



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

《计算机网络》

第2章 应用层

主讲人：聂兰顺



主要内容

本章学习目标

- ❖ 掌握网络应用体系结构
- ❖ 理解网络应用通信原理
- ❖ 掌握域名系统及域名解析过程
- ❖ 掌握FTP应用及应用层协议
- ❖ 掌握email应用及应用层协议
- ❖ 掌握Web应用及应用层协议
- ❖ 掌握Web响应时间优化方案
- ❖ 掌握P2P应用原理及特点
- ❖ 掌握P2P文件分发原理
- ❖ 掌握P2P索引技术
- ❖ 掌握Socket编程技术

主要内容

- ❖ 2.1 网络应用体系结构
- ❖ 2.2 网络应用通信原理
- ❖ 2.3 域名解析系统(DNS)
- ❖ 2.4 FTP应用
- ❖ 2.5 Email应用
- ❖ 2.6 Web应用
- ❖ 2.7 P2P应用
- ❖ 2.8 Socket编程





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

2.1 网络应用体系结构

聂生顺



网络应用的体系结构

2.1 网络应用体系结构

- ❖ 客户机/服务器结构(Client-Server, C/S)
- ❖ 点对点结构(Peer-to-peer, P2P)
- ❖ 混合结构(Hybrid)





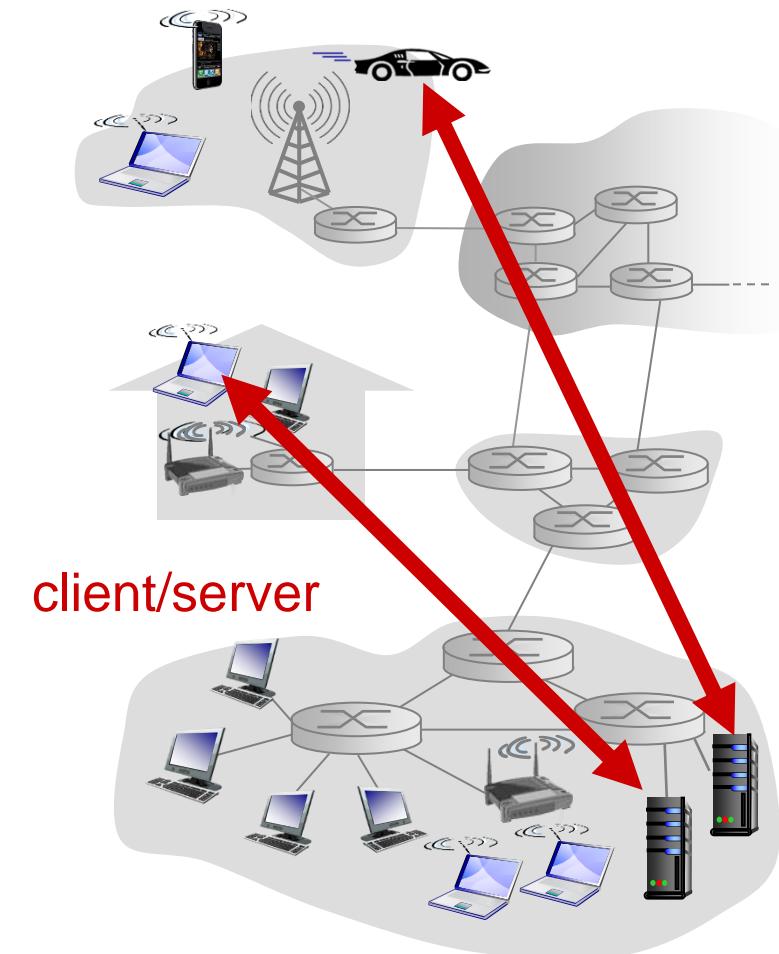
客户机/服务器结构

❖ 服务器 (server)

- 7*24小时提供服务
- 永久性可访问地址/域名
- 利用大量服务器实现可扩展性

❖ 客户机 (client)

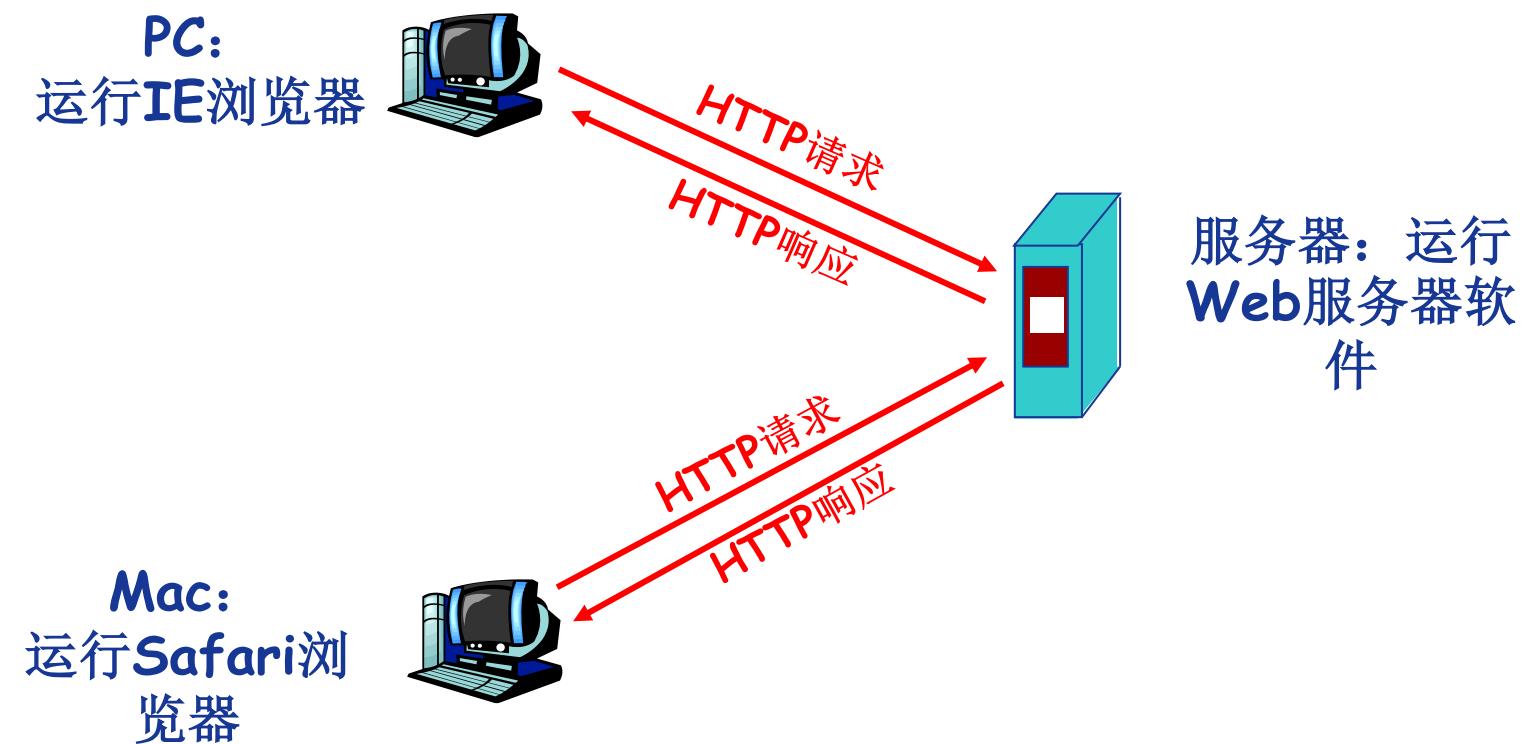
- 与服务器通信，使用服务器提供的服务
- 间歇性接入网络
- 可能使用动态IP地址
- 不会与其他客户机直接通信





客户机/服务器结构

例子：Web



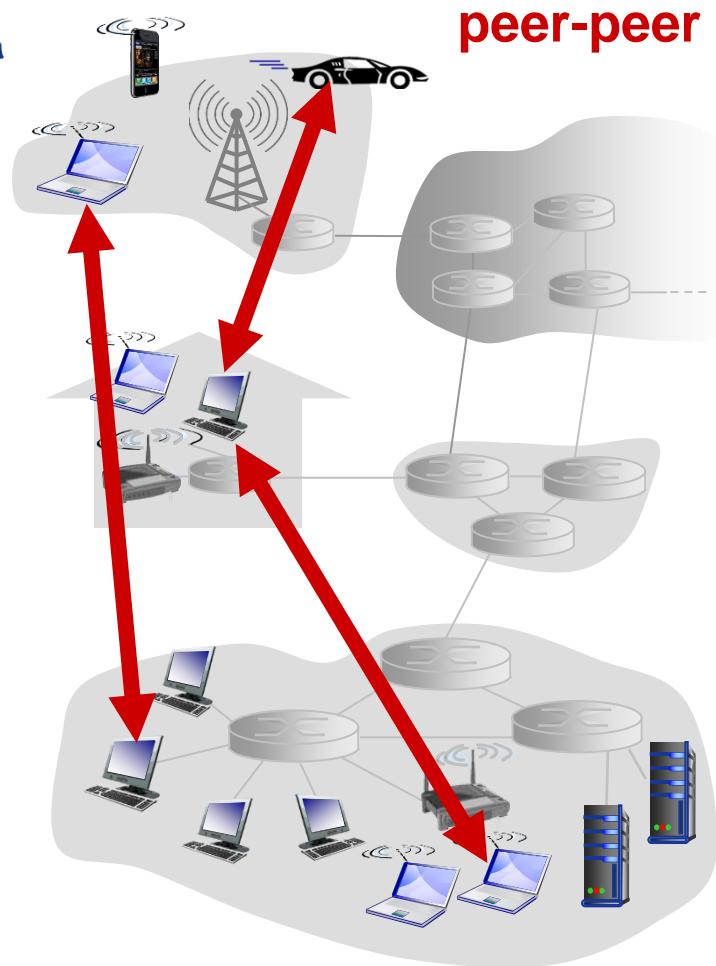


纯P2P结构

2.1 网络应用体系结构

- ❖ 没有永远在线的服务器
- ❖ 任意端系统/节点之间可以直接通讯
- ❖ 节点间歇性接入网络
- ❖ 节点可能改变IP地址

- ❖ 优点：高度可伸缩
- ❖ 缺点：难于管理





混合结构

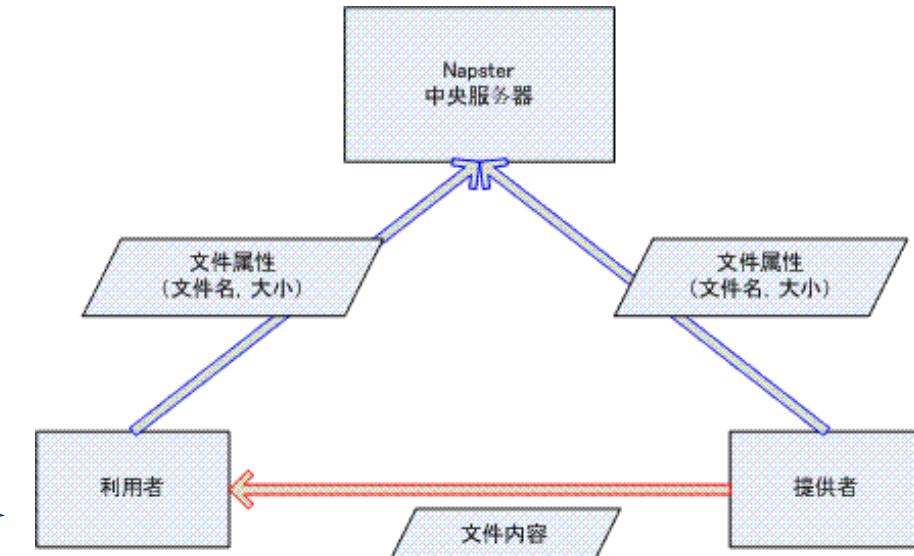


能否将两种结构混合在一起使用？

混合能够利用两者的优点同时规避两者的缺点吗？

❖ Napster

- 文件传输使用P2P结构
- 文件的搜索采用C/S结构——集中式
 - 每个节点向中央服务器登记自己的内容
 - 每个节点向中央服务器提交查询请求，查找感兴趣的内容





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

2.2 网络应用通信原理

聂生顺



2.1 网络应用体系结构

2.2 网络应用通信原理

网络应用的基础：进程间通信

❖ 进程：

- 主机上运行的程序

❖ 同一主机上运行的进程之间如何通信？

- 进程间通信机制
- 操作系统提供

❖ 不同主机上运行的进程间如何通信？

- 消息交换

客户机进程：发起通信的进程

服务器进程：等待通信请求的进程



采用P2P架构的应用
是否存在客户机进程/
服务器进程之分？

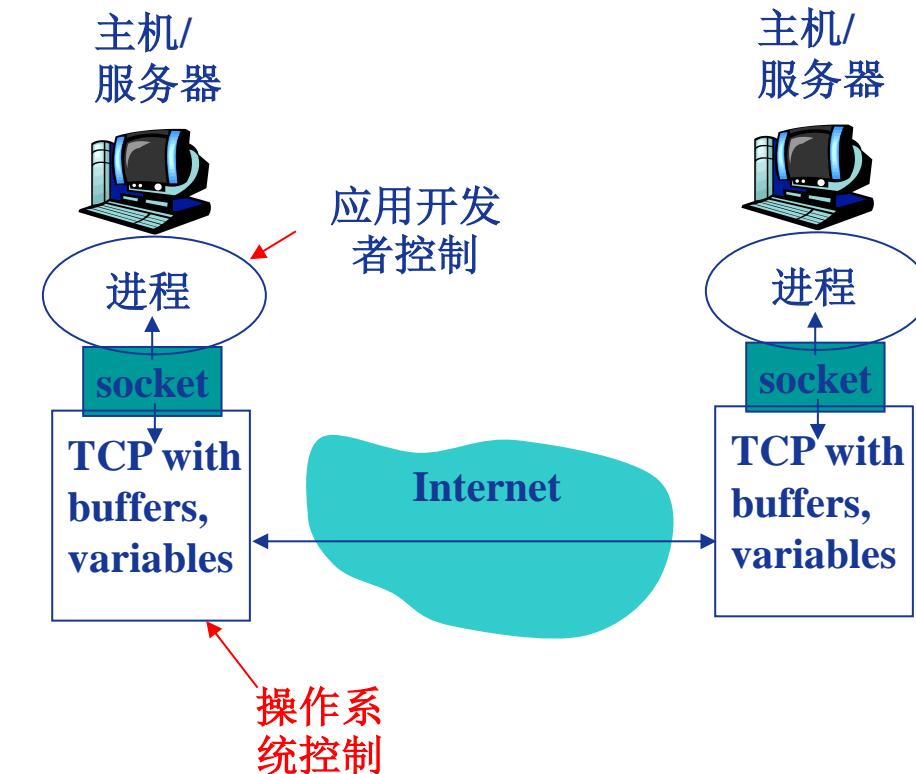


2.1 网络应用体系结构

2.2 网络应用通信原理

套接字: Socket

- ❖ 进程间通信利用socket发送/接收消息实现
- ❖ 类似于寄信
 - 发送方将消息送到门外邮箱
 - 发送方依赖（门外的）传输基础设施将消息传到接收方所在主机，并送到接收方的门外
 - 接收方从门外获取消息
- ❖ 传输基础设施向进程提供API
 - 传输协议的选择
 - 参数的设置





如何寻址进程？

- ❖ 不同主机上的进程间通信，那么每个进程必须拥有**标识符**
- ❖ 如何寻址主机？——**IP地址**
 - Q: 主机有了IP地址后，是否足以定位进程？
 - A: 否！同一主机上可能同时有多个进程需要通信。
- ❖ **端口号/Port number**
 - 为主机上每个需要通信的进程分配一个端口号
 - HTTP Server: 80
 - Mail Server : 25
- ❖ **进程的标识符**
 - **IP地址+端口号**

协议	本机IP地址: 端口号	外部IP地址: 端口号	状态
TCP	192.168.0.100:49225	202.108.23.105:5287	ESTABLISHED
TCP	192.168.0.100:49241	sinwnts1011813:https	ESTABLISHED



应用层协议

❖ 网络应用需遵循应用层协议

❖ 公开协议

- 由RFC(Request For Comments) 定义

- 允许互操作

- HTTP, SMTP,

❖ 私有协议

- 多数P2P文件共享应用

A screenshot of a web browser displaying the IETF homepage. The address bar shows 'www.ietf.org/rfc.html'. The page features the IETF logo and navigation links for Home, About the IETF, Internet-Drafts, RFC Pages, IANA Pages, and Working Groups. A sidebar on the right provides information about RFCs and their retrieval.

Request for Comments (RFC)

Memos in the Requests for Comments (RFC) document the Internet. They cover many aspects of computer networks as well as meeting notes, opinions, and sometimes humor. rfc-editor.org. Note that there is a brief time period where this site is the authoritative source page.

RFCs associated with an active IETF Working Group can be found at [Working Groups](#).

IETF Repository Retrieval

- Advanced search options are available at [IETF IANA](#)
- A text index of RFCs is available on the IETF website.
- To go directly to a text version of an RFC, type [NNNN](#) in your browser, where NNNN is the RFC number.

RFC Editor Repository Retrieval

- [RFC Search Page](#)
- [RFC Index \(HTML | TXT | XML \)](#)
- [Additional listings of RFCs](#)
- [RFC Editor Queue](#)



2.1 网络应用体系结构

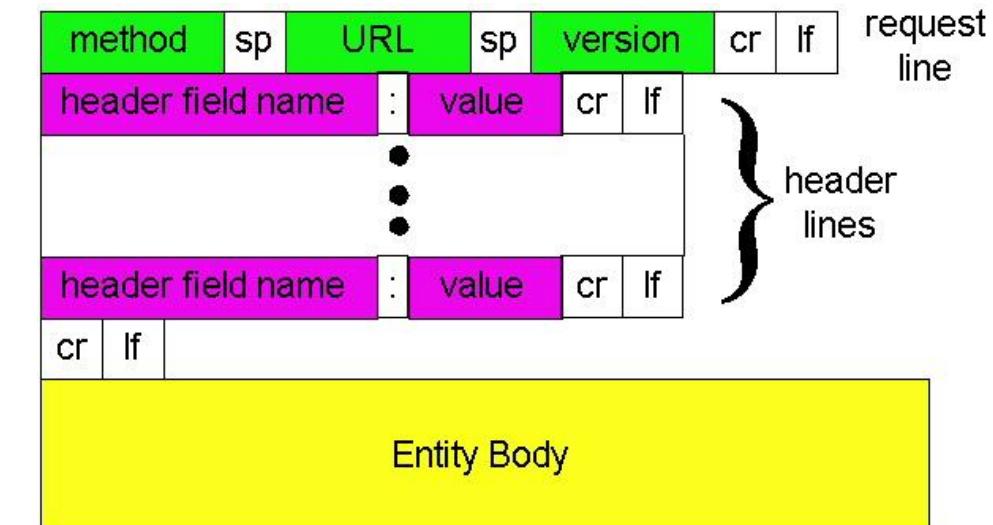
2.2 网络应用通信原理



应用层协议的内容

- ❖ 消息的类型(type)
 - 请求消息
 - 响应消息
- ❖ 消息的语法(syntax)/格式
 - 消息中有哪些字段(field) ?
 - 每个字段如何描述
- ❖ 字段的语义(semantics)
 - 字段中信息的含义
- ❖ 规则(rules)
 - 进程何时发送/响应消息
 - 进程如何发送/响应消息

HTTP请求消息的格式





2.1 网络应用体系结构

2.2 网络应用通信原理

网络应用对传输服务的需求

❖ 数据丢失(data loss)/可靠性(reliability)

- 某些网络应用能够容忍一定的数据丢失：网络电话
- 某些网络应用要求100%可靠的数据传输：文件传输, telnet

❖ 时间(timing)/延迟(delay)

- 有些应用只有在延迟足够低时才“有效”
- 网络电话/网络游戏

❖ 带宽(bandwidth)

- 某些应用只有在带宽达到最低要求时才“有效”：网络视频
- 某些应用能够适应任何带宽——弹性应用：email





典型网络应用对传输服务的需求

2.1 网络应用体系结构

2.2 网络应用通信原理

Application	Data loss	Bandwidth	Time Sensitive
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
Web documents	no loss	elastic	no
real-time audio/video	loss-tolerant	audio: 5kbps-1Mbps video:10kbps-5Mbps	yes, 100's msec
stored audio/video	loss-tolerant	same as above	yes, few secs
interactive games	loss-tolerant	few kbps up	yes, 100's msec
instant messaging	no loss	elastic	yes and no





Internet提供的传输服务

❖ TCP服务

- **面向连接**: 客户机/服务器进程间需要建立连接
- **可靠的传输**
- **流量控制**: 发送方不会发送速度过快，超过接收方的处理能力
- **拥塞控制**: 当网络负载过重时能够限制发送方的发送速度
- **不提供时间/延迟保障**
- **不提供最小带宽保障**

❖ UDP服务

- **无连接**
- **不可靠的数据传输**
- **不提供:**
 - 可靠性保障
 - 流量控制
 - 拥塞控制
 - 延迟保障
 - 带宽保障





典型网络应用所使用的传输层服务

2.1 网络应用体系结构

2.2 网络应用通信原理

	Application layer protocol	Underlying transport protocol
Application		
e-mail	SMTP [RFC 2821]	TCP
remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
file transfer	FTP [RFC 959]	TCP
streaming multimedia	proprietary (e.g. RealNetworks)	TCP or UDP
Internet telephony	proprietary (e.g., Vonage, Dialpad)	typically UDP





2.3 域名解析系统(DNS)

順生長



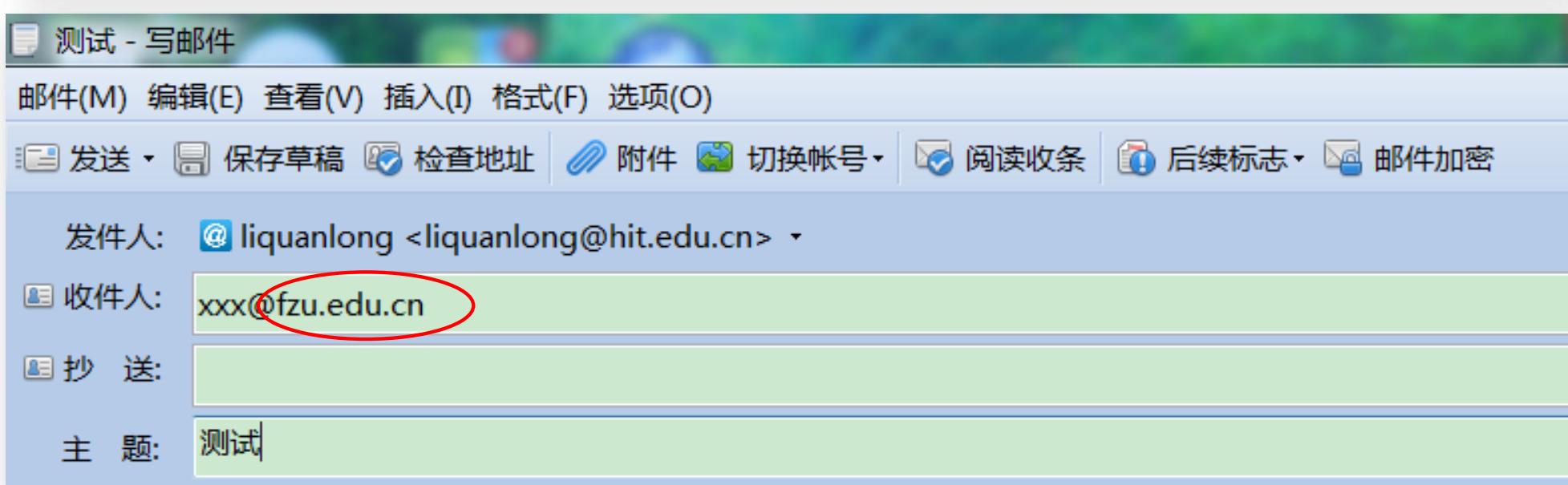
DNS: Domain Name System

2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

http://www.sina.com.cn/



测试 - 写邮件

邮件(M) 编辑(E) 查看(V) 插入(I) 格式(F) 选项(O)

发送 保存草稿 检查地址 附件 切换帐号 阅读收条 后续标志 邮件加密

发件人: @ liquanlong <liquanlong@hit.edu.cn>

收件人: xxx@fzu.edu.cn

抄 送:

主 题: 测试



2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)



DNS: Domain Name System

❖ Internet上主机/路由器的标识/寻址问题

- IP地址
- 域名: www.hit.edu.cn

❖ Q: 域名和IP地址之间如何映射?

