

第九章

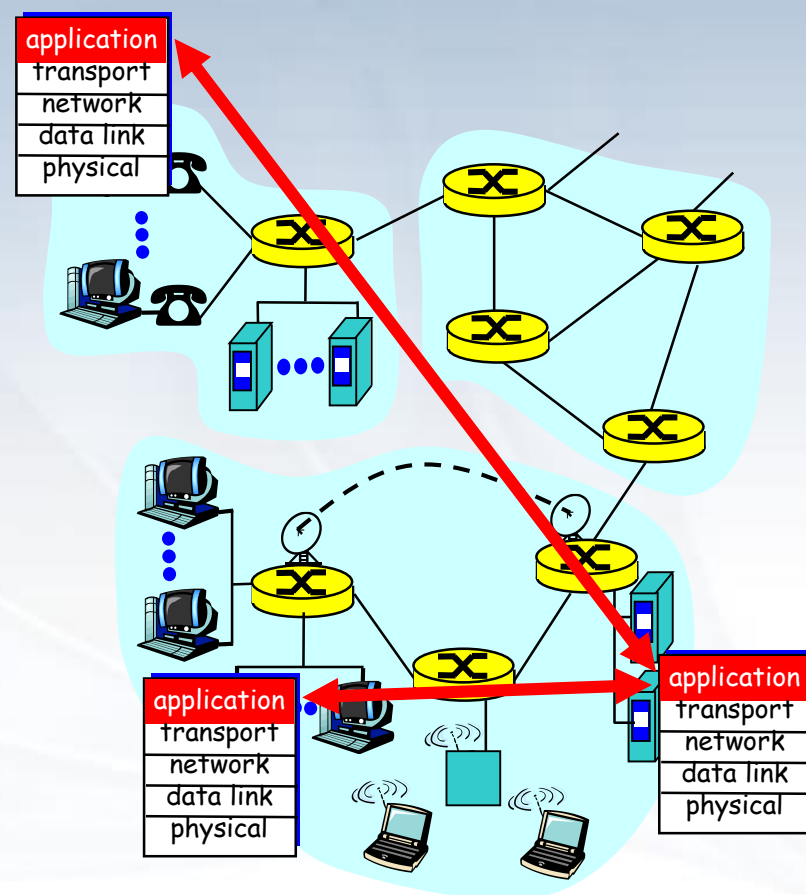
计算机网络应用

主要内容

- 9.1 应用层概述
- 9.2 客户/服务器模型
- 9.3 域名服务
- 9.4 简单网络管理协议SNMP
- 9.5 网络安全
- 9.6 电子邮件
- 9.7 文件传输协议FTP
- 9.8 WWW及其应用
- 9.9 P2P及其应用

9.1 应用层概述（1）

- 应用程序：互相通信的分布的进程
 - 在网络主机上的用户空间运行
 - 互相交换消息
 - 比如email、ftp和web
- 应用层协议
 - 应用程序的一部分
 - 定义应用程序直接交换的信息以及相应的动作
 - 利用底层协议提供的服务



9.1 应用层概述（2）

■ 应用层术语

- 一个**进程**是运行于主机上的一个程序。 .
- 在同一主机上的进程利用操作系统提供的IPC（interprocess communication）进行通信。
- 在不同主机上运行的进程利用应用层协议进行通信
- **用户代理（user agent）**是指用户和网络应用程序间的接口。比如**web**浏览器，流媒体播放器等

9.1 应用层概述（3）

- 应用程序编程接口API（application programming interface）
 - 定义应用程序和传输层之间的接口
 - socket: Internet API
 - 两个进程通过向socket写数据和读数据来通信
- Q: 进程如何指明它要与之通信的另一个进程
 - IP地址指明该进程所在的主机
 - 端口号指明该主机应该把收到的数据交给哪个当地进程

9.1 应用层概述（4）

- 应用程序需要什么样的传输服务
 - 数据丢失
 - 有的应用程序(比如音频)可以容忍一定程度的数据丢失
 - 有的应用程序(如文件传输)要求**100%**的可靠传输
 - 带宽
 - 有的程序(如多媒体)需要一定的带宽才能工作
 - 有的程序则使用它所能得到的全部带宽
 - 延迟
 - 有的程序要求低延迟，比如**IP**电话和交互游戏。

应用程序所需的传输服务

应用	数据丢失	带宽	时间敏感
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: 5Kb-1Mb video: 10Kb-5Mb	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
financial apps	no loss	elastic	yes and no

Internet传输协议提供的服务

TCP服务：

- 面向连接：用户端和服务
器需要建立连接
- 接收和发送进程间的可靠
传输
- 流量控制：发送方不会淹
没接收方
- 拥塞控制：网络负载过高
时限制发送方发送
- 不提供：延迟保证，最小
带宽保证

UDP服务：

- 接收和发送进程间的不可
靠传输
- 不提供：连接建立，可
靠性、流量控制、拥塞
控制和带宽保证

Internet应用和使用的相应协议

应用 应用层协议 底层传输层协议

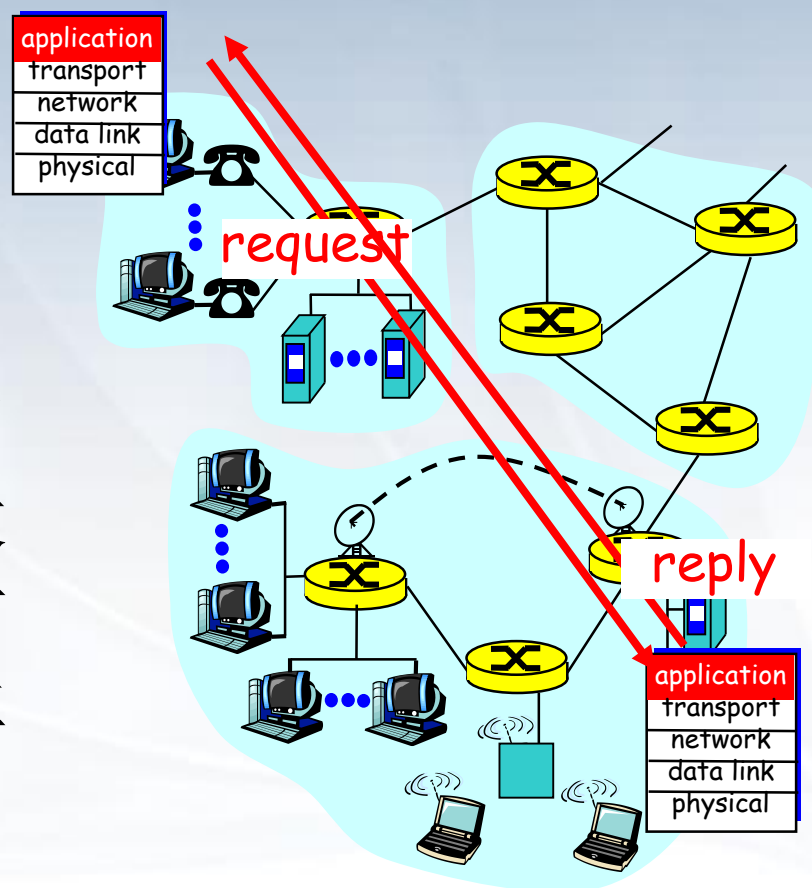
	e-mail	smtp [RFC 821]	TCP
remote	terminal access	telnet [RFC 854]	TCP
	Web	http [RFC 2068]	TCP
	file transfer	ftp [RFC 959]	TCP
streaming	multimedia	proprietary (e.g. RealNetworks)	TCP or UDP
	remote file server	NSF	TCP or UDP
Internet	telephony	proprietary (e.g., Vocaltec)	typically UDP

9.2 客户/服务器模型（1）

■ 基本概念

- 客户/服务器模型是所有网络应用的基础。
- 客户/服务器分别指参与一次通信的两个应用实体，客户方主动地发起通信请求，服务器方被动地等待通信的建立。

。



9.2 客户/服务器模型（2）

■ 客户软件

- 任何一个应用程序当需要进行远程访问时成为客户，这个应用程序也要完成一些本地的计算；
- 一般运行于用户的个人计算机上；
- 向服务器主动发起通信请求；
- 可以访问多个服务器，但一次只能访问一个；
- 不需要特殊的硬件和复杂的操作系统。

■ 服务器软件

- 是专用的提供某种服务的特权程序，可以同时处理多个远程客户；
- 一般在系统启动时被执行，并连续运行以处理多次会话；
- 被动的等待远程客户发起通信；
- 需要特殊的硬件和复杂的操作系统。

9.2 客户/服务器模型（3）

- 数据在客户和服务端之间是双向流动的，一般是客户发出请求，服务器给出响应。
- 服务器软件的并发性
 - 由于服务器软件要支持多个客户的同时访问，它必须具备并发性。服务器软件为每个新到的客户创建一个进程或线程来处理 and 这个客户的通信。服务器方传送层实体使用客户的源端口号和服务的端口号来确定正确的服务器软件进程（线程）
- 服务器软件的组成
 - 服务器软件一般分为两部分：一部分用于接受请求并创建新的进程或线程，另一部分用于处理实际的通信过程。

9.2 客户/服务器模型（4）

- 客户/服务器之间使用的传送层协议
 - 可以是基于连接的**TCP**协议，要求建立和释放连接，适用于可靠的交互过程；
 - 也可以是无连接的**UDP**协议，适用于可靠性要求不高的或实时的交互过程；
 - 同时使用**TCP**和**UDP**的服务，有两种服务器软件的实现或服务器软件同时和**TCP**、**UDP**协议交互，不对客户做限制。

9.2 客户/服务器模型（5）

- 客户和服务器的交互
 - 支持协议：在**INTERNET**中，客户和服务器的交互通过使用**TCP/IP**协议栈来完成。因此，客户和服务服务器所在的机器要求支持完全的协议栈。客户/服务器通过套接字访问传送层服务
 - 多种服务：一台计算机上可以运行多个服务器软件，但是要求计算机有强大的硬件资源(服务器级别的计算机)和多任务操作系统(**UNIX**和**WIN95/98/2000/NT**)。
 - 服务的标识：客户是通过服务的标识来访问某种服务的，比如在**INTERNET**中，服务是用端口号来标识，**UNIX**在**/etc/services**文件中定义。服务器软件启动时将其标识通知传送层实体。

9.3 域名服务（1）

■ 产生原因

- 32比特的IP地址难于记忆，应该使用符号地址，但是网络本身是使用IP地址的，因此需要一个完成二者之间相互转换的机制。比如用netlab.cs.tsinghua.edu.cn表示166.111.69.241。
- 当网络规模比较小时，例如ARPANET，每台主机只需查找一个文件（UNIX的host），该文件中列出了主机与IP地址的对应关系。
- 当网络规模很大时，上述方法就不适用了，因此产生了域名系统DNS（Domain Name System）

9.3 域名服务（2）

■ DNS概述

- 域名系统是一个典型的客户/服务器交互系统；
- 域名系统是一个多层次的、基于域的命名系统，并使用分布式数据库实现这种命名机制；
- 当应用程序需要进行域名解析时(从符号名到IP地址)，它成为域名系统的一个客户。它向本地域名服务器发出请求(调用**resolver**)，请求以**UDP**包格式发出，域名服务器找到对应的IP地址后，给出响应。当本地域名服务器无法完成域名解析，它临时变成其上级域名服务器的客户，递归解析，直到该域名解析完成。
- RFC 1034, 1035

9.3 域名服务（3）

■ 域名的结构

- **INTERNET**的顶级域名分为组织结构和地理结构两种。每个域对它下面的子域和机器进行管理。
- **DNS**中，每台计算机的名字是由“.”所分开的字符、数字串所组成的。例如**www.tsinghua.edu.cn**
- 域名是大小写无关的，“**edu**”和“**EDU**”相同。域名最长**255**个字符，每部分最长**63**个字符。

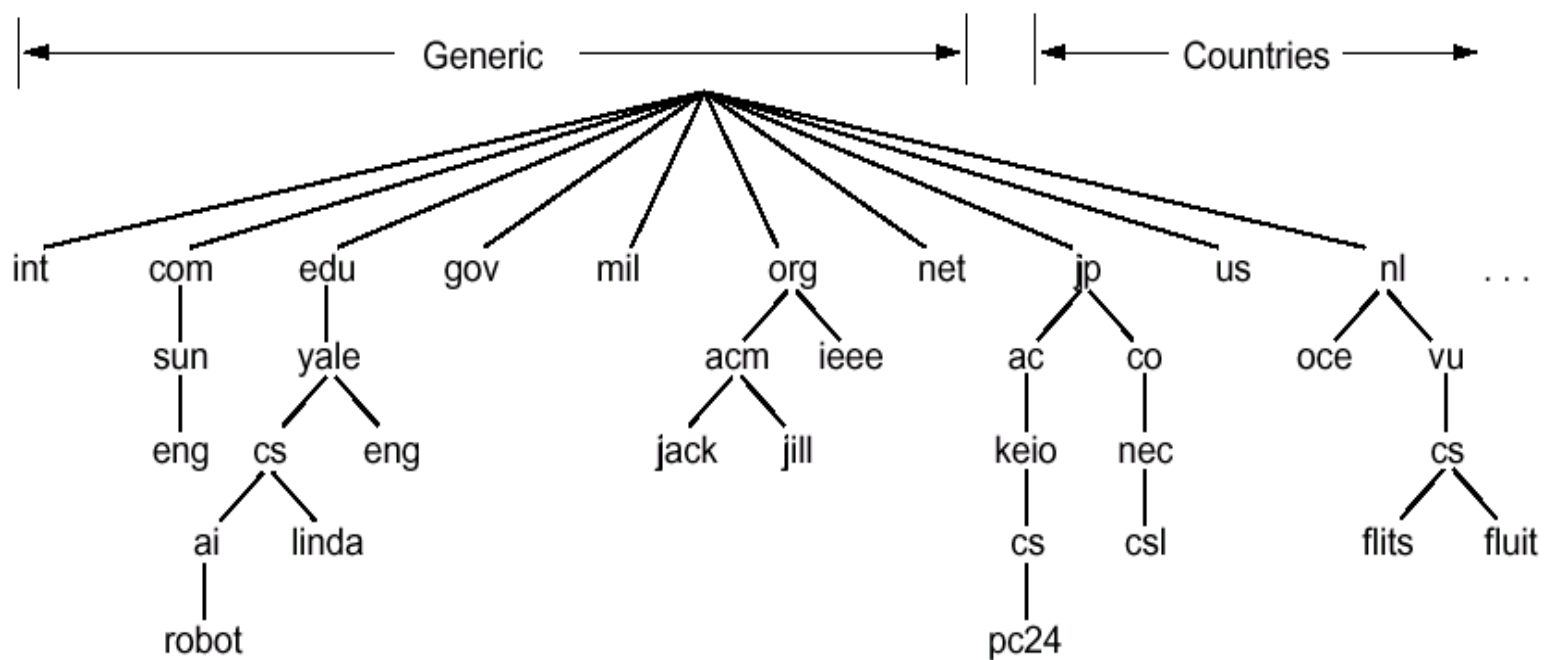


Fig. 7-25. A portion of the Internet domain name space.

