

[XJTUSE Computer Network]—Chapter02 Application Layer

概述

一、应用层协议原理

1、网络应用程序体系结构

客户端-服务器体系结构C/S

P2P（对等）体系结构

混合C/S加P2P的体系结构

2、进程通信

客户和服务进程

进程与计算机网络之间的接口

进程寻址

3、应用层协议的定义

4、互联网上的QoS(服务质量)要求

5、因特网提供的传输服务

TCP传输控制协议（Transmission Control Protocol）服务

UDP用户数据报协议（User Datagram Protocol）服务

流行的因特网应用机器应用层协议和支撑的传输协议

二、Web和HTTP

1、HTTP概况

2、非持续连接和持续连接

非持久的HTTP：Non-Persistent HTTP

持续连接的HTTP：Persistent HTTP

3、HTTP报文格式

HTTP请求报文request message

HTTP响应报文response message

4、用户和服务器的交互：cookie

5、Web缓存

proxy cache

Client Cache: Conditional Get

distributed cache

server cache:cluster

三、因特网的电子邮件

1、FTP—— file transfer protocol

独立的控制连接和数据连接

2、电子邮件

用户代理

邮件服务器

SMTP

3、与HTTP的对比

4、邮件报文格式

5、邮件访问协议

- 5、非ASCII数据的MIME扩展
- 四、DNS：域名服务
 - 1、DNS提供的服务
 - 2、DNS工作机理概述
 - 分布式、层次数据库
 - DNS域名解析示例
 - DNS缓存和更新记录
 - 3、DNS记录和报文
 - DNS协议，报文
- 五、搜索引擎
- 六、Socket编程
 - 使用WinSock.h
- 练习题

[XJTUSE Computer Network]——

Chapter02 Application Layer

概述

1 网络应用协议的概念和实现方面

传输层服务模型

客户端-服务器模式

点对点模式

2 通过研究流行的应用程序级协议来了解协议

HTTP

FTP

SMTP / pop3 / imap

DNS

3 网络应用程序编程

套接字socket API

一、应用层协议原理

广播、电视、报纸、网站：四大媒介

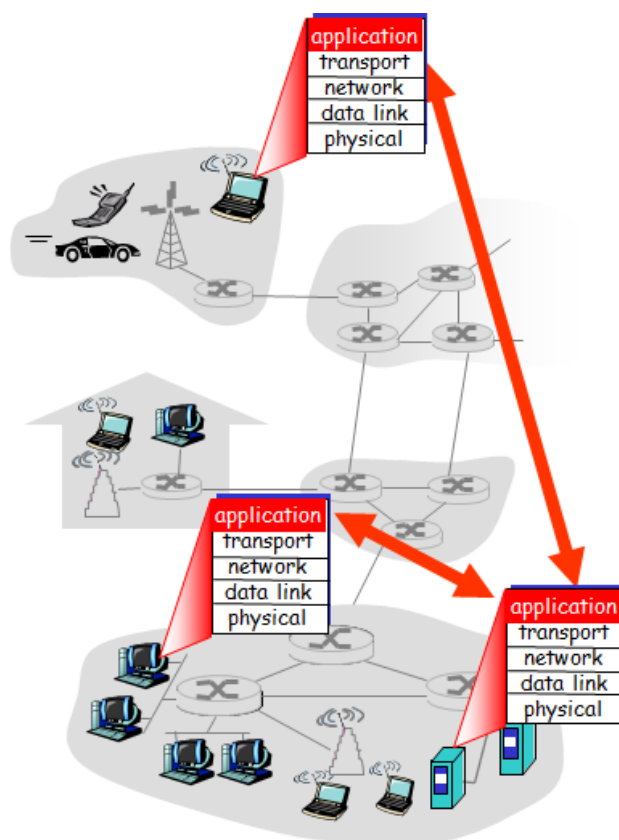
网络核心设备并不在应用层上起作用，而是在较低层起作用。将应用软件限制在端系统，促进了大量的网络应用程序的迅速研发和部署。

研发网络应用程序的核心是写出能够运行在不同的端系统和通过网络彼此通信的程序。

客户端-服务端模式

P2P模式

混合模式



1、网络应用程序体系结构

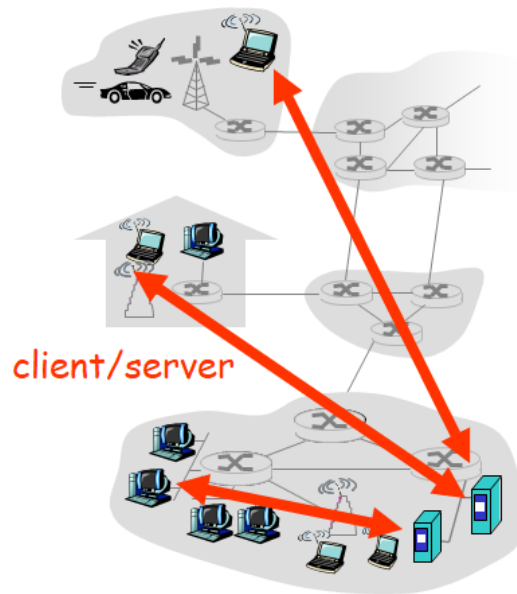
有两种主流的结构：客户端-服务器体系结构与P2P（对等）体系结构

客户端-服务器体系结构C/S

服务器：不间断的主机；永久的IP地址；用于扩展的服务器群

客户端：与服务器通信；可能是间歇性连接；可能有动态IP地址

Web、FTP、Telnet和电子邮件

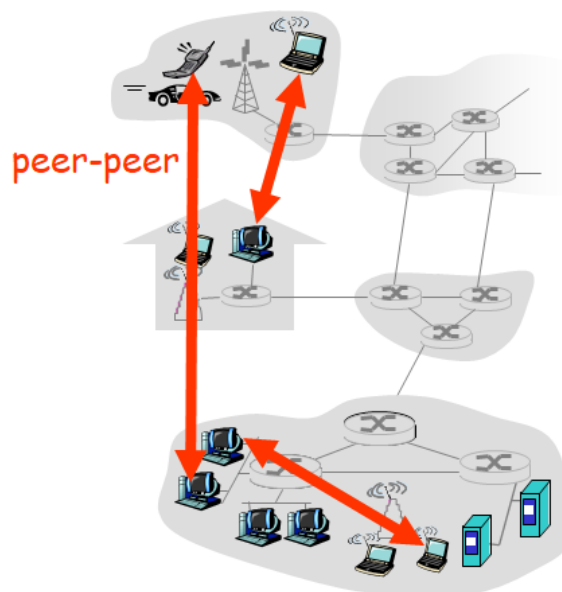


P2P（对等）体系结构

没有不间断服务器；任意端系统直接通信；对等体之间间歇式连接，IP地址变化

高度可扩展但难以管理

适用于流量密集型的应用



混合C/S加P2P的体系结构

1 Skype

基于ip的P2P应用程序

集中式服务器：查找远程方地址

客户端-客户端连接:直接(不通过服务器)