❖ A: 域名系统-DNS

- 多层命名服务器构成的**分布式数据库**
- **应用层协议:** 完成名字的解析
 - Internet核心功能
 - 应用层协议实现





2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

❖ DNS服务

- 域名向IP地址的翻译
- 主机别名
- 邮件服务器别名
- 负载均衡：Web服务器

❖ 问题：为什么不使用集中式的DNS？

- 单点失败问题
- 流量问题
- 距离问题
- 维护性问题

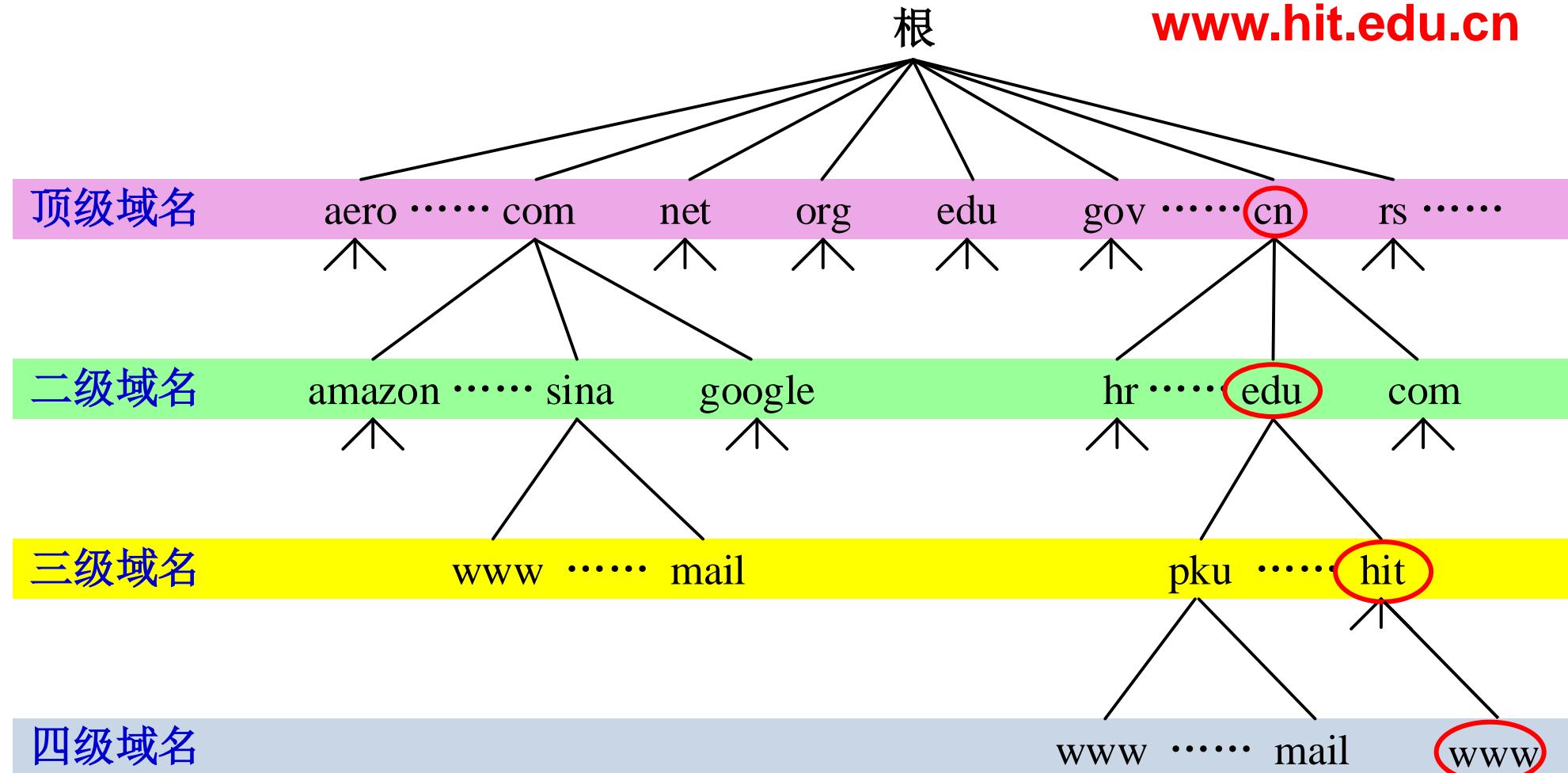




DNS—域名层次结构

2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)**www.hit.edu.cn**

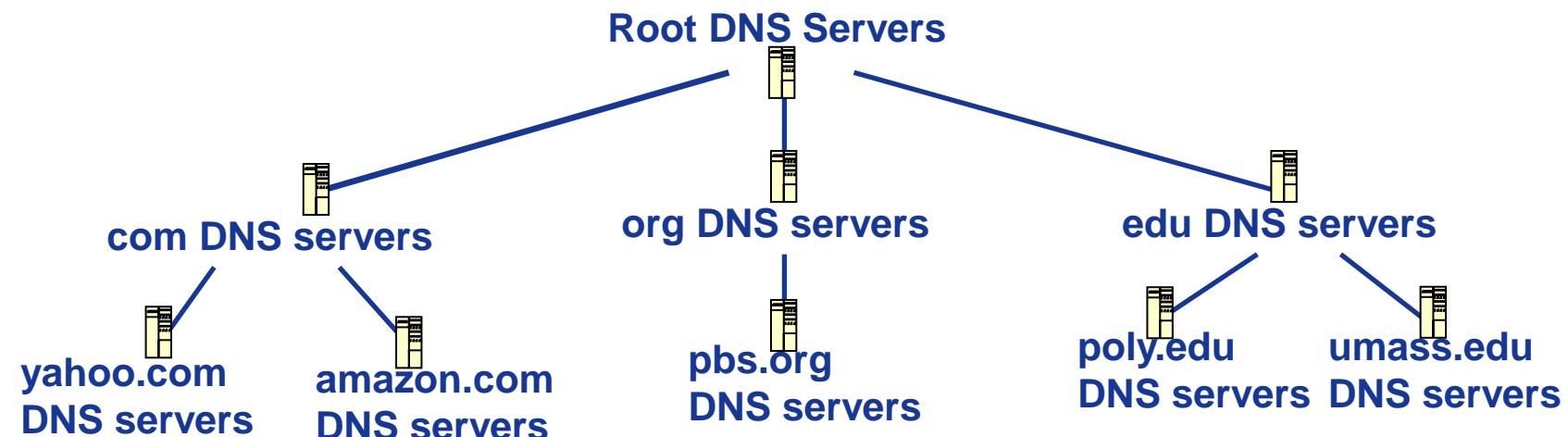


分布式层次式数据库

2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)



❖ 客户端想要查询www.amazon.com的IP

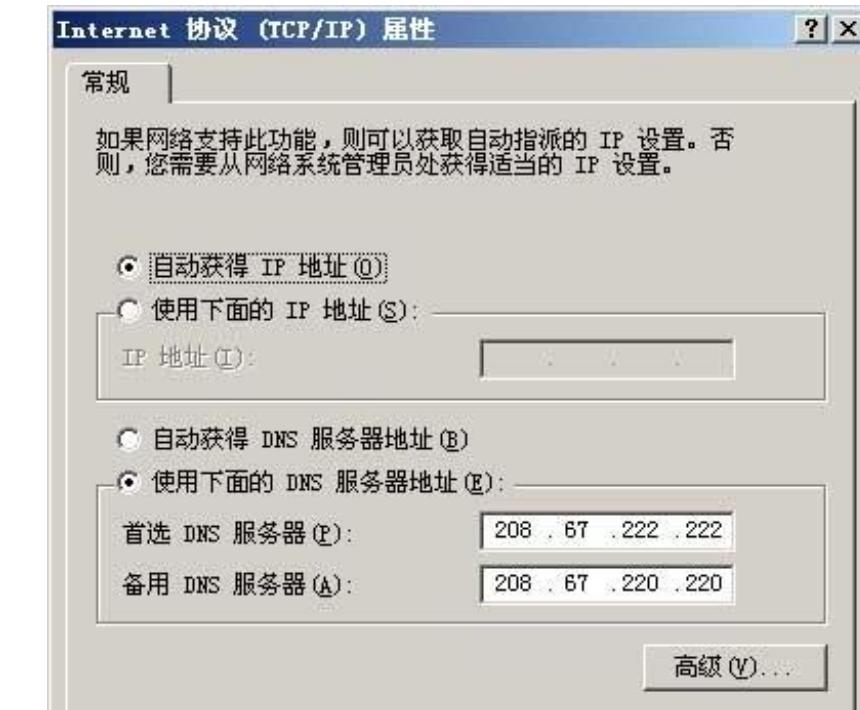
- 客户端查询根服务器，找到com域名解析服务器
- 客户端查询com域名解析服务器，找到amazon.com域名解析服务器
- 客户端查询amazon.com域名解析服务器，获得www.amazon.com的IP地址





本地域名解析服务器

- ❖ 不严格属于层级体系
- ❖ 每个ISP至少有一个本地域名服务器
 - 默认域名解析服务器
- ❖ 当主机进行DNS查询时，查询首先被发送到本地域名服务器
 - 作为代理(proxy)，将查询转发给（层级式）域名服务器系统





DNS根域名服务器

2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

❖ 本地域名服务器无法解析域名时，查询根域名服务器

❖ 根域名服务器

- 如果不知道映射，则查询顶级域名服务器，……，查询**权威域名服务器**
- 获得映射
- 向本地域名服务器返回映射



全球有13个
根域名服务器





TLD和权威域名解析服务器

- ❖ **顶级域名服务器(TLD, top-level domain):** 负责 com, org, net, edu 等顶级域名和国家顶级域名, 例如 cn, uk, fr 等
 - Network Solutions 维护 com 顶级域名服务器
 - Educause 维护 edu 顶级域名服务器
- ❖ **权威(Authoritative)域名服务器:** 组织的域名服务器, 提供组织内部服务器的解析服务
 - 组织负责维护
 - 服务提供商负责维护

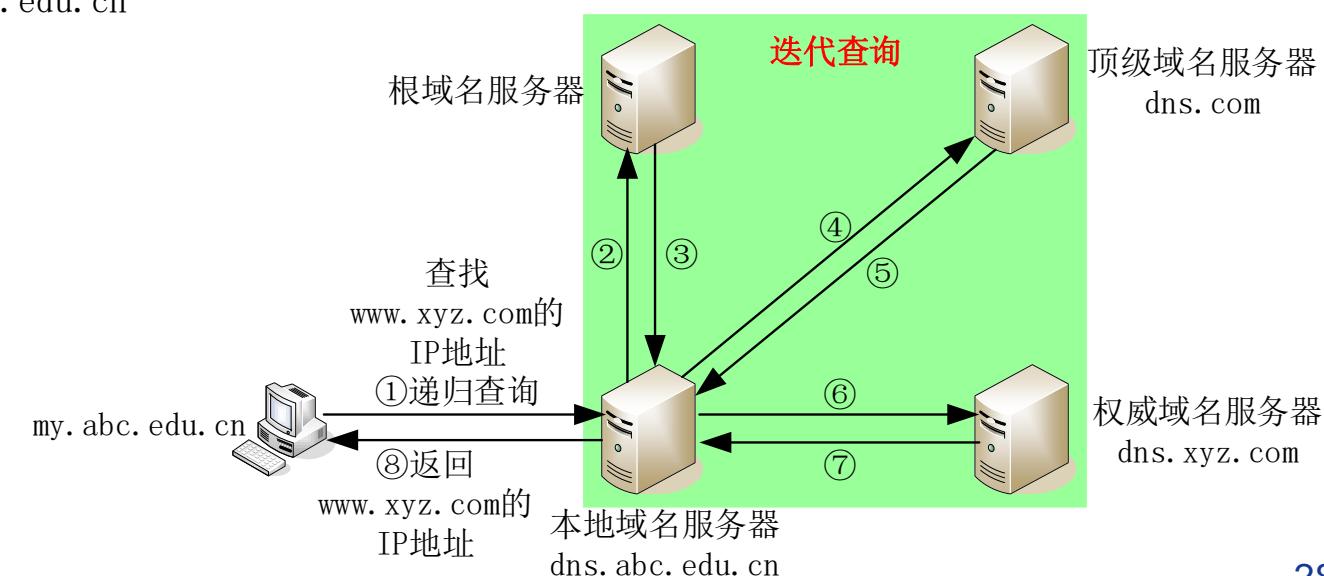
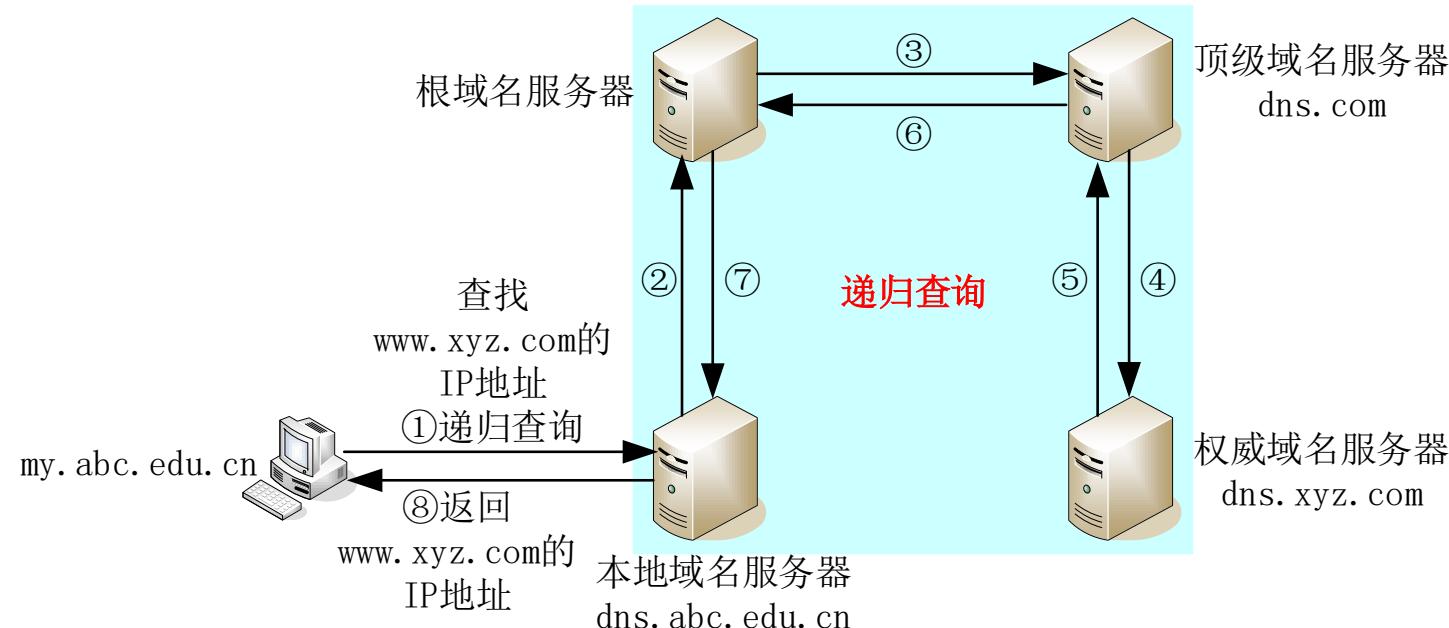


2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

DNS查询过程





2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

DNS记录缓存和更新

❖ 只要域名服务器获得域名—IP映射，即缓存这一映射

- 一段时间过后，缓存条目失效（删除）
- 本地域名服务器一般会缓存顶级域名服务器的映射
 - 因此根域名服务器不经常被访问

❖ 记录的更新/通知机制

- RFC 2136
- Dynamic Updates in the Domain Name System (DNS UPDATE)





DNS记录

❖ 资源记录(RR, resource records)

❖ Type=A

- Name: 主机域名
- Value: IP地址

❖ Type=NS

- Name: 域(edu.cn)
- Value: 该域权威域名
服务器的主机域名

RR format: (name, value, type, ttl)

❖ Type=CNAME

- Name: 某一真实域名的别名
 - www.ibm.com – servereast.backup2.ibm.com
- Value: 真实域名

❖ Type=MX

- Name: 邮件服务器别名
- Value: 邮件服务器域名



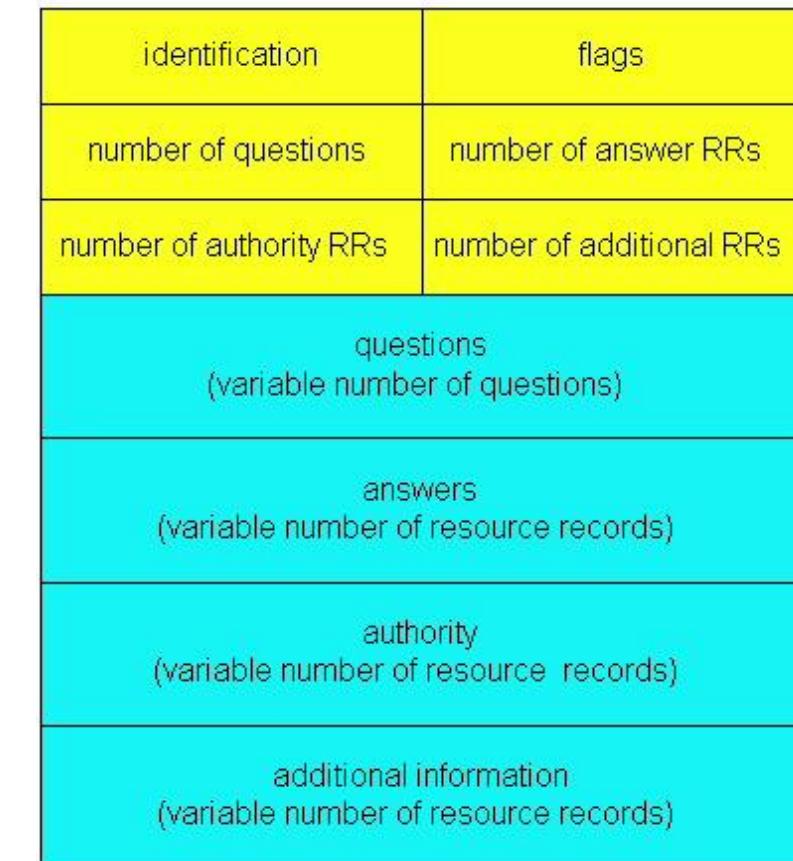
DNS协议与消息

❖ DNS协议：

- 查询(query)和回复(reply)消息
- 消息格式相同

❖ 消息头部

- Identification: 16位查询编号，
回复使用相同的编号
- flags
 - 查询或回复
 - 期望递归
 - 递归可用
 - 权威回答

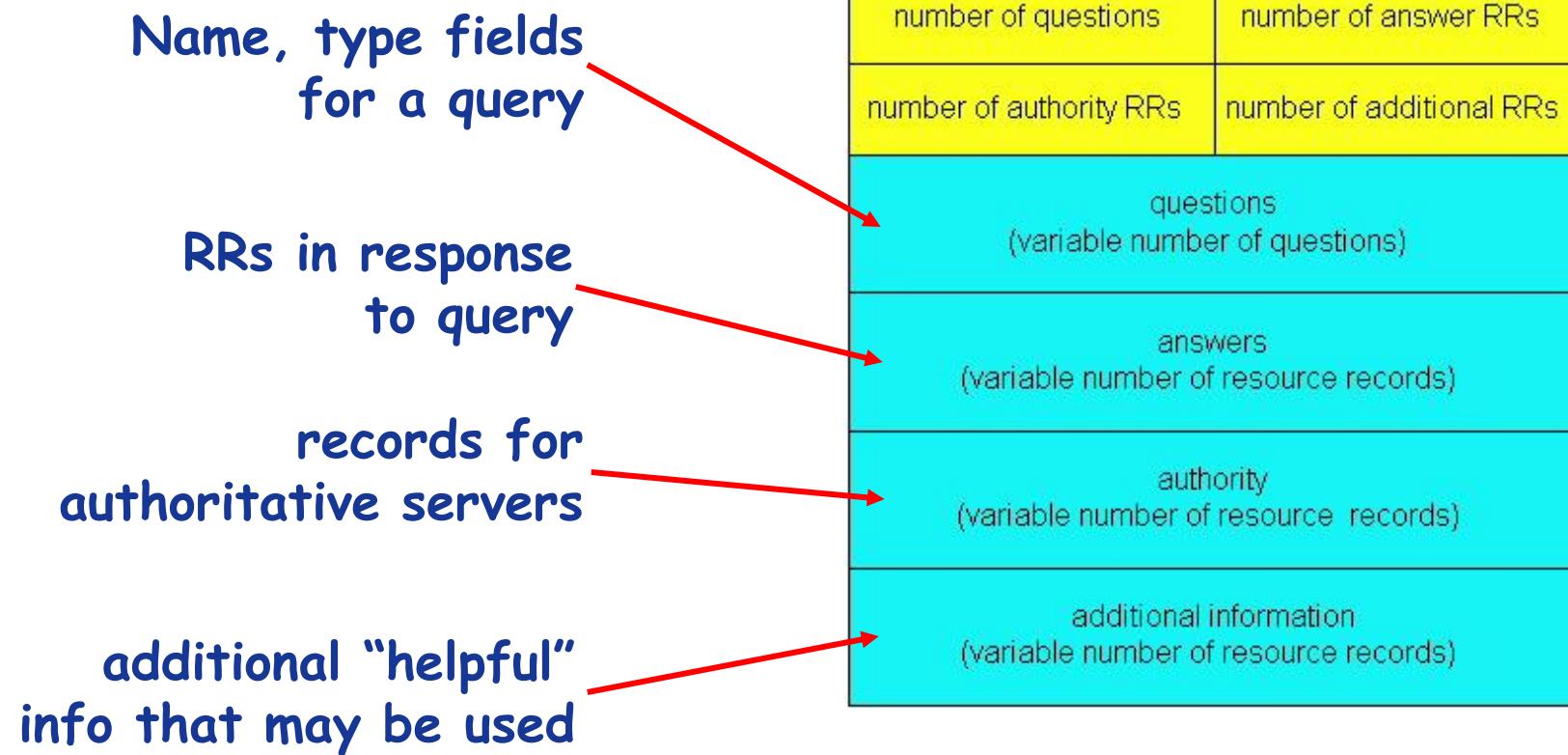




DNS协议与消息

2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)



2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

如何注册域名？

- ❖ 例子：你刚刚创建了一个公司“**Network Utopia**”
- ❖ 在域名管理机构(如**Network Solutions**)注册域名
networkutopia.com
 - 向域名管理机构提供你的权威域名解析服务器的名字和IP地址
 - 域名管理机构向com顶级域名解析服务器中插入两条记录
 - (**networkutopia.com, dns1.networkutopia.com, NS**)
 - (**dns1.networkutopia.com, 212.212.212.1, A**)
- ❖ 在权威域名解析服务器中：
 - 为**www.networkutopia.com**加入Type A记录
 - 为**networkutopia.com**加入Type MX记录、 Type A记录
 -





例题

【例1】如果本地域名服务器无缓存，当采用递归方法解析另一网络某主机域名时，用户主机、本地域名服务器发送的域名请求消息数分别为

- A. 一条、一条
- B. 一条、多条
- C. 多条、一条
- D. 多条、多条

【解析】域名递归解析过程中，主机向本地域名服务器发送**DNS**查询，被查询的域名服务器代理后续的查询，然后返回结果。所以，递归查询时，如果本地域名服务器无缓存，则主机和本地域名服务器都仅需要发送一次查询，故正确答案为**A**。



单选题 1分

2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)



一个DNS资源记录(RR)为(h.edu.cn, d.h.edu.cn, NS, 250),
则d.h.edu.cn是

- A 邮件服务器的域名
- B 邮件服务器的别名
- C 本地域名服务器的域名
- D 权威域名服务器的域名

提交



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

2.4 FTP应用

聂兰顺



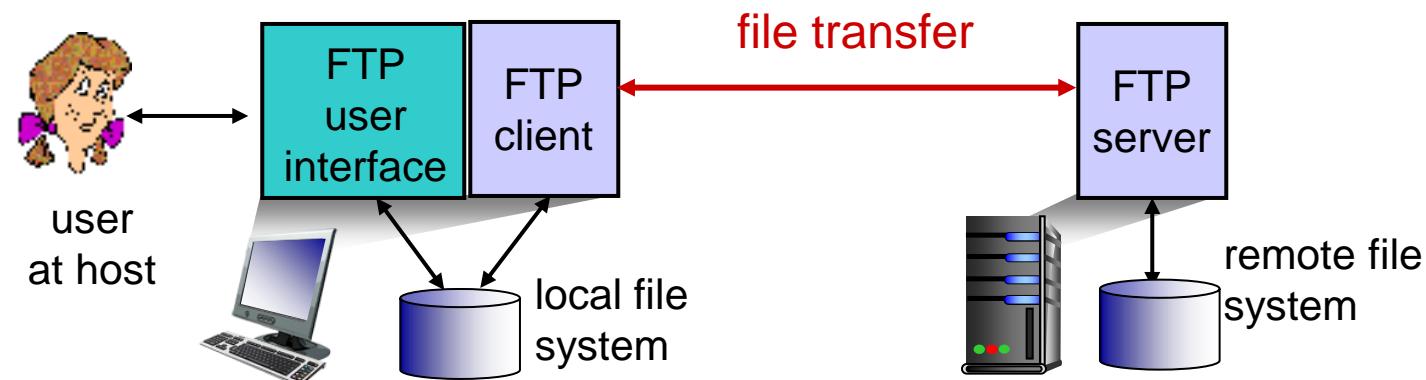
FTP: the file transfer protocol

2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用



- ❖ transfer file to/from remote host
- ❖ client/server model
 - *client*: side that initiates transfer (either to/from remote)
 - *server*: remote host
- ❖ ftp: RFC 959
- ❖ ftp server: port 21





FTP: separate control, data connections

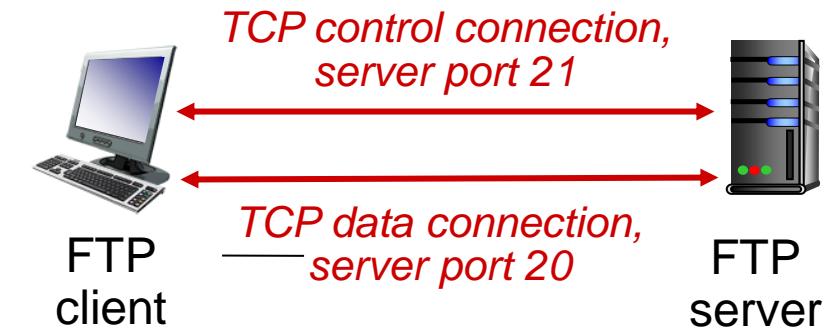
2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

- ❖ FTP client contacts FTP server at port 21, **using TCP**
- ❖ client authorized over **control connection**
- ❖ client browses remote directory, sends commands over control connection
- ❖ when server receives file transfer command, **server opens 2nd TCP data connection** (for file) **to client**
- ❖ after transferring one file, server closes data connection



- ❖ server opens another TCP data connection to transfer another file
- ❖ control connection: "**out of band**"
- ❖ FTP server maintains "**state**": current directory, earlier authentication



2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用



FTP commands, responses

sample commands:

- ❖ sent as ASCII text over control channel
- ❖ **USER *username***
- ❖ **PASS *password***
- ❖ **LIST** return list of file in current directory
- ❖ **RETR *filename*** retrieves (gets) file
- ❖ **STOR *filename*** stores (puts) file onto remote host

sample return codes

- ❖ status code and phrase (as in HTTP)
- ❖ **331 Username OK, password required**
- ❖ **125 data connection already open; transfer starting**
- ❖ **425 Can't open data connection**
- ❖ **452 Error writing file**

单选题 1分

2.1 网络应用体系结构
2.2 网络应用通信原理
2.3 域名解析系统(DNS)

2.4 FTP应用



FTP客户和服务器间传递FTP命令时，使用的连接是

- A 建立在TCP之上的控制连接
- B 建立在TCP之上的数据连接
- C 建立在UDP之上的控制连接
- D 建立在UDP之上的数据连接

提交



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

2.5 Email应用

聂兰顺



Email应用的构成(1)

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用

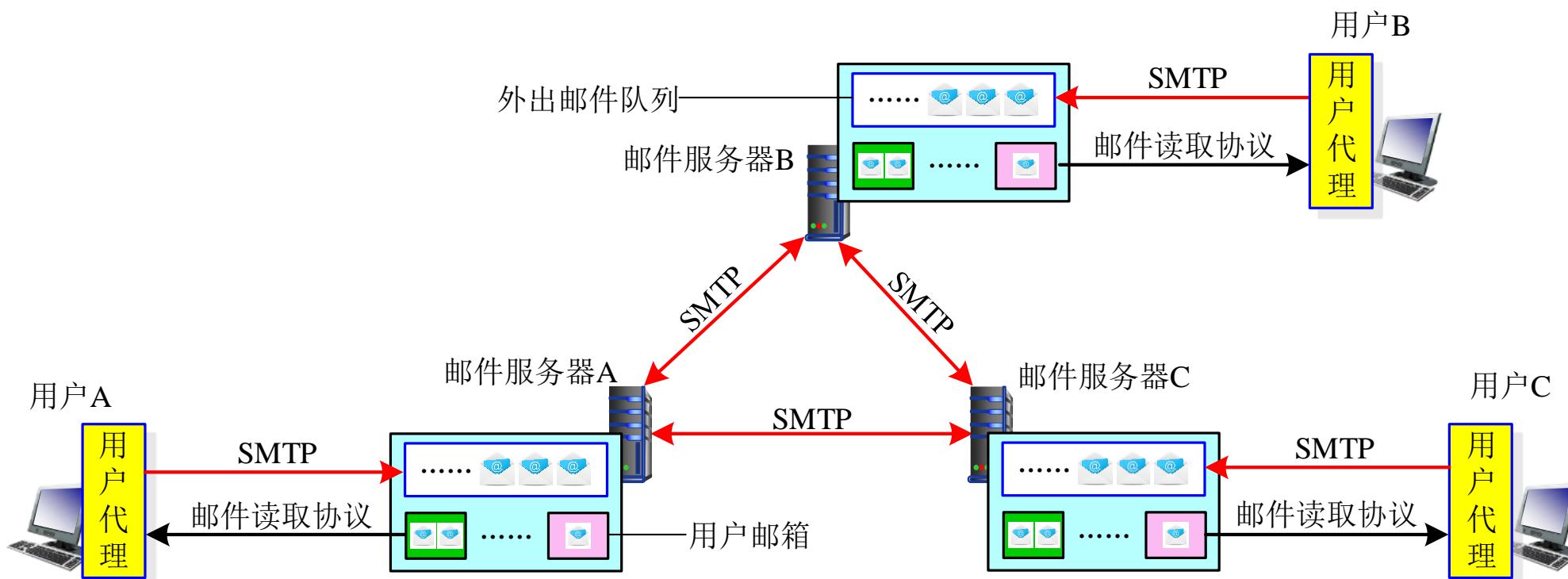
2.5 Email应用

◆ Email应用的构成组件

- 邮件客户端(user agent)
- 邮件服务器
- SMTP协议(Simple Mail Transfer Protocol)

◆ 用户代理（邮件客户端）

- 读、写Email消息
- 与服务器交互，收、发Email消息
- Outlook, Foxmail, Thunderbird
- Web客户端





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用

2.5 Email应用



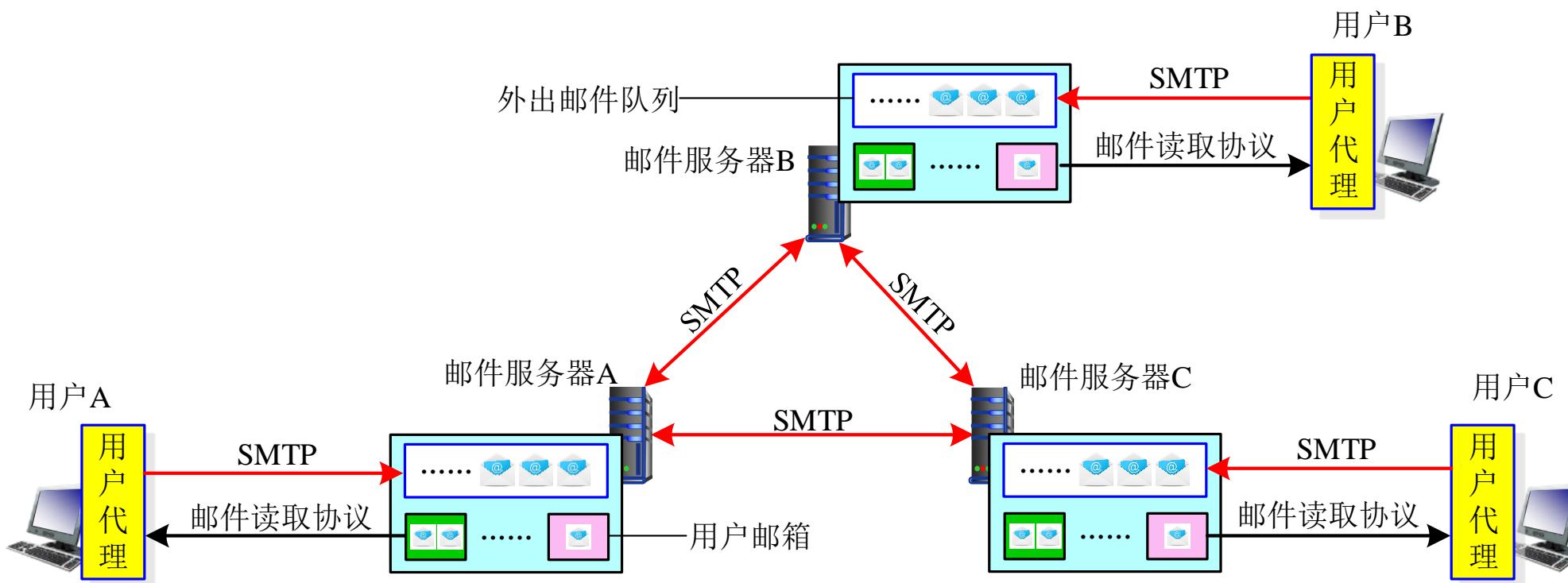
Email应用的构成(2)