9.3 域名服务（4）

■ 资源记录

- 在**DNS**的数据库中用资源记录来表示主机和子域的信息，当应用程序进行域名解析时，得到的便是域名所对应的资源记录。
- 资源记录是一个五元式

Domain_name Time_to_live Type Class Value

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept email
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

Fig. 7-26. The principal DNS resource record types.

; Authoritative data for cs.vu.nl

cs.vu.nl. 86400 IN SOA star boss (952771,7200,7200,2419200,86400)

cs.vu.nl. 86400 IN TXT "Faculteit Wiskunde en Informatica."

cs.vu.nl. 86400 IN TXT "Vrije Universiteit Amsterdam."

cs.vu.nl. 86400 IN MX 1 zephyr.cs.vu.nl.

cs.vu.nl. 86400 IN MX 2 top.cs.vu.nl.

flits.cs.vu.nl.86400 IN HINFO Sun Unix

flits.cs.vu.nl.86400 IN A 130.37.16.112

flits.cs.vu.nl.86400 IN A 192.31.231.165

flits.cs.vu.nl.86400 IN MX 1 flits.cs.vu.nl.

flits.cs.vu.nl.86400 IN MX 2 zephyr.cs.vu.nl.

flits.cs.vu.nl.86400 IN MX 3 top.cs.vu.nl.

www.cs.vu.nl.86400 IN CNAMEstar.cs.vu.nl

ftp.cs.vu.nl. 86400 IN CNAMEzephyr.cs.vu.nl

rowboat IN A 130.37.56.201

IN MX 1 rowboat

IN MX 2 zephyr

IN HINFO Sun Unix

little-sister IN A 130.37.62.23

IN HINFO Mac MacOS

laserjet IN A 192.31.231.216

IN HINFO "HP Laserjet III Si" Proprietary

Fig. 7-27. A portion of a possible DNS database for *cs.vu.nl*

9.3 域名服务（5）

■ 域名服务器

- 区域划分

- DNS将域名空间划分为许多无重叠的区域(zone)，每个区域覆盖了域名空间的一部分并设有域名服务器对这个区域的域名进行管理。
- 每个区域有一个主域名服务器和若干个备份域名服务器，区域的边界划分是人工设置的，比如：`edu.cn` `tsinghua.edu.cn`
`cs.tsinghua.edu.cn`是三个不同的区域，分别有各自的域名服务器。

9.3 域名服务（6）

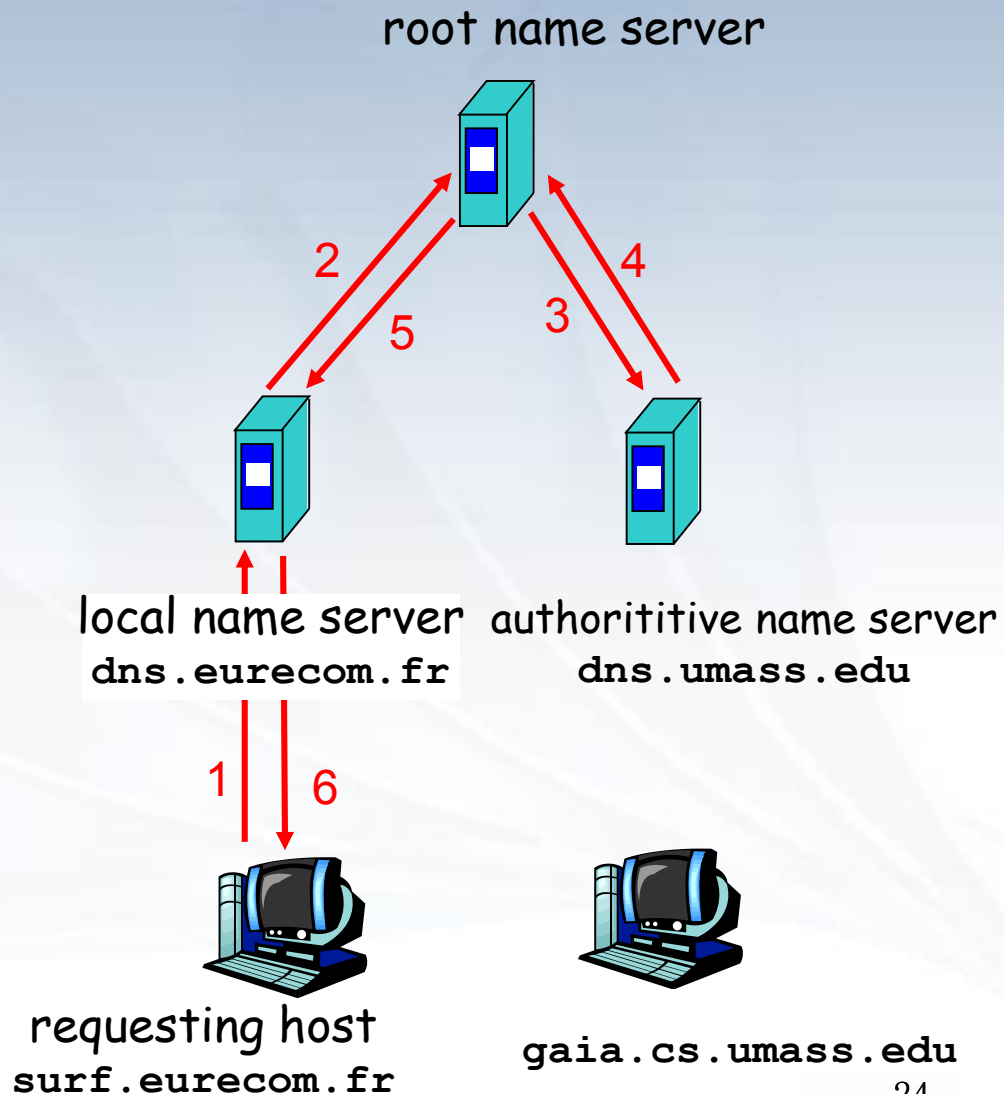
- 域名解析

- 一个区域内机器上的应用程序进行域名解析时，首先向该区域的域名服务器发出解析请求，若查找到，则返回域名对应的资源记录。
- 若找不到，该域名服务器向所查找域名的顶级域的域名服务器发出解析请求，
- 顶级域的域名服务器通过向下的层次查询得到对应的资源记录，返回给该域名服务器，
- 最后资源记录被返回给发起域名解析的机器，并在该区域的域名服务器中做缓存，超时后删除。

Simple DNS example

主机 **surf.eurecom.fr** 需要 **gaia.cs.umass.edu** 的 IP 地址

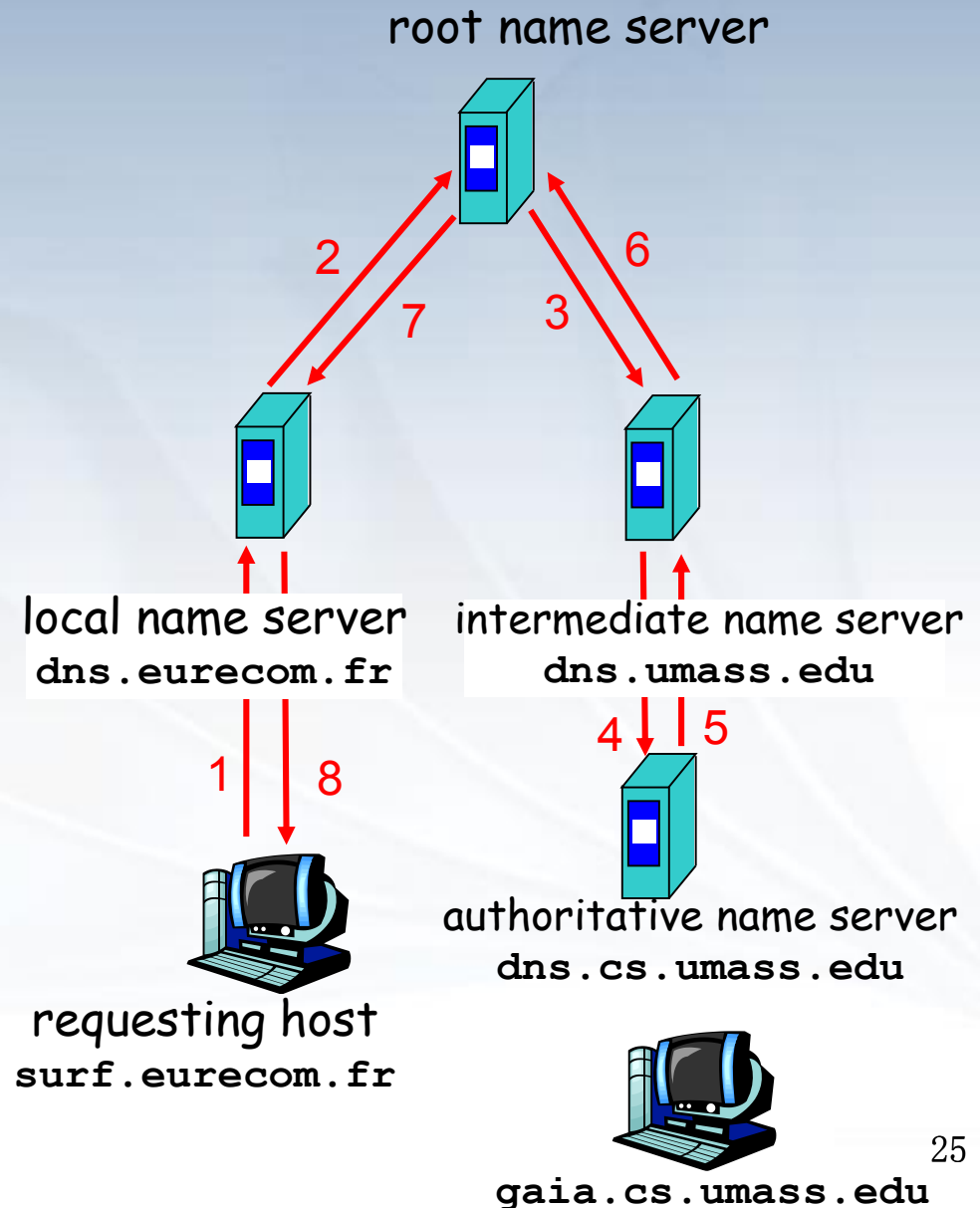
1. 与本地DNS服务器 **dns.eurecom.fr** 联系
2. 如有必要 **dns.eurecom.fr** 与根域名服务器联系
3. 如有必要，根域名服务器联系 authoritative 域名服务器 **dns.umass.edu**。



DNS example

根域名服务器:

- 可能不知道 authoritative 域名服务器
- 可能知道中间的域名服务器，中间域名服务器知道如何与 authoritative 域名服务器联系



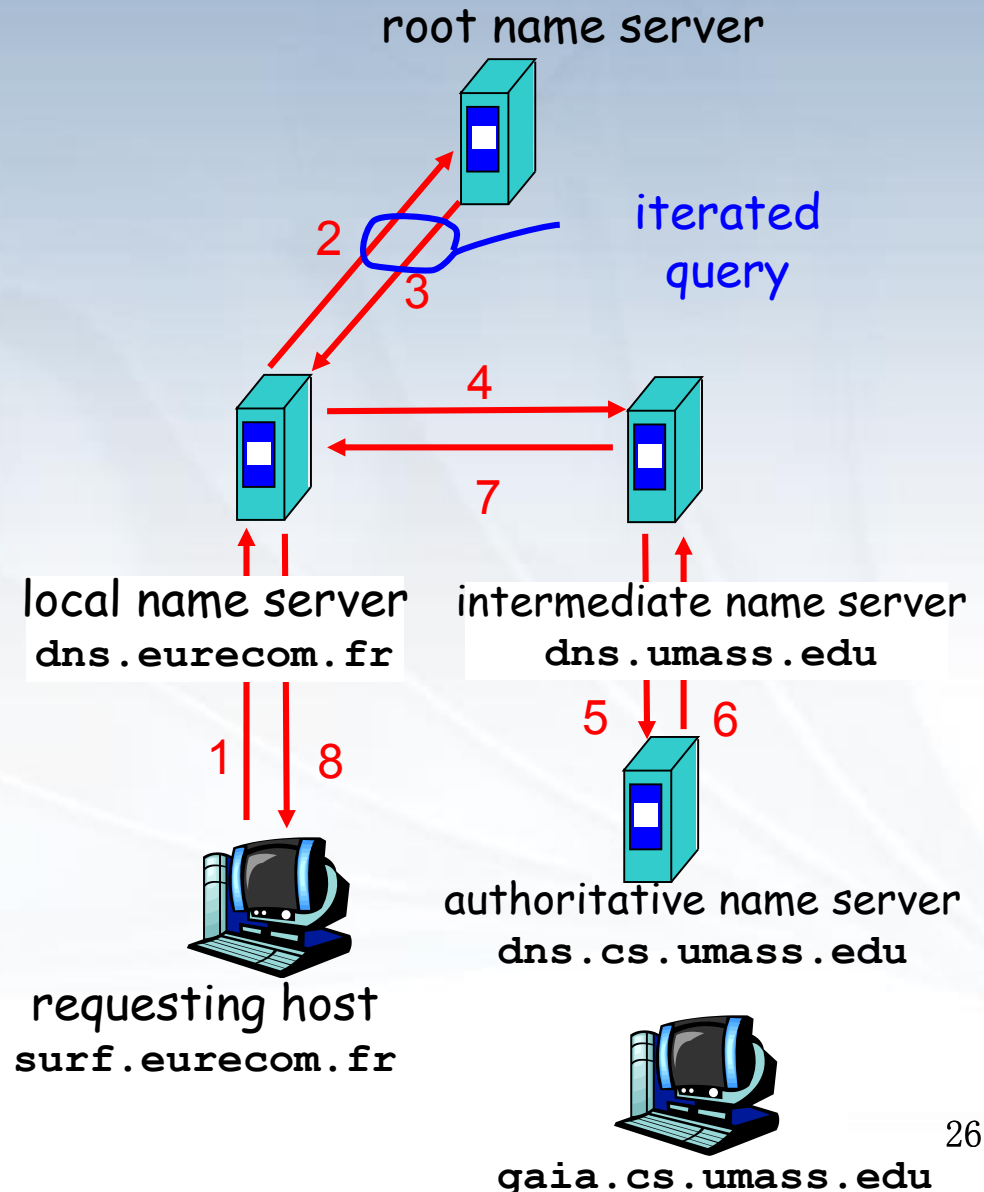
DNS: iterated queries

recursive query:

- puts burden of name resolution on contacted name server
- heavy load?

iterated query:

- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"



9.4 简单网络管理协议SNMP（1）

- SNMP（Simple Network Management Protocol）的产生
 - 早期网络，如ARPANET，规模很小，可以通过执行“PING”命令来发现网络故障；
 - 网络规模变大，需要一个好的工具来管理网络。
1990年发布RFC 1157，定义了SNMP v1；
 - SNMP v2，RFC 1441 ~ 1452。
- 网络管理的五个基本管理功能：性能管理、故障管理、配置管理、记帐管理和安全管理。
- SNMP是基于UDP的

9.4 简单网络管理协议SNMP（2）

■ SNMP模型

- 被管理节点

- 运行SNMP代理程序（SNMP agent），维护一个本地数据库，描述节点的状态和历史，并影响节点的运行。

- 管理工作站

- 运行专门的网络管理软件（manager），使用管理协议与被管理节点上的SNMP代理通信，维护管理信息库。

9.4 简单网络管理协议SNMP（3）

- 管理信息

- 每个站点使用一个或多个变量描述自己的状态，这些变量称为“对象（objects）”，所有的对象组成管理信息库MIB（Management Information Base）。

- 管理协议（SNMP）

- 管理协议用于管理工作站查询和修改被管理节点的状态，被管理节点可以使用管理协议向管理站点产生“陷阱（trap）”报告。

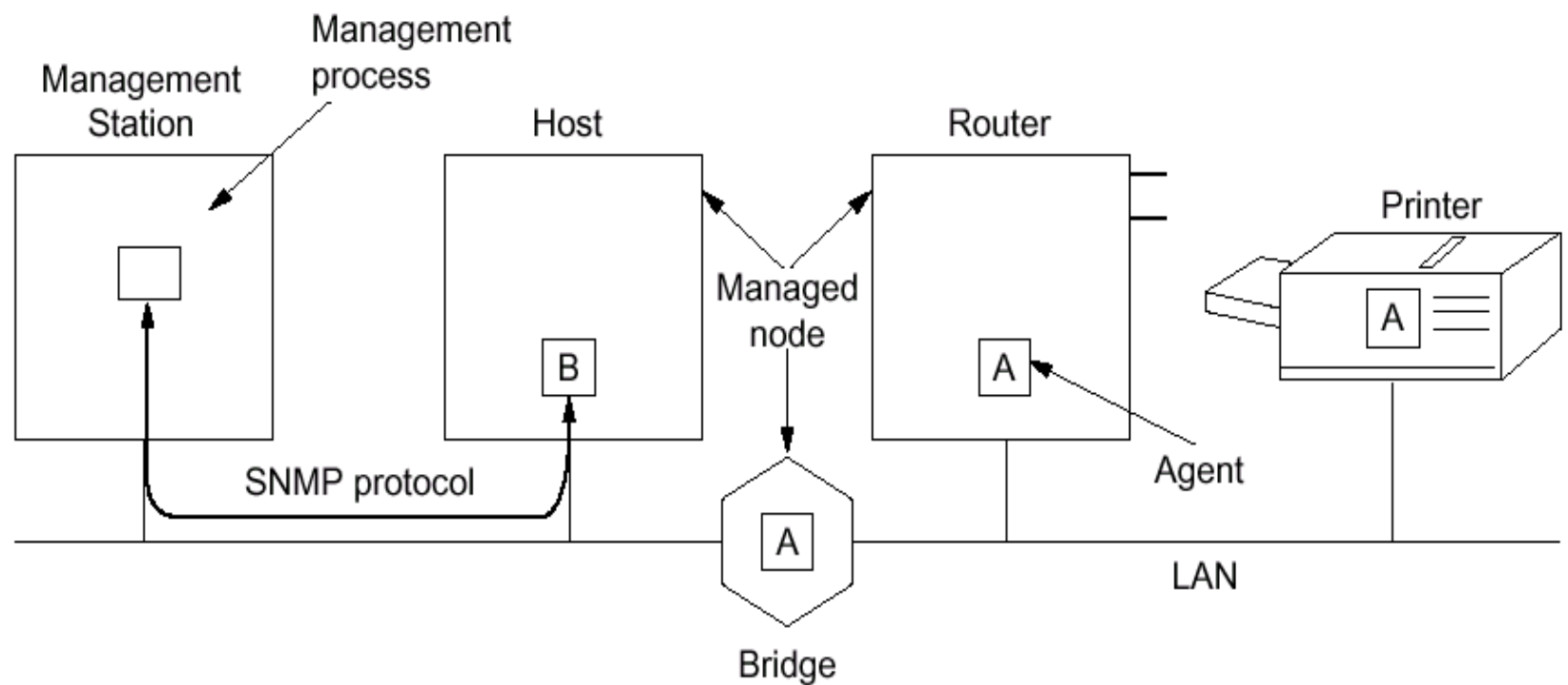


Fig. 7-30. Components of the SNMP management model.

9.4 简单网络管理协议SNMP（4）

- 抽象语法表示法1（ASN.1）
 - 定义：抽象语法表示法1是一种标准的对象定义语言，它分为数据描述定义（ISO 8824）和传输语法定义（ISO 8825）两部分。ASN.1可以作为异种计算机设备之间“对象”的描述和传输机制。
 - ASN.1的基本数据类型

Primitive type	Meaning	Code
INTEGER	Arbitrary length integer	2
BIT STRING	A string of 0 or more bits	3
OCTET STRING	A string of 0 or more unsigned bytes	4
NULL	A place holder	5
OBJECT IDENTIFIER	An officially defined data type	6

Fig. 7-31. The ASN.1 primitive data types permitted in SNMP.

9.4 简单网络管理协议SNMP（5）

- 对象命名树

- 对象命名树使用编码，唯一的确定每个标准中的对象。基于对象命名树，任何标准中的任意对象都可以用如下的对象表示符表示。
- {iso(1) identified-organizations(3) dod(6) internet(1) mgmt(2) mib-2(1) ..tcp(6)..}
- 或者是 {1 3 6 1 2 1 6}

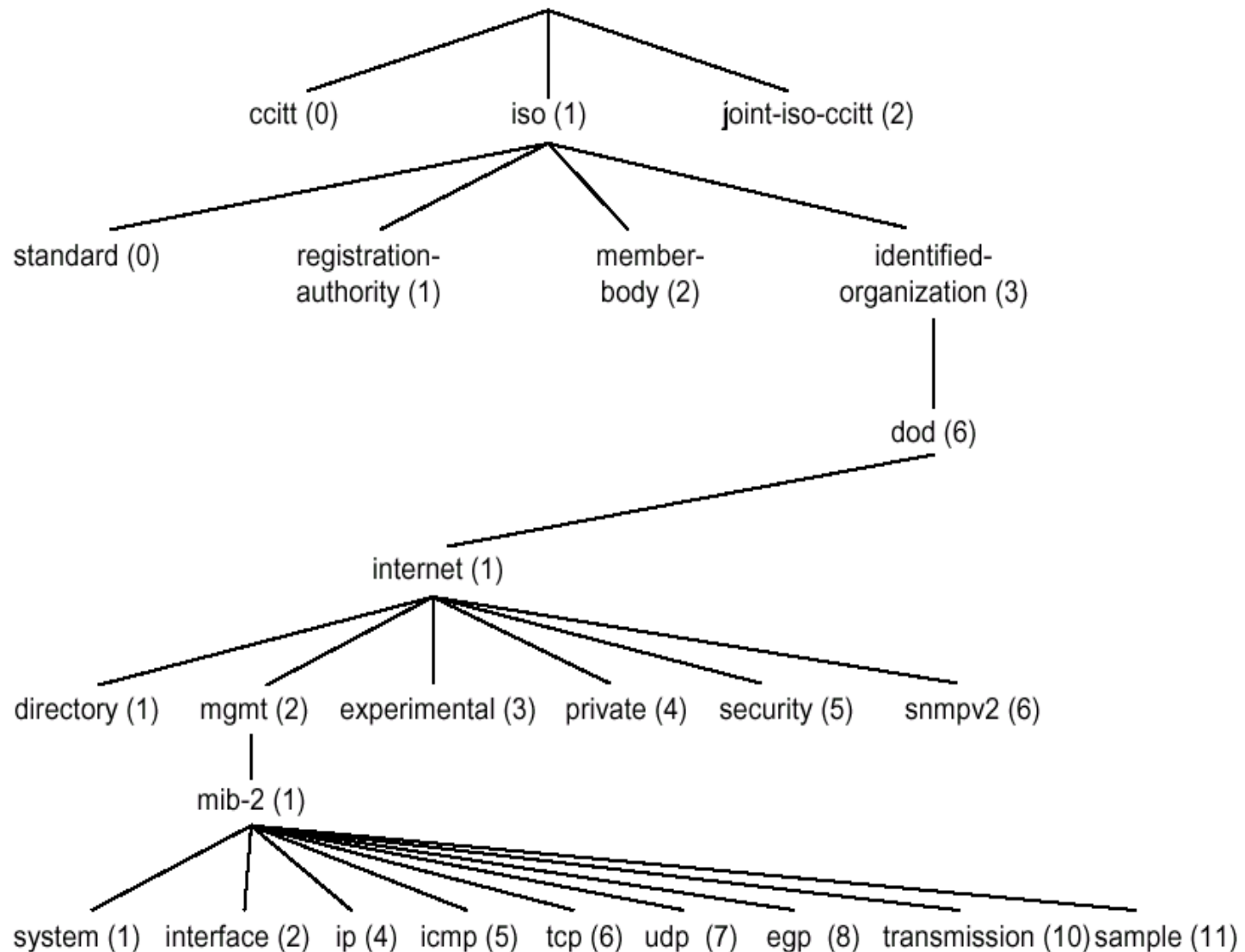


Fig. 7-32. Part of the ASN.1 object naming tree.