2 即时通信

两个用户之间的聊天是P2P

集中式服务：客户端在线检测/位置

当用户上线时，向中央服务器注册IP地址

用户联系中央服务器，查找好友的IP地址

2、进程通信

进程:在主机上运行的程序。

1 在同一个主机中，两个进程使用 **进程间通信(由操作系统定义)** 进行通信。

2 不同主机上的进程通过 **交换消息** 进行通信



客户和服务进程

客户端进程:发起通信的进程

服务器进程:等待联系的进程

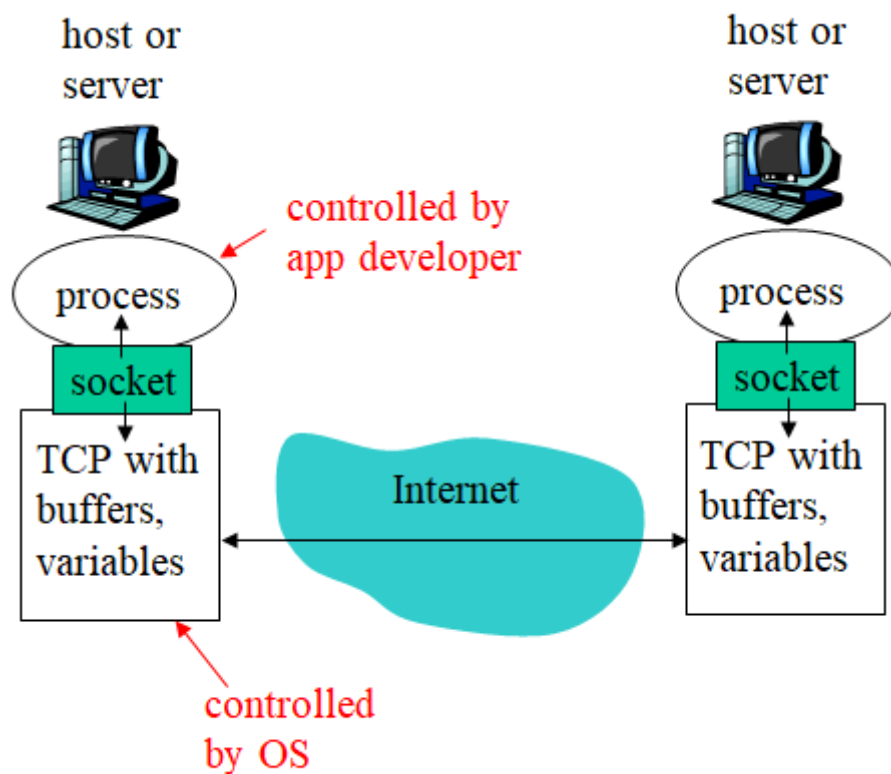
进程与计算机网络之间的接口

进程通过一个称为socket套接字的软件接口向网络发送报文和从网络接受报文

进程可类比于一座房子 ，而它的套接字可以类比于它的门 。当一个进程想向位于另外一台主机上的另一个进程发送报文时，它把报文推出该门(套接字)。

该发送进程假定该门到另外一侧之间有运输的基础设施，该设施将把报文传送到目的进程的门口。

一旦该报文抵达目的主机，它通过接收进程的套接字(套接字)传递，然后接收进程对该报文进行处理。



套接字是同一台主机内应用层与传输层之间的接口。由于该套接字是建立网络应用程序的可编程接口，因此套接字也称为应用程序和网络之间的应用程序编程接口(API)

进程寻址

为了接受报文，进程需要有标识符

❓ 运行进程的主机的IP地址(32位)是否足以识别进程？

❌ 不，许多进程可以运行在同一平台上

标识符包括与主机上的进程关联的 **IP地址和端口号**。

例如端口号：

HTTP服务器:80

邮件服务器:25

3、应用层协议的定义

应用层协议定义了：

- 1 交换的报文**类型**，例如,请求、响应
- 2 消息的**语法**：消息中的哪些字段&字段是如何描述的
- 3 字段的**语义**：字段中信息的含义
- 4 进程何时以及如何发送和响应报文的**规则**

RTP：实时传输协议

种类：

- 1 公共协议：在RFC[请求注解（Request For Comments）]定义；允许互操作性
例如，HTTP, SMTP, BitTorrent
- 2 专用协议：例如,Skype, ppstream

应用层只是网络应用的一部分

4、互联网上的QoS(服务质量)要求

1 丢包率data loss

一些应用程序(如音频)可以承受一些损失

其他应用程序(例如，文件传输，telnet)需要100%可靠的数据传输

2 实时性timing

一些应用程序(如网络电话、互动游戏)需要低延迟才能“有效”。

3 吞吐量throughput

一些应用程序(如多媒体)需要一定的吞吐量才能“有效”：带宽敏感应用

其他应用程序(“弹性应用程序”)利用他们所获得的任何吞吐量

4 安全security

加密的数据，数据的完整性

一些常用应用的需求

应用程序	丢包	吞吐量	时间敏感性
文件传输	不允许	弹性的	不敏感
e-mail	不允许	弹性的	不敏感
Web文档	不允许	弹性的	不敏感
实时音频/视频	容忍一定的丢包	音频: 5kbps-1Mbps 视频:10kbps-5Mbps	敏感: 100ms
存储式音频/视频	容忍一定的丢包	音频: 5kbps-1Mbps 视频:10kbps-5Mbps	敏感: 几秒
互动游戏	容忍一定的丢包	超过几kbs	敏感: 100ms
实时发信息	不允许	弹性的	是和不是

5、因特网提供的传输服务

TCP传输控制协议（Transmission Control Protocol）服务

- 1 连接管理：客户端和服务端进程之间需要设置
- 2 可靠性控制：发送和接收过程之间的可靠传输
- 3 流量控制：发送方不会淹没接收方
- 4 拥塞控制：网络过载时对发送方进行节流

不提供：实时性，最小吞吐量保证，安全性

因特网界研制了TCP的加强版。称为安全套接字层Secure Socket Layer (SSL)，这种强化是在应用层上实现的

UDP用户数据报协议（User Datagram Protocol）服务

发送和接收过程之间不可靠的数据传输

不提供：连接管理，可靠性，流量控制，拥塞控制，实时性，吞吐量保证，安全性

流行的因特网应用机器应用层协议和支撑的传输协议

应用	应用层协议	支撑的传输层协议
电子邮件	SMTP [RFC 2821]	TCP
远程终端访问	Telnet [RFC 854]	TCP

应用	应用层协议	支撑的传输层协议
网页	HTTP [RFC 2616]	TCP
文件传输	FTP [RFC 959]	TCP
流式多媒体	HTTP (eg Youtube), RTP [RFC 1889]	TCP or UDP(局域网内)
网络电话	SIP, RTP, proprietary	通常是UDP