❖ 邮件服务器(Mail Server)

- 邮箱: 存储发给该用户的Email
- 消息队列(message queue): 存储等待发送的Email

❖ SMTP协议

- 邮件服务器之间传递消息所使用的协议
- 客户端: 发送消息的服务器
- 服务器: 接收消息的服务器





SMTP协议: RFC 2821

2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

2.5 Email应用



- ❖ 使用TCP进行email消息的可靠传输
- ❖ 端口25
- ❖ 传输过程的三个阶段:
 - 握手
 - 消息的传输
 - 关闭
- ❖ 命令/响应交互模式
 - 命令(command): ASCII文本
 - 响应(response): 状态代码和语句
- ❖ Email消息只能包含**7位ASCII码**



SMTP交互示例

2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

2.5 Email应用



```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
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?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用

2.5 Email应用



Email消息格式

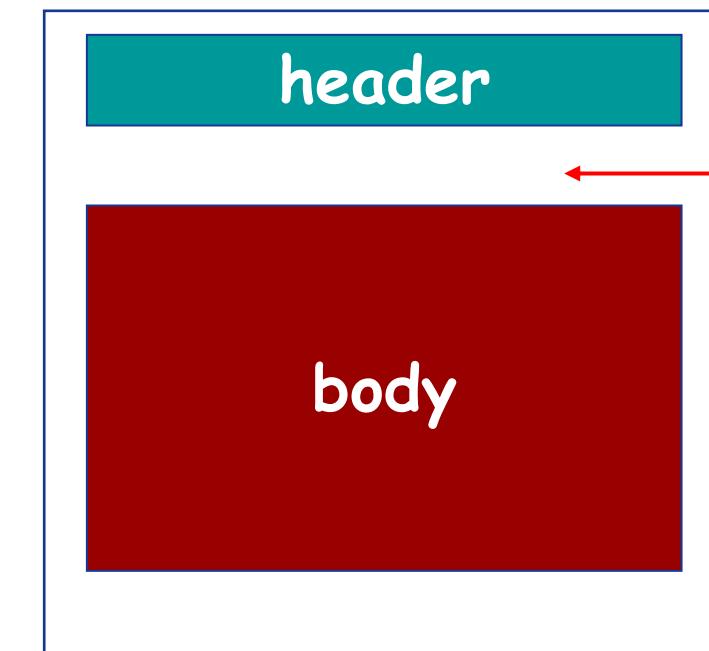
- ❖ SMTP: email消息的传输/交换协议
- ❖ RFC 822: 文本消息格式标准

- 头部行(header)

- To
- From
- Subject

- 消息体(body)

- 消息本身
- 只能是ASCII字符





Email消息格式：多媒体扩展

❖ MIME: 多媒体邮件扩展 RFC 2045, 2056

- 通过在邮件头部增加额外的行以声明MIME的内容类型

MIME版本

数据编码方法

多媒体数据的类型，
子类型以及参数声明

编码后的数据

```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

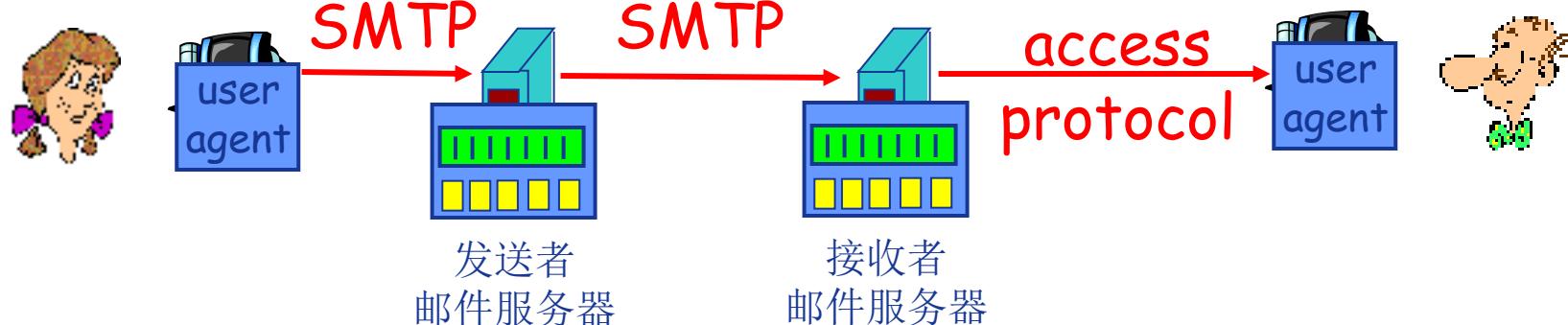
base64 encoded data .....
.....
.....base64 encoded data
```





邮件访问协议

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用**



❖ 邮件访问协议：从服务器获取邮件

- POP: Post Office Protocol [RFC 1939]
 - 认证/授权(客户端↔服务器)和下载
- IMAP: Internet Mail Access Protocol [RFC 1730]
 - 更多功能
 - 更加复杂
 - 能够操纵服务器上存储的消息
- HTTP: Web Mail





POP协议

❖ 认证过程

- 客户端命令 →
- User: 声明用户名
- Pass: 声明密码
- 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
```

❖ 事务阶段 →

- List: 列出消息数量
- Retr: 用编号获取消息
- Dele: 删除消息
- Quit

```
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
C: quit
S: +OK POP3 server signing off
```



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用

2.5 Email应用



POP协议

- ❖ “下载并删除”模式
 - 用户如果换了客户端软件，无法重读该邮件
- ❖ “下载并保持”模式
 - 不同客户端都可以保留消息的拷贝
- ❖ POP3是无状态的



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用

2.5 Email应用



IMAP协议

- ❖ 所有消息统一保存在一个地方：服务器
- ❖ 允许用户利用文件夹组织消息
- ❖ IMAP支持跨会话(Session)的用户状态：
 - 文件夹的名字
 - 文件夹与消息ID之间的映射等

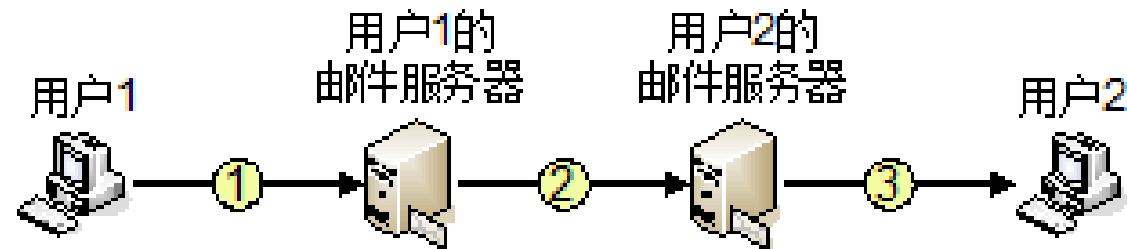
单选题 1分

2.1 网络应用体系结构
2.2 网络应用通信原理
2.3 域名解析系统(DNS)
2.4 FTP应用
2.5 Email应用

2.5 Email应用



若用户1与用户2之间发送和接收电子邮件的过程如下图所示，则图中①、②、③阶段分别使用的应用层协议可以是



- A SMTP、SMTP、SMTP
- B POP3、SMTP、POP3
- C POP3、SMTP、SMTP
- D SMTP、SMTP、POP3

提交



2.6 Web应用

聂兰顺



Web与HTTP

❖ World Wide Web: Tim Berners-Lee

- 网页
- 网页互相链接



Baidu 百科

❖ 网页(Web Page)包含多个对象(objects)

- 对象: HTML文件、JPEG图片、视频文件、动态脚本等
- 基本HTML文件: 包含对其他对象引用的链接

❖ 对象的寻址(addressing)

- URL (Uniform Resource Locator): 统一资源定位器 (RFC1738)
- Scheme://host:port/path

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

host name

path name

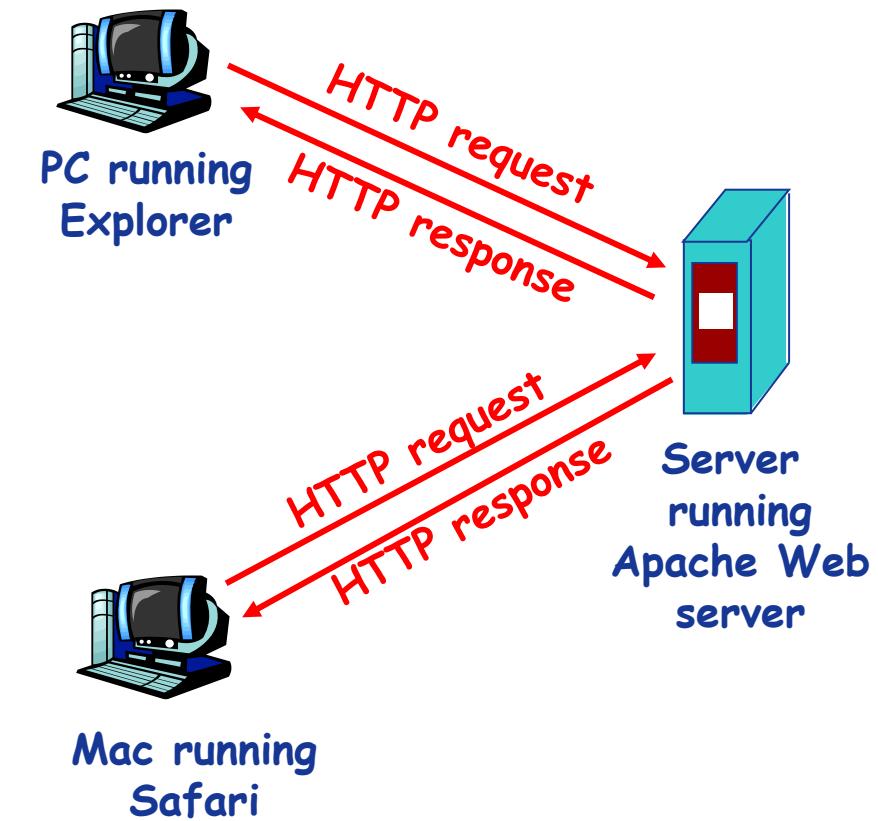


- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



HTTP协议概述(1)

- ❖ 万维网应用遵循什么协议?
- ❖ 超文本传输协议
 - HyperText Transfer Protocol
- ❖ C/S结构
 - 客户—Browser: 请求、接收、展示Web对象
 - 服务器—Web Server: 响应客户的请求, 发送对象
- ❖ HTTP版本:
 - 1.0: RFC 1945
 - 1.1: RFC 2068





HTTP概述(2)

❖ 使用TCP传输服务

- 服务器在80端口等待客户的请求
- 浏览器发起到服务器的TCP连接(创建套接字Socket)
- 服务器接受来自浏览器的TCP连接
- 浏览器(HTTP客户端)与Web服务器(HTTP服务器)交换HTTP消息
- 关闭TCP连接

❖ 无状态(stateless)

- 服务器不维护任何有关客户端过去所发请求的信息



有状态的协议更复杂：

- 需维护状态(历史信息)
- 如果客户或服务器失效，会产生状态的不一致，解决这种不一致代价高

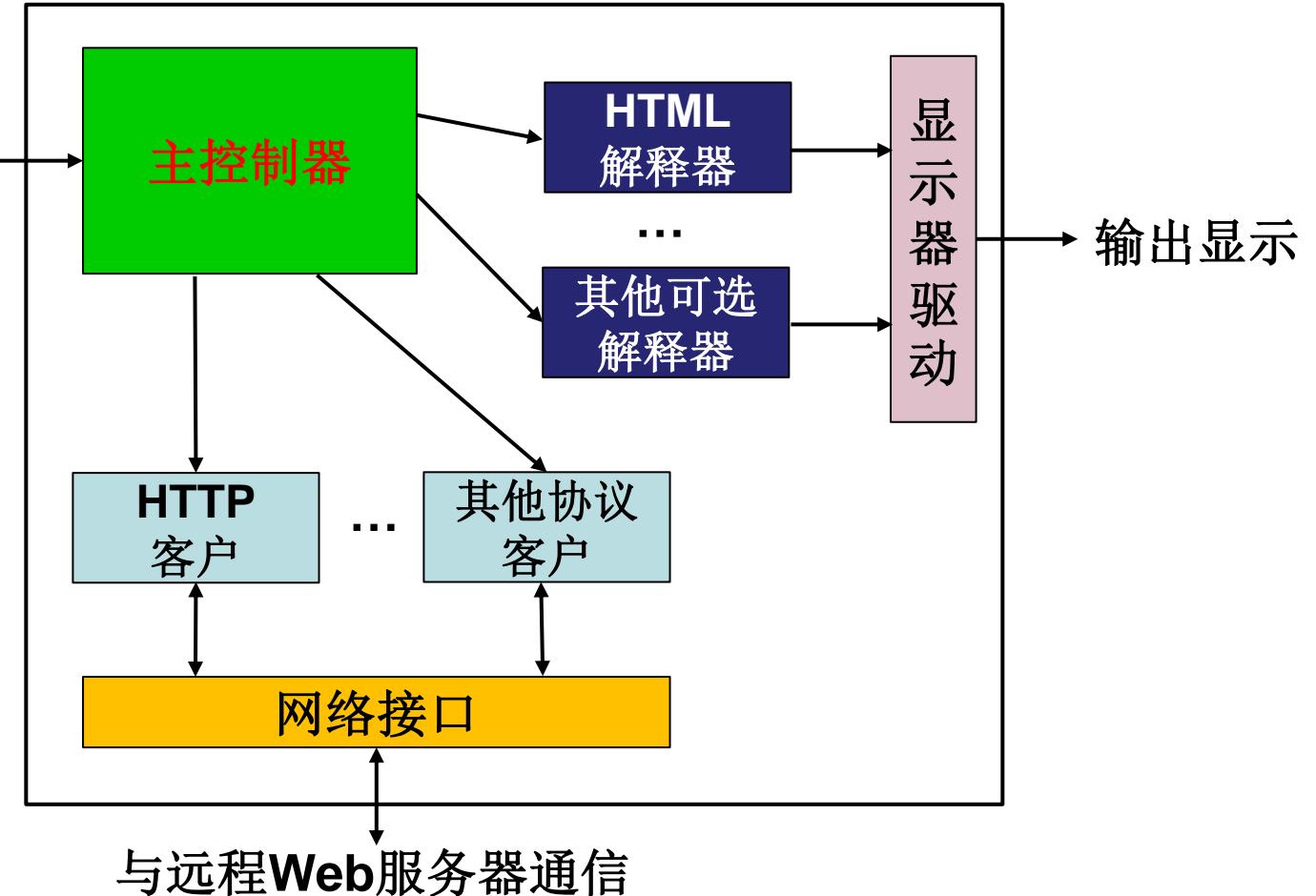
浏览器



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



鼠标和键盘输入





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



HTTP请求消息

❖ HTTP协议有两类消息

- 请求消息(request)
- 响应消息(response)

❖ 请求消息

- ASCII: 人直接可读

request line
(GET, POST,
HEAD commands)

header lines

Carriage return,
line feed indicates
end of message

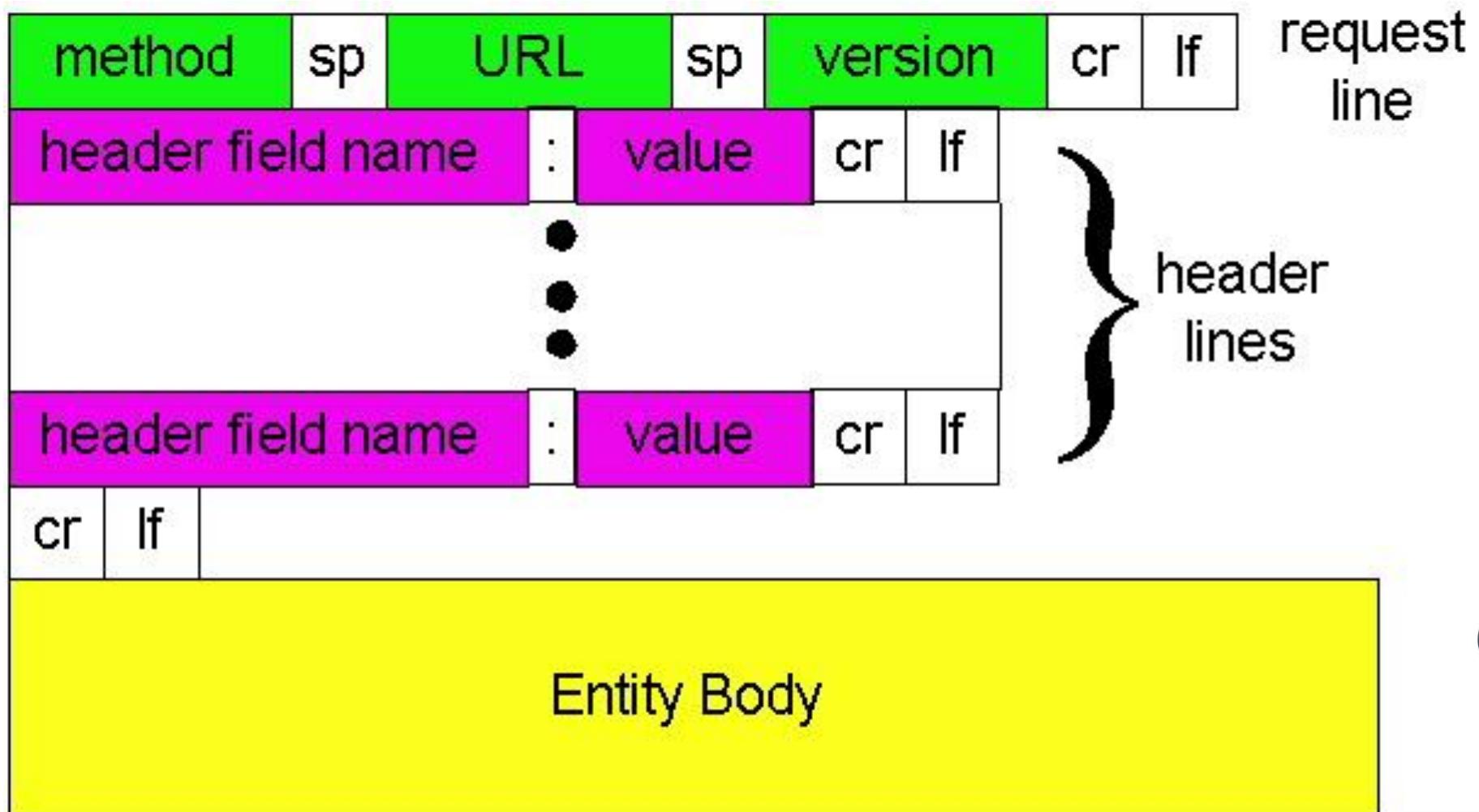
```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
User-agent: Mozilla/4.0
Connection: close
Accept-language: fr
```

(extra carriage return, line feed)



HTTP请求消息的通用格式

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**





上传输入的方法

❖ POST方法

- 网页经常需要填写表单 (form)
- 在请求消息的消息体(entity body)中上传客户端的输入

❖ URL方法

- 使用GET方法
- 输入信息通过request行的URL字段上传

www.somesite.com/animalsearch?monkeys&banana



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



方法的类型

❖ HTTP/1.0

- GET
- POST
- HEAD
 - 请Server不要将所请求的对象放入响应消息中

❖ HTTP/1.1

- GET, POST, HEAD
- PUT
 - 将消息体中的文件上传到URL字段所指定的路径
- DELETE
 - 删除URL字段所指定的文件



HTTP响应消息

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



status line
(protocol
status code
status phrase)

header
lines

data, e.g.,
requested
HTML file

HTTP/1.1 200 OK
Connection: close
Date: Thu, 06 Aug 1998 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 1998
Content-Length: 6821
Content-Type: text/html

data data data data data ...



HTTP响应状态代码

❖ 响应消息的第一行

404 Not Found

❖ 示例

- 200 OK
- 301 Moved Permanently
- 400 Bad Request
- 404 Not Found
- 505 HTTP Version Not Supported

nginx



无法找到该网页

最可能的原因是：

- 在地址中可能存在键入错误。
- 当您点击某个链接时，它可能已过期。

您可以尝试以下操作：

- 重新键入地址。
- 返回到上一页。



2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

2.5 Email应用

2.6 Web应用

HTTP/1.0 请求-响应过程

❖ HTTP/1.0

- 每个TCP连接传输一个对象
- 非持久性连接(**Nonpersistent HTTP**)
- 版本使用非持久性连接





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



HTTP/1.0 请求-响应过程(1)

假定用户在浏览器中输入URL:

`www.someSchool.edu/someDepartment/home.index`

`home.index`包含文本和指向10个jpeg图片的链接

1a. HTTP客户端向地址为 `www.someSchool.edu` 的服务器上的HTTP服务器进程(端口80)发起TCP连接请求。

1b. HTTP服务器在端口80等待TCP连接请求，接受连接并通知客户端。

2. HTTP客户端将HTTP请求消息(包含URL地址)通过TCP连接的套接字发出，消息中所含的URL表明客户端需要对象 `someDepartment/home.index`

3. HTTP服务器收到请求消息，解析，产生包含所需要对象的响应消息，并通过套接字发给客户端。

时间
↓



HTTP/1.0 请求-响应过程(2)

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**

时间 ↓
2. 为每个jpeg对象重复步骤1-5。

4. HTTP服务器关闭TCP连接。

5. HTTP客户端收到响应消息，解析html文件，显示html文件，发现有10个指向jpeg对象的超连接

- ❖ HTTP/1.0
- 每个TCP连接传输**1个**对象
 - 非持久性连接(**Nonpersistent HTTP**)





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



HTTP/1.0响应时间分析

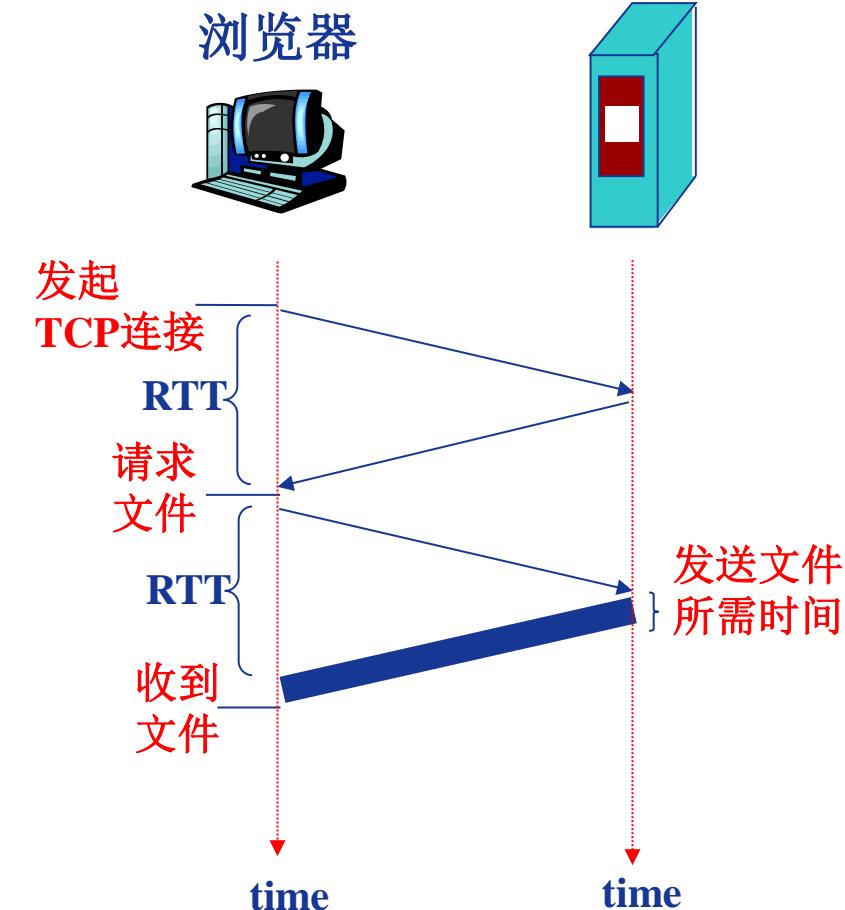
❖ RTT(Round Trip Time)

- 从客户端发送一个很小的数据包到服务器并返回所经历的时间

❖ 响应时间(Response time)

- 发起、建立TCP连接: 1个RTT
- 发送HTTP请求消息到HTTP响应消息的前几个字节到达: 1个RTT
- 响应消息传输时间
- 响应时间 :
 $=2\text{RTT} + \text{响应消息发送时间}$

Web服务器





HTTP响应时间优化-并行连接

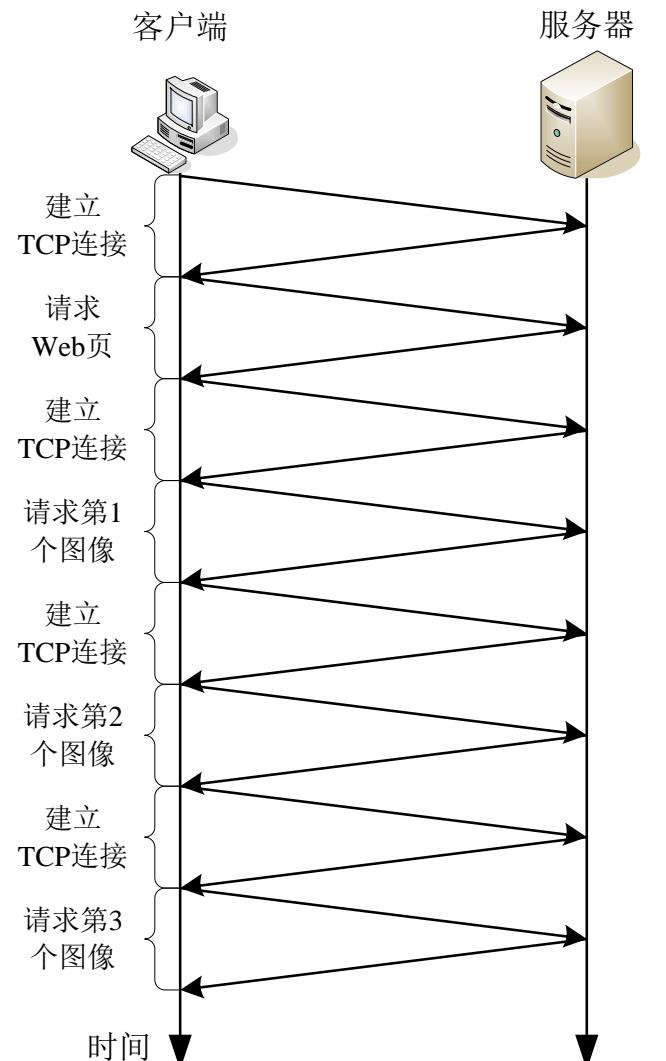
❖ 非持久性连接的问题

- 每个对象需要2个RTT
- 操作系统需要为每个TCP连接开销资源(**overhead**)
- 浏览器可以怎么做?
 - 打开多个并行**TCP连接**以获取网页所需对象
 - 给服务器端造成什么影响?
- 例如: 请求浏览一个引用3个**JPEG小图像**的Web页