9.4 简单网络管理协议SNMP（6）

- ASN.1定义了5种方法构造新的类型
 - SEQUENCE：多种类型的有序序列；
 - SEQUENCE OF：一种类型的一维有序序列；
 - SET：多种类型的无序集合；
 - SET OF：一种类型的无序集合；
 - CHOICE：创建一些类型的共同体（UNION）。
- 构造新类型的另一种方法是重新标记一个老的类型
 - 类似C语言中定义新的类型（#define ...）
 - 标签有四类：universal, application-wide, context-specific, private
 - 例如，Counter32 ::= [APPLICATION 1] INTEGER (0..4294967295)

9.4 简单网络管理协议SNMP（7）

- ASN.1的传输语法
 - 基本编码规则BER（Basic Encoding Rules）定义了如何将ASN.1类型的值表示为无二义的字节序列
 - 需要传输的内容
 - 标志符 (type or tag)
 - 数据长度域
 - 数据域

9.4 简单网络管理协议SNMP（8）

- 标志符 (type or tag)
 - 包括三个子域
 - 当tag值在0 ~ 30之间时，用低5位表示；
 - 当tag值大于30时，低5位为“11111”，用后面字节表示。每个标识字节包括7个数据位，最后一个字节高位为“1”，其它字节高位为“0”。

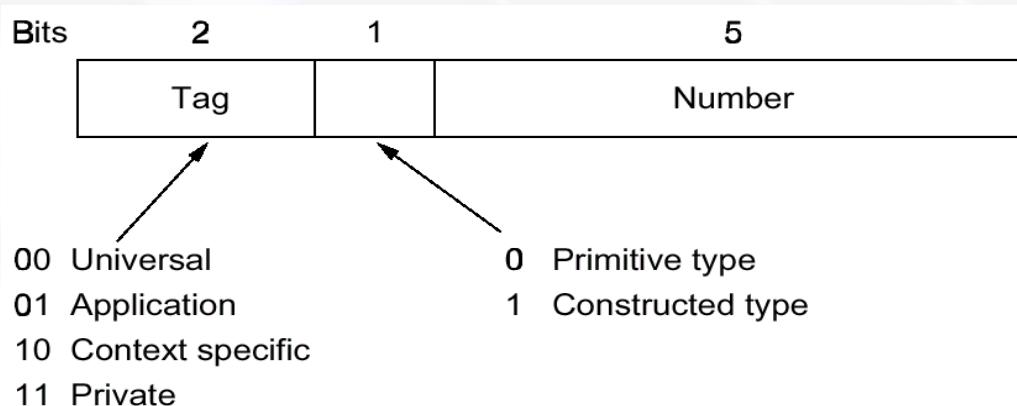


Fig. 7-33. The first byte of each data item sent in the ASN.1 transfer syntax.

9.4 简单网络管理协议SNMP（9）

- 数据域长度

- 当长度 < 128 字节时，用一个字节表示长度，高位为“0”；
- 当长度 ≥ 128 字节时，第一个字节高位为“1”，低7位表示后面表示长度的字节个数，后面的若干个（ ≤ 127 ）字节表示长度
- 例，数据长度1000字节，则长度域包括3个字节，第一个字节为“10000010”，后两个字节为“00000011”和“11101000”

9.4 简单网络管理协议SNMP（10）

- 数据域

- **INTEGER**: 二进制编码;
- **BIT STRING**: 编码表示不变, 长度域表示字节个数, 并在传位串前先传一个字节表示位串最后一个字节不用的位数。
 - 例, 位串 “010011111” 传输时变为 “07 4f 80” (十六进制)
- **OCTET STRING**: 编码表示不变;
- **NULL**: 长度域为0, 不传数据;
- **OBJECT IDENTIFIER**: 按照命名树的编码整数序列编码, 前两个数 a, b 可用一个字节编码, 值为 $40a + b$
- 如果数据长度未知, 需要有结束标志

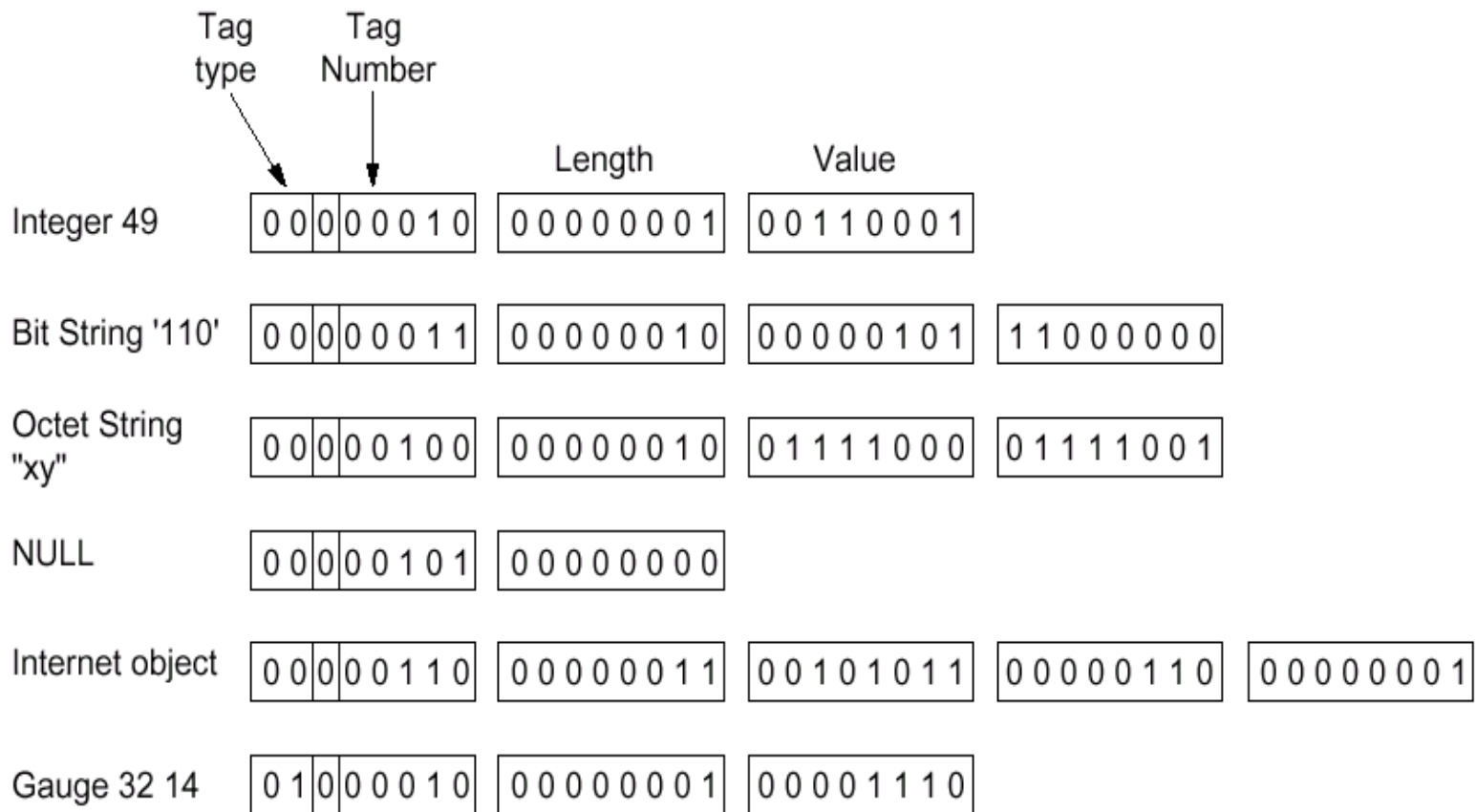


Fig. 7-34. ASN.1 encoding of some example values.

9.4 简单网络管理协议SNMP（11）

- 管理信息结构SMI和管理信息库MIB
 - 定义
 - SNMP在ASN.1的基础上，定义了四个宏，八个新数据类型来定义SNMP的数据结构，被称为管理信息结构SMI。
 - SNMP使用SMI首先将变量定义为“对象”（object），相关的对象被集成“组”（group），组最后被汇集成“模块”（module）。
 - 管理信息库
 - SNMP的MIB包含10个组。网络管理工作站通过使用SNMP协议，向被管理节点中的SNMP代理发出请求，查询这些对象的值。

9.4 简单网络管理协议SNMP（12）

- 每个对象有以下四个属性：
 - 对象类型(object type): 定义了对象的名字
 - 语法(syntax): 指定了数据类型。
 - 存取(access): 表示了对象的存取级别，合法的值有只读、只写、读写和不可存取
 - 状态(status): 定义了对象的实现需要，
 - ◆ 必备的：被管理结点必须实现该对象
 - ◆ 可选的：被管理结点可能实现该对象
 - ◆ 已经废弃的：被管理结点不需要实现该对象

9.4 简单网络管理协议SNMP（13）

■ SNMP协议

- 定义了网络管理工作站和SNMP代理之间的通信过程和协议数据单元。
- 网络管理工作站发往SNMP代理的数据请求
Get-request, Get-next-request, Get-bulk-request
- 网络管理工作站发往SNMP代理的数据更新请求
Set-request
- 网络管理工作站与网络管理工作站之间的MIB交换
Inform-request
- SNMP代理发往网络管理工作站的陷阱报告
SnmpV2-trap

9.5 网络安全（1）

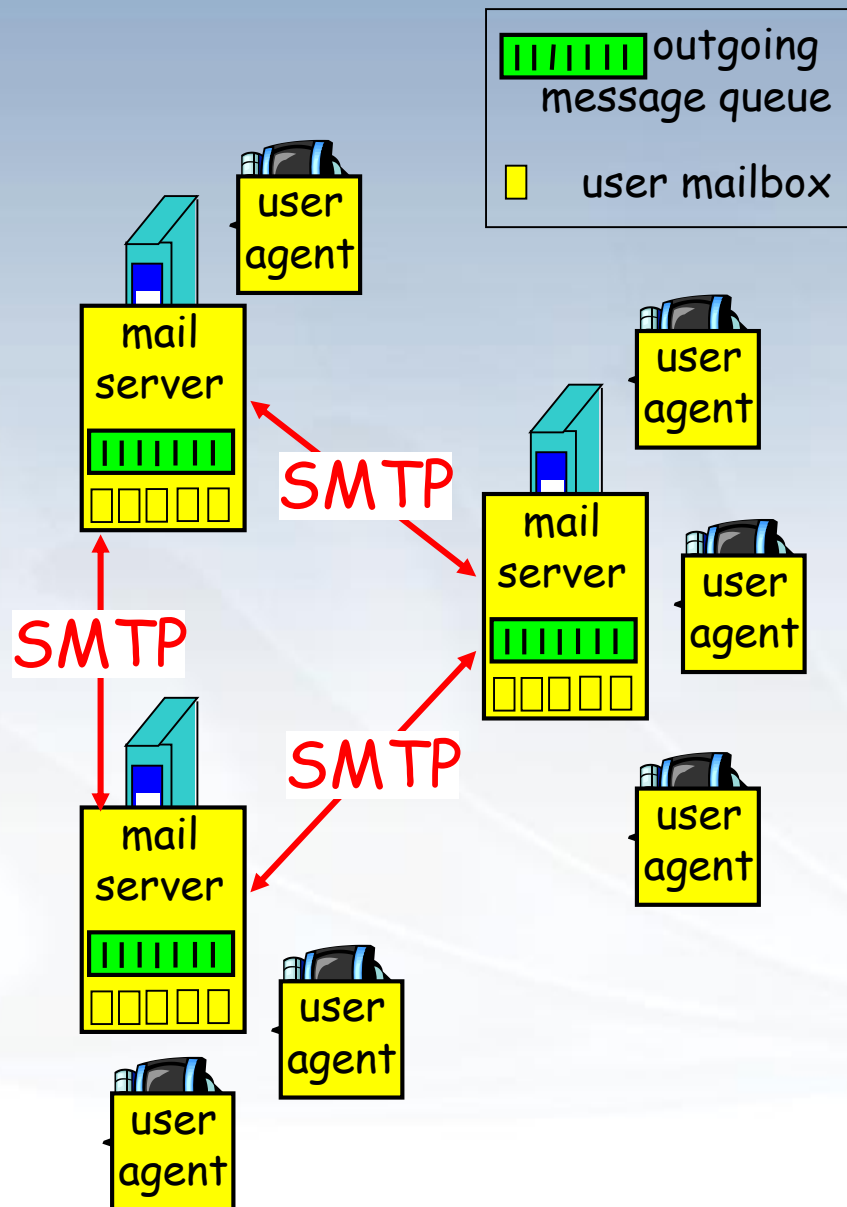
- 自学
- 要求掌握以下知识点
 - 网络安全问题
 - 加密
 - 身份验证
 - 数字签名
 - 完整性控制
 - 加密模型
 - 明文P，密钥K，密文C
 - 加密过程： $C = E_k(P)$
 - 解密过程： $P = D_k(C)$

9.5 网络安全（2）

- 加密方法
 - 加密由两类基本方法：替换加密法和换位加密法
- 加密算法的标准
 - 基于密钥算法的加密标准：**DES**，使用**56**位的密钥，目前被广泛地使用；
 - 基于公钥算法的加密标准：**RSA**，每个人有一个公钥**E_k**和一个密钥**D_k**，只能用密钥**D_k**解开会用公钥**E_k**加密的报文。

9.6 电子邮件（1）

- 相关协议标准
 - 1982年ARPANET提出了RFC821(传输协议), RFC822(消息格式)作为电子邮件协议;
 - 1984年CCITT提出了X.400建议,但是没有得到普及
- 体系结构
 - 用户代理: 允许用户阅读和发送电子邮件, 一般为用户进程;
 - 消息传输代理: 将消息从源端发送至目的端, 一般为系统的后台进程;
 - 简单邮件传输协议SMTP(Simple Mail Transfer Protocol)



9.6 电子邮件（2）

- 电子邮件系统提供的五大基本功能
 - 成文：指创建消息或回答消息的过程；
 - 传输：指将消息从发送者传出至接收者；
 - 报告：将消息的发送情况报告给消息发送者；
 - 显示：使用相应的工具软件将收到的消息显示给接收者
 - 处理：接收者对接收到的消息进行处理，存储/丢弃/转发等等。
- 其它高级功能
 - 自动转发、自动回复；
 - mailbox，创建邮箱存储邮件；
 - mailing list；
 - 抄送（cc）、高优先级、加密。