二、Web和HTTP

Web页面由对象组成：对象可以是HTML文件，JPEG图像，Java applet，音频文件，…

Web页面由**基本HTML(超文本标记语言)文件**组成，其中包括几个引用的对象

每个对象都可通过URL寻址：

$$\underbrace{\text{www.someschool.edu}}_{\text{主机名}} / \underbrace{\text{someDept/pic.gif}}_{\text{路径名}}$$

someschool.edu是域名

1、HTTP概况

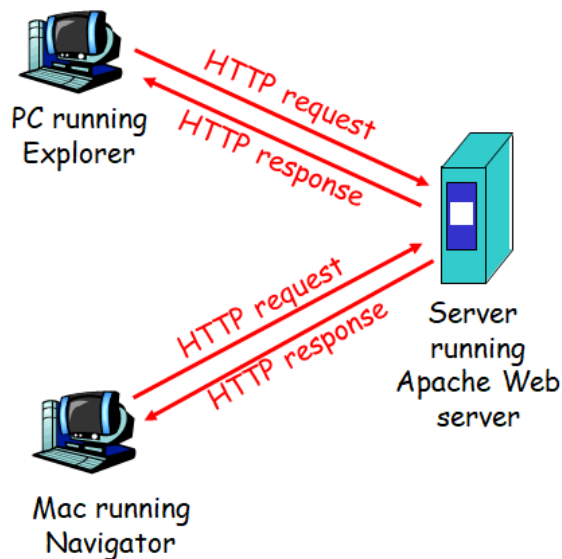
HTTP: hypertext transfer protocol超文本传输协议

Web的应用层协议：用于通信

客户机/服务器模型

客户端：请求、接收、显示Web对象的浏览器

服务端：Web服务器发送对象来响应请求



HTTP使用TCP作为它的支撑传输协议

- 1 客户端发起TCP连接(创建套接字)到服务器，端口80
- 2 服务器接受来自客户端的TCP连接
- 3 浏览器(HTTP客户端)和Web服务器(HTTP服务器)之间交换的HTTP消息(应用层协议消息)
- 4 TCP连接关闭

HTTP是一个**无状态的协议**：服务器不保存以前的客户端的请求信息，因为保存这些状态十分复杂

2、非持续连接和持续连接

非持久的HTTP：Non-Persistent HTTP

非持续连接：每个请求/响应对是经过一个单独的TCP连接发送；一个TCP连接最多发送一个对象。

假设用户进入网址：`www.someSchool.edu/someDepartment/home.index`，其中包含了对十张JPEG图形的引用，这十一个对象位于同一个服务器中。

📎 JPEG（Joint Photographic Experts Group）即联合图像专家组，是用于连续色调静态**图像压缩**的一种标准，**文件后缀名**为.jpg或.jpeg，是最常用的图像文件格式

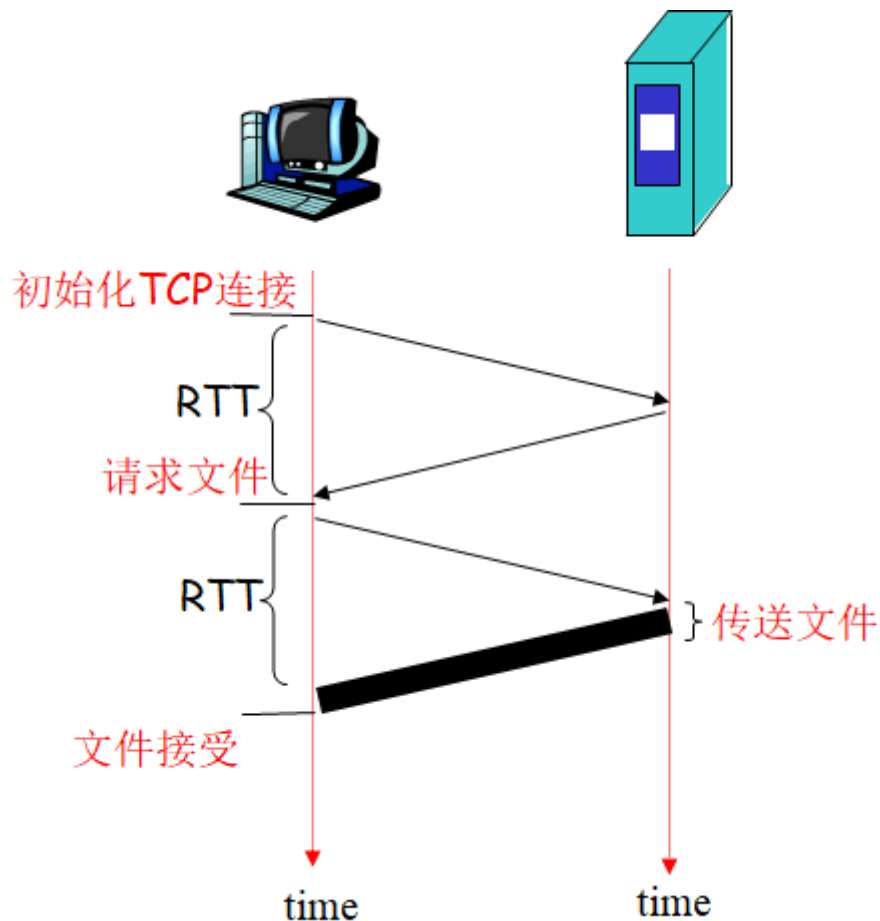
过程如下：

- 1 HTTP客户进程在端口号80发起一个到服务器 `www.someSchool.edu` 的TCP连接, 该端口号是HTTP的默认端口。在客户和服务器的套接字与该连接相关联。
- 2 HTTP 客户经它的套接字向该服务器发送一个HTTP请求报文。请求报文中包含了路径名 `/someDepartment/home.index` 。
- 3 HTTP 服务器进程经它的套接字接收该请求报文, 从其存储器(RAM 或磁盘)中检索出对象 `www.someSchool.edu/someDepartment/home.index` , 在一个HTTP响应报文中封装对象, 并通过其套接字向客户发送响应报文。
- 4 HTTP 服务器进程通知TCP断开该TCP连接。(但是直到TCP确认客户已经完整地收到响应报文为止, 它才会实际中断连接。)
- 5 HTTP 客户接收响应报文, TCP连接关闭。该报文指出封装的对象是一个HTML文件, 客户从响应报文中提取出该文件, 检查该HTML文件, 得到对10个JPEG图形的引用。
- 6 对每个引用的JPEG图形对象重复前4个步骤。