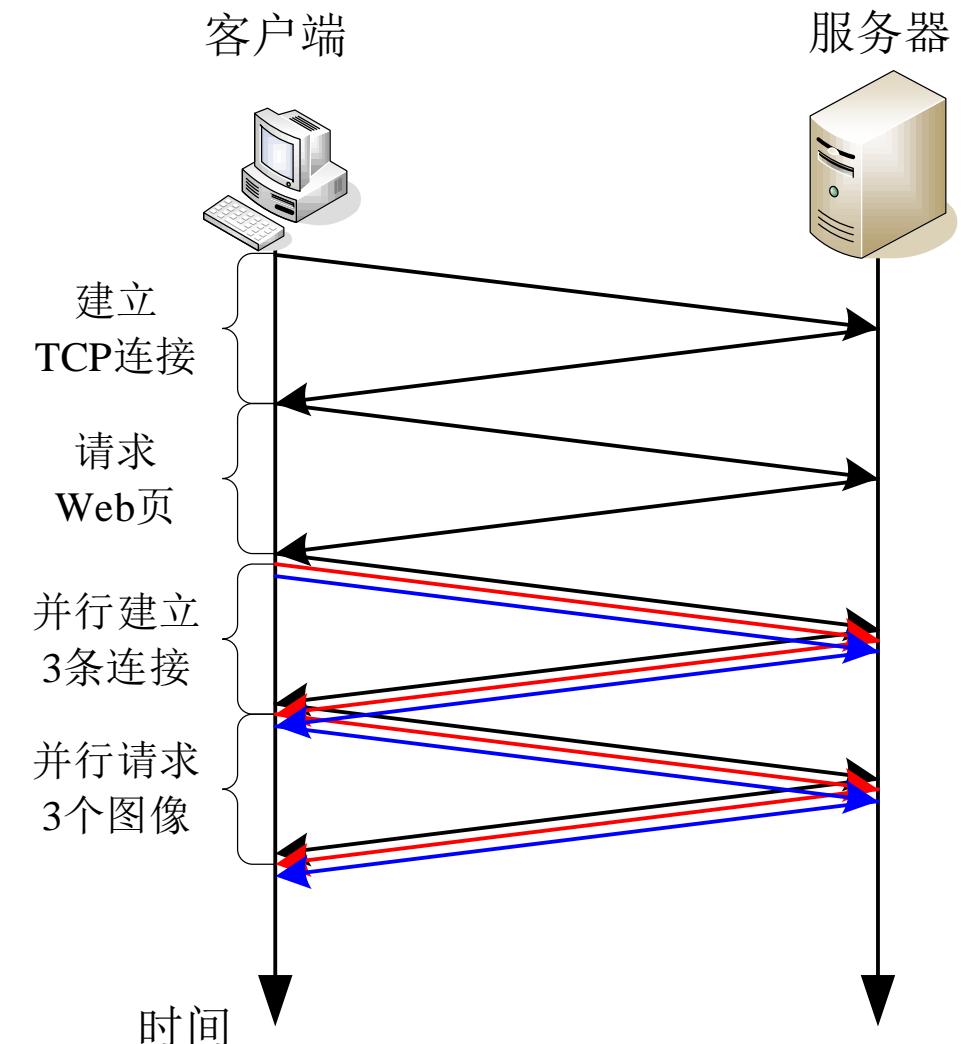


HTTP响应时间优化-并行连接

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



非持久连接（默认）的HTTP/1.0



使用并行连接的HTTP/1.0



HTTP响应时间优化-持久连接

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



❖ 持久连接(Persistent HTTP)

- 发送响应后，服务器保持TCP连接打开
- 后续HTTP消息可以通过这个连接发送
- 每个TCP连接允许传输多个对象

❖ 非流水(pipelining) 持久连接

- 客户端只有收到前一个响应后才发送新的请求
- 每个被引用的对象耗时1个RTT

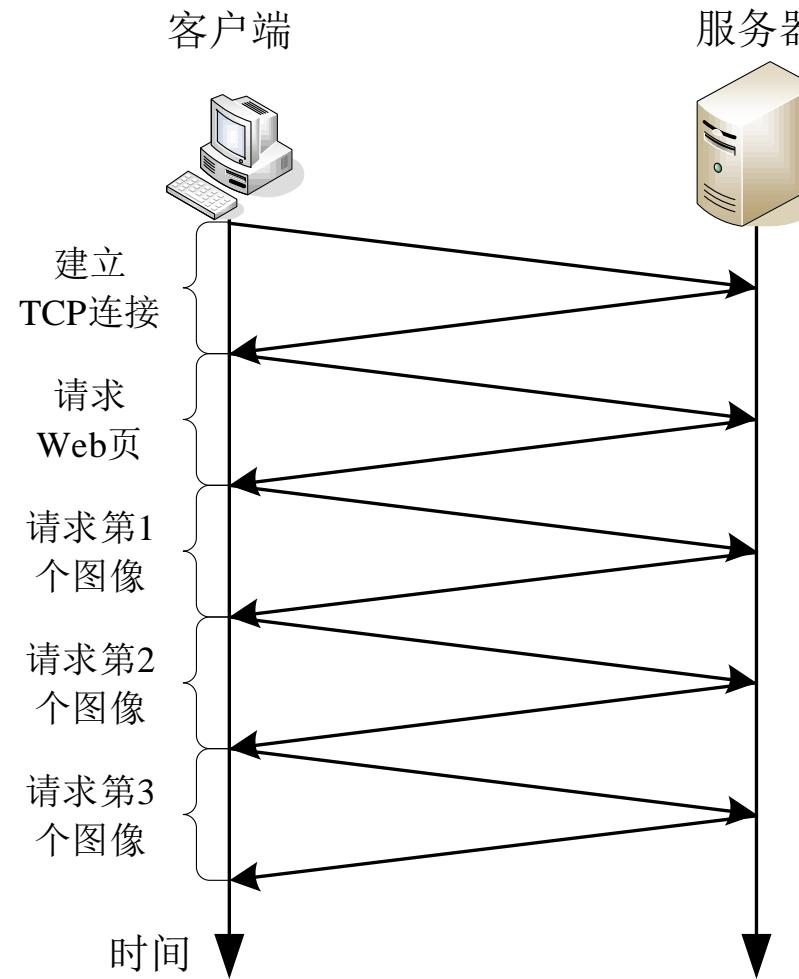
❖ 流水的持久连接

- 客户端只要遇到一个引用对象就尽快发出请求
- 理想情况下，收到所有的引用对象只需耗时约1个RTT
- **HTTP/1.1**版本默认使用流水的持久连接
- 例如：请求浏览一个引用3个JPEG小图像的Web页

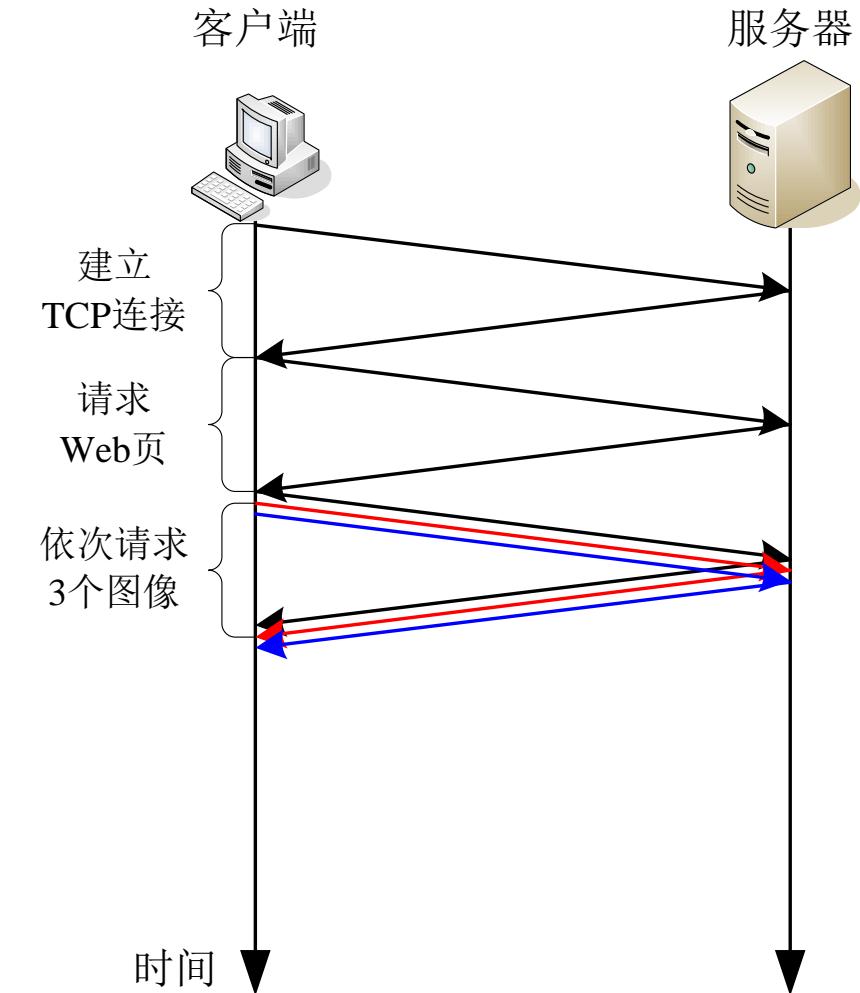


HTTP响应时间优化-持久连接

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



非流水持久连接的HTTP/1.1



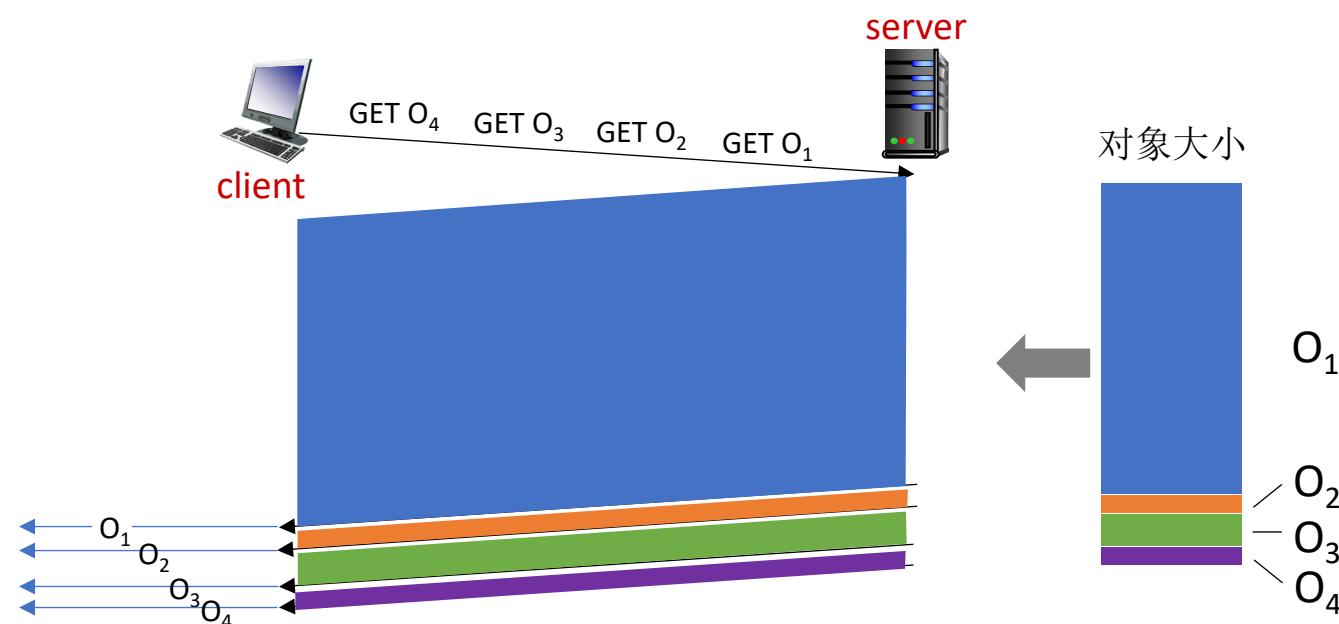
流水持久连接的HTTP/1.1



HTTP响应时间优化- HTTP/2

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**

- ❖ HTTP/1.1有什么不足？考虑通过HTTP/1.1连续请求多个大小不同的对象，可能会发生什么问题？
 - 例：通过HTTP 1.1连续请求1个大对象(如视频文件)和3个小对象





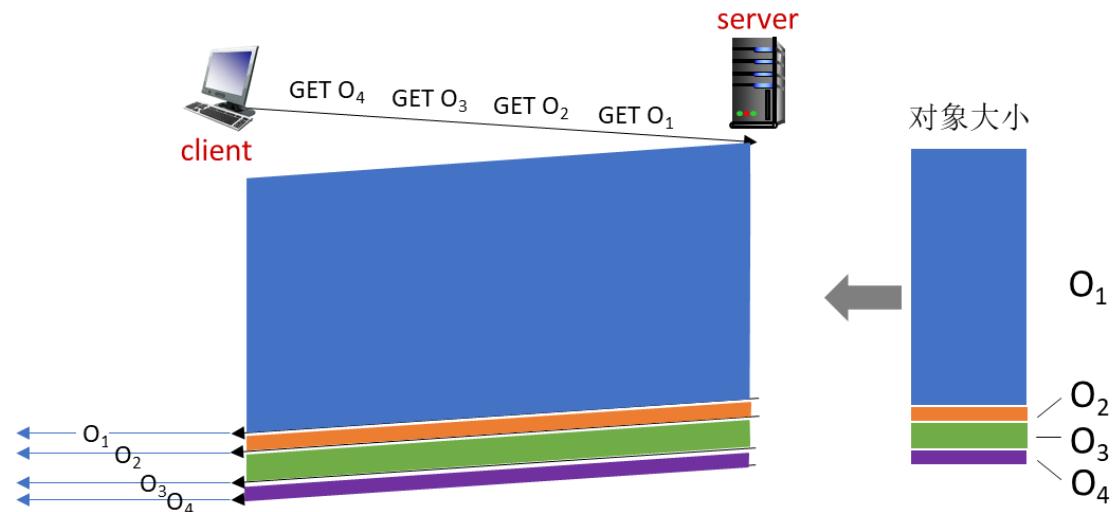
- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



HTTP响应时间优化- HTTP/2

❖ HTTP/1.1

- 服务器顺序响应 (FCFS调度)
- 在大对象之后请求的小对象，需要等待大对象的传输 (**head-of-line (HOL) blocking**)
- 若发生数据丢失会暂停对象传输





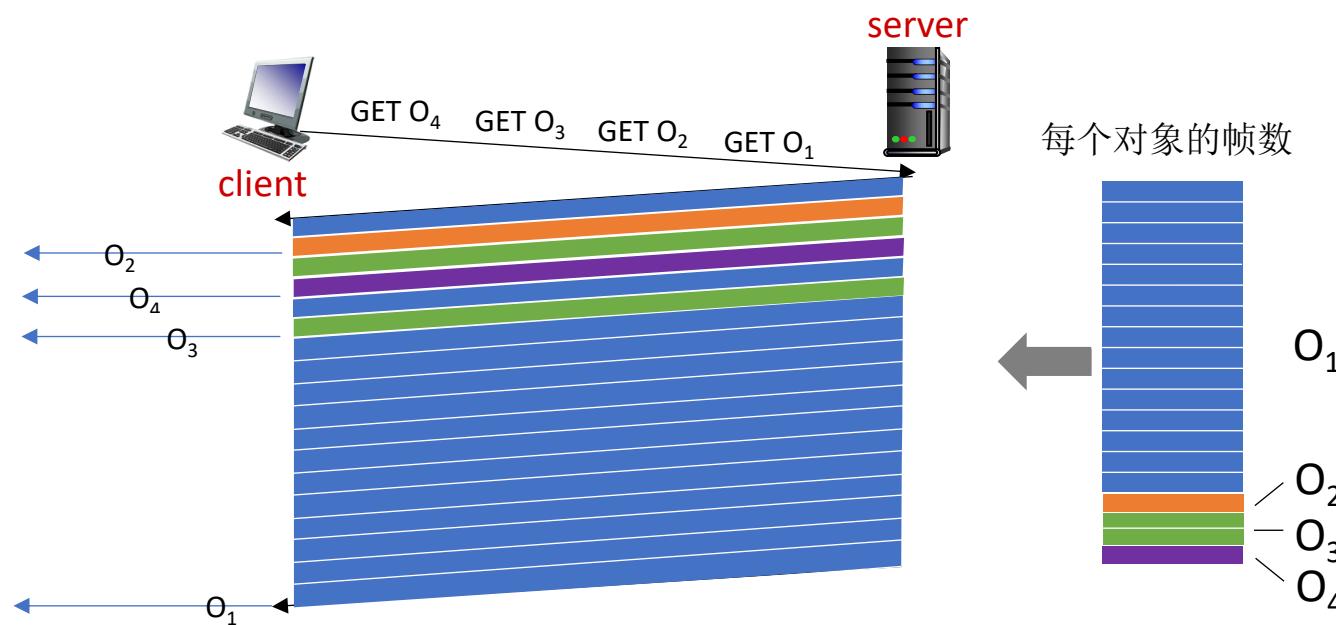
HTTP响应时间优化- HTTP/2

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



❖ **HTTP/2 [RFC 7540, 2015]**：减少多对象请求的时延（响应时间）

- 将对象分割为系列“帧”，不同对象的帧交替传输
- 与HTTP/1.1相比，方法、状态码、多数首部行等没变
- 对象传输顺序基于客户定义的对象优先级（非FCFS调度）
- 推送（push）客户未请求的对象





HTTP响应时间优化- HTTP/3

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



❖ HTTP/2进一步优化?

- 使用单一的**TCP**连接
- 丢包恢复仍然会暂停对象传输

❖ 传输层功能?

- **TCP、UDP**: 主要传输协议，已超过40年
- 针对特定场景，**TCP**的具有不同“风格”：

场景	挑战
长、胖管道（大量数据传输）	许多数据包在“飞行”；丢包将关闭管道
无线网络	由于无线链路噪声、移动等造成的丢包； TCP 将此视为拥塞丢包
长延迟链路	超长的 RTT
数据中心网络	延迟敏感
后台流量	低优先级的“后台” TCP 流

❖ 在 UDP 之上实现应用，将传输层功能移到应用层

- **HTTP/3: QUIC**

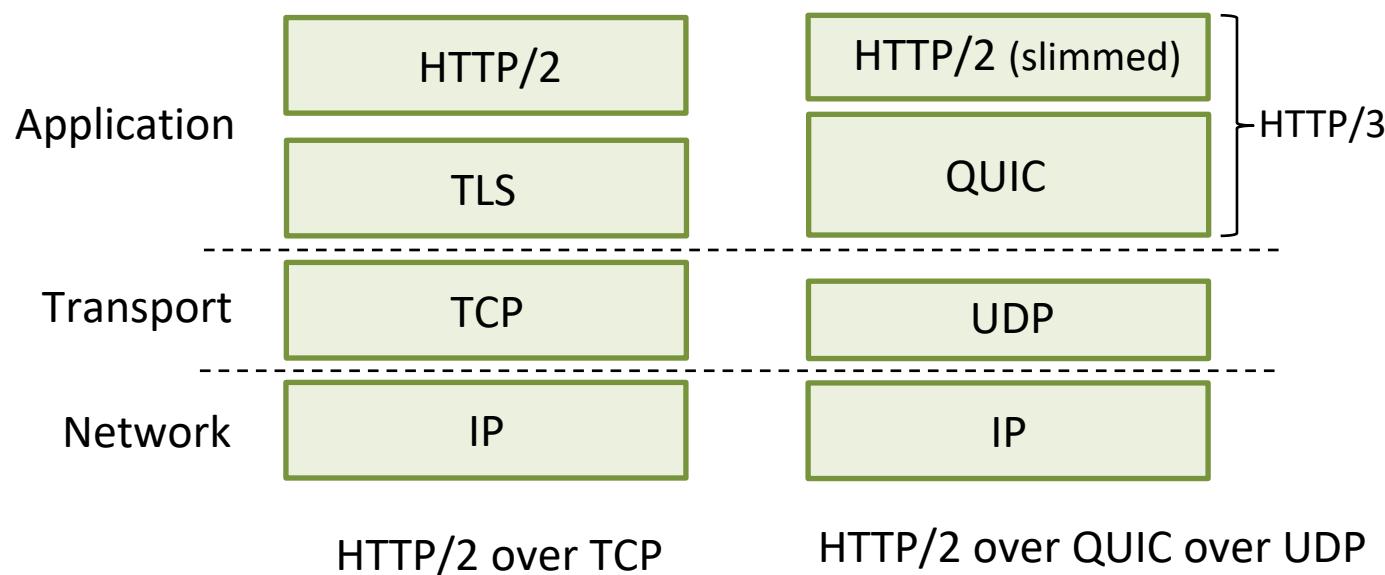


HTTP响应时间优化- HTTP/3

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



- ❖ QUIC: Quick UDP Internet Connections?
- ❖ 基于UDP的应用层协议
 - 提高 HTTP 的性能
 - 部署在许多谷歌服务器、APP应用程序（Chrome，移动 YouTube 应用程序）上





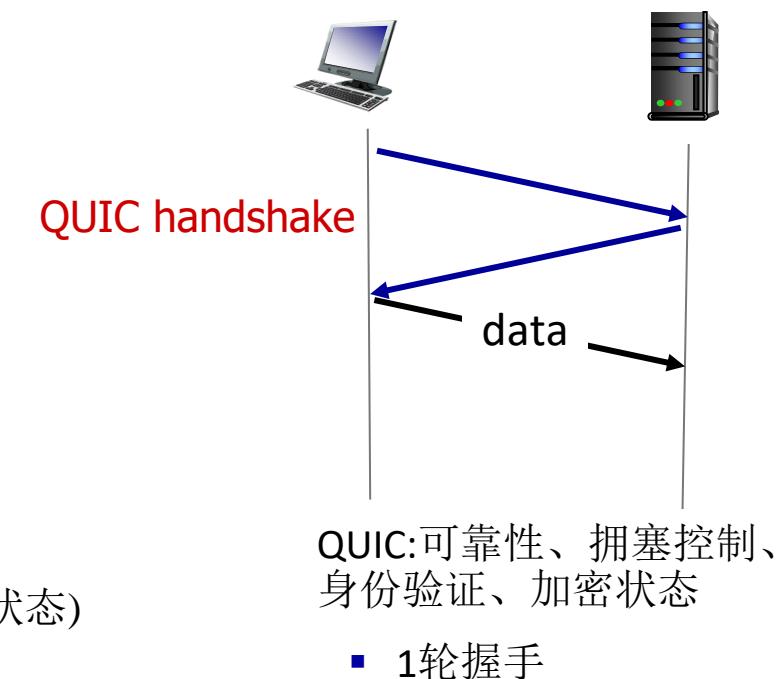
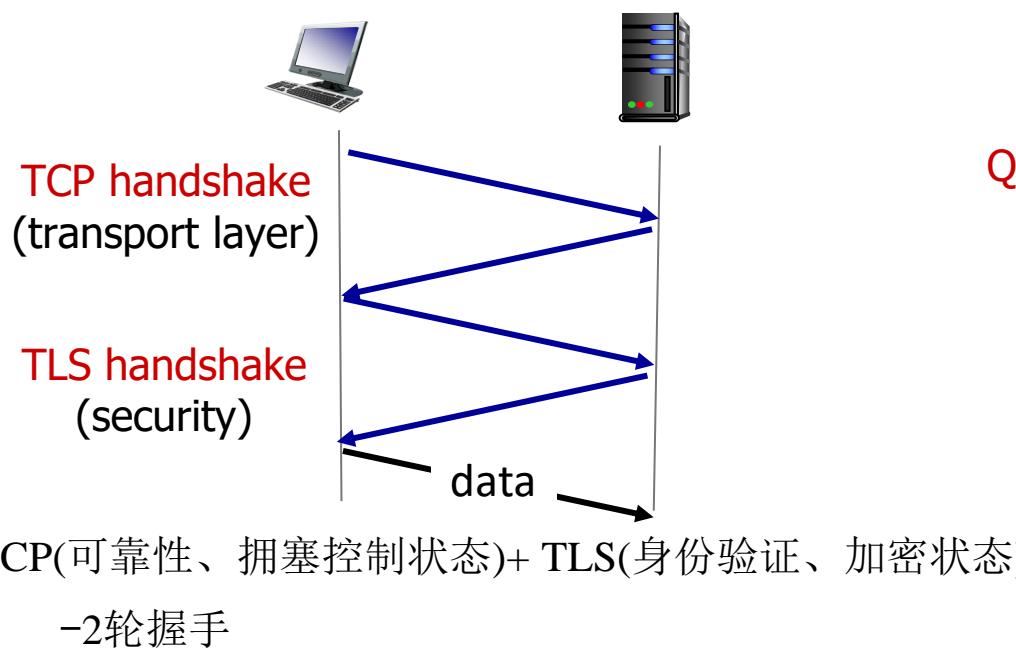
HTTP响应时间优化- HTTP/3

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



❖ QUIC: Quick UDP Internet Connections ?

