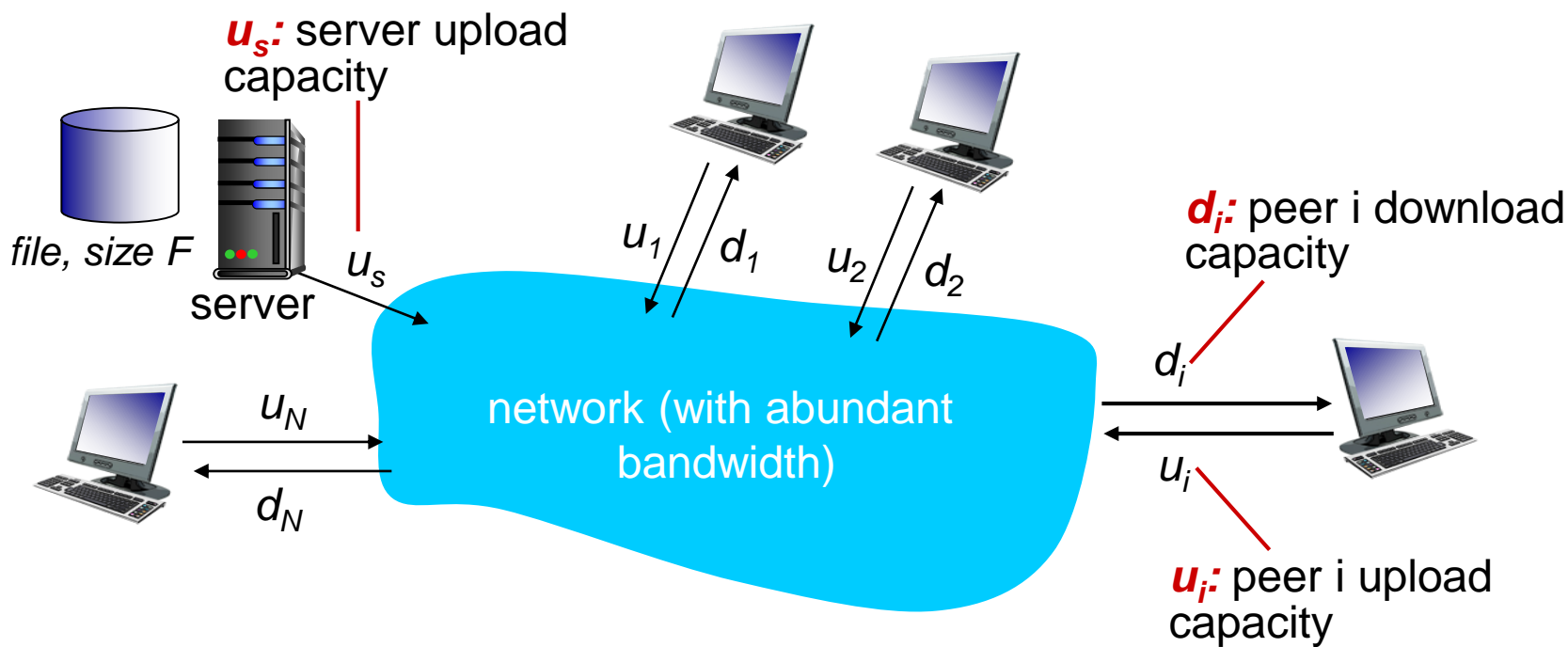
➤ 例如：

- 文件分发
- 流媒体
- VoIP



文件分发: CS vs P2P

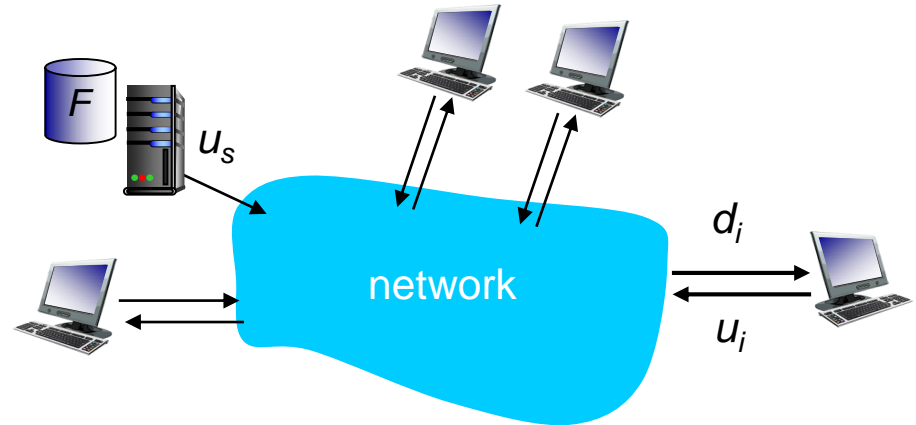
- Q: 将文件 (大小为 F) 从服务器分发到 N 个节点, 需要多少时间?
- 每个节点上传/下载的数据容量是有限的