非持续性连接中每个TCP连接只传输一个请求报文和响应报文, 因此上例中当用户请求该Web页面时需要产生11个TCP连接

RTT(round trip time) 的定义:往返时间, 一个小数据包在客户端和服务端之间往返传输的时间。

全部的时间 = 2RTT+文件发送时间



持续连接的HTTP：Persistent HTTP

持续连接：客户端和服务端之间可以通过单个TCP连接发送多个对象。

非持久的HTTP问题：

每个对象需要2个RTT

每个TCP连接需要操作系统开销

浏览器经常打开并行TCP连接来获取被引用的对象

持续的HTTP

服务器在发送响应后**保持连接打开**

在同一客户端/服务器之间通过打开的连接发送的后续HTTP消息

客户端一遇到引用的对象就发送请求

对于所有被引用的对象，只有一个RTT：**一开始连接需要两个RTT，后面的对象获取只用1个RTT**

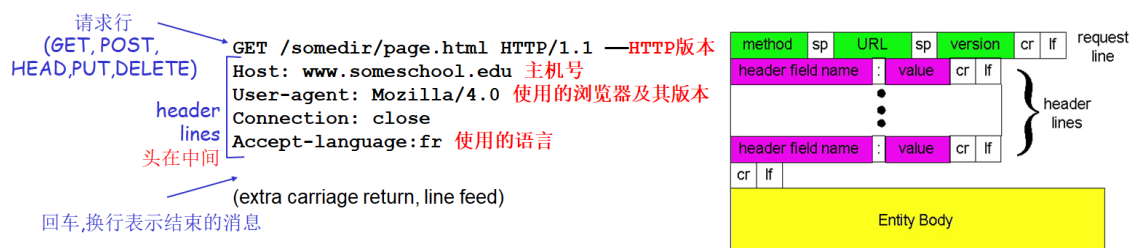
3、HTTP报文格式

两种类型的HTTP消息:请求request, 响应response

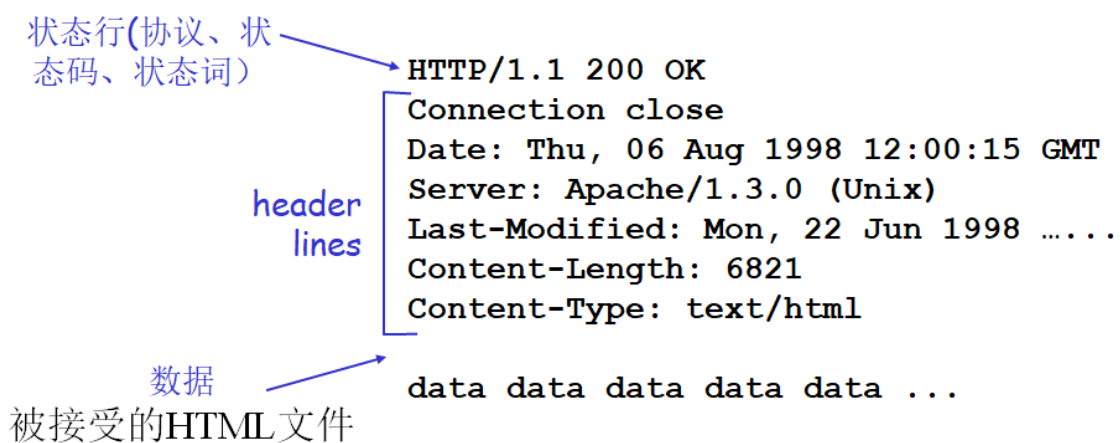
考试只了解由三部分组成, 头在中间

HTTP请求报文request message

ASCII ((American Standard Code for **Information Interchange**): 美国信息交换标准代码) 是基于 **拉丁字母** 的一套电脑 **编码** 系统, 主要用于显示现代 **英语** 和其他 **西欧** 语言



HTTP响应报文response message



4、用户和服务器的交互: cookie

许多主要网站使用cookie:存储在客户端中的小文件; 用于身份验证

四个组件:

- 1 在HTTP响应报文中的一个cookie首部行;
- 2 在HTTP请求报文中的一个cookie首部行;
- 3 在用户端系统中保留有一个cookie文件, 并由用户的浏览器进行管理;
- 4 位于Web站点的一个后端数据库。

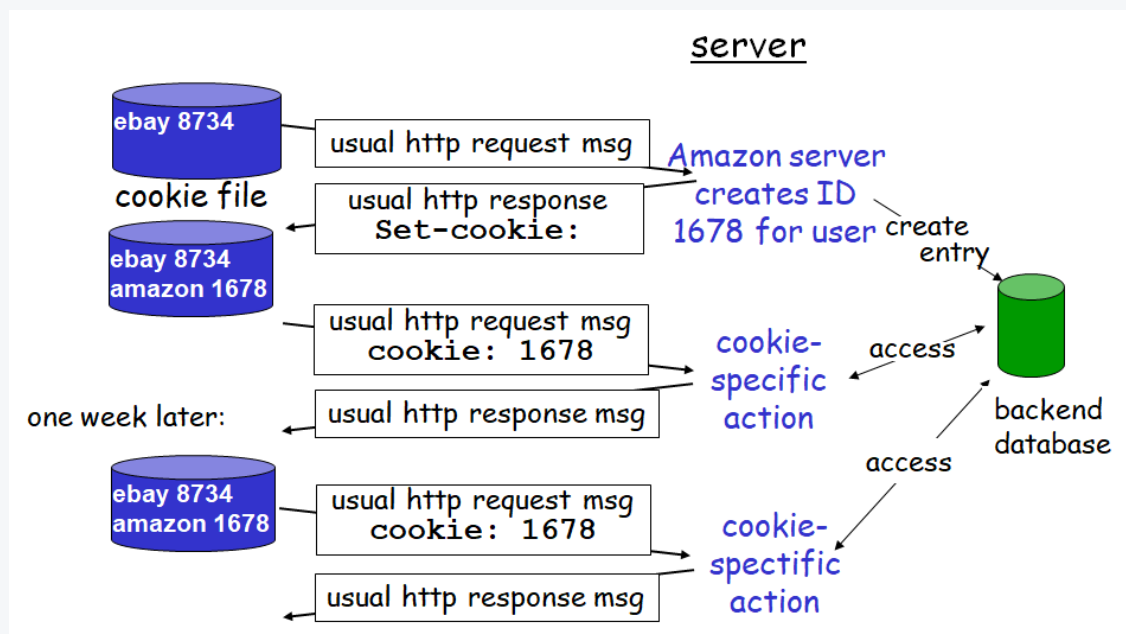
举例

假设Susan总是从家中PC使用InternetExplorer 上网，她首次与Amazon.com 联系。我们假定过去她已经访问过eBay站点。当请求报文到达该 Amazon Web 服务器时，该 Web 站点将产生一个唯一识别码，并以此作为索引在它的后端数据库中产生一个表项。接下来 Amazon Web 服务器用一个包含 Set-cookie:首部的HTTP 响应报文对Susan的浏览器进行响应，其中Set-cookie: 首部 含有该识别码。

例如， 该首部行可能是Set-cookie:1678

当Susan的浏览器收到了该HTTP响应报文时，它会看到该Set-cookie: 首部。该浏览器在它管理的特定cookie文件中添加一行，该行包含服务器的主机名和在Set-cookie: 首部中的识别码。值得注意的是该cookie 文件已经有了用于 eBay 的表项，因为Susan 过去访问过该站点。当Susan 继续浏览 Amazon 网站时，每请求一个Web页面，其浏览器就会查询该cookie 文件并抽取她对这个网站的识别码，并放到 HTTP 请求报文中包括识别码的 cookie 首部行中。

如果Susan再次访问Amazon站点，比如说一个星期后，她的浏览器会在其请求报文中继续放入首部行cookie: 1678。Amazon将根据Susan过去在Amazon访问的网页向她推荐产品。



5、Web缓存

使用Web缓存的原因：减少客户端请求的响应的时间；改善用户体验；节省主干带宽的流量

Web cache的类别：代理cache：proxy cache；客户端cache：client cache；分布式cache：distributed cache；服务端cache：cluster（集群）