- 应用层实现连接建立、错误控制、拥塞控制等功能
 - 差错控制和拥塞控制: 采用类似于 TCP的丢失检测和拥塞控制的算法
 - 连接建立: 可靠性、拥塞控制、身份验证、加密等在1个 RTT 建立状态
- 多个应用级“流”复用一个 QUIC 连接
 - 可靠数据传输与安全分离
 - 常见的拥塞控制



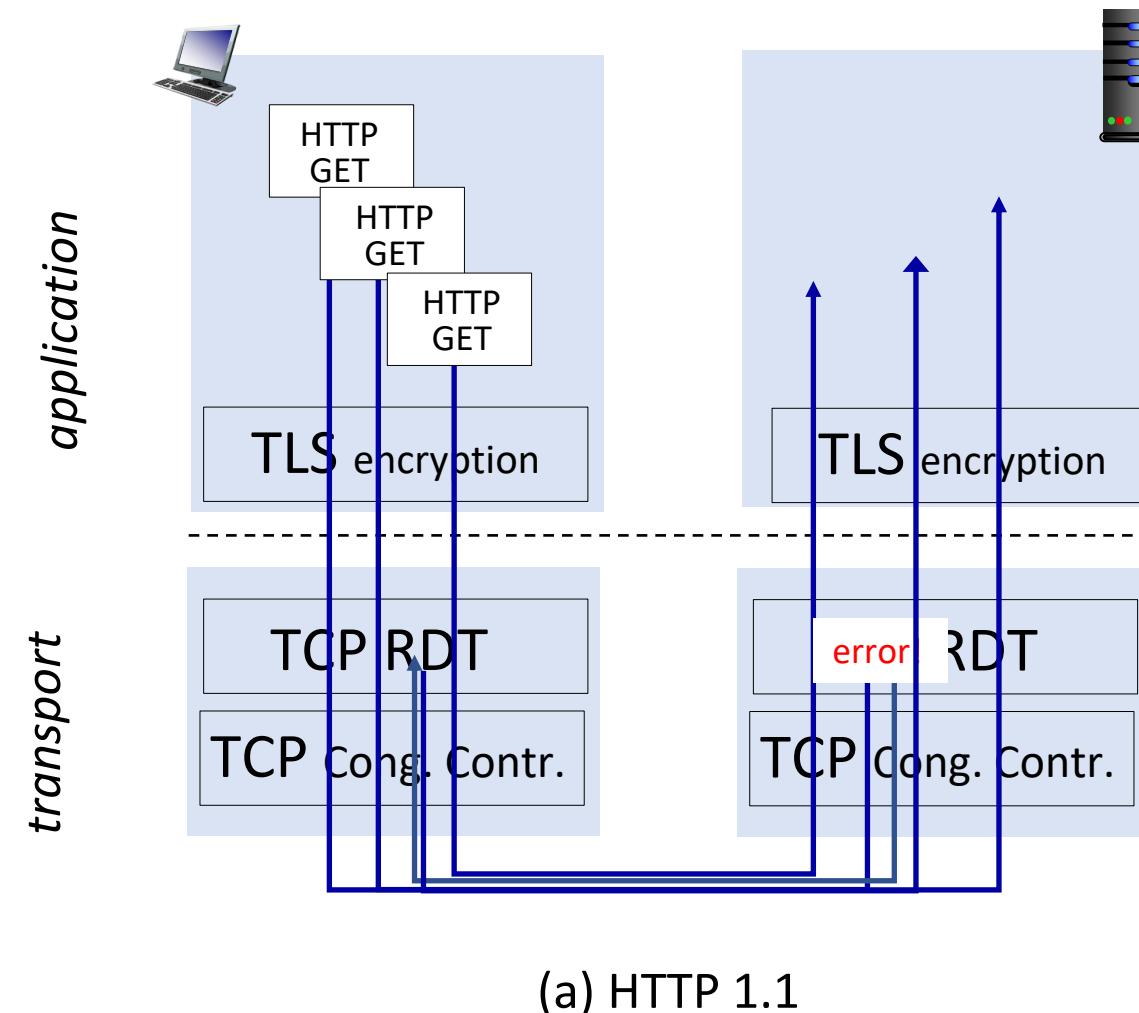


HTTP响应时间优化- HTTP/3

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



❖ QUIC流：并行性、无 HOL 阻塞

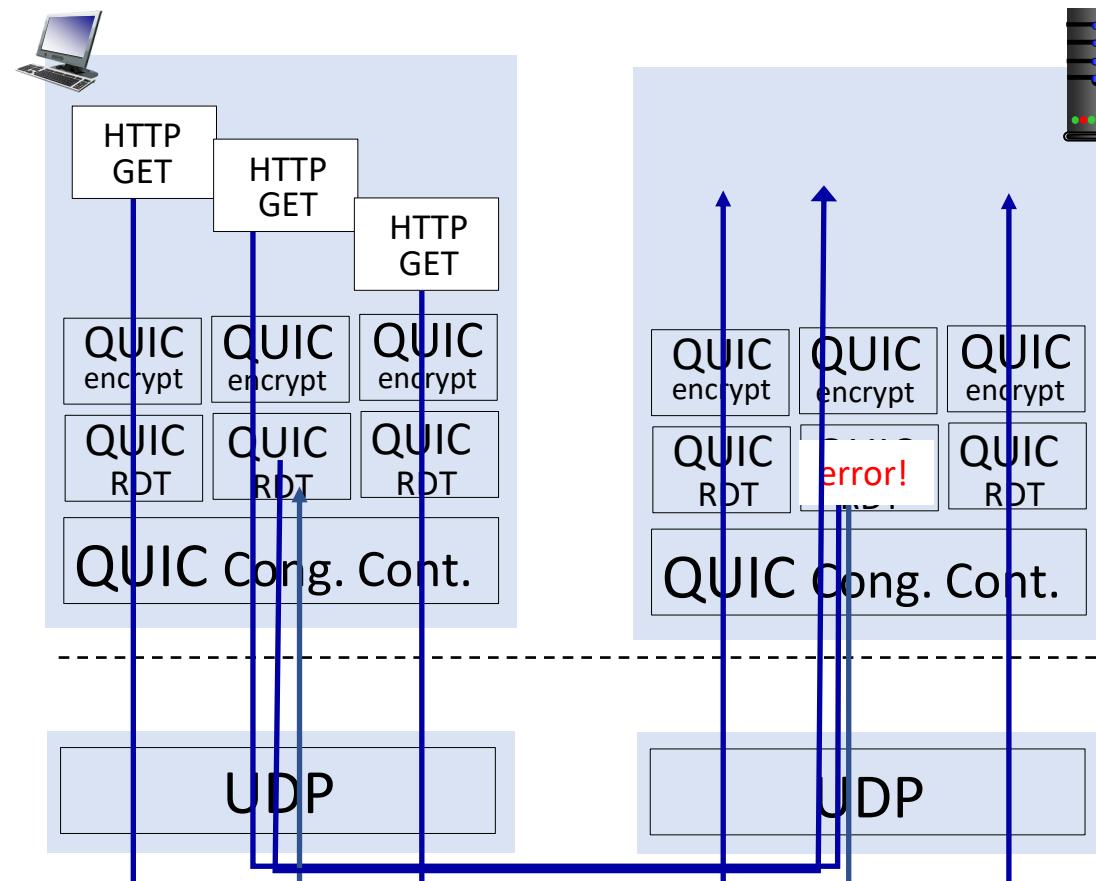




HTTP响应时间优化- HTTP/3

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**

❖ QUIC流：并行性、无 HOL 阻塞



(b) HTTP/2 with QUIC: no HOL blocking





HTTP响应时间优化-Web缓存/代理服务器

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



❖ 目标

- 在不访问源服务器的前提下满足客户端的HTTP请求。

❖ 为什么要发明这种技术？

- 缩短客户请求的响应时间
- 减少机构/组织的流量
- 在大范围内(Internet)实现有效的内容分发



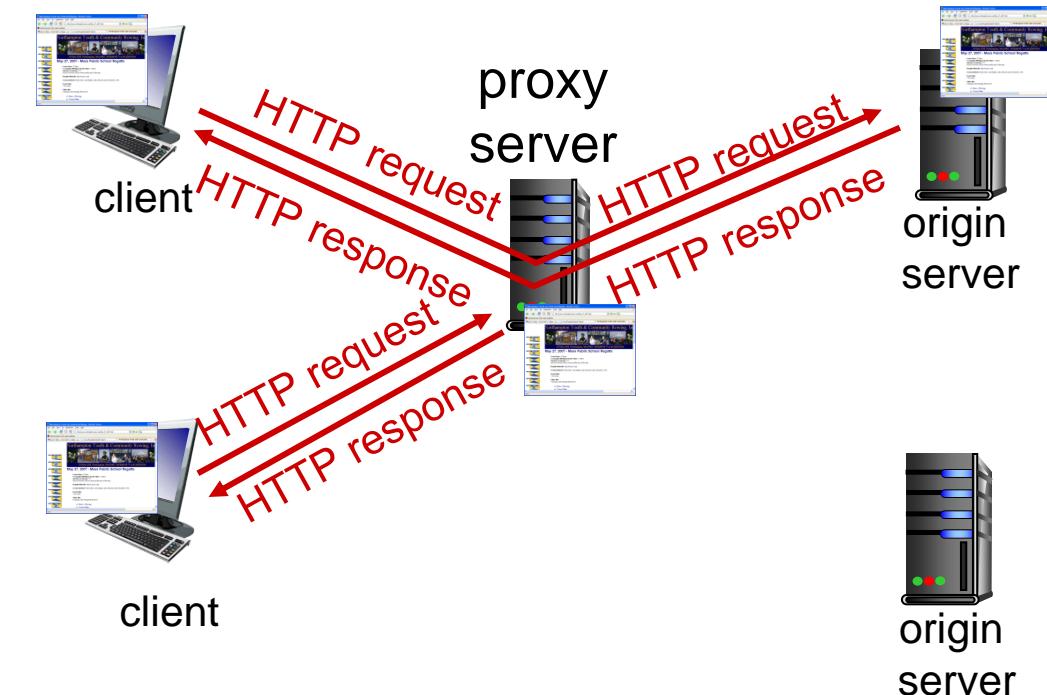


Web响应时间优化-Web缓存/代理服务器

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



- ❖ Web缓存/代理服务器
 - 用户设定浏览器通过缓存进行Web访问
 - 浏览器向缓存/代理服务器发送所有的HTTP请求
 - 如果所请求对象在缓存中，缓存返回对象
 - 否则缓存服务器向原始服务器发送HTTP请求，获取对象，然后返回给客户端并保存该对象
- ❖ 缓存既充当客户端，也充当服务器
- ❖ 一般由ISP(Internet服务提供商)架设





HTTP响应时间优化-Web缓存/代理服务器

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



❖ Web 缓存同时充当客户端和服务器

- 原始请求客户端的服务器
- 客户端到源服务器

❖ 服务器告诉缓存有关响应标头中对象允许的缓存:

```
Cache-Control: max-age=<seconds>
```

```
Cache-Control: no-cache
```

为什么使用 Web 缓存?

❖ 减少客户端请求的响应时间

- 缓存更靠近客户端

❖ 减少机构访问链路上的流量

❖ 互联网密集部署缓存

- 使“poor”内容提供商能够更有效地交付内容



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



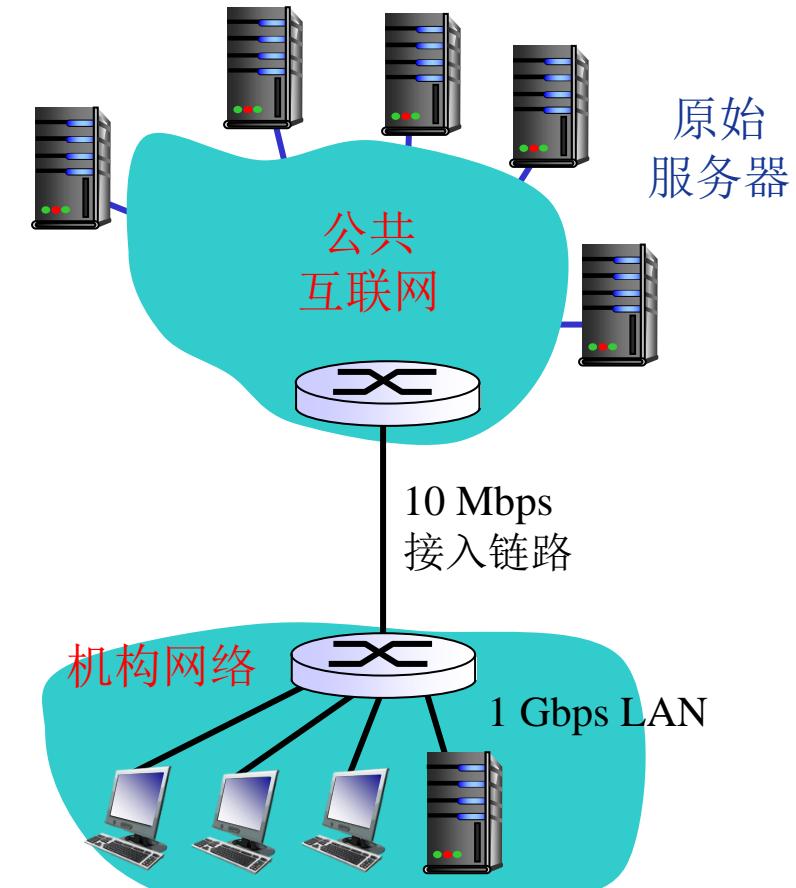
Web缓存示例(1)

❖ 假定:

- 平均对象大小=1 Mbits
- 机构网络中浏览器平均每秒有10个/秒到原始服务器的请求
- 从机构路由器到原始服务器的往返延迟 RTT=2秒
- 接入链路带宽: 10 Mbps

❖ 网络性能分析:

- 局域网(LAN)的利用率=1%
- 接入互联网的链路利用率=100%
- 总的延迟=互联网上的延迟+访问延迟+局域网延迟=2秒+几分钟+几微妙



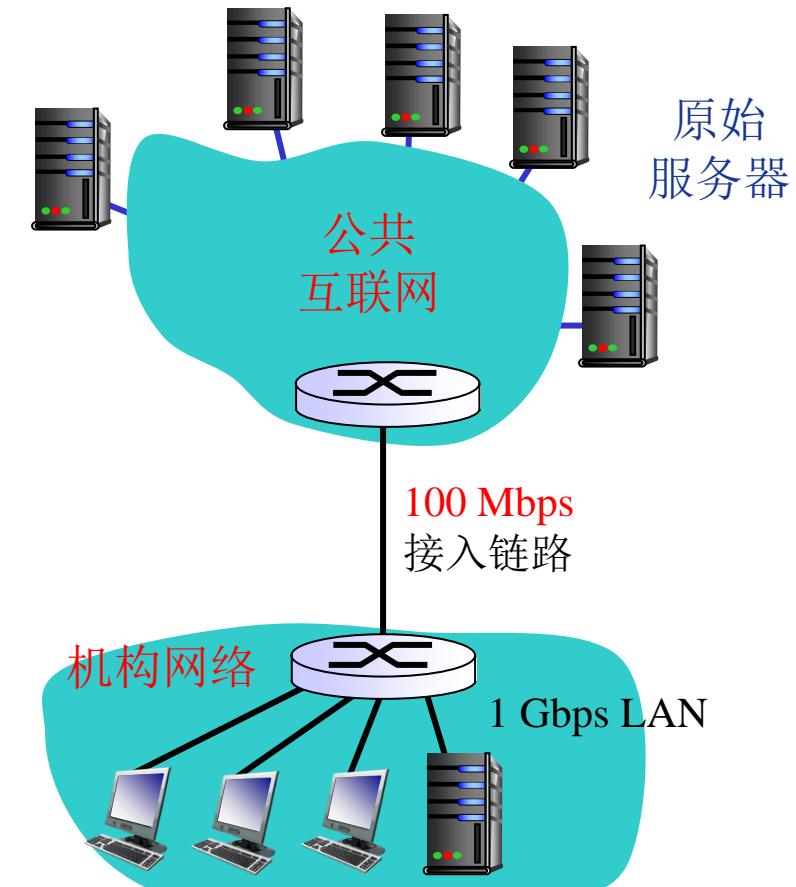


Web缓存示例(2)

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



- ❖ 解决方案1:
 - 提升互联网接入带宽=100Mbps
- ❖ 网络性能分析:
 - 局域网(LAN)的利用率=1%
 - 接入互联网的链路的利用率=10%
 - 总的延迟=互联网上的延迟+访问延迟+局域网延迟=2秒+几微妙+几微妙
- ❖ 问题:
 - 成本太高





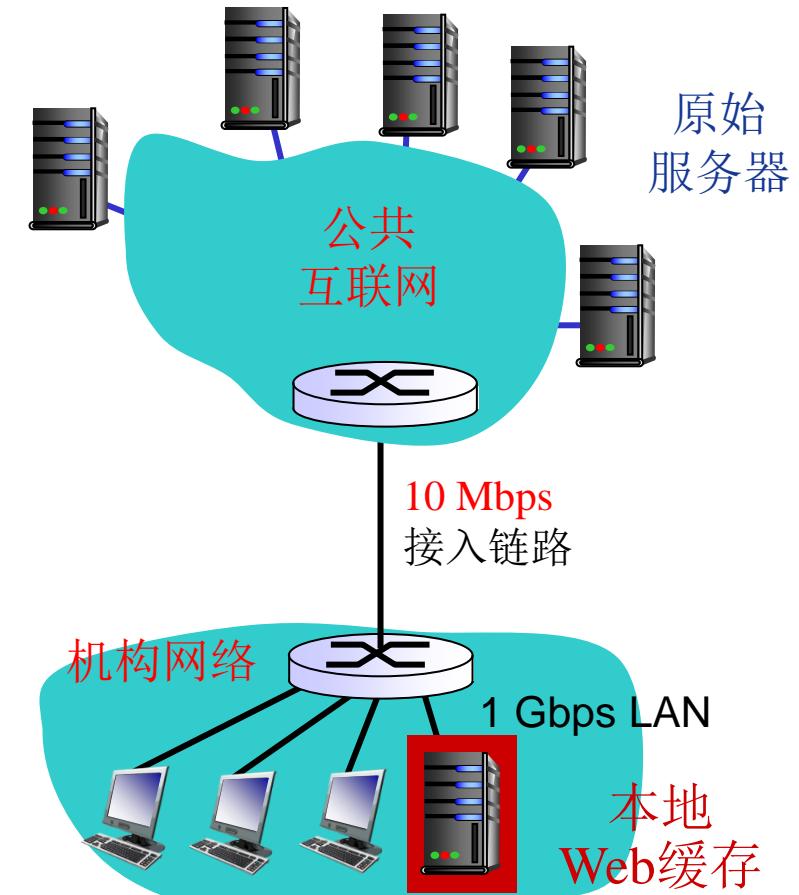
Web缓存示例(3)

❖ 解决方案2:

- 部署本地Web缓存
- 假定缓存命中率=0.4

❖ 网络性能分析:

- 40% 的请求立刻得到满足
- 60% 的请求通过原始服务器满足
- 接入链路的利用率下降到60%，
从而其延迟可以忽略不计，例如
10微秒
- 总的平均延迟=互联网上的延迟+
访问延迟+局域网延迟=0.6×2.0秒
+0.4×n微秒≈1.2秒





Web应用响应时间优化

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**

❖ 讨论：其他优化Web应用响应时间的方案？

❖ CDN?





条件性GET方法

❖ 目标:

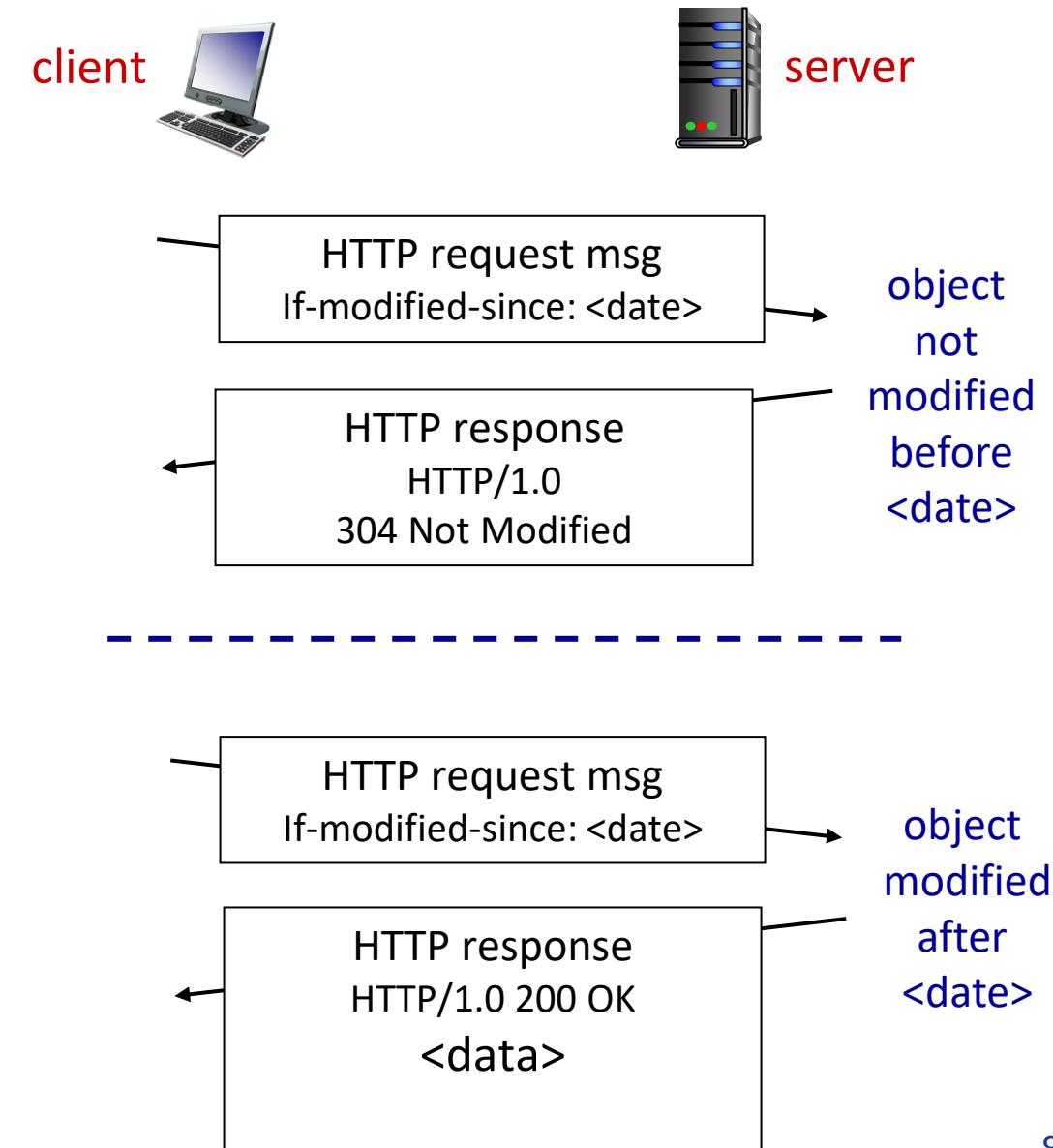
- 如果缓存有最新的版本，则不需要发送请求对象

❖ 缓存:

- 在HTTP请求消息中声明所持有版本的日期
- `If-modified-since: <date>`

❖ 服务器:

- 如果缓存的版本是最新的，则响应消息中不包含对象
- `HTTP/1.0 304 Not Modified`





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



为什么需要Cookie ?

讨论:

- 无状态HTTP协议会带来什么问题?
- 很多应用需要服务器掌握客户端的状态, 如网上购物, 如何实现?





2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

2.5 Email应用

2.6 Web应用

Cookie技术

❖Cookie技术

- 某些网站为了辨别用户身份、进行**session**跟踪而储存
在用户本地终端上的数据（通常经过加密）。
- RFC6265

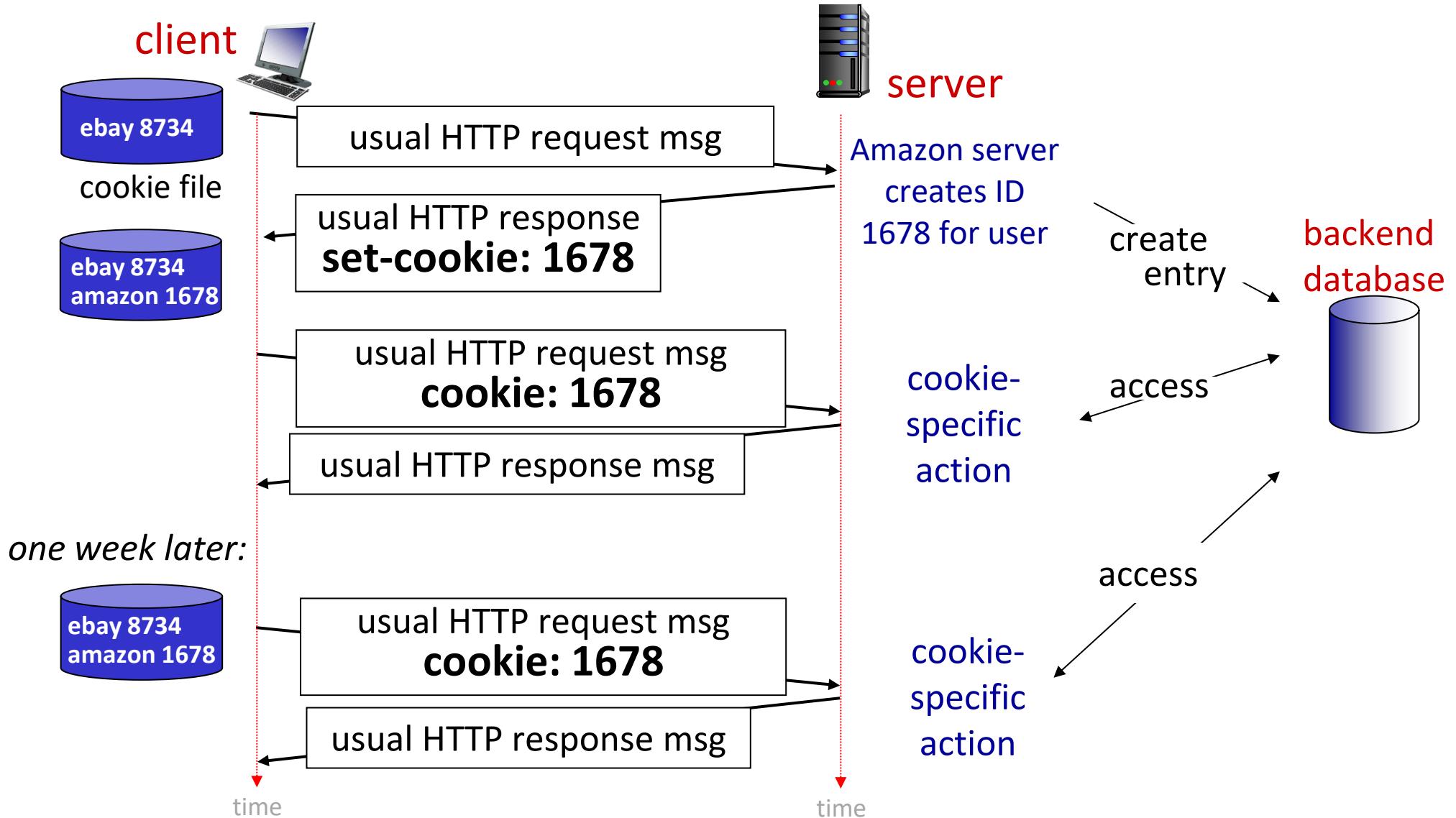
❖Cookie的组件

- HTTP响应消息的cookie头部行
- HTTP请求消息的cookie头部行
- 保存在客户端主机上的cookie文件，由浏览器管理
- Web服务器端的后台数据库



Cookie的原理

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**





2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

2.5 Email应用

2.6 Web应用

Cookie的作用

❖ Cookie能够用于：

- 身份认证
- 购物车
- 推荐
- Web e-mail
-

❖ 隐私问题

cookie时代终结 苹果谷歌微软抢占新标准滩头

腾讯科技 [微博] 晨曦 2013年11月05日16:25

分享 ▾

[导读]科技巨头均已经开始研发或者采用替代cookie的技术，并把重点放在了移动端。





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



HTTP vs SMTP

❖ SMTP:

- 推式(push)
- 使用持久性连接
- 要求消息必须由7位ASCII码构成
- 利用CRLF,CRLF确定消息的结束
- 多个对象在由多个部分构成的消息中发送

❖ HTTP:

- 拉式(pull)
- 每个对象封装在独立的响应消息中

❖ 共同点:

- 都使用命令/响应交互模式
- 命令和状态代码都是ASCII码

单选题 1分

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用**



某浏览器发出的HTTP请求报文如下：

```
GET /index.html HTTP/1.1
```

```
Host: www.test.edu.cn
```

```
Connection: Close
```

```
Cookie: 123456
```

下列叙述中，错误的是

- A 该浏览器请求浏览index.html
- B index.html存放在www.test.edu.cn上
- C 该浏览器请求使用持续连接
- D 该浏览器曾经浏览过www.test.edu.cn