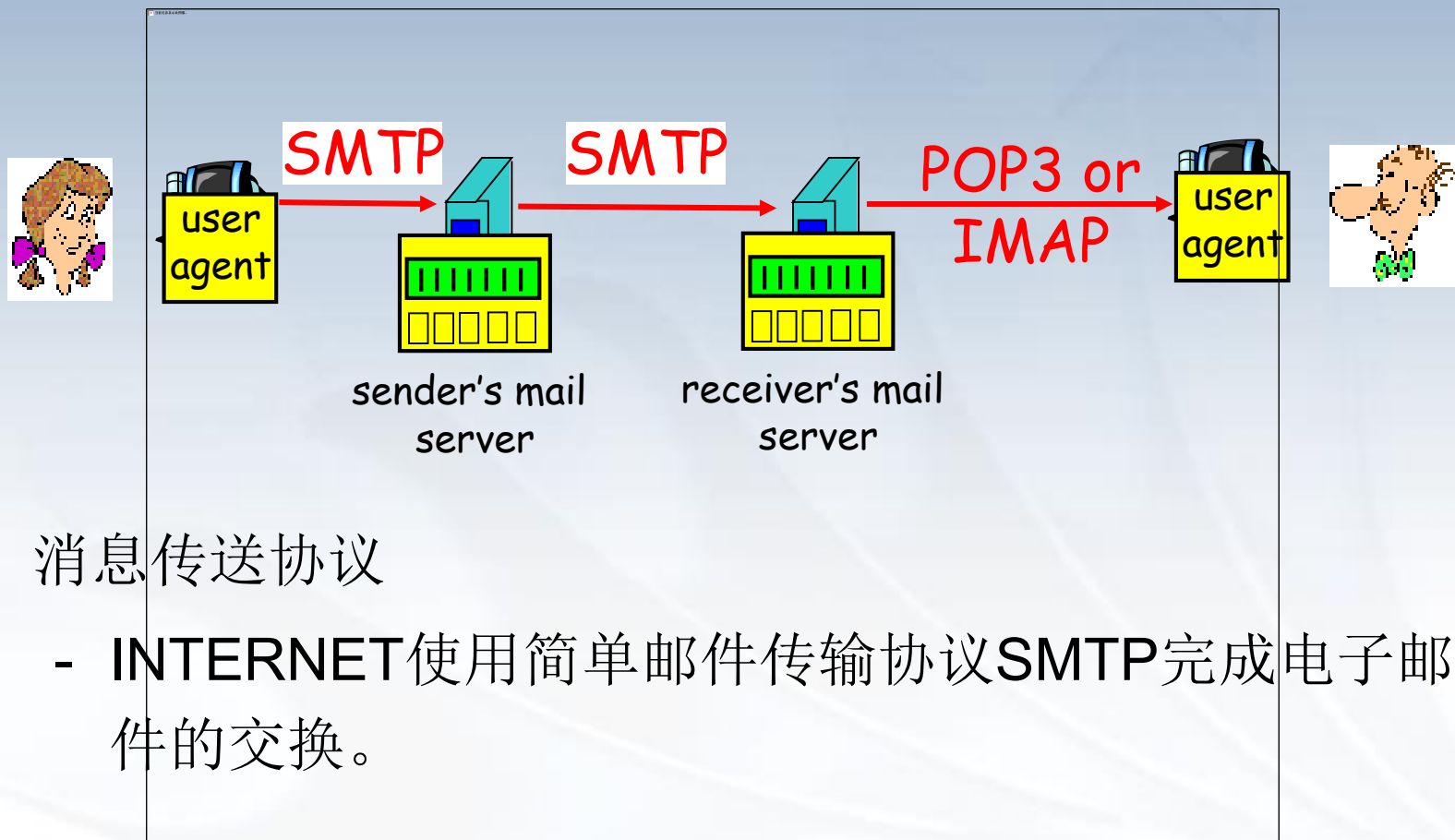
9.6 电子邮件（3）

- 电子邮件的组成
 - 信封：接收方的信息，如名字、地址、邮件的优先级和安全级别；
 - 信件内容：由信头和信体组成，信头包含了用户代理所需的控制信息，信体是真正的内容。
- 用户代理
 - 发送电子邮件
 - email地址，例如，webmaster@www.tsinghua.edu.cn
 - mailing list，例如，students@cs.tsinghua.edu.cn
 - X.400地址，例如，/C=US/SP=MASSACHUSETTS/L=CAMBRIDGE/PA=360 MEMORIAL DR./CN=KEN SMITH/

9.6 电子邮件（4）

- 阅读电子邮件
 - 用户代理在启动时检查用户的**mailbox**，通知用户是否有新邮件到来。并摘要性的显示邮件的主题、发送者及其邮件的状态。
- 信件格式
 - **RFC822**
 - 信件包括信封、若干信头域、一个空行和信体
 - 电子邮件的扩展
 - **MIME**（**Multipurpose Internet Mail Extensions**），增加了对图像、声音、视频、可执行文件等的支持。使用不同的编码方法将信息转化为**ASCII**字符流

9.6 电子邮件（5）



9.6 电子邮件（6）

- 过程如下

- 消息传输代理在源端主机和目的主机的**25**号端口之间建立一条**TCP**连接，使用简单邮件传输协议**SMTP**协议进行通信；
- 在**TCP**连接建立好之后，作为客户的邮件发送方等待作为服务器的邮件接收方首先传输信息
- 服务器首先发出准备接收的**SMTP**消息，客户向服务器发出**HELO**消息，服务器回答以**HELO**消息，双方进入邮件传输状态；

9.6 电子邮件（7）

- 邮件传输过程：客户首先发出邮件的发信人地址(MAIL FROM)，然后发出收信人的地址(RCPT TO)，服务器确认收信人存在后，发出可以继续发送的指示，客户发送真正的消息(DATA)，以 ‘.’作为结束；
- 当客户方邮件发送完之后，服务器开始发送邮件至客户，过程同上；
- 两个方向的发送完成之后，释放TCP连接(QUIT)。
- Persistent方式

9.6 电子邮件（8）

- 注意
 - 消息以7-比特ASCII码为单位
 - 某些特殊字符串(如CRLF.CRLF)不允许在消息中出现，需要编码(例如，base64)
- 其它协议
 - POP3（Post Office Protocol），RFC 1939，用户代理和邮箱不在同一机器上，用户代理使用此协议将邮箱中的信件取回本地；
 - IMAP（Internet Mail Access Protocol），RFC 1730，收信人使用多个用户代理访问同一邮箱，邮件始终保持在邮箱中。
 - 加密电子邮件协议：PGP与PEM协议。

9.6 电子邮件（9）

- try smtp interaction for yourself
 - telnet servername 25
 - see 220 reply from server
 - enter HELO, MAIL FROM, RCPT TO, DATA, QUIT commands
 - above lets you send email without using email client (reader)

9.6 电子邮件（10）

■ Smtplib交互实例

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?

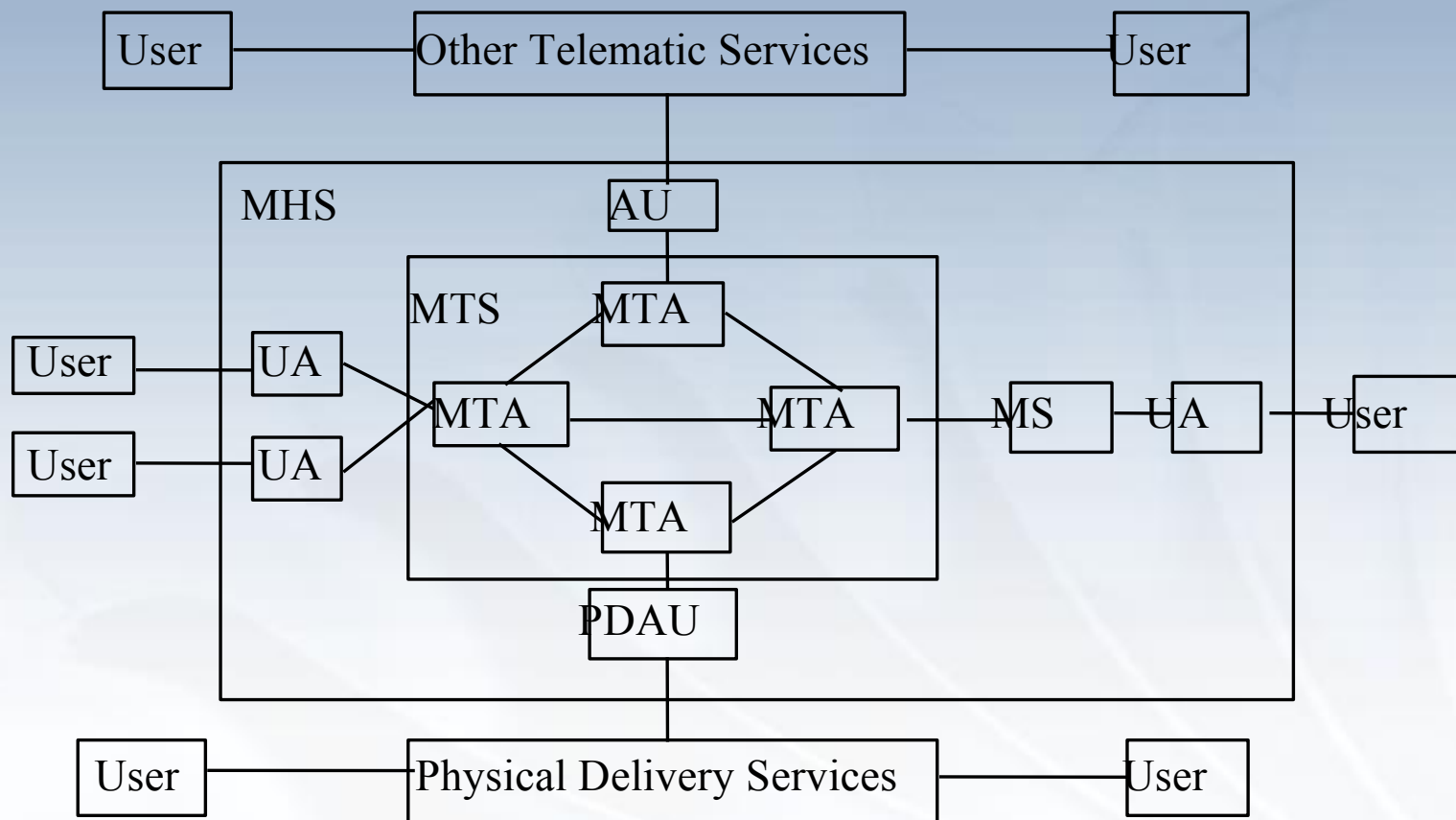
C: How about pickles?

C: .

S: 250 Message accepted for delivery

C: QUIT

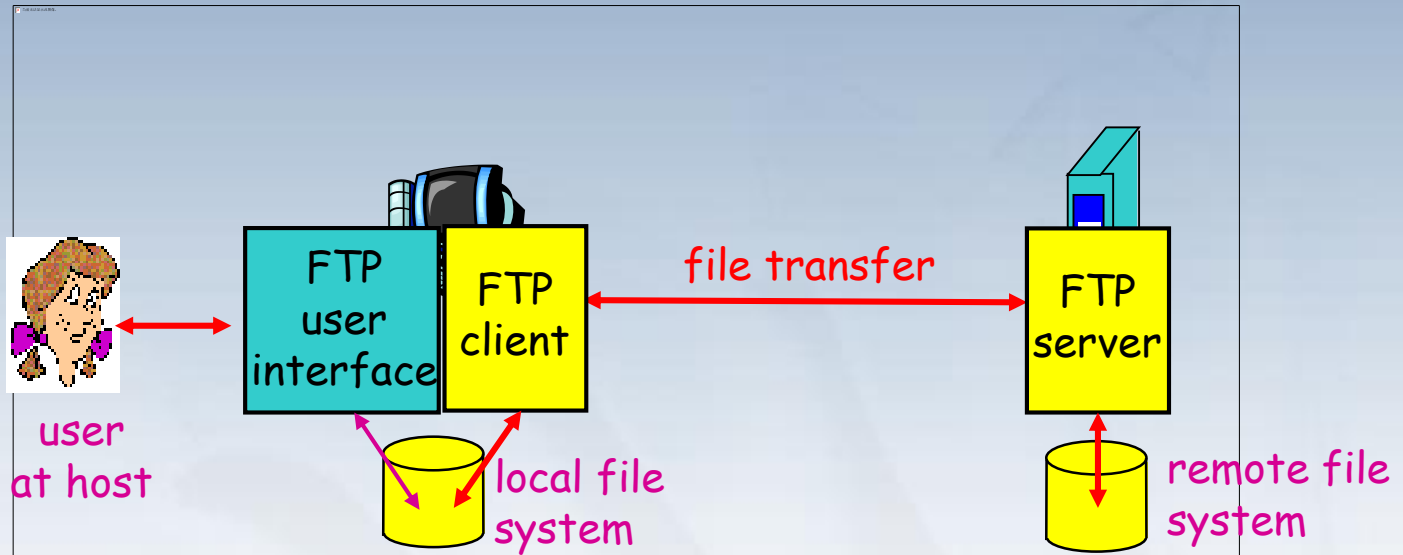
S: 221 hamburger.edu closing connection



X.400 MHS Function Model

UA	----	User Agent	MTA	----	Message Transfer Agent
MS	----	Message Store	AU	----	Access Unit
PDAU	----	Physical Delivery Access Unit			