普遍用的是客户端cache和服务端cache

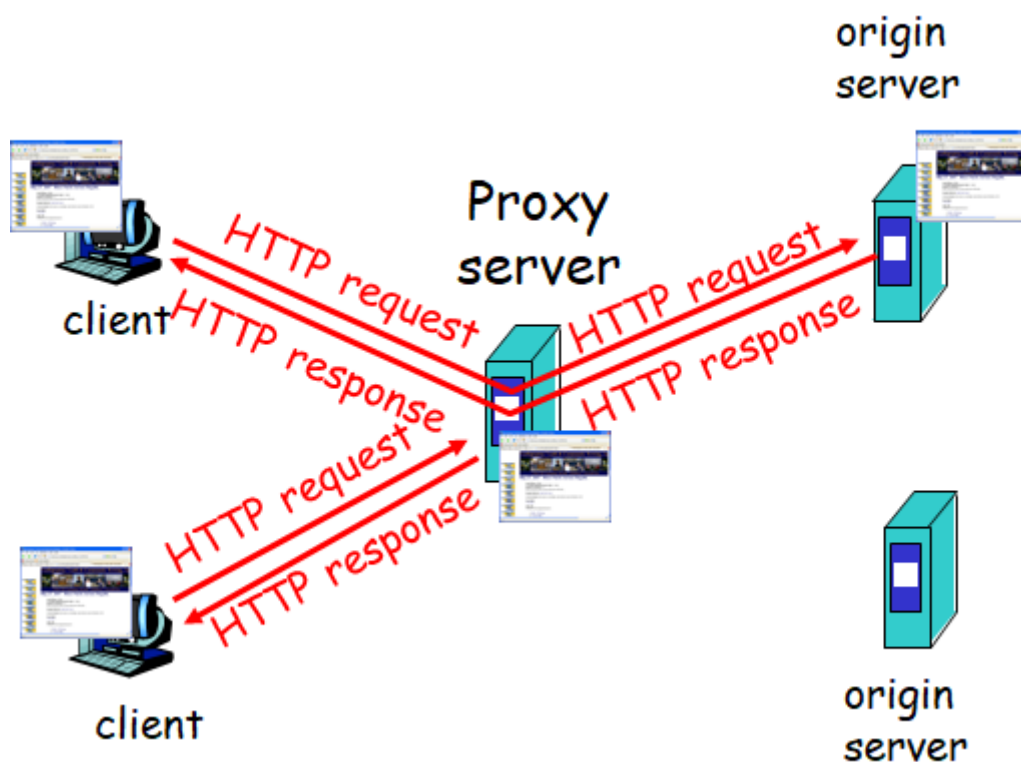
proxy cache

目标:在不涉及源服务器的情况下满足客户端请求

用户设置浏览器:通过缓存访问Web

浏览器将所有的HTTP请求发送到缓存

对象在缓冲中则让缓存返回对象；否则从源服务器缓存请求对象，然后返回对象给客户端



Client Cache: Conditional Get

保存在服务器中的对象自该副本缓存在客户上以后可能已经被修改了。HTTP协议有一种机制，允许缓存器证实它的对象是最新的。这种机制就是条件GET

(conditional GET) 方法。如果：①请求报文使用 GET方法；并且②请求报文中包含一个 “If-Modified-Since;” 首部行。那么，这个HTTP 请求报文就是一个条件GET 请求报文。

缓存器在将对象转发到请求的浏览器的同时，也会在本地缓存了该对象，同时会存储最后修改日期。只有当缓存对象被修改了才从服务器中发送对象，否则直接读取cache中的对象。

distributed cache

许多缓存是合作的

本地访问丢失，缓存链接邻居

通过http或ICP

如果邻居没有数据，则访问源服务器

操作不便，一般采用镜像服务器

server cache:cluster

多台服务器以集群方式构造

内容相同或不同的内容

连接被传输到轻载服务器(缓存)

高并行性、可靠性

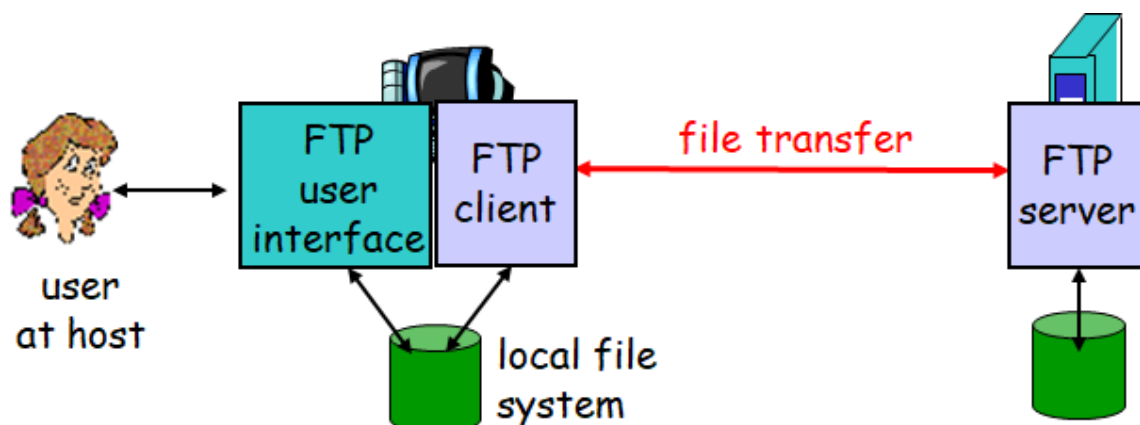
负载均衡需要

所需目标定位算法

受到广泛的采用

三、因特网的电子邮件

1、FTP—— file transfer protocol



向远程主机传输文件

客户机/服务器模型

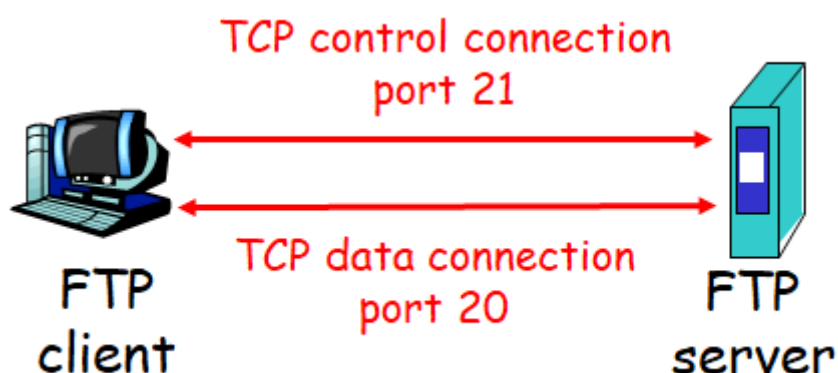
客户端:发起传输(远程传输或远程传输)的端

服务器:远程主机

ftp: RFC 959

FTP server: 21端口，用于控制连接

独立的控制连接和数据连接



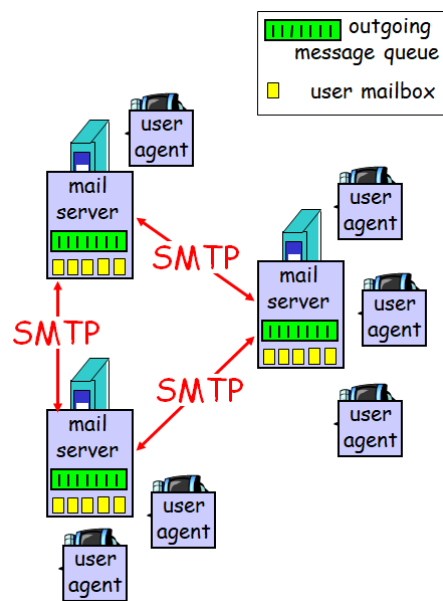
2、电子邮件

三个主要组件:

用户代理 (如浏览器) user agent

邮件服务器 mail server

简单邮件传输协议:SMTP



以下通过Alice发电子邮件给接收方Bob的场景进行说明。

用户代理

用户代理允许用户阅读、回复、转发、保存和撰写报文。微软的 Outlook 和 Apple Mail 是电子邮件用户代理的例子。当Alice 完成邮件撰写时，她的邮件代理向其邮件服务器发送邮件，此时 **邮件放在邮件服务器** 的外出报文队列中。当 Bob 要阅读报文时，他的用户代理在其邮件服务器的邮箱中取得该报文。

邮件服务器

邮件服务器形成了电子邮件体系结构的核心。每个接收方（如Bob）在其中的某个邮件服务器上有一个**邮箱**（mailbox）。Bob 的邮箱管理和维护着发送给他的报文。一个典型的邮件发送过程是：从发送方的用户代理开始，传输到发送方的邮件服务器，再传输到接收方的邮件服务器，然后在这里被分发到接收方的邮箱中。当 Bob 要在他的邮箱中读取该报文时，包含他邮箱的邮件服务器（使用用户名和口令）来鉴别 Bob。

当邮件无法发送成功，在邮件服务器的一个报文队列中保持该报文并在以后再次尝试发送，多次尝试失败后则进行删除并通知给发送方。

SMTP协议在邮件服务器之间发送邮件消息

客户端:发送邮件服务器

服务端:接收邮件服务器

SMTP

使用TCP在客户端和服务端之间可靠地传输邮件消息，**端口号为25**

直连:发送服务器到接收服务器，没有通过中间服务器

命令:ASCII文本

响应:状态码和短语

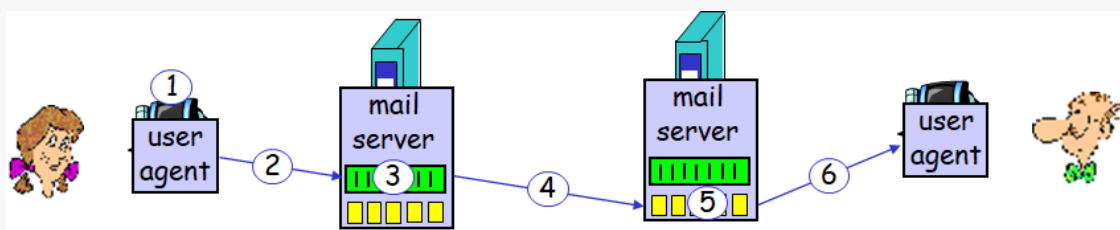
消息必须是7位ASCII码——**只是英文邮件**

SMTP使用的是持续连接

📁 具体的场景说明

Alice发邮件给Bob

- 1 Alice调用她的邮件代理程序并提供Bob的邮件地址（例如bob@some school.edu），撰写报文，然后指示用户代理发送该报文。
- 2 Alice的用户代理把报文发给她的邮件服务器，在那里该报文被放在报文队列中。
- 3 运行在Alice的邮件服务器上的SMTP客户端发现了报文队列中的这个报文，它就创建一个到运行在 Bob 的邮件服务器上的SMTP 服务器的 TCP 连接。
- 4 在经过一些初始SMTP握手后，SMTP客户通过该TCP连接发送Alice的报文。
- 5 在Bob的邮件服务器上，SMTP的服务器端接收该报文。Bob的邮件服务器然后将该报文放入 Bob 的邮箱中。
- 6 在Bob方便的时候，他调用用户代理阅读该报文



3、与HTTP的对比

相同点

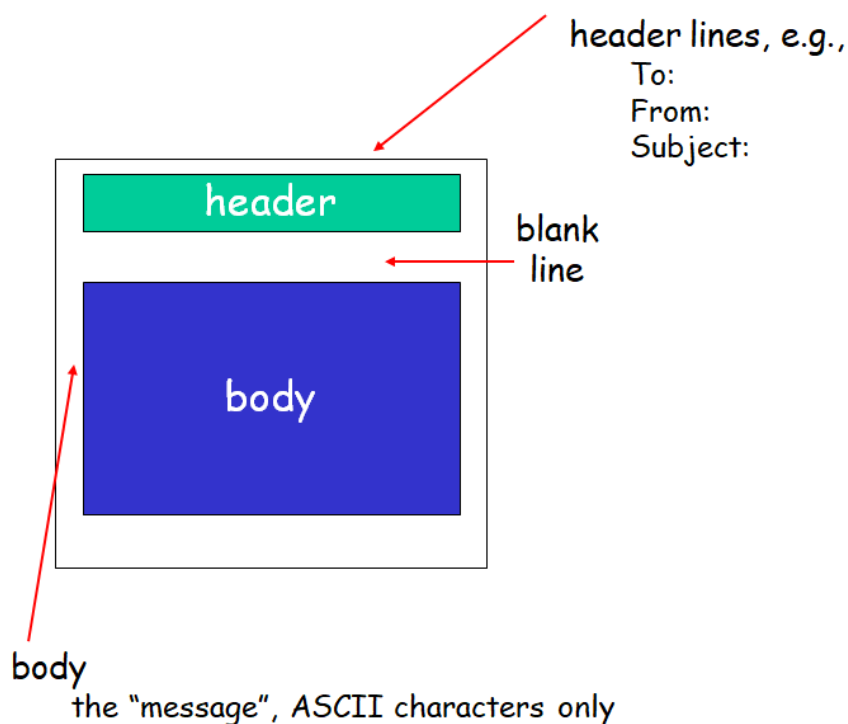
这两个协议都用于从一台主机向另一台主机传送文件

当进行文件传输时，持续的HTTP和SMTP都使用持续连接

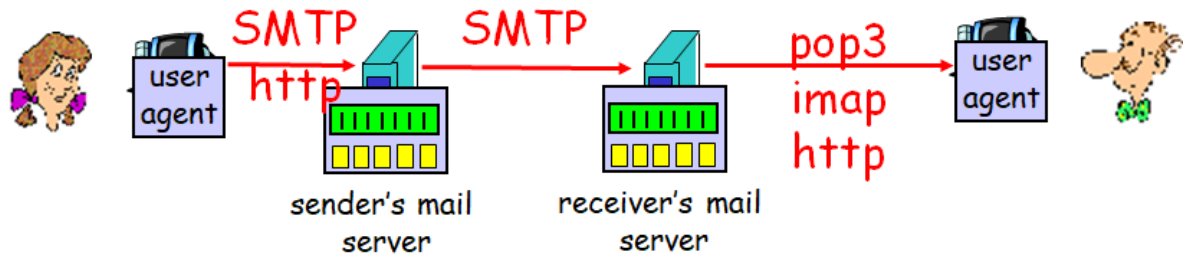
区别

- 1 HTTP是一个拉协议(pull protocol): TCP连接是由想要接收文件的机器发起的; SMTP是一个推协议(push protocol): TCP连接是由要发送该文件的机器发起的
- 2 SMTP要求每个报文使用7bitASCII码格式, HTTP数据不受这种限制
- 3 HTTP将每个对象封装到他自己的HTTP响应报文中, 而SMTP将所有报文对象都放在一个报文中