提交



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

2.7 P2P应用

聂兰顺



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用

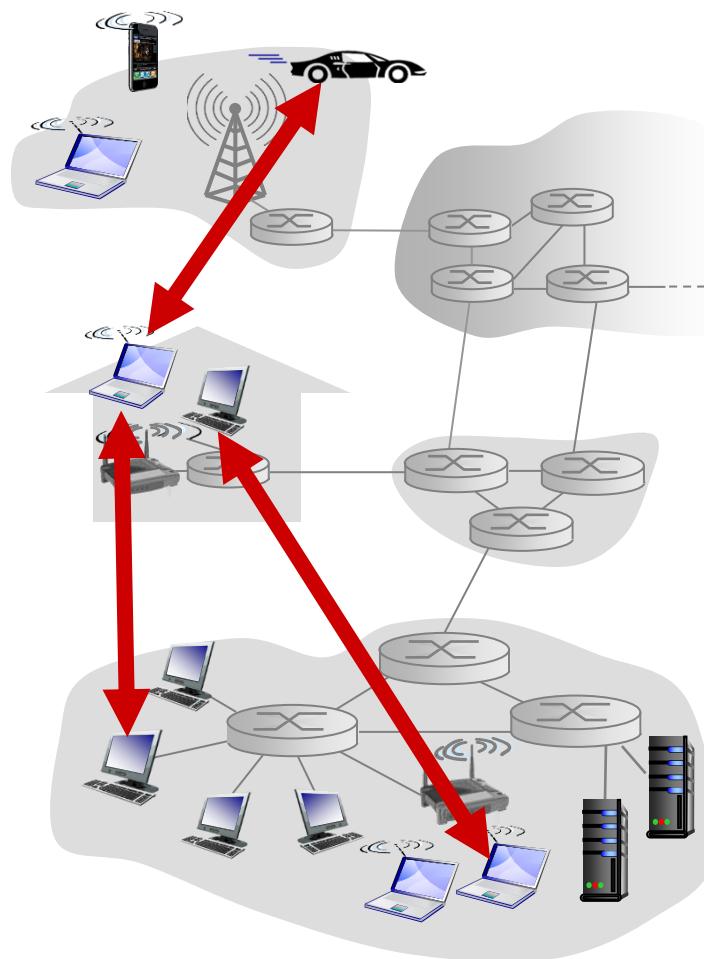
2.7 P2P应用



纯P2P架构

❖ Peer-to-Peer

- ❖ 没有服务器
- ❖ 任意端系统之间直接通信
- ❖ 节点阶段性接入Internet
- ❖ 节点可能更换IP地址
- ❖ 以具体应用为例讲解



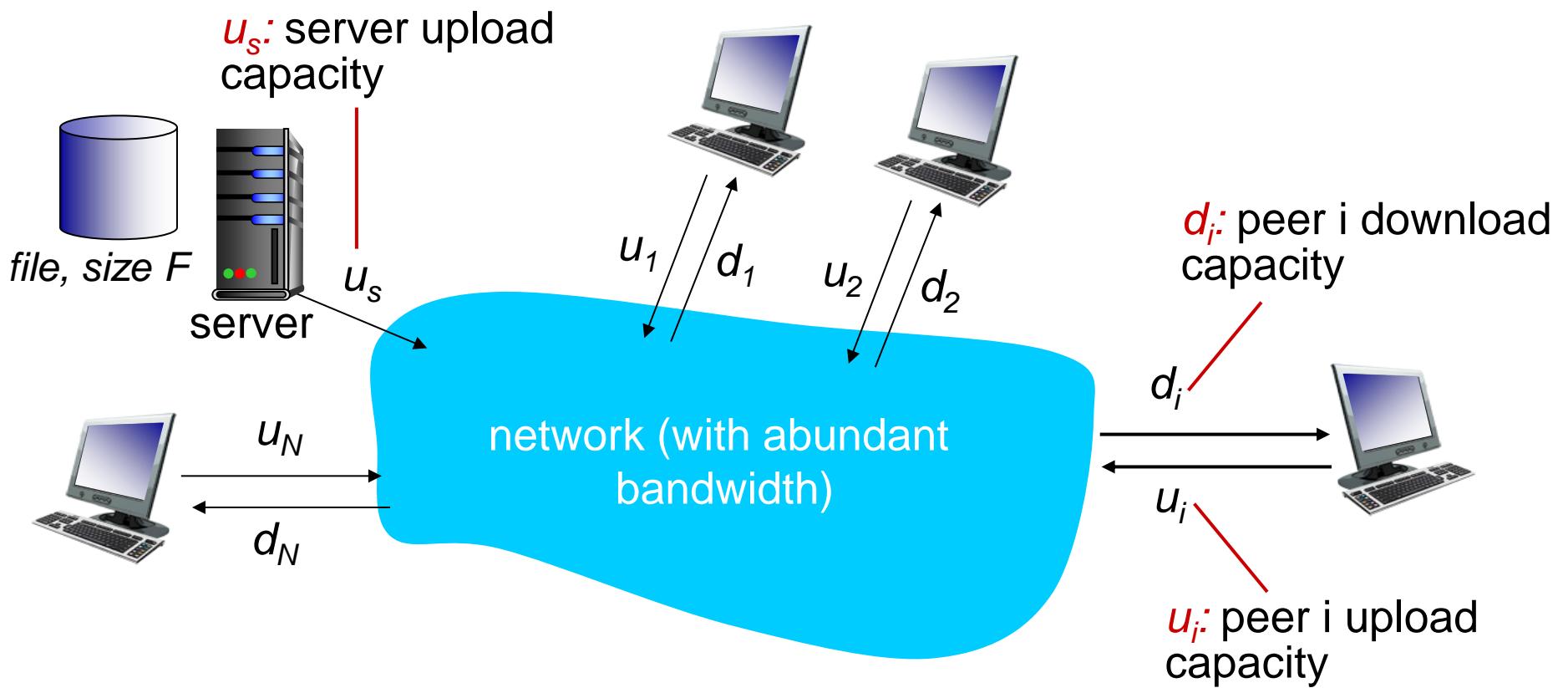


文件分发：客户机/服务器 vs. P2P

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



问题：从一个服务器向N个节点分发一个文件需要多长时间？



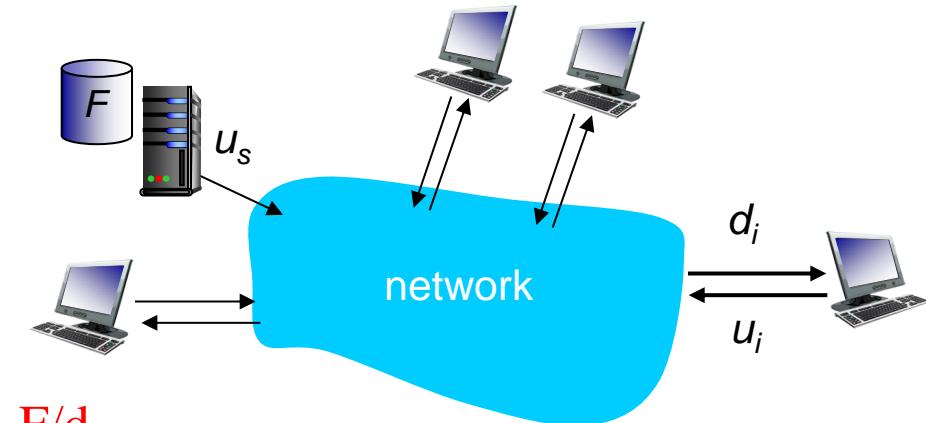


- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



文件分发：客户机/服务器

- ❖ 服务器串行地发送N个副本
 - 时间: NF/u_s
- ❖ 每个客户机必须下载1个文件副本
 - 客户*i*下载1个副本时间: F/d_i
 - 下行带宽最小的客户下载1个副本时间: F/d_{min}



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

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

increases linearly in N



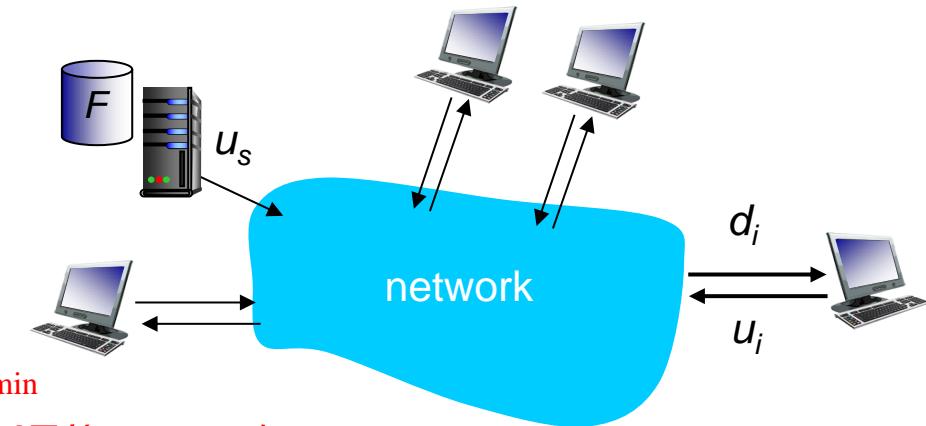
- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用

2.7 P2P应用



文件分发：P2P

- ❖ 服务器必须至少上传1个副本
 - 时间: F/u_s
- ❖ 每个客户机必须下载1个文件副本
 - 客户*i*下载1个副本时间: F/d_i
 - 下行带宽最小的客户下载1个副本时间: F/d_{min}
- ❖ 服务器与客户共同上传NF比特数据时间: $NF/(u_s + \sum u_i)$



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

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

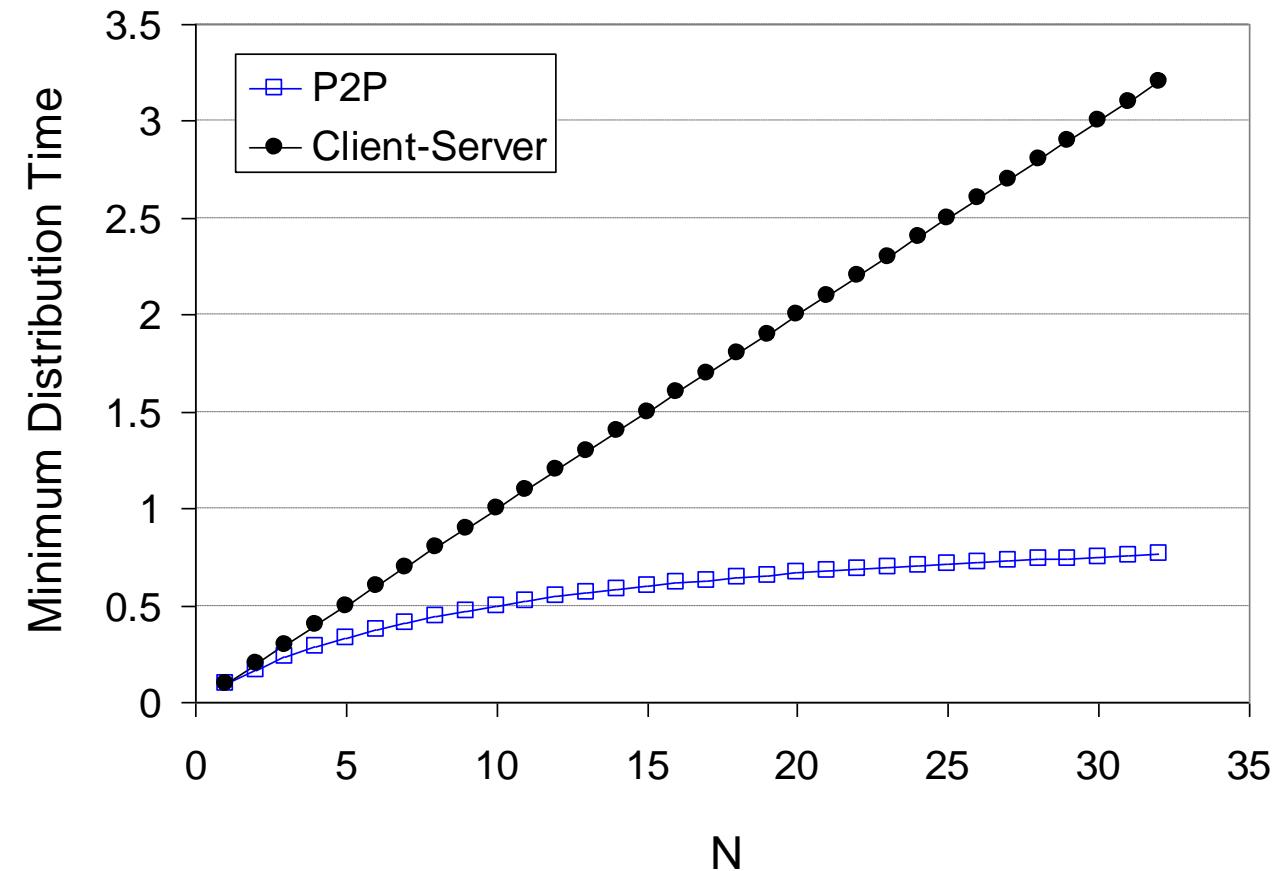
increases linearly in N ...
... but so does this, as each peer brings service capacity



客户机/服务器 vs. P2P: 例子

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**

客户端上传速率 = u , $F/u = 1$ 小时, $u_s = 10u$, $d_{min} \geq u_s$





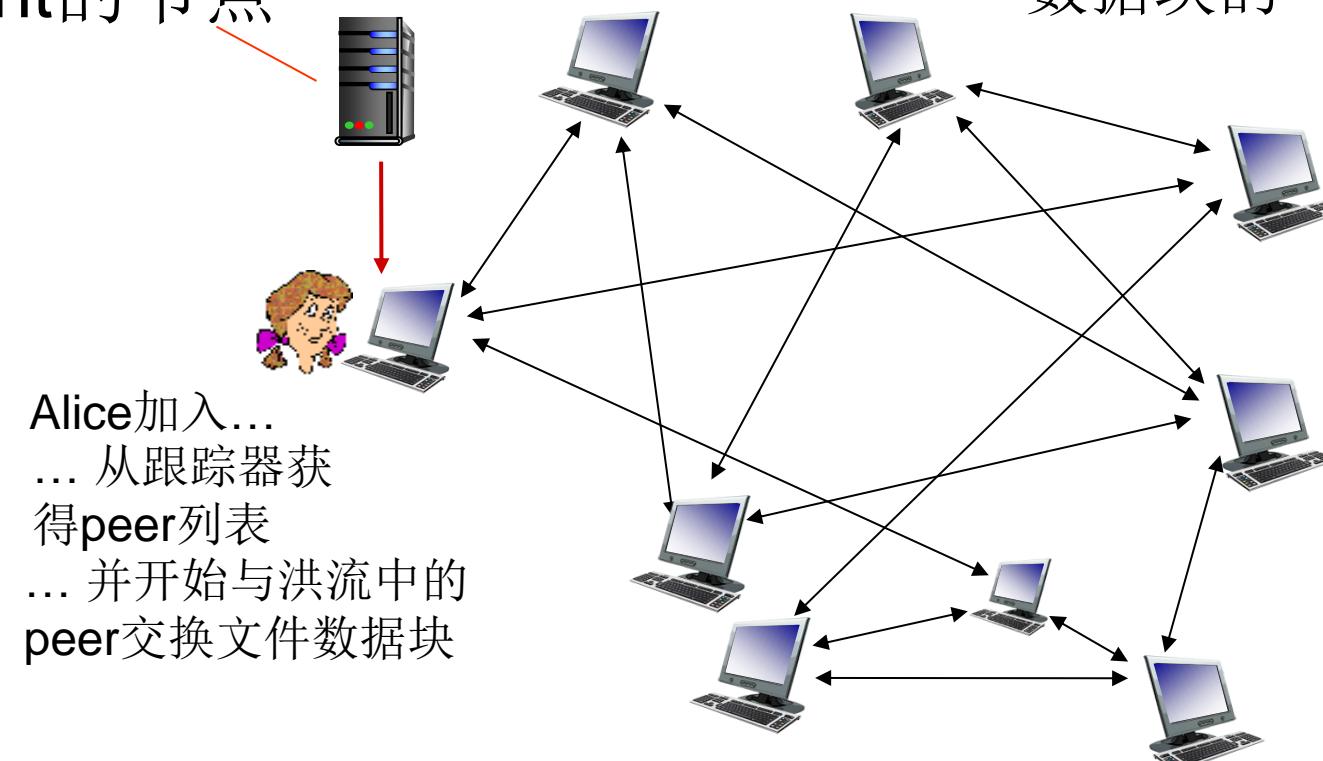
- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



文件分发：BitTorrent

- 文件被分割成大小为256Kb数据块（chunks）
- 洪流中的对等端间发送/接收文件数据块

tracker: 跟踪参与torrent的节点

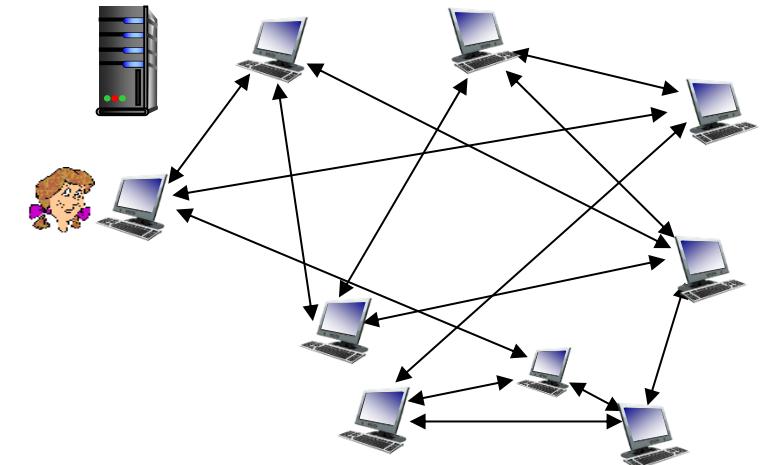


torrent: 交换文件数据块的一组peer



BitTorrent (1)

- ❖ 文件划分为256KB的chunk
- ❖ 节点加入torrent:
 - 没有chunk，但是会逐渐积累
 - 向tracker注册以获得节点清单，与某些节点（“邻居”）建立连接
- ❖ 下载的同时，节点需要向其他节点上传chunk
- ❖ 节点可能加入或离开
- ❖ 一旦节点获得完整的文件，它可能（自私地）离开或（无私地）留下





BitTorrent (2)

❖ 获取chunk

- 给定任一时刻，不同的节点持有文件的不同**chunk**集合
- 节点(Alice)定期查询每个邻居所持有的**chunk**列表
- 节点发送请求，请求获取缺失的**chunk**
 - 稀缺优先

❖ 发送chunk: 互惠 (tit-for-tat)

- Alice向4个邻居发送**chunk**: 正在向其发送**Chunk**, 速率最快的4个
 - 每10秒重新评估top 4
- 每30秒随机选择一个其他节点，向其发送**chunk**
 - 新选择节点可能加入top 4
 - “optimistically unchoke” (乐观疏通)



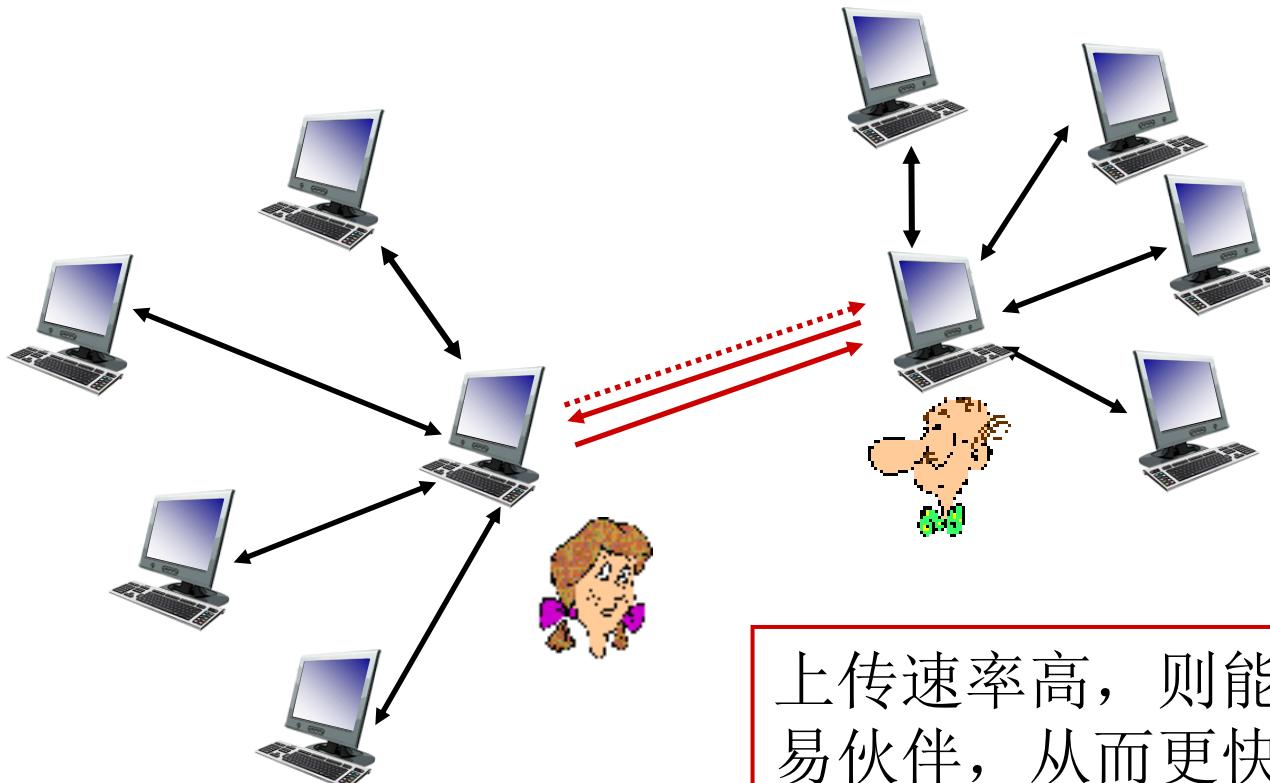
- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用

2.7 P2P应用



BitTorrent: 互惠 (Tit-for-tat)

- (1) Alice “乐观疏通” 向Bob发送数据
- (2) Alice变成Bob的top-four数据提供方之一; Bob投桃报李
- (3) Bob变成Alice的top-four数据提供方之一



上传速率高，则能够找到更好的交易伙伴，从而更快地获取文件！



P2P: 搜索信息

❖ P2P系统的索引：信息到节点位置(IP地址+端口号)的映射

❖ 文件共享(电驴)

- 利用索引动态跟踪节点所共享的文件的位置
- 节点需要告诉索引它拥有哪些文件
- 节点搜索索引，从而获知能够得到哪些文件



❖ 即时消息(QQ)

- 索引负责将用户名映射到位置
- 当用户开启IM应用时，需要通知索引它的位置
- 节点检索索引，确定用户的IP地址





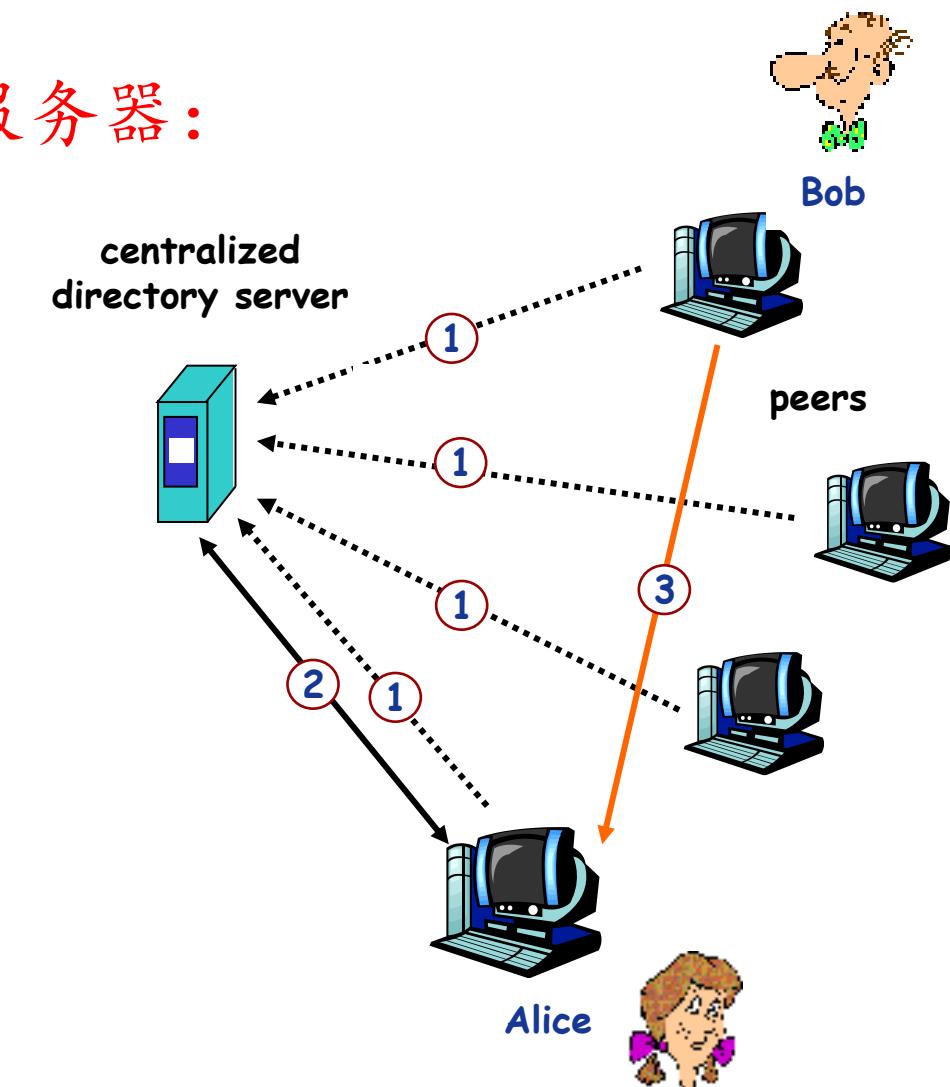
- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



集中式索引

❖ Napster最早采用这种设计

- 1) 节点加入时，通知中央服务器：
 - IP地址
 - 内容
- 2) Alice查找“Hey Jude”
- 3) Alice从Bob处请求文件





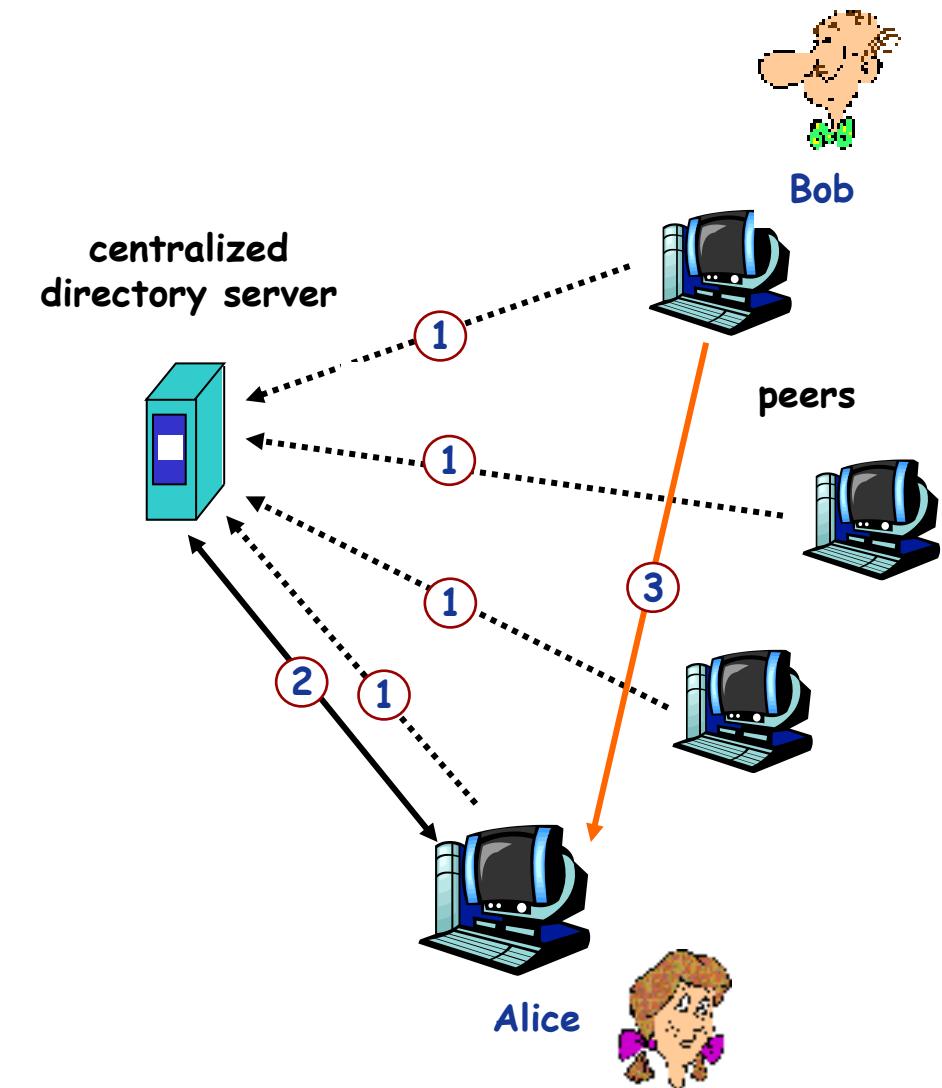
- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



集中式索引的问题

内容和文件传输是分布式的，
但是内容定位是高度集中式的

- ❖ 单点失效问题
- ❖ 性能瓶颈
- ❖ 版权问题





洪泛式查询: Query flooding

- ❖ 完全**分布式**架构
- ❖ Gnutella采用这种架构
- ❖ 每个节点对它共享的文件进行索引，且只对它共享的文件进行索引

- 覆盖网络(overlay network): Graph**
- ❖ 节点X与Y之间如果有TCP连接，那么构成一个边
 - ❖ 所有的活动节点和边构成覆盖网络
 - ❖ 边：虚拟链路
 - ❖ 节点一般邻居数少于10个