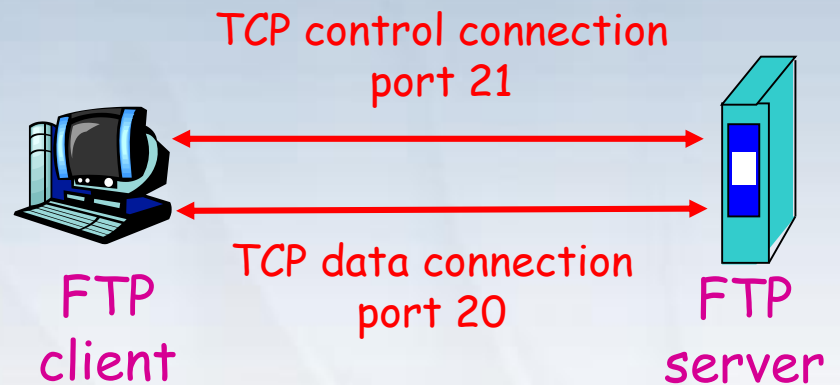
9.7 文件传输协议FTP（1）



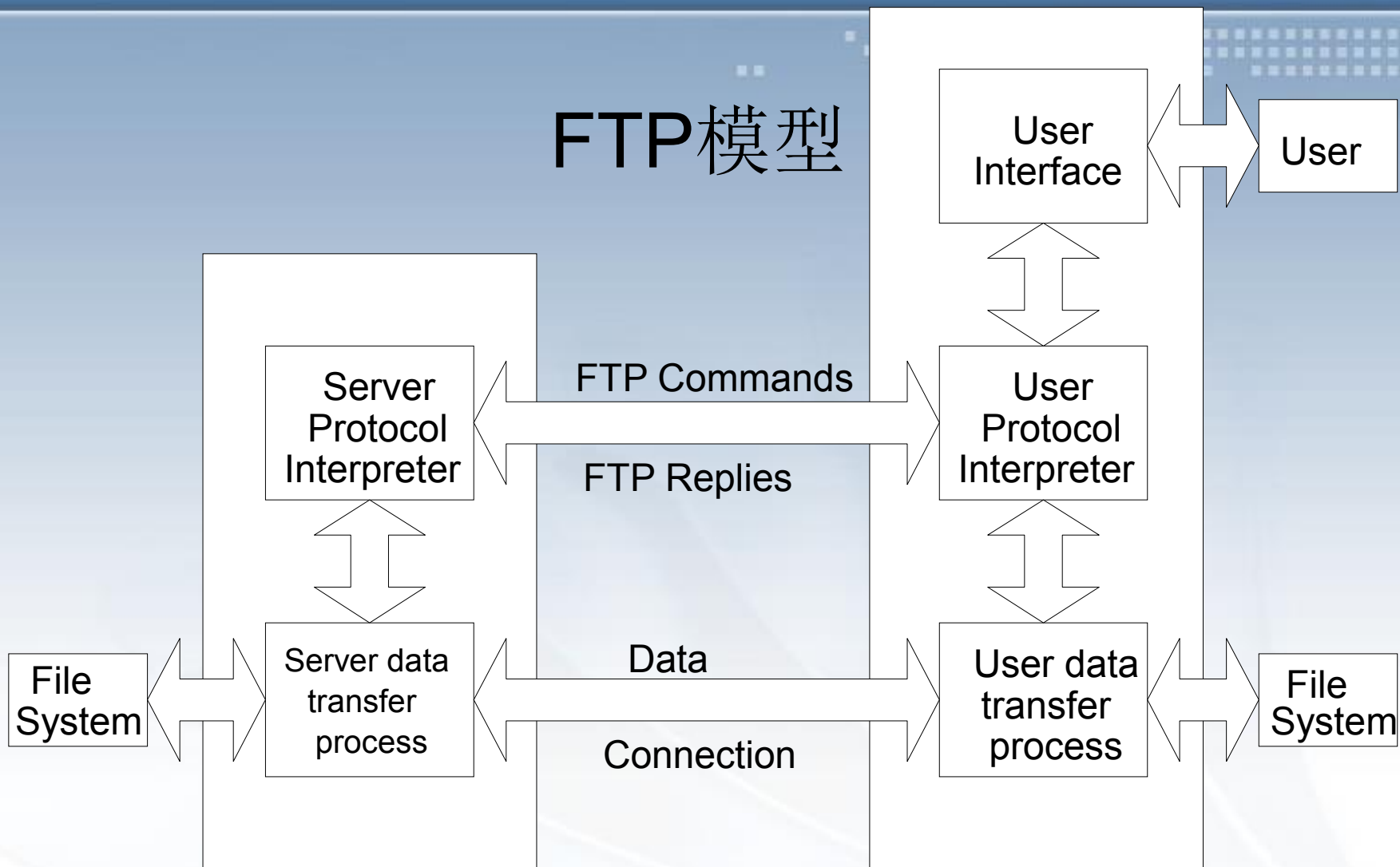
- 在两个主机之间传输文件
- 客户/服务器模式：由客户端发起文件传输(上传或下载)
- ftp: RFC 959
- ftp server: port 21

9.7 文件传输协议FTP（2）

- 客户端连接到ftp服务器
TCP的21号端口
- 建立两个并行的TCP连接：
 - **控制**: 在客户端和服务
器之间交换命令、响应
 - **数据**: 传递文件数据
- Ftp服务器维护状态：当前
目录，身份认证



FTP模型



- 注意:
1. 数据连接可以双向使用.
 2. 数据连接不必始终存在.
 3. 控制连接采用的是telnet.

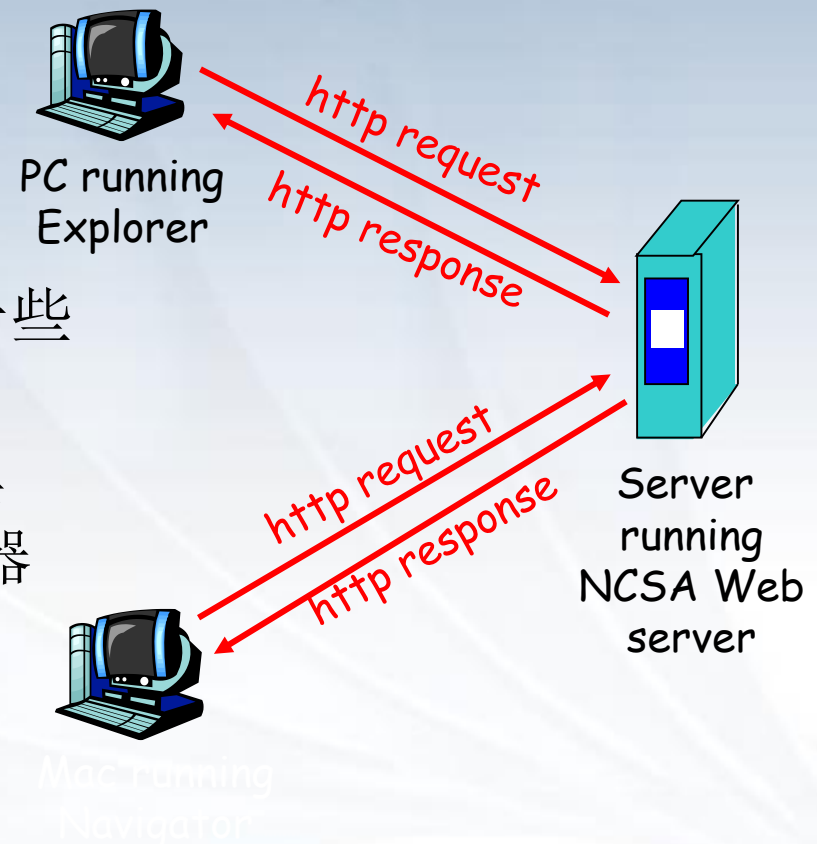
9.8 WWW (1)

- WWW(World Wide Web)是用于访问遍布于INTERNET上的相互链接在一起的超文本的一种结构框架。
- 历史
 - 1989年，设计WWW的思想产生于欧洲核研究中心CERN；
 - 1991年，第一个原型在美国的Hypertext '91会议上展示；
 - 1993年，第一个图形化浏览器，Mosaic；
 - 1994年，Andreessen创建NETSCAPE公司，开发WEB的客户和服务软件；
 - 同年，CERN和MIT共同创建WWW论坛，制定相关的协议标准，<http://www.w3.org>。

9.8 WWW (2)

■ 术语

- 网页：
 - 由“object”组成
 - 用URL标识地址
- 大部分网页包括：
 - 基本的HTML页面和一些引用的object
- 用户用来访问网页的工具 (user agent) 被称为浏览器 (browser)：
 - MS Internet Explorer
 - Netscape Navigator
- Web服务器：



9.8 WWW (3)

■ 用户眼中的WEB

- WEB是由互相链接在一起的网页构成的，这些网页是由普通文本、超文本Hypertext，以及图表、照片等构成；
- 用户通过称为浏览器的软件来观看网页，浏览器取回所请求的网页，解释其中所含的文本和格式命令，并正确的显示出来；
- 网页中的文本串若指向其它的网页（此指针称为超级链接Hyperlink，此文本串称为超文本），会被特别地显示出来（加下划线），用户若选择此超级链接，浏览器会将此超级链接所指的网页取回；
- 当超文本网页中包含声音、动画等其它媒体时，网页被称为是超媒体的。浏览器一般通过外挂的帮助程序来显示这些超媒体信息。

9.8 WWW (4)

■ WEB的客户/服务器模型

- 在每个**WEB**服务器上有一个服务进程在**TCP**的**80**端口上监听由浏览器发来的建立连接请求；在连接建立之后，浏览器和服务器之间使用超文本传输协议**HTTP**协议进行信息传输
- 超级链接是使用**URL**（统一资源定位符）来找到目标网页的。**URL**由三部分组成：
 - 协议类型（**HTTP**、**FTP**、**TELNET**等）；
 - 网页所在机器的地址（域名或**IP**地址）；
 - 包含网页的文件名称。
- 浏览器确定**URL**，通过**DNS**解析**IP**地址，建立**TCP**连接。向服务器发出**HTTP**的**GET**请求取回网页，释放连接，并显示网页中所有的文本。然后，浏览器逐一取回网页中的图象。

9.8 WWW (5)

■ 超文本传输协议HTTP

- HTTP协议由一套从浏览器发往服务器的请求和一套从服务器发往浏览器的响应组成。
- 请求分为两类
 - 简单请求: **GET**网页文件名, 服务器回答以网页内容;
 - 完全请求: **GET**网页文件名、协议版本号, 服务器回答以使用**MIME**格式编码的网页内容。
- http1.0: RFC 1945; http1.1: RFC 2068

9.8 WWW (6)

- 超文本标记语言HTML
 - HTML使用一套标记符号来实现对文本的格式化;
 - HTML 2.0引入了对表格的支持, 允许用户填充信息并将其送回服务器。服务器方使用一个CGI程序进行处理。

http example

Suppose user enters URL

www.someSchool.edu/someDepartment/home.index

(contains text,
references to 10
jpeg images)

1a. http client initiates TCP
connection to http server (process)
at www.someSchool.edu. Port 80
is default for http server.

1b. http server at host
www.someSchool.edu waiting for TCP
connection at port 80. "accepts"
connection, notifying client

2. http client sends http *request message*
(containing URL) into TCP connection
socket

3. http server receives request message,
forms *response message* containing
requested object
(someDepartment/home.index), sends
message into socket

time



http example (cont.)

4. http server closes TCP connection.

5. http client receives response message containing html file, displays html. Parsing html file, finds 10 referenced jpeg objects

6. Steps 1-5 repeated for each of 10 jpeg objects

time



非持久连接和持久连接

非持久连接（Non-persistent）

- HTTP/1.0
- 服务器解析请求，发出响应报文后关闭连接
- 每个object的取得都需要两个RTT
- 每个object的传输都要经历慢启动

But most 1.0 browsers use parallel TCP connections.