4、邮件报文格式



5、邮件访问协议



SMTP:发送/存储到接收者的服务器

邮件访问协议:从服务器取回邮件

1 POP3: Post Office Protocol-version 3 第三版邮局协议

授权(代理<—>服务器)和下载, 端口:110

POP3跨会话是 无状态的

2 IMAP: Internet邮件访问协议

更多功能(更复杂), 端口:143

操作服务器上存储的MSGs

IMAP保持用户跨会话的状态: 有状态的

3 HTTP: gmail, Hotmail, Yahoo邮件等。

5、非ASCII数据的MIME扩展

MIME: Multipurpose Internet Mail Extension多用途互联网邮件扩展, RFC 2045, 2056

对于非ASCII文本, 需要在msg中添加额外的头信息

消息头中的其他行声明MIME内容类型

内容类型:提醒接收器使用哪个显示程序

内容传输编码: ASCII编码时使用的编码类型

四、DNS：域名服务

主机的一种标识方法是用它的主机名hostname，例如 www.baidu.com

也可以使用IP地址进行标识，例如192.168.1.202

域名：baidu.com

IP和域名是多对多的关系

ps：DNS属于网络内核的功能，但是放在网络边缘

因特网：“瘦内核，胖边缘”

ATM网络：“胖内核，瘦边缘”

1、DNS提供的服务

DNS是 [1] 一个由分层的DNS服务器实现的分布式数据库 [2] 一个使得主机能够查询分布式数据库的应用层协议

DNS运行在UDP上，使用53号端口

提供的服务如下：

[1] 主机名到IP地址的转换

[2] 主机别名:规范,别名

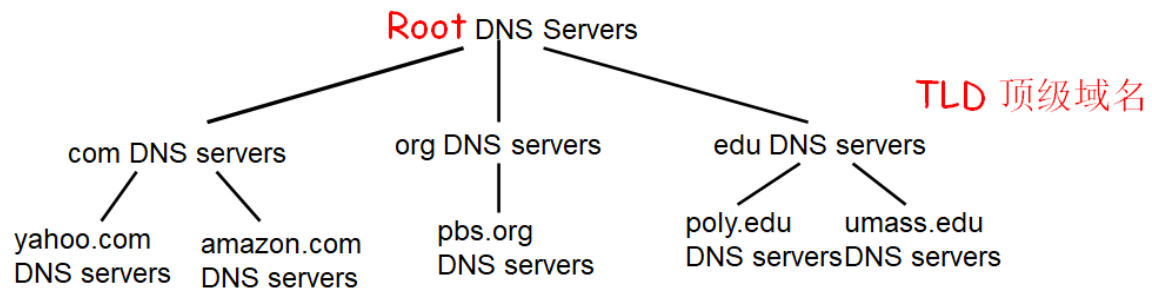
[3] 邮件服务器别名

[4] 负载均衡：DNS在冗余的服务器之间进行负载分配，繁忙的站点，如cnn.com被冗余分布在多个服务器上，每个服务器运行在不同的端系统上，每个有不同的IP地址。一个主机对应有一个IP地址集合，DNS数据库存储着这些IP地址集合。当客户对映射到某地址集合的名字发出一个DNS请求时，该服务器用IP地址的整个集合进行响应，但在每个回答中循环这些地址次序。因为客户通常总是向IP地址排在最前面的服务器发送HTTP请求报文，所以DNS就在所有这些冗余的Web服务器之间循环分配了负载。DNS的循环同样可以用于邮件服务器，因此，多个邮件服务器可以具有相同的别名。

2、DNS工作机制概述

集中式的DNS服务器缺点：单点故障、通信容量、远距离的集中式数据库造成延时长、维护困难

分布式、层次数据库



根域名服务器：400多个，13个组织管理

顶级域(TLD)服务器

注册DNS服务器：记录项一直存在，只要不欠费

本地DNS服务器：不属于上面的层次结构，通常与主机位于同一个局域网中。

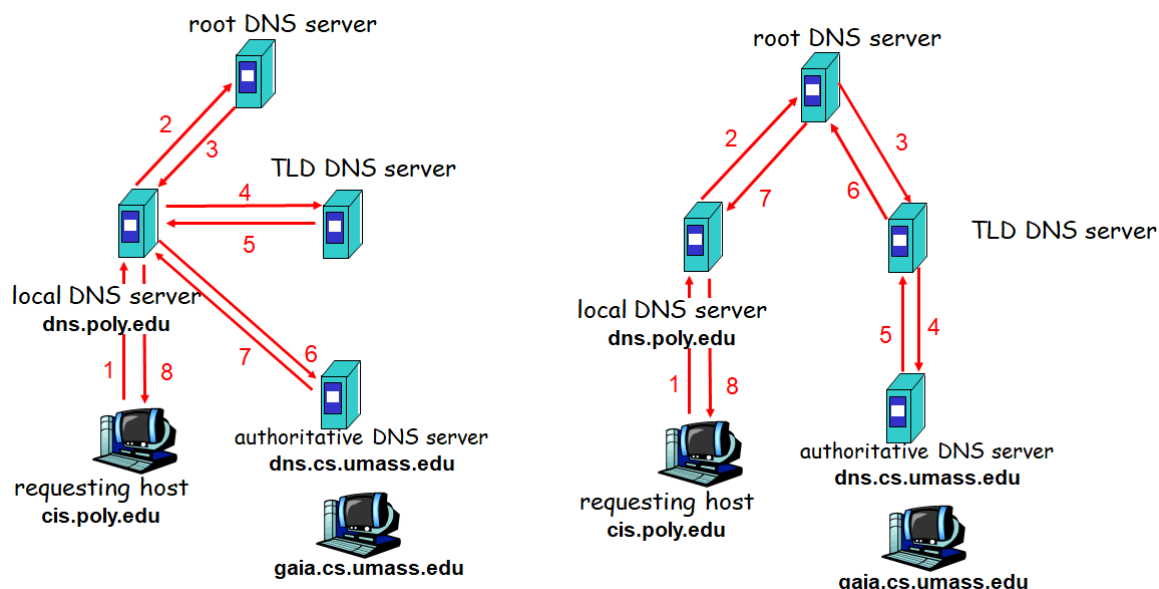
DNS域名解析示例

主机在cis.poly.edu需要IP地址gaia.c.s.umass.edu

1 迭代式查询（接近广度查询）

被联系的服务器用要联系的服务器名称回复“我不知道这个名字，但是问问这个服务器”