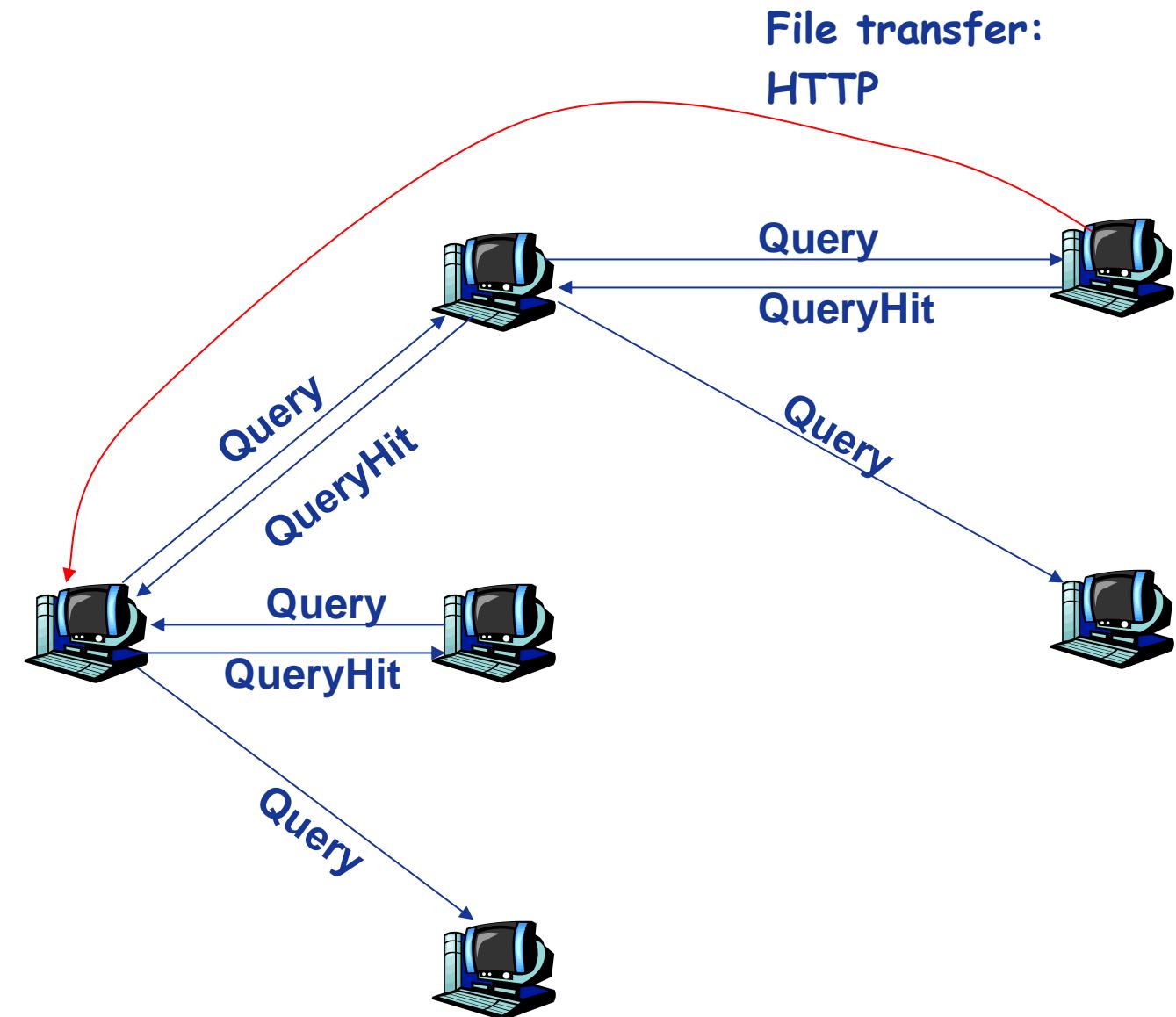


洪泛式查询: Query flooding

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用

2.7 P2P应用

- ❖ 查询消息通过已有的TCP连接发送
- ❖ 节点转发查询消息
- ❖ 如果查询命中，则利用反向路径发回查询节点





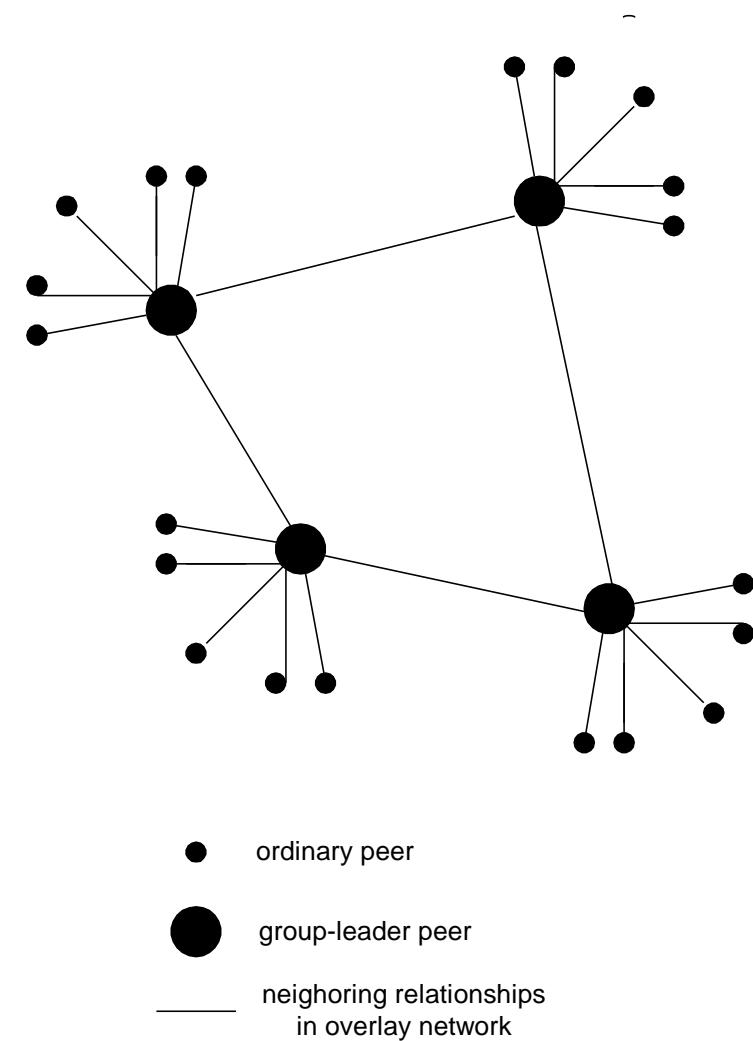
- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用

2.7 P2P应用



层次式覆盖网络

- ❖ 介于集中式索引和洪泛查询之间的方法
- ❖ 每个节点或者是一个**超级节点**，或者被**分配**一个超级节点
 - 节点和超级节点间维持TCP连接
 - 某些超级节点对之间维持TCP连接
- ❖ 超级节点负责跟踪子节点的内容



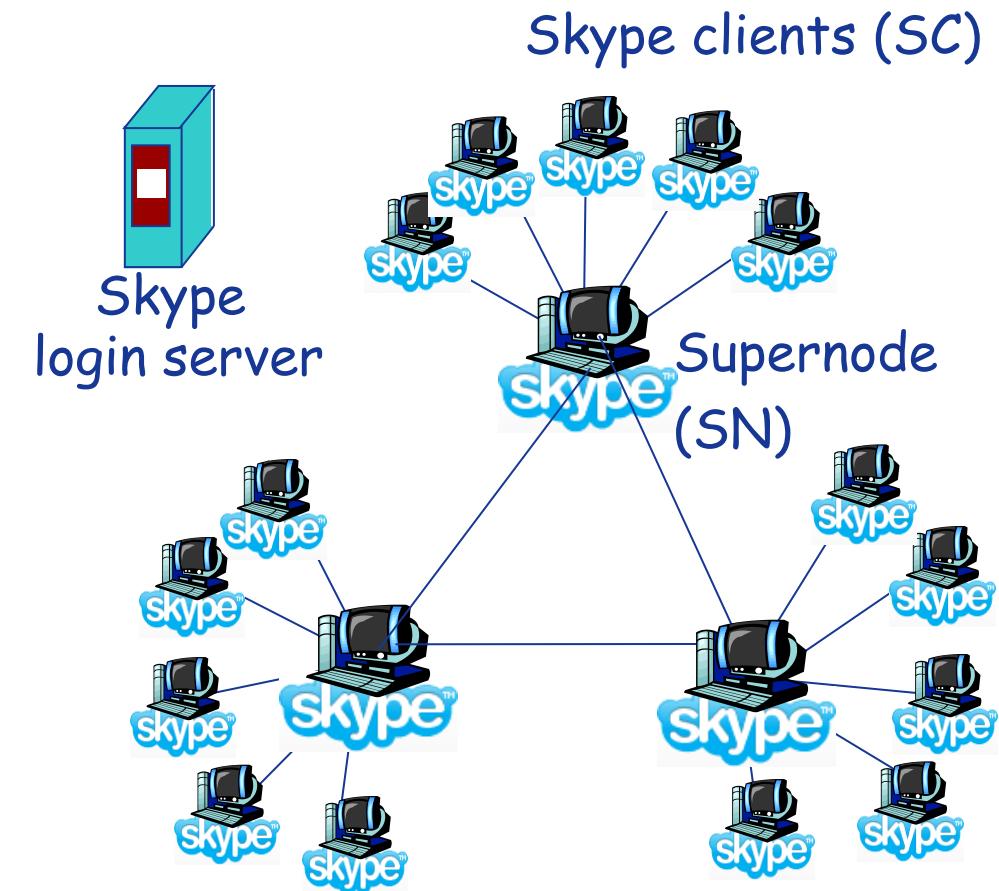


- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



P2P案例应用：Skype

- ❖ 本质上是P2P的：用户/节点对之间直接通信
- ❖ 私有应用层协议
- ❖ 采用层次式覆盖网络架构
- ❖ 索引负责维护用户名与IP地址间的映射
- ❖ 索引分布在超级节点上





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



分布式哈希表 (DHT)

- ❖ DHT (Distributed Hash Table) : 一个**分布式P2P数据库**
- ❖ 存储(**key, value**)对；例如：
 - **Key**: 身份证号码; **value**: 人名
 - **Key**: 电影标题; **value**: IP 地址
- ❖ Peer通过“**key**”查询DHT
 - DHT返回与键匹配的值
- ❖ Peers也可以向DHT插入(**key, value**)
 - ❖ (**key, value**)对分布在（数百万个对等体）上



Q:如何将key分配到对等方?

❖核心问题:

- 分配(key, value)对到对等方.

❖基本思想:

- 将每个key转换为一个整数
- 为每个对等方分配一个整数 (ID)
- 将(key, value)对存放到ID “距离” key最近的Peer上



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



DHT标识符

- ❖ 为每个对等方分配整数标识符，取值范围:[0, 2^n-1]
 - 每个标识符为 n 位数
- ❖ 要求每个键是同一范围内的整数
- ❖ 为获取整数键，散列原始键
 - e.g., key = hash(“我和我的祖国”)
 - 这就是为什么称之为分布式哈希表



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用

2.7 P2P应用



将key分配给对等方

- ❖ 规则: 将key分配给与 ID最近的对等方。
- ❖ 本课程约定: “最近” 是指key的直接后继者
- ❖ e.g., $n=4$; Peers: 1,3,4,5,8,10,12,14;
 - key = 13, 则直接后继peer = 14
 - key = 15, 则直接后继peer = 1

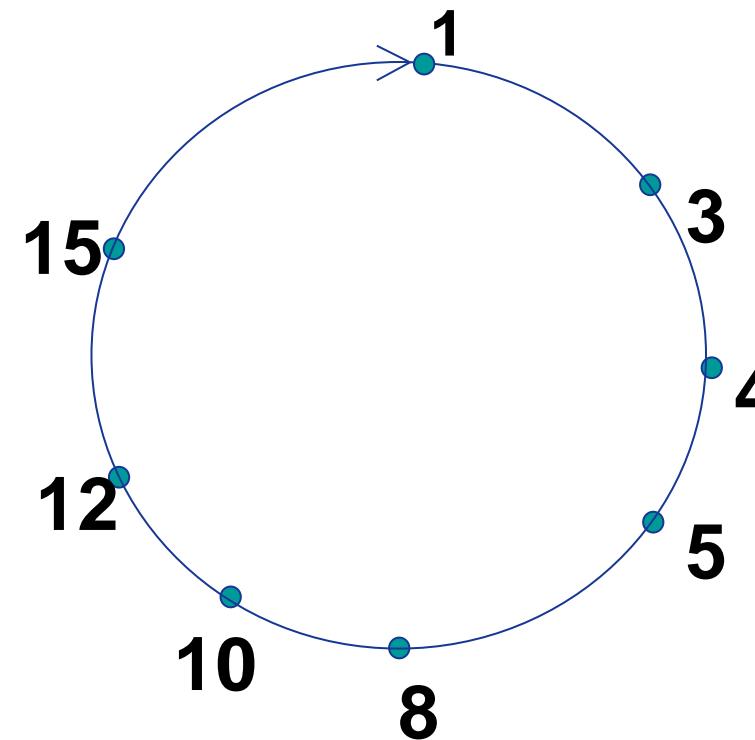


- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



圆形DHT (1)

- ❖ 每个对等方只知道直接的继任者和前任.
- ❖ “覆盖网络”



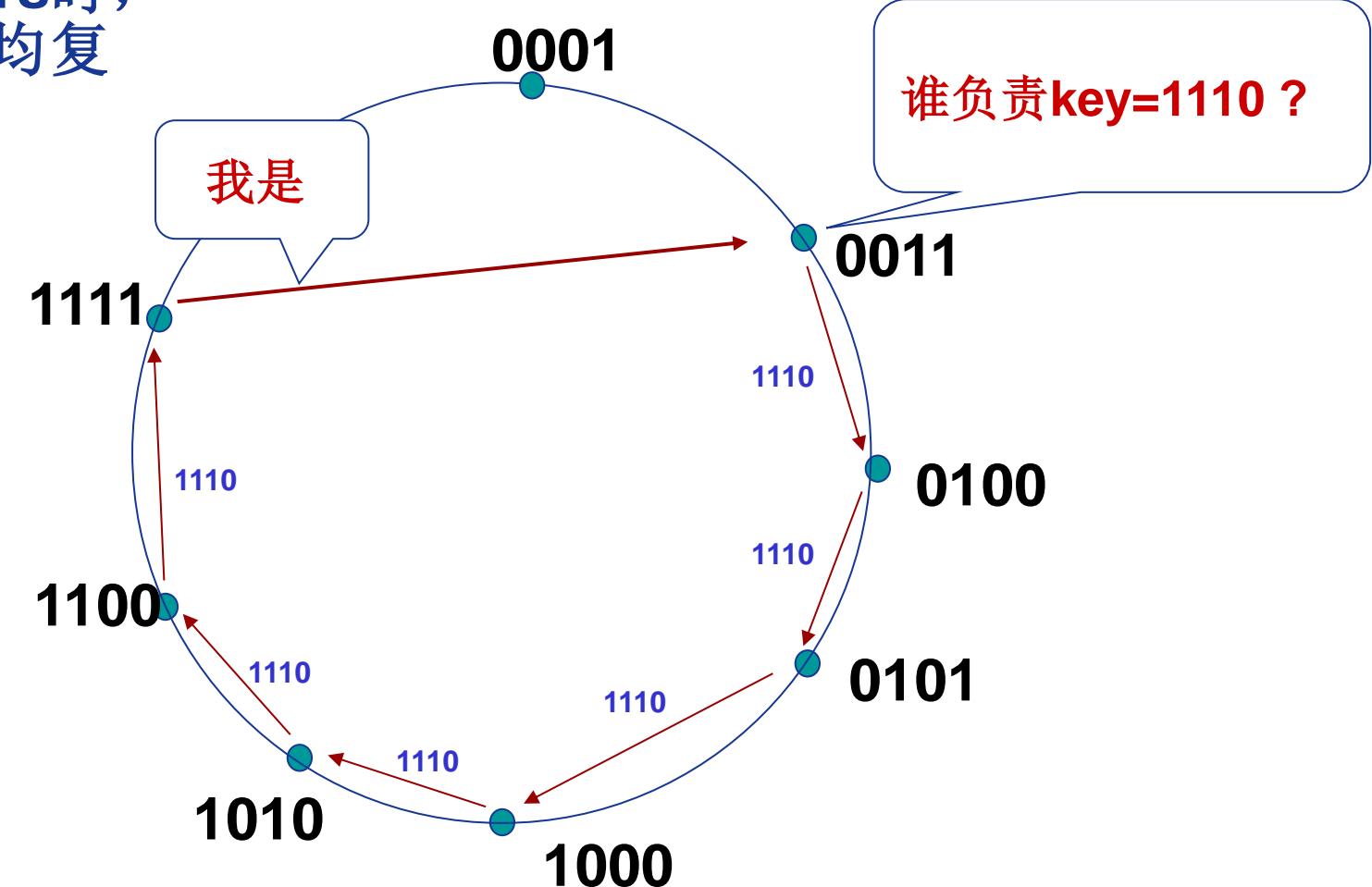


- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



圆形DHT (1)

当存在**N个Peers**时，
查询报文的平均复
杂度为**O(N)**

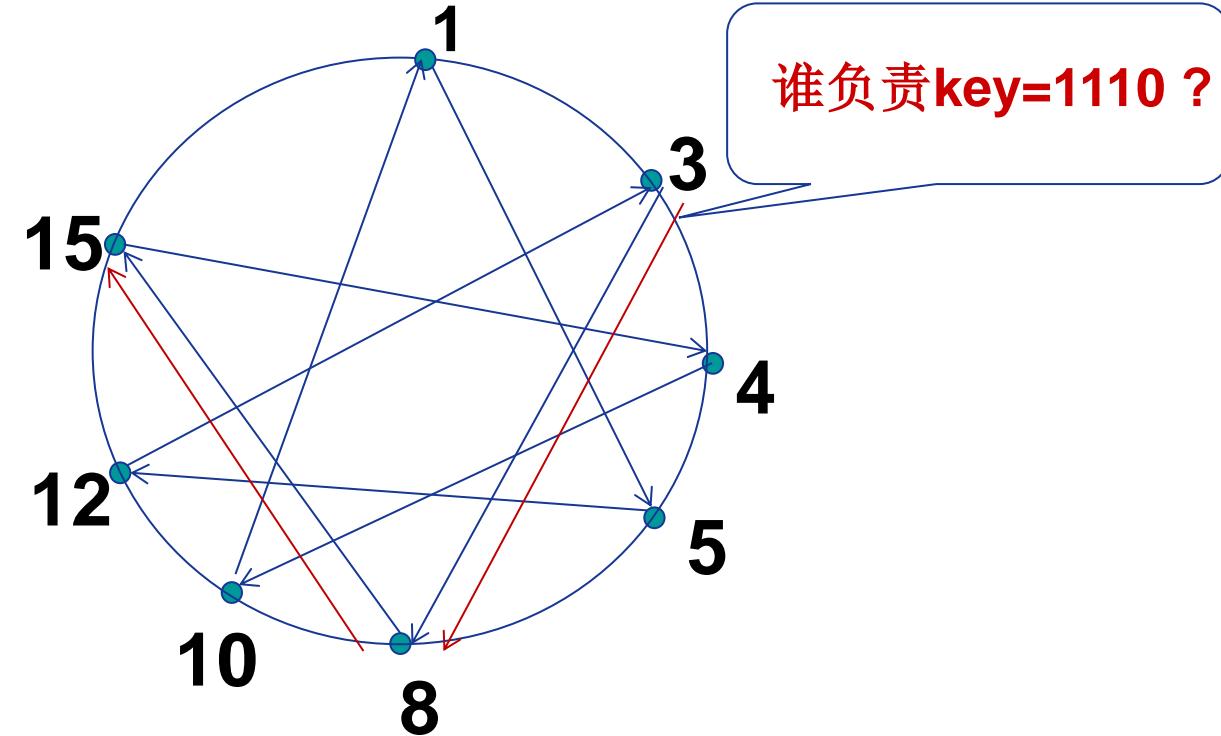


定义“最近”
为最近的后继



带有捷径的圆形 DHT

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



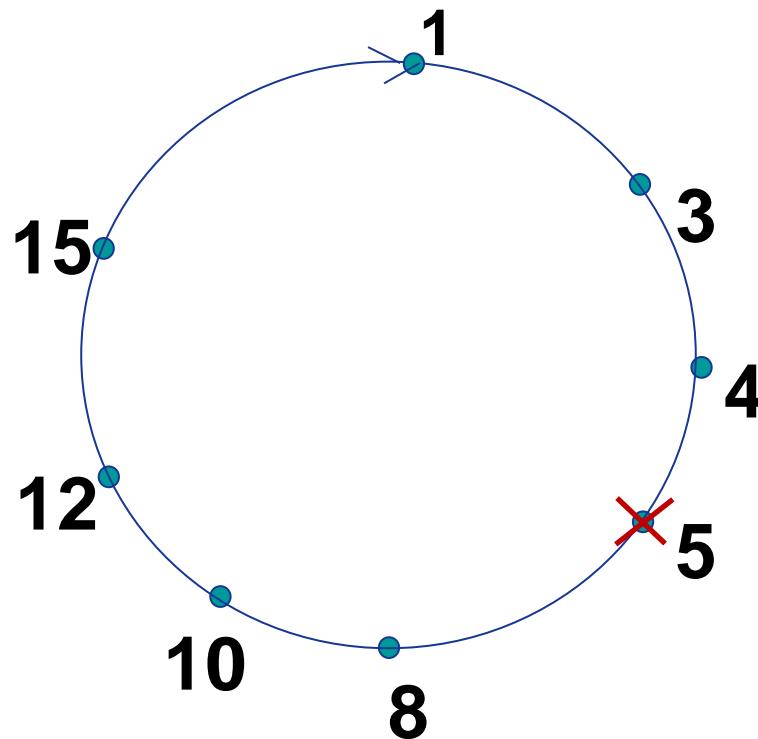
- ❖ 每个对等体都跟踪前续、后继者以及捷径的IP地址
- ❖ 消息数从 6 条减少到 2 条
- ❖ 可以设计与 $O(\log N)$ 的邻居建立捷径，使查询消息为 $O(\log N)$



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用**



Peer流失



处理对等方流失:

- ❖ Peers 可能来来去去(churn)
- ❖ 每个对等体都知道其两个后继Peers的地址
- ❖ 每个对等体定期 ping 其两个后续节点以检查活动状态
- ❖ 如果直接后继者离开，则选择简介后继者作为新的直接后继者

示例：Peer5突然离开

Peer4检测到Peer5离开；则使Peer8成为其直接后继者；并询问Peer8谁是它的直接后继者；然后使Peer8的直接后继者成为Peer4的第二后继者。

- ❖ 如果Peer13想加入怎么办？



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

2.8 Socket编程（略）

聂兰顺

通过MOOC学习：<https://www.icourse163.org/course/HIT-154005>

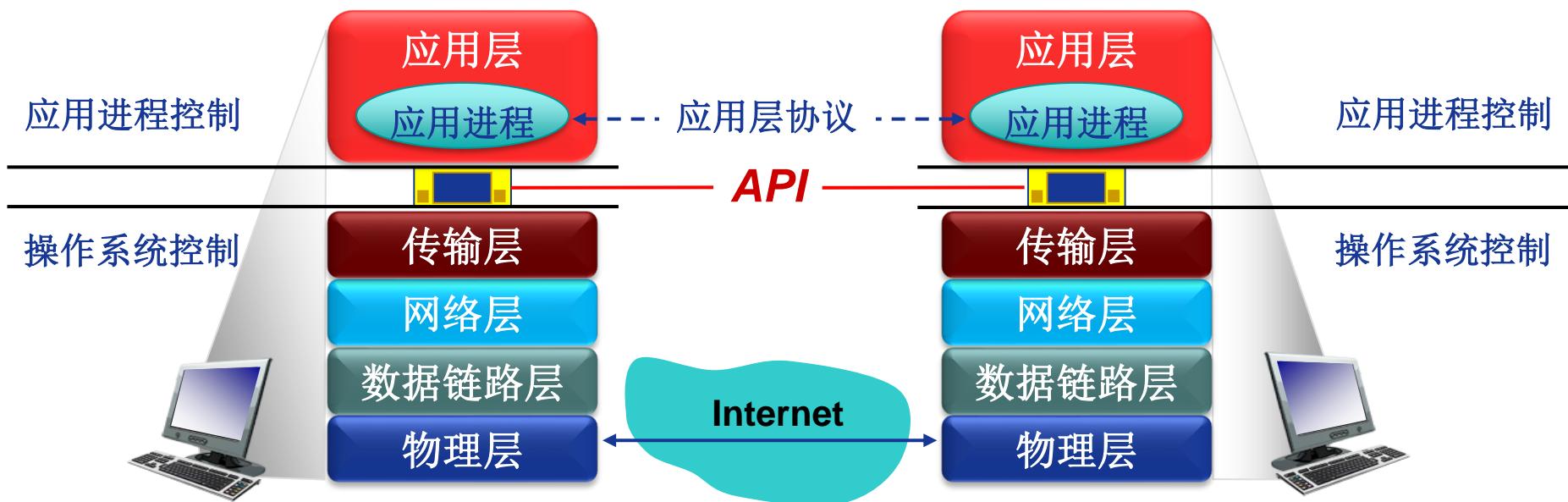


- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



应用编程接口 API

应用编程接口 API (Application Programming Interface)



应用编程接口API:就是应用进程的控制权和操作系统的控制权进行转换的一个系统调用接口.



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用

2.8 Socket编程



几种典型的应用编程接口

- ❖ Berkeley UNIX 操作系统定义了一种 API，称为套接字接口(socket interface)，简称套接字 (socket)。
- ❖ 微软公司在其操作系统中采用了套接字接口 API，形成了一个稍有不同的 API，并称之为 Windows Socket Interface，WINSOCK。
- ❖ AT&T 为其 UNIX 系统 V 定义了一种 API，简写为 TLI (Transport Layer Interface)。



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



Socket API

- ❖ 最初设计
 - 面向BSD UNIX-Berkley
 - 面向TCP/IP协议栈接口
- ❖ 目前
 - 事实上的工业标准
 - 绝大多数操作系统都支持
- ❖ Internet网络应用最典型的API接口
- ❖ 通信模型
 - 客户/服务器（C/S）
- ❖ 应用进程间通信的抽象机制

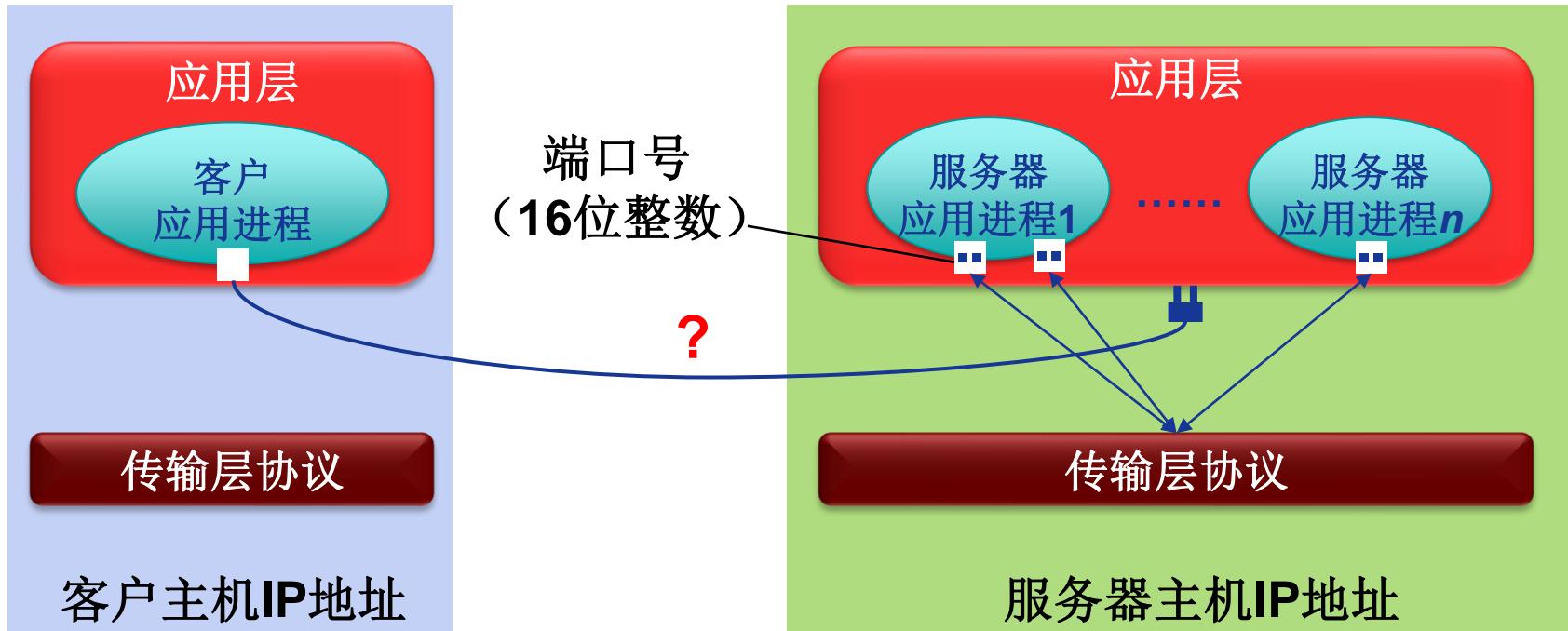




- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



Socket API



- ❖ 标识通信端点（对外）：
 - IP地址+端口号
- ❖ 操作系统/进程如何管理套接字（对内）？
 - 套接字描述符（socket descriptor）
 - 小整数