File distribution time: client-server

➤ 文件分发时间：客户-服务器模式

- 服务器处理：必须顺序的发送（上传） N 个拷贝： NF/u_s
- 客户端：每个客户端下载文件拷贝： F/d_{\min}



*time to distribute F
to N clients using
client-server approach*

$$D_{c-s} \geq \max\{NF/u_s, F/d_{\min}\}$$

Application Layer

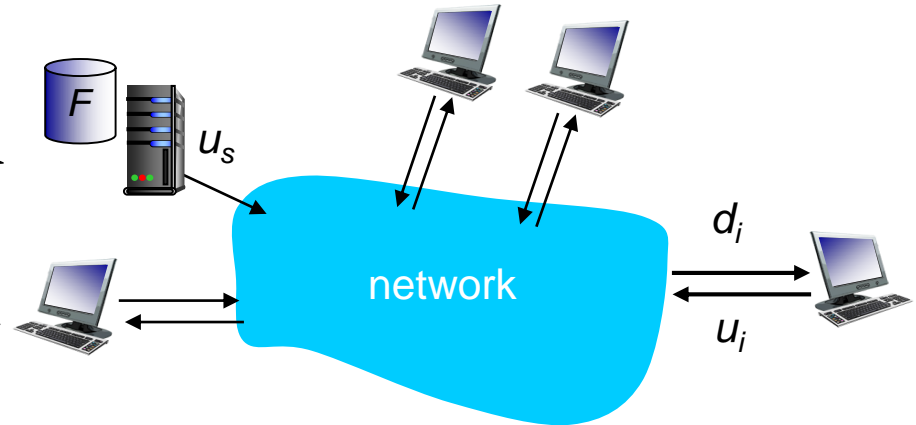
increases linearly in N

2-68

File distribution time: P2P

文件分发时间：P2P

- 服务器处理：至少上传一个文件拷贝
- 客户端：每个客户端下载一个文件拷贝
- 全部客户端总下载量为 NF bits



*time to distribute F
to N clients using
P2P approach*

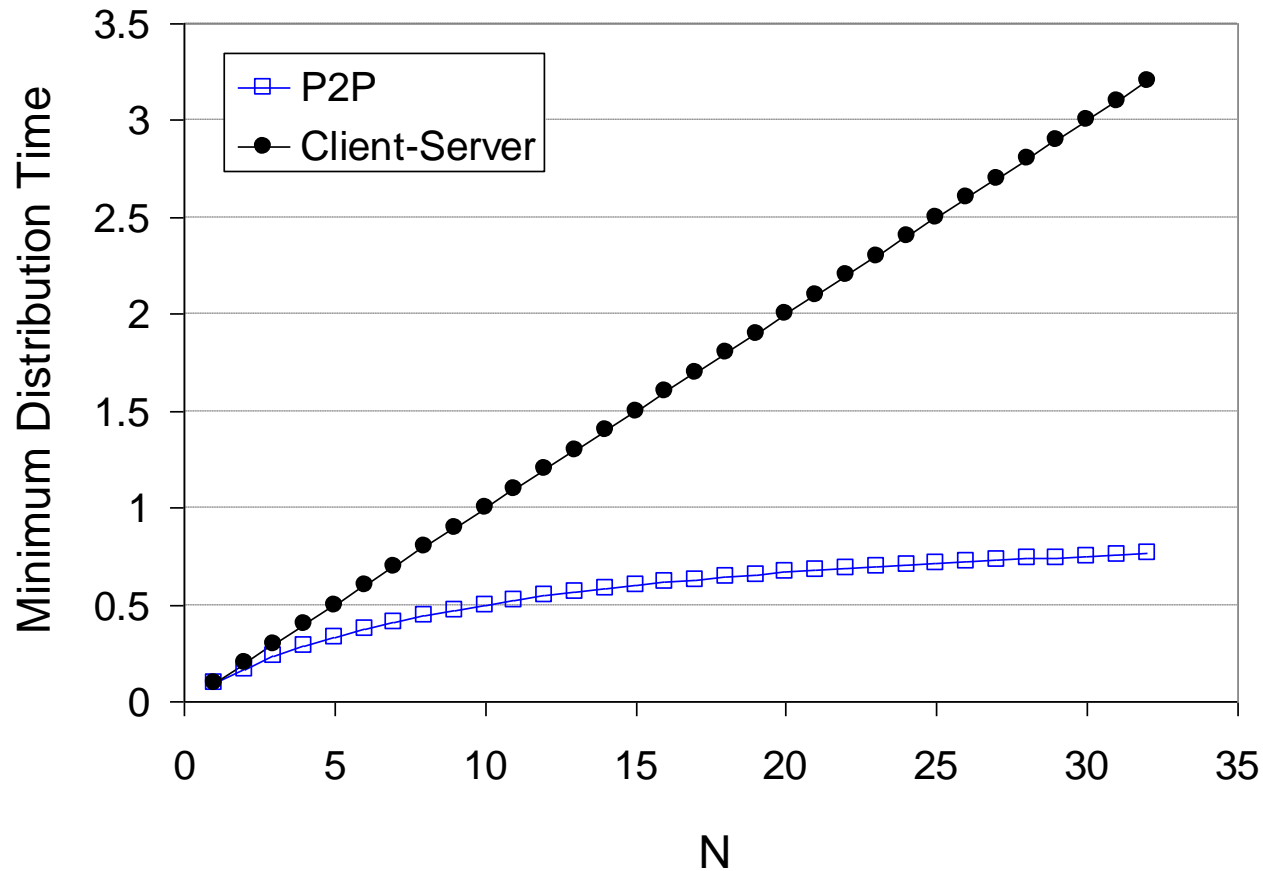
$$D_{P2P} \geq \max\{F/u_s, F/d_{min}, NF/(u_s + \sum u_i)\}$$

increases linearly in N ...

... but so does this, as each peer brings service capacity

Client-server vs. P2P: example

client upload rate = u , $F/u = 1$ hour, $u_s = 10u$, $d_{min} \geq u_s$



本章小结与作业

重要内容:

- 网络应用程序体系结构。
- 各个应用层协议 (HTTP,FTP,SMTP) 的作用及端口。
- HTTP的持久连接与非持久连接。
- DNS的作用, 层次, 域名解析过程。
- 访问网站的步骤。

作业:

- R5, R12, R15, R19, R21
- P4, P5, P22
- 思考题: R11,