持久连接（Persistent）

- default for HTTP/1.1
- 在同一个TCP连接上：服务器解析请求并响应，再解析新的请求...
- 客户端一旦得到基本的HTML文件就发出请求索取全部object
- 较少的RTT和慢启动时间

Trying out http (client side) for yourself

1. Telnet to your favorite Web server:

```
telnet www.eurecom.fr 80
```

Opens TCP connection to port 80 (default http server port) at www.eurecom.fr. Anything typed in sent to port 80 at www.eurecom.fr

2. Type in a GET http request:

```
GET /~ross/index.html HTTP/1.0
```

By typing this in (hit carriage return twice), you send this minimal (but complete) GET request to http server

3. Look at response message sent by http server!

9.8 WWW (7)

■ JAVA

- 产生的原因

- HTML只能处理包含文本、表格和图象的静态页面，以及浏览器和服务端之间使用CGI的有限交互。利用JAVA设计交互式的主页。

- 工作过程

- 网页可以指向一个JAVA的applet（小应用程序），当浏览器请求时，applet会被下载至浏览器，并被浏览器中所包含的JAVA解释器执行。

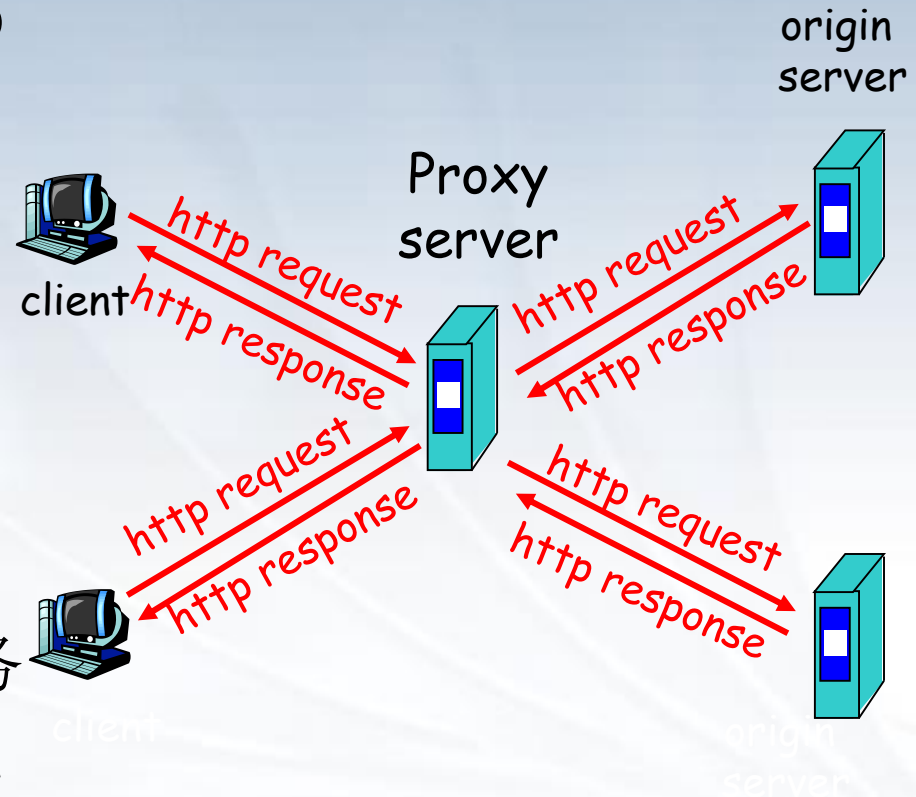
- JAVA系统包含三个部分

- JAVA到字节代码的编译器
- 支持applet的浏览器
- 字节代码的解释器

Web Caches (proxy server)

目标：不访问原服务器而满足用户的请求

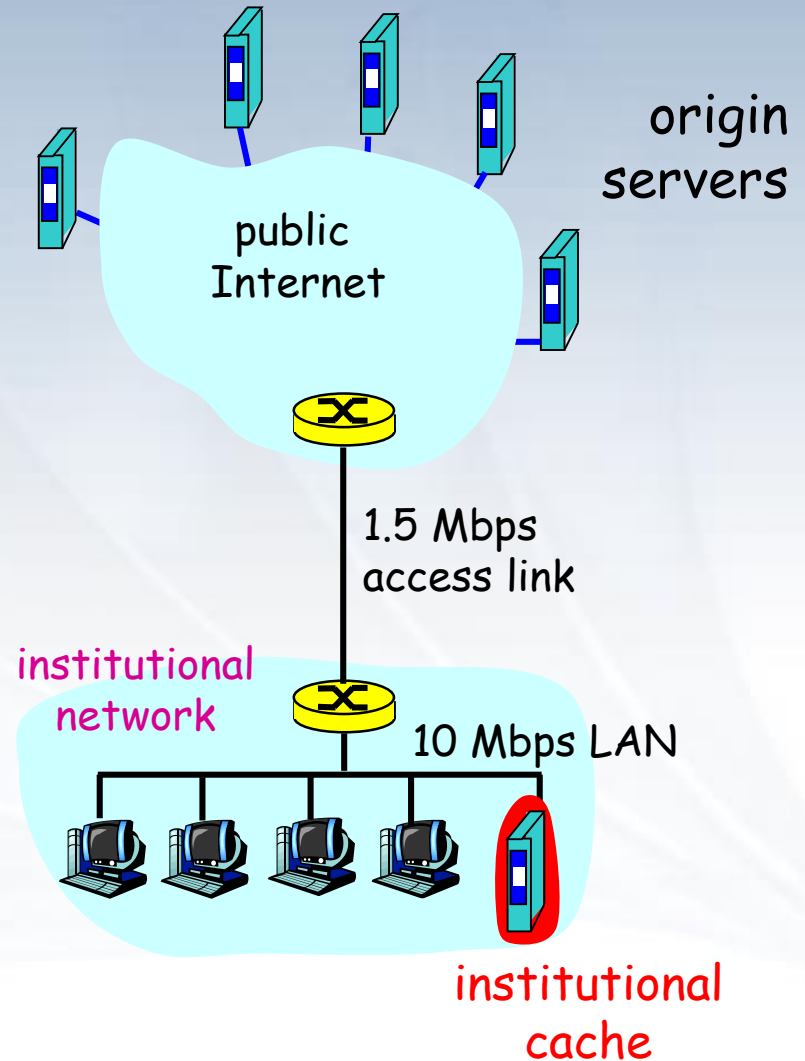
- 用户设置浏览器通过Web缓存来访问Web
- 浏览器将所有http请求发给Web缓存
 - 如果所请求object在缓存中，该object被立即通过http回答返回
 - 否则Web缓存向原服务器发请求获得该object，再响应浏览器的请求



为什么需要Web缓存

假定：缓存离用户端近，
比如在同一网络

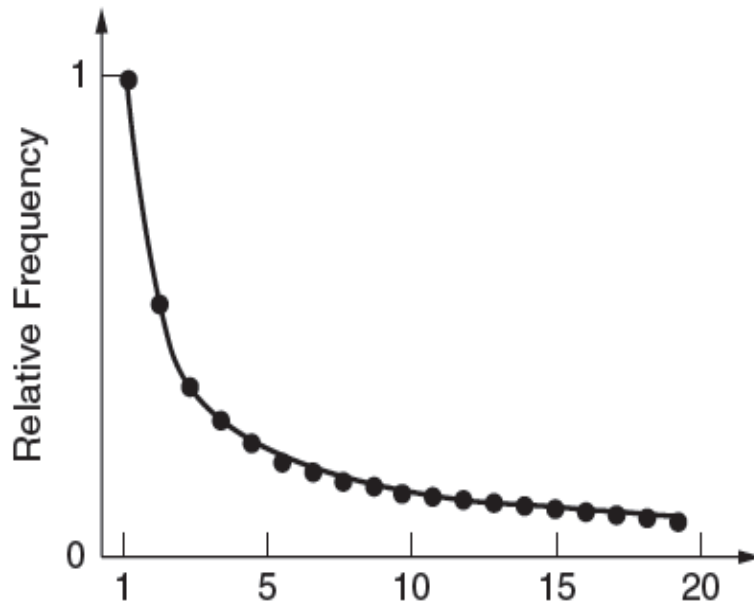
- 较小的响应时间
- 减轻通往远端服务器的连接的流量



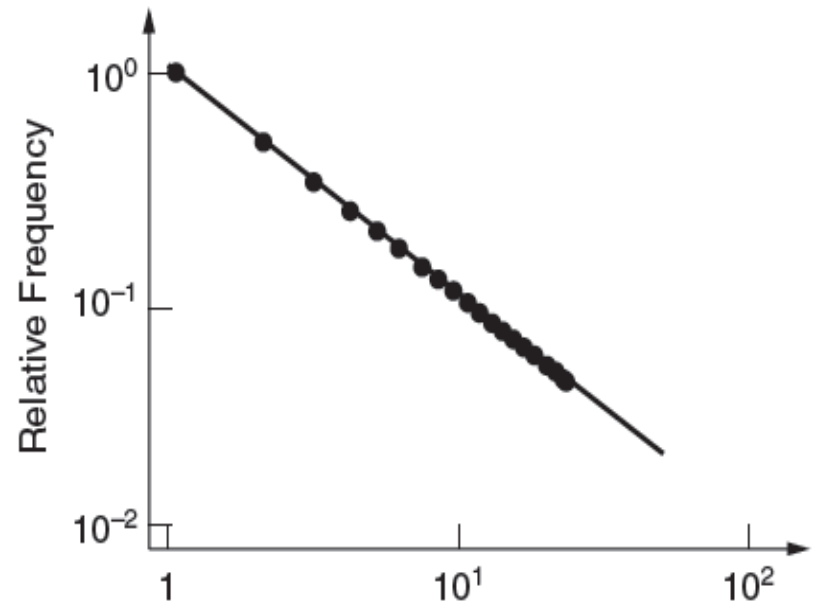
Content Delivery

- Content and internet traffic
- Server farms and web proxies
- Content delivery networks
- Peer-to-peer networks

Content and Internet Traffic



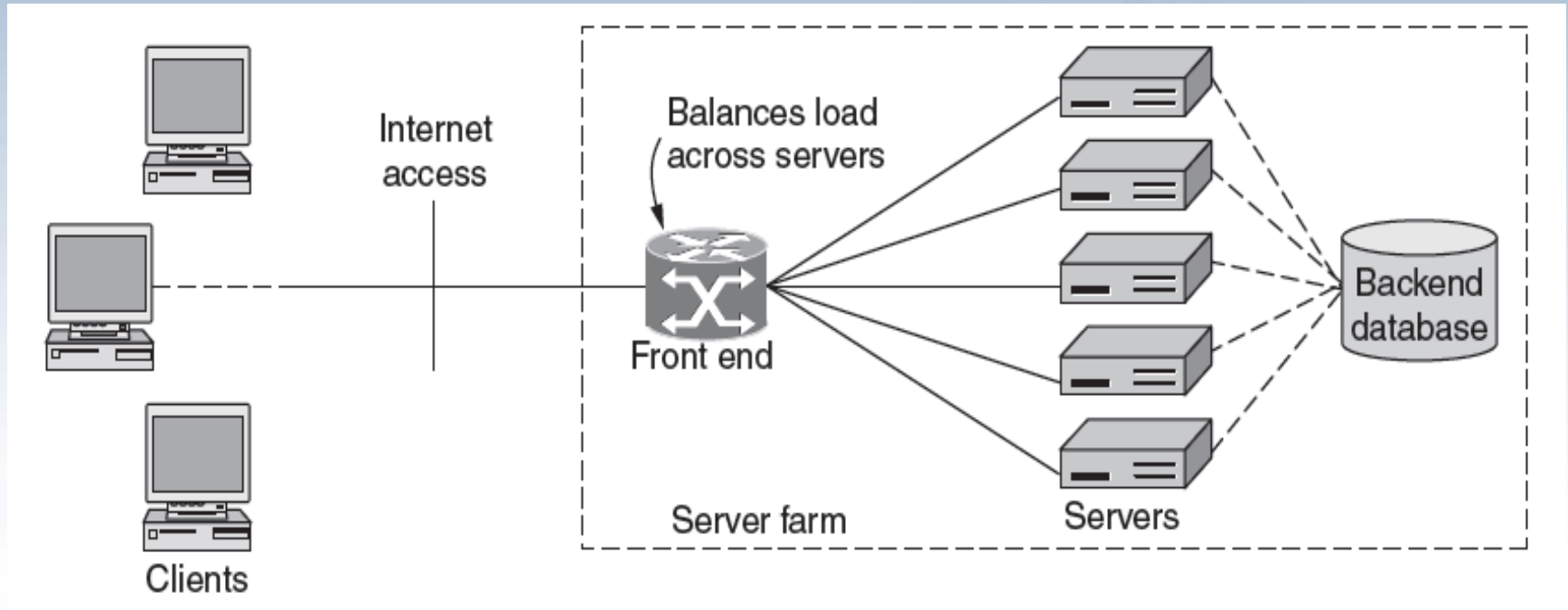
(a)



(b)

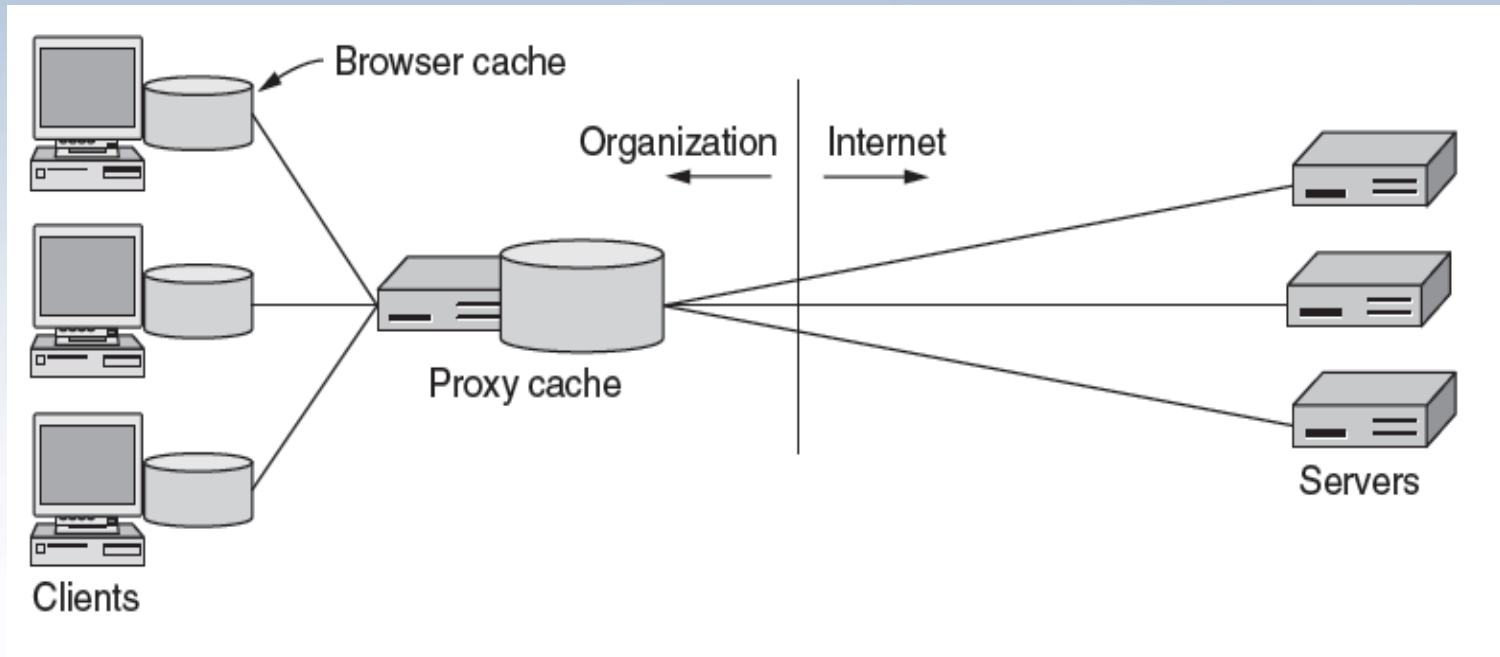
Zipf distribution (a) On a linear scale. (b) On a log-log scale.