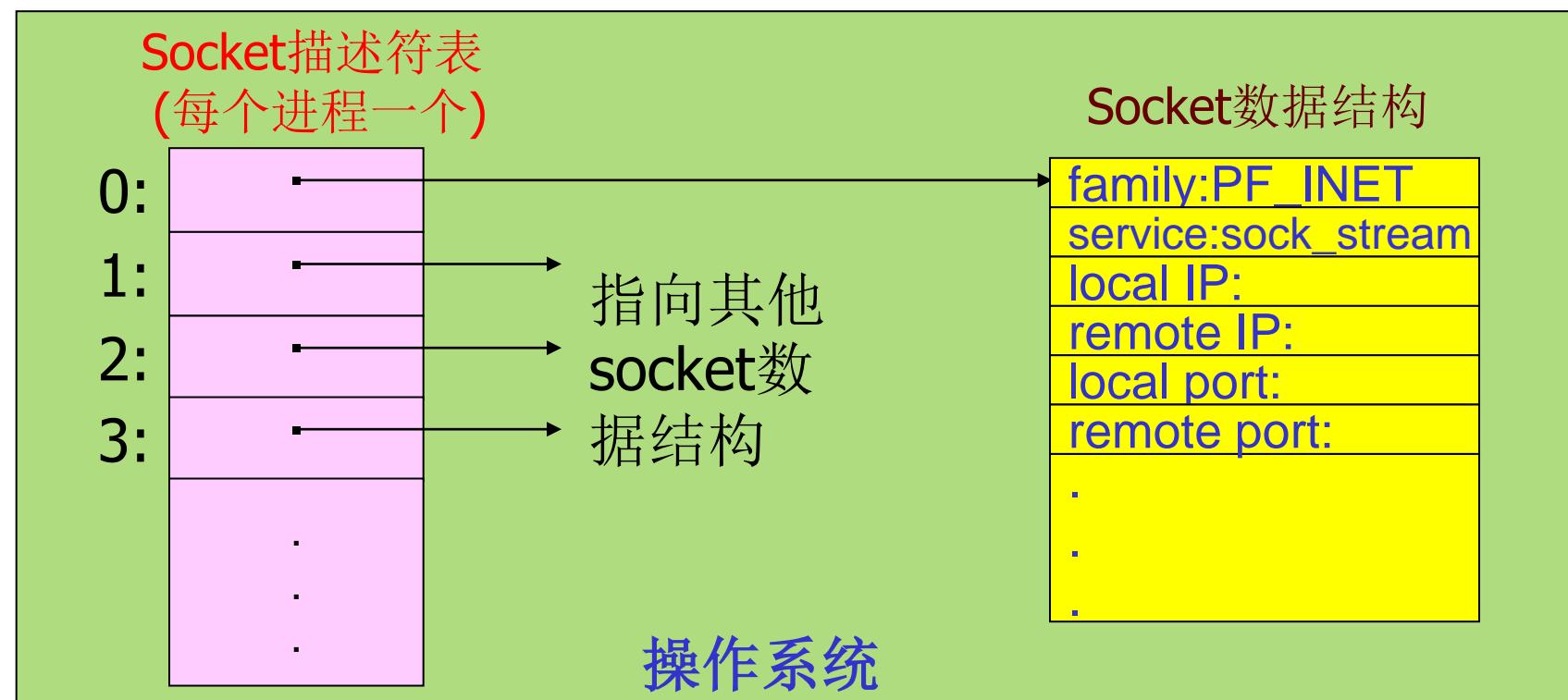


- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



Socket抽象

- ❖ 类似于文件的抽象
- ❖ 当应用进程创建套接字时，操作系统分配一个数据结构存储该套接字相关信息
- ❖ 返回套接字描述符





地址结构

- ❖ 已定义结构 ***sockaddr_in***:

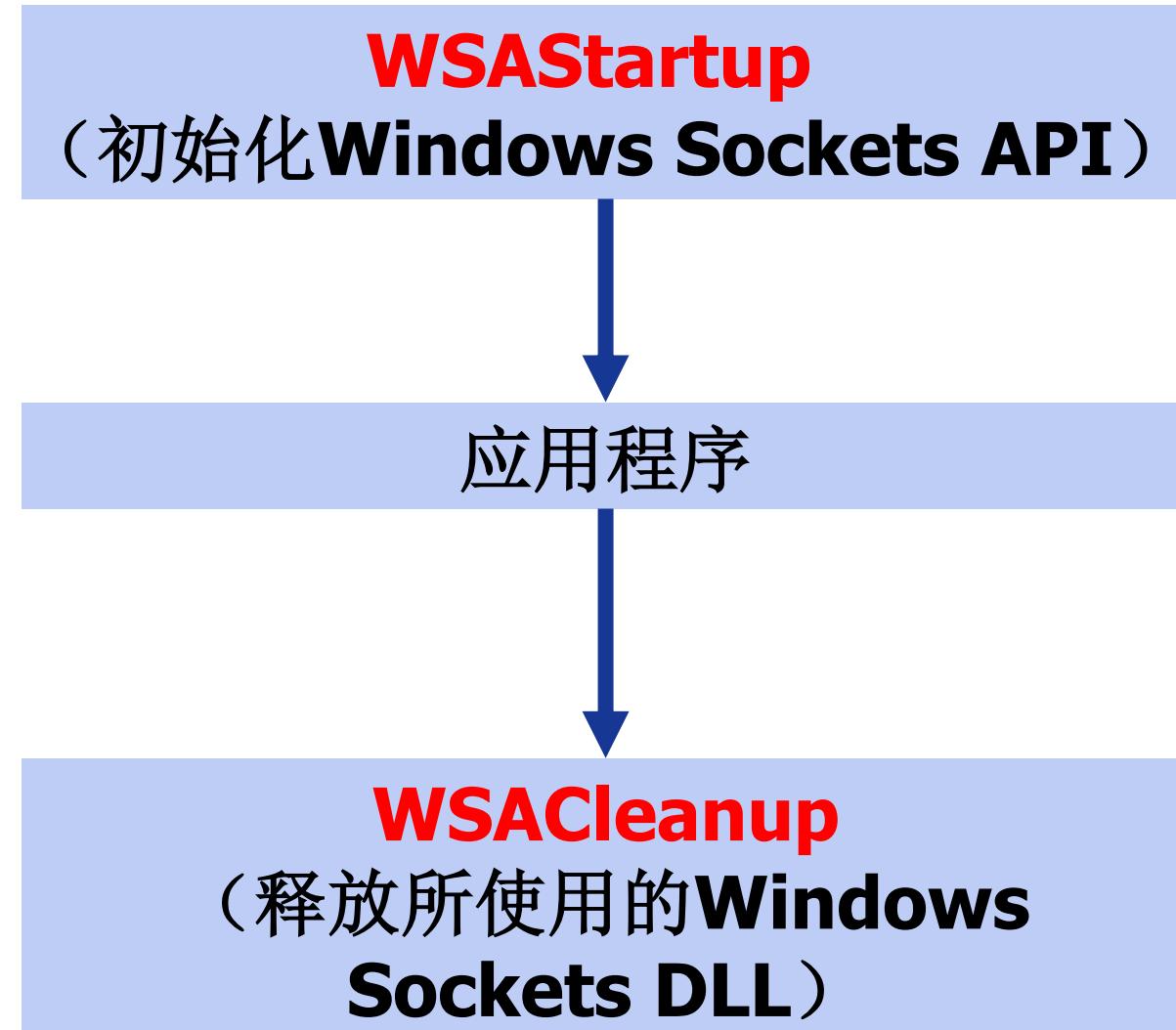
```
struct sockaddr_in
{
    u_char sin_len;          /*地址长度 */
    u_char sin_family;       /*地址族(TCP/IP: AF_INET) */
    u_short sin_port;        /*端口号 */
    struct in_addr sin_addr; /*IP地址 */
    char sin_zero[8];         /*未用(置0) */
}
```

- ❖ 使用 **TCP/IP** 协议簇的网络应用程序声明端点地址变量时，使用结构 ***sockaddr_in***



Socket API函数 (WinSock)

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用

2.8 Socket编程



Socket API函数

WSAStartup

```
int WSAStartup(WORD wVersionRequested, LPWSADATA lpWSAData);
```

- ❖ 使用Socket的应用程序在使用Socket之前必须首先调用 **WSAStartup** 函数
- ❖ 两个参数：
 - 第一个参数指明程序请求使用的WinSock版本，其中高位字节指明副版本、低位字节指明主版本。
 - 十六进制整数，例如0x102表示2.1版
 - 第二个参数返回实际的WinSock的版本信息
 - 指向 WSADATA 结构的指针
- ❖ 例：使用2.1版本的WinSock的程序代码段

```
wVersionRequested = MAKEWORD( 2, 1 );
err = WSAStartup( wVersionRequested, &wsaData );
```



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



Socket API函数

WSACleanup

```
int WSACleanup (void);
```

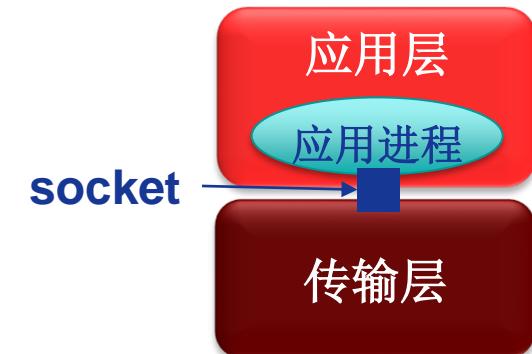
- ❖ 应用程序在完成对请求的**Socket**库的使用，**最后要调用**
WSACleanup函数
- ❖ 解除与**Socket**库的绑定
- ❖ 释放**Socket**库所占用的系统资源



Socket API函数

socket

```
sd = socket(protofamily,type,proto);
```



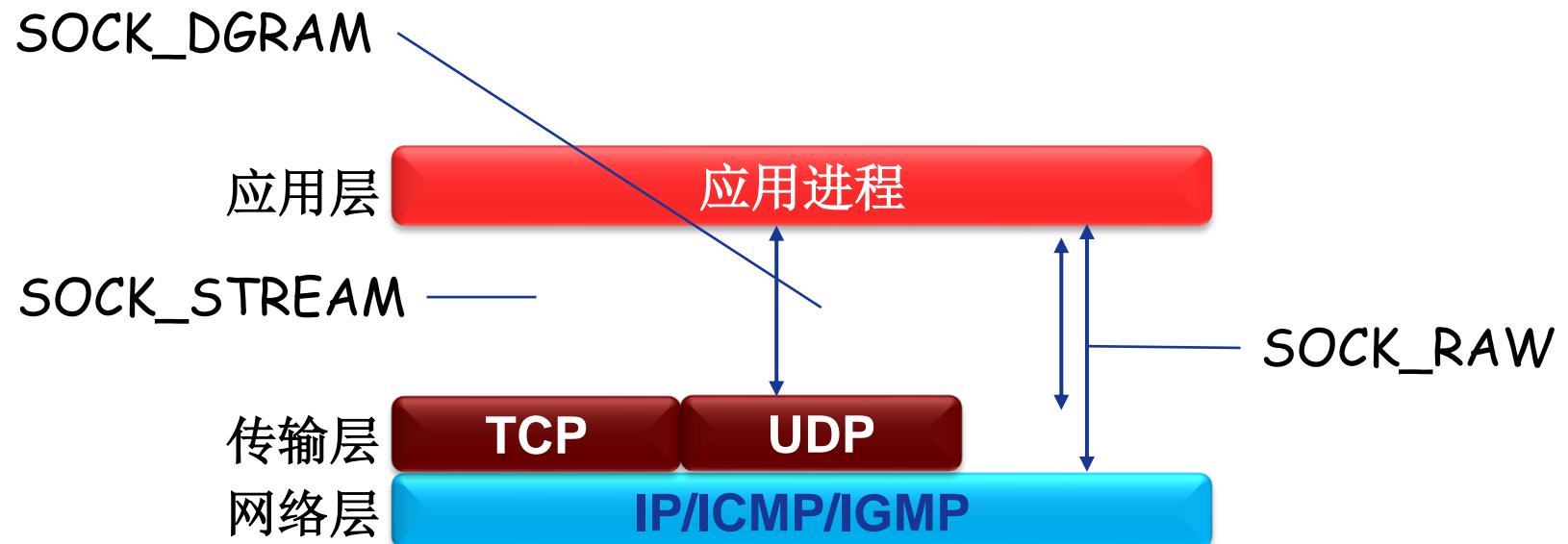
- ❖ 创建套接字
- ❖ 操作系统返回套接字描述符（sd）
- ❖ 第一个参数(协议族): **protofamily** = **PF_INET** (TCP/IP)
- ❖ 第二个参数(套接字类型):
 - **type** = **SOCK_STREAM**,**SOCK_DGRAM** or **SOCK_RAW** (TCP/IP)
- ❖ 第三个参数(协议号):0为默认
- ❖ 例：创建一个流套接字的代码段

```
struct protoent *p;  
p=getprotobyname("tcp");  
SOCKET sd=socket(PF_INET,SOCK_STREAM,p->p_proto);
```



Socket面向TCP/IP的服务类型

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



- ❖ **TCP:** 可靠、面向连接、字节流传输、点对点
- ❖ **UDP:** 不可靠、无连接、数据报传输





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



Socket API函数

Closesocket

```
int closesocket(SOCKET sd);
```

- ❖ 关闭一个描述符为sd的套接字
- ❖ 如果多个进程共享一个套接字，调用*closesocket*将套接字引用计数减1，减至0才关闭
- ❖ 一个进程中的多线程对一个套接字的使用无计数
 - 如果进程中的一个线程调用*closesocket*将一个套接字关闭，该进程中的其他线程也将不能访问该套接字
- ❖ 返回值：
 - 0：成功
 - SOCKET_ERROR：失败



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



Socket API函数

bind

```
int bind(sd, localaddr, addrlen);
```

- ❖ 绑定套接字的本地端点地址
 - IP地址+端口号
- ❖ 参数:
 - 套接字描述符 (sd)
 - 端点地址 (localaddr)
 - 结构 *sockaddr_in*
- ❖ 客户程序一般不必调用bind函数
- ❖ 服务器端?
 - 熟知端口号
 - IP地址?





- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



Socket API函数

❖ 考虑如下情形：



- ❖ 服务器应该绑定哪个地址？
- ❖ 问题？
- ❖ 解决方案
 - 地址通配符: **INADDR_ANY**



2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

2.5 Email应用

2.6 Web应用

2.7 P2P应用

2.8 Socket编程

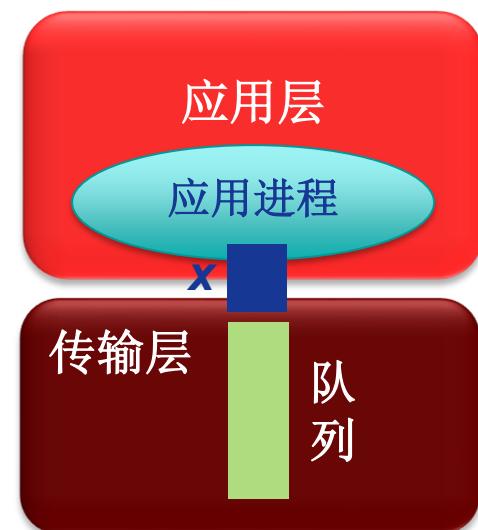


Socket API函数

listen

```
int listen(sd, queuesize);
```

- ❖ 置服务器端的流套接字处于监听状态
 - 仅服务器端调用
 - 仅用于面向连接的流套接字
- ❖ 设置连接请求队列大小（queuesize）
- ❖ 返回值：
 - 0: 成功
 - SOCKET_ERROR: 失败





2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

2.5 Email应用

2.6 Web应用

2.7 P2P应用

2.8 Socket编程

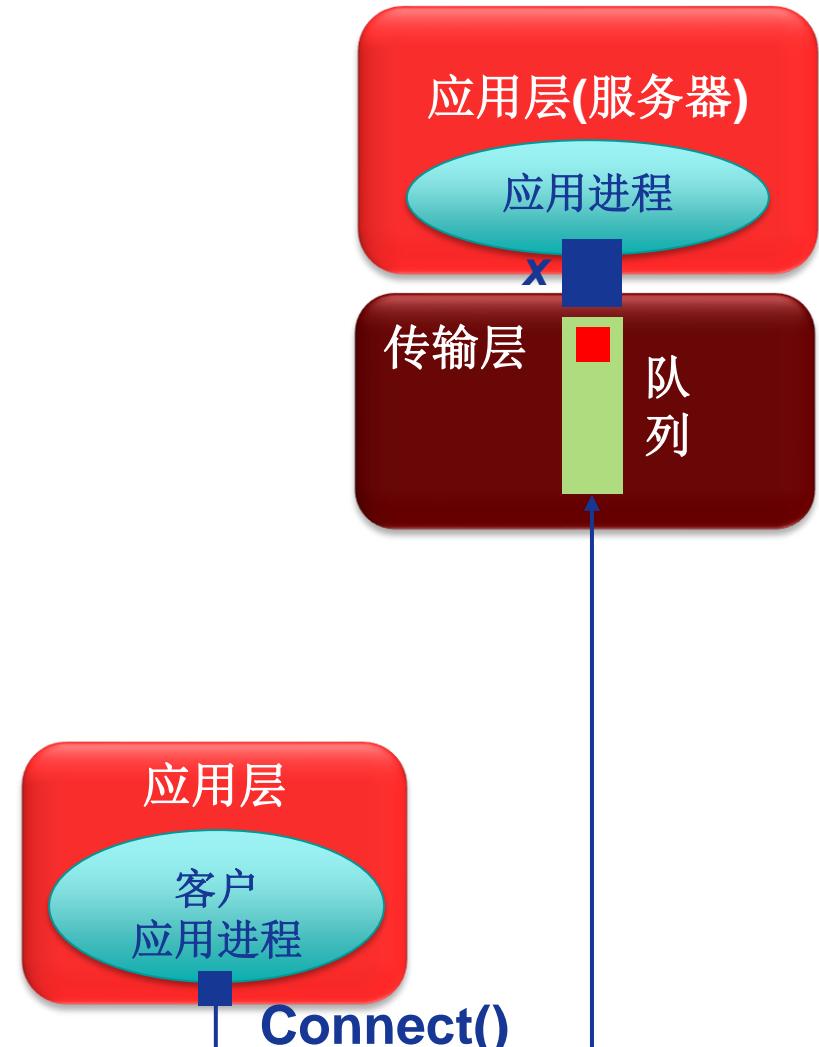


Socket API函数

connect

```
connect(sd, saddr, saddrlen);
```

- ❖ 客户程序调用**connect**函数来使客户套接字（**sd**）与特定计算机的特定端口（**saddr**）的套接字（服务）进行连接
- ❖ 仅用于客户端
- ❖ 可用于**TCP**客户端也可以用于**UDP**客户端
 - **TCP**客户端：建立**TCP**连接
 - **UDP**客户端：指定服务器端点地址





2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

2.5 Email应用

2.6 Web应用

2.7 P2P应用

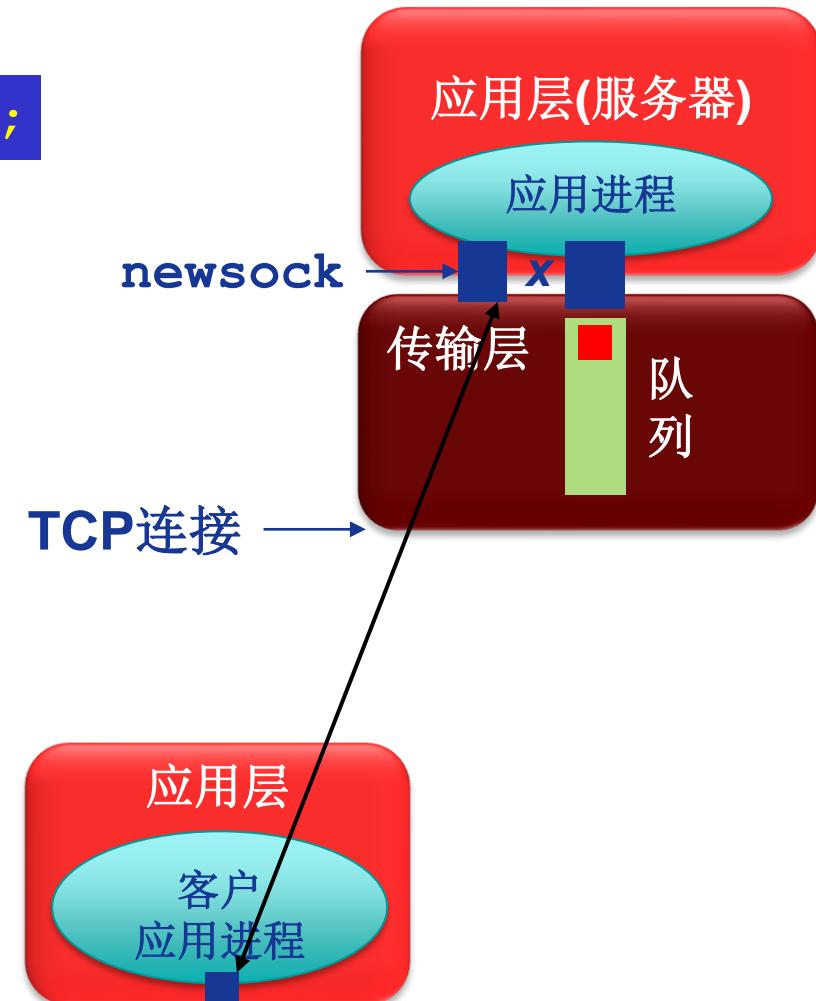
2.8 Socket编程

Socket API函数

accept

```
newsock = accept(sd,caddr,caddrlen);
```

- ❖ 服务程序调用**accept**函数从处于监听状态的流套接字sd的客户连接请求队列中取出排在最前的一个客户请求，并且创建一个新的套接字来与客户套接字创建连接通道
 - 仅用于**TCP**套接字
 - 仅用于服务器
- ❖ 利用新创建的套接字(**newsock**)与客户通信





2.1 网络应用体系结构

2.2 网络应用通信原理

2.3 域名解析系统(DNS)

2.4 FTP应用

2.5 Email应用

2.6 Web应用

2.7 P2P应用

2.8 Socket编程



Socket API函数

send, sendto

```
send(sd,*buf,len,flags);
```

```
sendto(sd,*buf,len,flags,destaddr,addrulen);
```

- ❖ **send**函数TCP套接字（客户与服务器）或调用了**connect**函数的**UDP**客户端套接字
- ❖ **sendto**函数用于**UDP**服务器端套接字与未调用**connect**函数的**UDP**客户端套接字



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用

2.8 Socket编程



Socket API函数

recv, recvfrom

```
recv(sd, *buffer, len, flags);
```

```
recvfrom(sd, *buf, len, flags, senderaddr, saddrhlen);
```

- ❖ **recv**函数从TCP连接的另一端接收数据，或者从调用了**connect**函数的UDP客户端套接字接收服务器发来的数据
- ❖ **recvfrom**函数用于从UDP服务器端套接字与未调用**connect**函数的UDP客户端套接字接收对端数据



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用

2.8 Socket编程



Socket API函数

setsockopt, getsockopt

```
int setsockopt(int sd, int level, int optname, *optval, int optlen);
```

```
int getsockopt(int sd, int level, int optname, *optval, socklen_t *optlen);
```

- ❖ **setsockopt()**函数用来设置套接字sd的选项参数
- ❖ **getsockopt()**函数用于获取任意类型、任意状态套接口的选项当前值，并把结果存入optval



Socket API函数小结

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用

2.8 Socket编程



- ❖ **WSAStartup**: 初始化socket库(仅对**WinSock**)
- ❖ **WSACleanup**: 清楚/终止socket库的使用 (**仅对WinSock**)
- ❖ **socket**: 创建套接字
- ❖ **connect**: “连接”远端服务器 (**仅用于客户端**)
- ❖ **closesocket**: 释放/关闭套接字
- ❖ **bind**: 绑定套接字的本地IP地址和端口号 (**通常客户端不需要**)
- ❖ **listen**: 置服务器端TCP套接字为监听模式，并设置队列大小 (**仅用于服务器端TCP套接字**)
- ❖ **accept**: 接受/提取一个连接请求，创建新套接字，通过新套接 (**仅用于服务器端的TCP套接字**)
- ❖ **recv**: 接收数据 (**用于TCP套接字或连接模式的客户端UDP套接字**)



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



Socket API函数小结

- ❖ **recvfrom**: 接收数据报（用于非连接模式的**UDP套接字**）
- ❖ **send**: 发送数据（用于**TCP套接字**或连接模式的**客户端UDP套接字**）
- ❖ **sendto**:发送数据报（用于非连接模式的**UDP套接字**）
- ❖ **setsockopt**: 设置套接字选项参数
- ❖ **getsockopt**: 获取套接字选项参数



- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**



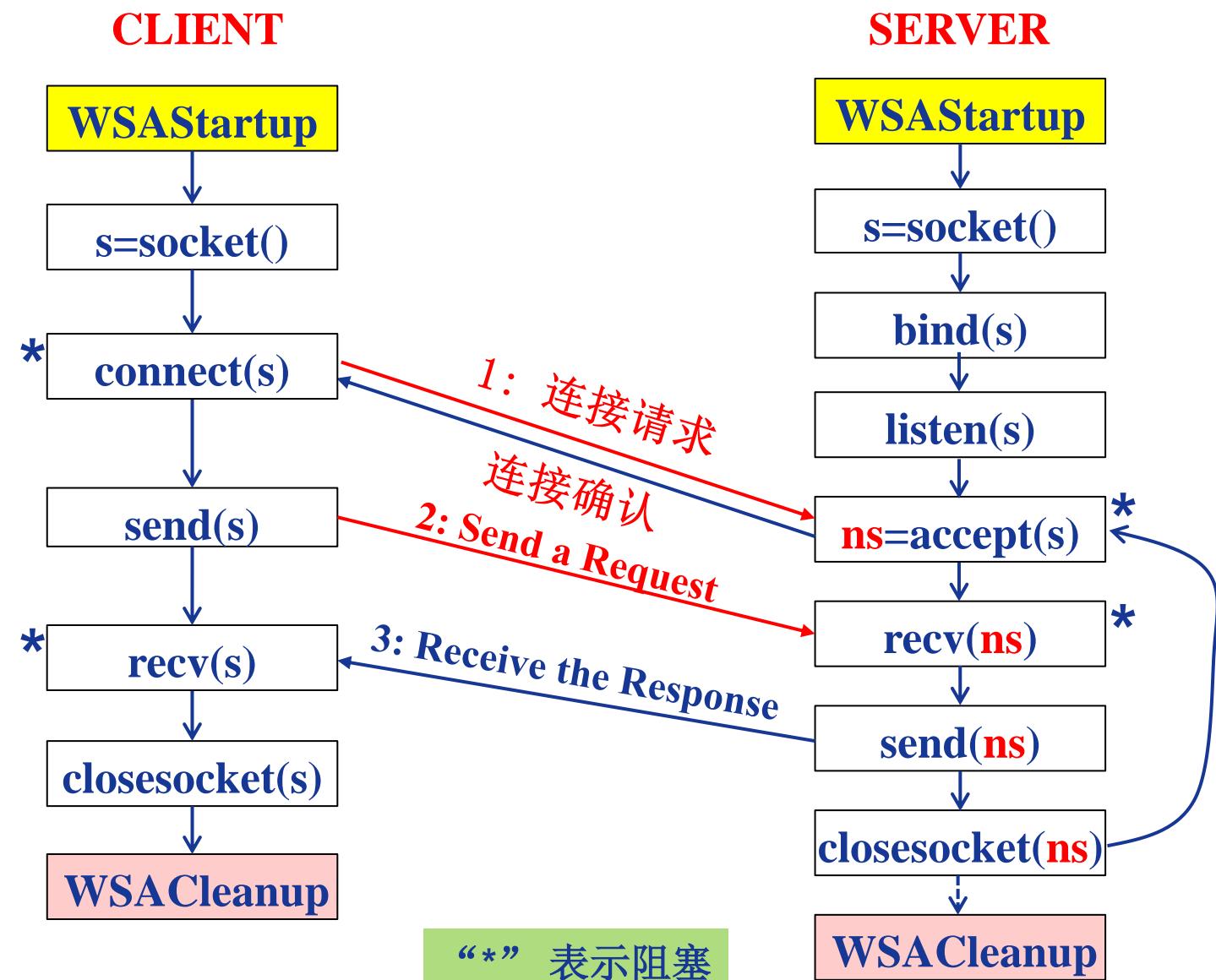
关于网络字节顺序

- ❖ TCP/IP 定义了标准的用于协议头中的二进制整数表示：
网络字节顺序 (network byte order)
- ❖ 某些 **Socket API** 函数的参数需要存储为网络字节顺序 (如 IP 地址、端口号等)
- ❖ 可以实现本地字节顺序与网络字节顺序间转换的函数
 - **htons**: 本地字节顺序 → 网络字节顺序(16bits)
 - **ntohs**: 网络字节顺序 → 本地字节顺序(16bits)
 - **htonl**: 本地字节顺序 → 网络字节顺序(32bits)
 - **ntohl**: 网络字节顺序 → 本地字节顺序(32bits)



网络应用的Socket API(TCP)调用基本流程

- 2.1 网络应用体系结构
- 2.2 网络应用通信原理
- 2.3 域名解析系统(DNS)
- 2.4 FTP应用
- 2.5 Email应用
- 2.6 Web应用
- 2.7 P2P应用
- 2.8 Socket编程**





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场

謝 謝！