2 递归式查询（接近深度查询）



3 混合式查询

DNS缓存和更新记录

在一个请求链中，当某个DNS服务器接收到一个回答，他能缓存包含在该回答中的任何信息

举一个例子，假定主机 `apricot.nyu.edu` 向 `dns.nyu.edu` 查询主机名 `cnn.com` 的 IP 地址。此后，假定过了几个小时，纽约大学的另外一台主机如 `kiwi.nyu.edu` 也向 `dns.nyu.edu` 查询相同的主机名。因为有了缓存，该本地 DNS 服务器可以立即返回 `cnn.com` 的 IP 地址，而不必查询任何其他 DNS 服务器。本地 DNS 服务器也能够缓存 TLD 服务器的 IP 地址，因而允许本地 DNS 绕过查询链中的根 DNS 服务器。事实上，因为缓存，除了少数 DNS 查询以外，根服务器被绕过了

3、DNS记录 and 报文

共同实现DNS分布式数据库的所有DNS服务器存储了资源记录（RR）

RR的格式如下：

(Name, Value, Type, TTL)

TTL是该记录的生存时间，它决定了资源记录应当从缓存中删除的时间。在下面给出的记录例子中，我们忽略掉TTL字段。Name和Value的值取决于Type

1 如果Type =A，则Name是主机名，Value是该主机名对应的IP地址。因此，一条类型为A 的资源记录提供了标准的主机名到IP 地址的映射。例如 (`relay1.bar.foo.com, 145.37.93.126, A`) 就是一条类型A记录。

② 如果Type = NS，则Name是个域（如foo.com），而Value是个知道如何获得该域中主机 IP 地址的权威 DNS 服务器的主机名。这个记录用于沿着查询链来路由 DNS 查询。例如（foo.com, dns.foo.com, NS）就是一条类型为NS的

③ 记录如果Type = CNAME，则Value 是别名为Name的主机对应的规范主机名。该记录能够向查询的主机提供一个主机名对应的规范主机名，例如（foo.com, relay1.bar.foo.com, CNAME）就是一条CNAME类型的记录。

④ 如果Type = MX，则Value是个别名为Name的**邮件服务器**的规范主机名。举例来说，（foo.com, mail.bar.foo.com, MX）就是一条MX记录。MX记录允许邮件服务器主机名具有简单的别名。值得注意的是，通过使用 MX 记录，一个公司的邮件服务器和其他服务器（如它的 Web 服务器）可以使用相同的别名。为了获得邮件服务器的规范主机名，DNS客户应当请求一条 MX 记录；而为了获得其他服务器的规范主机名，DNS客户应当请求CNAME记录。

DNS协议，报文

DNS协议:查询和应答报文，报文格式相同

标识符	标志	12个字节
问题数	回答RR数	
authority RR数	附加RR数	
问题（问题的变量数）		查询的名字和类型字段
回答（资源记录的变量数）		响应查询的RR
authority（资源记录的变量数）		注册服务器的记录
附加信息（资源记录的变量数）		可被使用的附加有用信息

前面12个字节是首部区域header

五、搜索引擎

万维网可以视作一个大的图：页面是一个结点；url是一个边

索引

将页面的关键字作为索引

关键字<-->url

需要三种数据结构

- 1 线性数组：存储发现的url指针和标题/页面指针
- 2 堆：存储可变长度的标题/页面和url
- 3 哈希表：将url哈希成更短的条目，避免重复访问

索引创建具体过程

- 1 第一:搜索（广度优先搜索）

使用递归过程:process_url，输入url，散列url，决定url是否在url_table

如果url在url_table中，选择下一个url，否则访问页面并将url和标题/页面放入堆中，然后哈希url并处理指针

process_url如上所示处理页面中的所有url(超链接)

- 2 第二:索引

对于url_table中的每个条目，提取标题和页面中的**关键字**

将关键字指向对应的url_table项

六、Socket编程

服务器必须在客户端可以发送任何东西之前处于运行状态。

服务器必须有一个socket(门)，通过它接收和发送段segment

类似地，客户端需要一个套接字

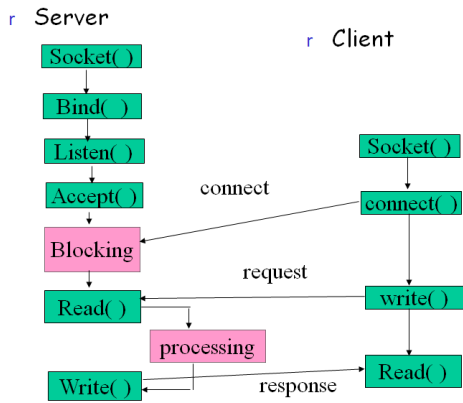
套接字在本地由一个端口号标识

类似于建筑物中的apt #

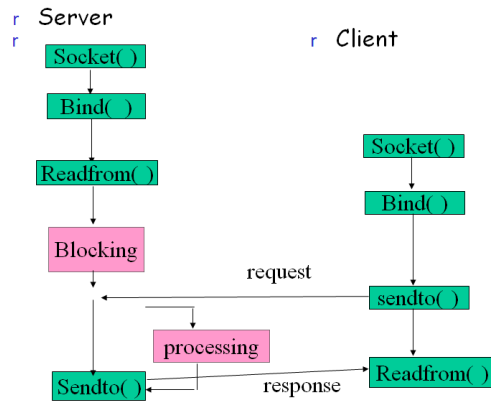
客户端需要知道服务器的IP地址和socket端口号

使用WinSock.h

TCP-based C/S model timing



UDP-based C/S model timing



练习题

1 What are features of the TCP/IP Transport layer? (Choose two.)

path determination

handles representation, encoding and dialog control

uses TCP and UDP protocols ☒

packet switching

reliability, flow control and error correction ☒

2 Which OSI layer defines the functions of a router?

physical

data link

network ☒

transport

session

3 Which type of institution does the domain suffix .org represent?

government

education

network

non-profit ☒

4 What is established during a connection-oriented file transfer between computers? (Choose two.)

a temporary connection to establish authentication of hosts

a connection used for ASCII or binary mode data transfer ☒

a connection used to provide the tunnel through which file headers are transported

a command connection which allows the transfer of multiple commands directly to the remote server system

a control connection between the client and server ☒

5 Which of the following services is used to translate a web address into an IP address?

DNS ☒

WINS

DHCP

Telnet

6 Which part of the URL <http://www.awsb.ca/teacher> gives the name of the domain?

www

http://

/teacher

awsb.ca ☒

7 Using the data transfer calculation $T=S/BW$, how long would it take a 4MB file to be sent over a 1.5Mbps connection?

52.2 seconds

21.3 seconds ☒

6.4 seconds

2 seconds

0.075 seconds

0.0375 seconds

If a network administrator needed to download files from a remote server, which protocols could the administrator use to remotely access those files? (Choose two.)

NFS

ASCII

TFTP ☒

IMAP

FTP ☒

UDP

Which of the following protocols are used for e-mail transfer between clients and servers? (Choose three.)

TFTP

SNMP

POP3 ☒

SMTP ☒

IMAP4 ☒

postoffice