Server Farms and Web Proxies (1)



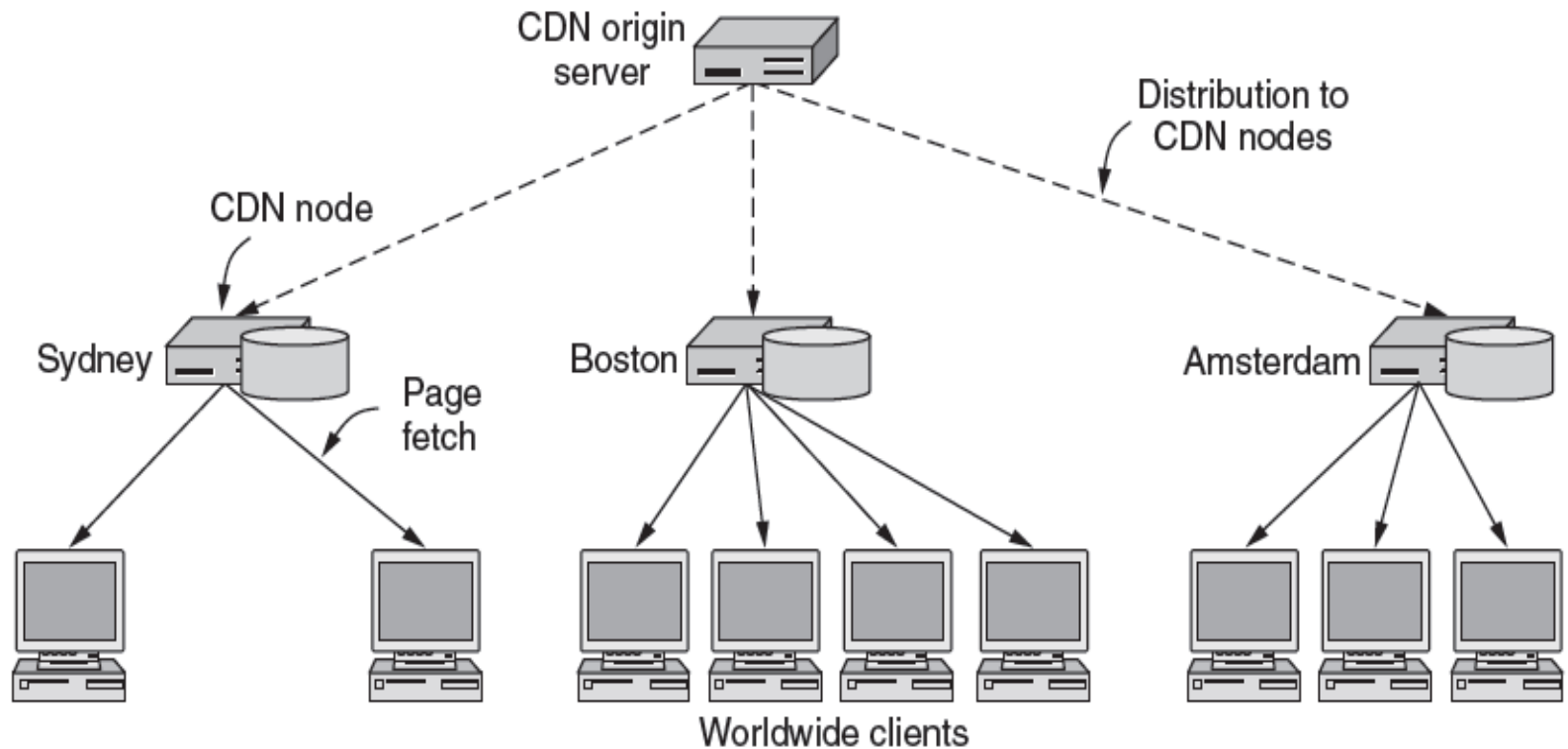
A server farm.

Server Farms and Web Proxies (2)



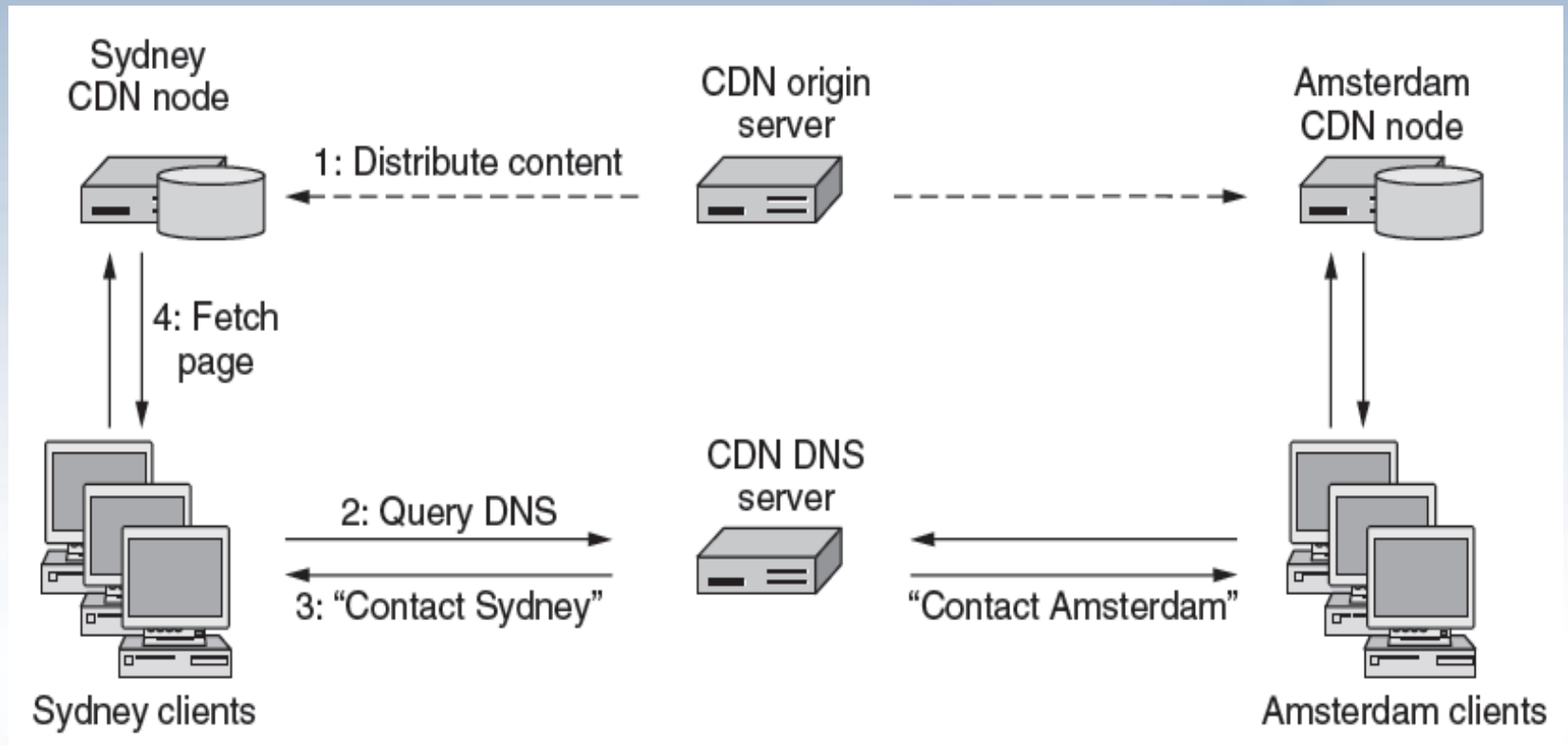
A proxy cache between Web browsers and Web servers.

Content Delivery Networks (1)



CDN distribution tree.

Content Delivery Networks (2)



Directing clients to nearby CDN nodes using DNS.

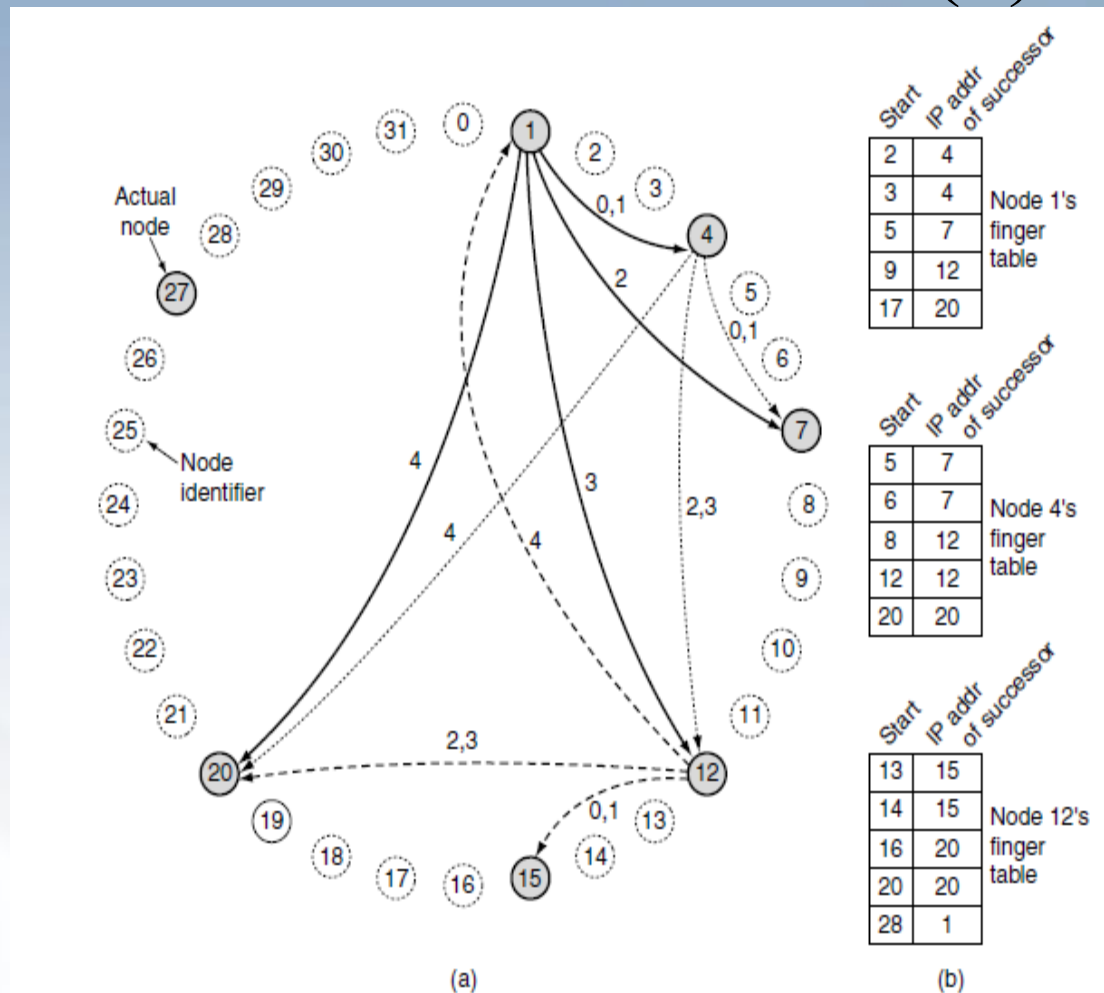
Peer-to-Peer Networks (1)

Problems to be solved with BitTorrent sharing

1. How does a peer find other peers
2. How is content replicated by peers to provide high-speed downloads
3. How do peers encourage each other to upload content to others



Peer-to-Peer Networks (3)



- (a) A set of 32 node identifiers arranged in a circle. The shaded ones correspond to actual machines. The arcs show the fingers from nodes 1, 4 and 12. The labels on the arcs are the table indices. (b) Examples of the finger tables.

课程结束，谢谢大家！