第六章 应用层

刘 轶
北京航空航天大学 计算机学院

本章内容

6.1 套接字编程接口

6.2 域名系统 DNS

6.3 文件传送协议

6.4 万维网WWW

6.5 电子邮件

应用层协议特点

- 应用层协议是为了解决某一类应用问题
  - 由于网络应用的多样性，应用层的协议也种类繁多
- 应用层的许多协议都是基于客户/服务器方式
  - 客户(client)和服务器(server)都是指通信中所涉及的两个应用进程
  - 客户/服务器方式所描述的是进程之间服务和被服务的关系
  - 客户是服务请求方，服务器是服务提供方

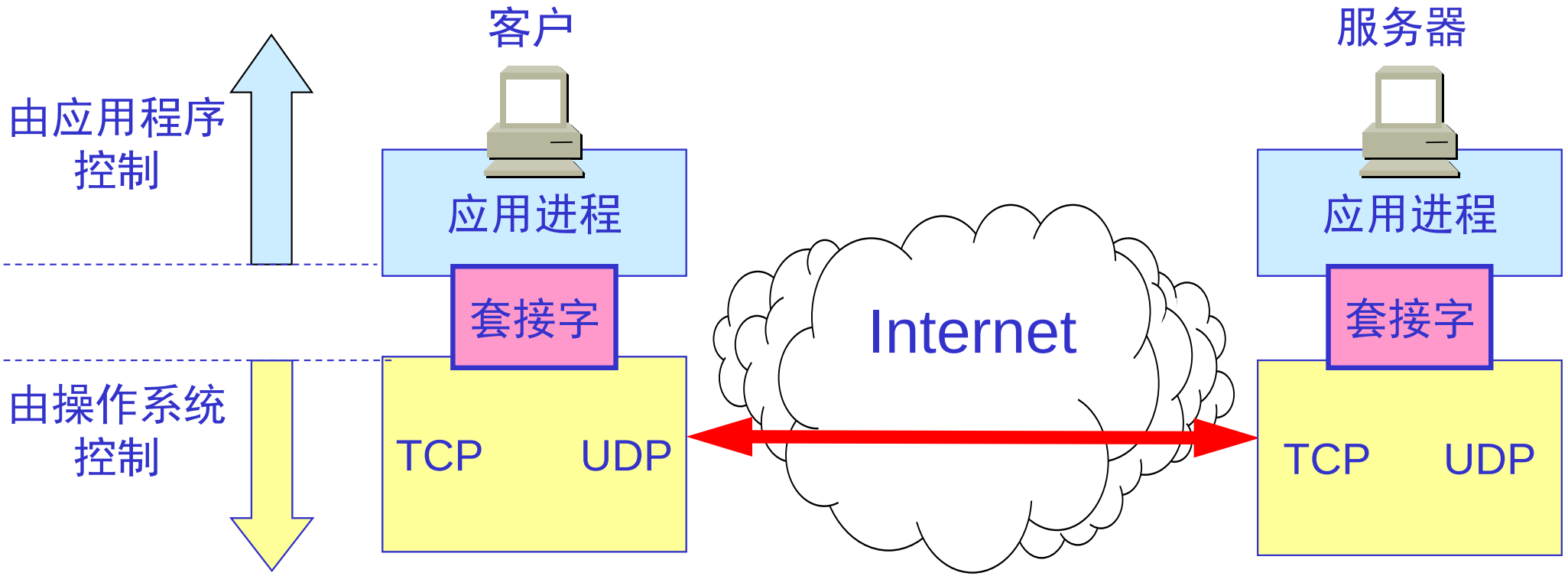
6.1 套接字编程接口

6.1 套接字编程接口

- 网络子系统作为操作系统的一部分，以应用编程接口(API—Application Programming Interface)的形式向应用程序提供调用接口
- 套接字(socket)是最常用的应用层编程接口
  - 名称起源于Berkeley UNIX操作系统
- 经典的socket编程接口采用同步调用方式(又称阻塞式)
  - 例：调用*recv()*时，调用进程将被阻塞，直到收到数据为止
- Windows这种基于事件驱动的系统中，程序的执行由事件驱动，需要异步事件驱动方式的套接字编程接口
  - Microsoft推出WinSock编程接口，在阻塞式调用基础上，增加了异步调用方式
  - 在阻塞式调用模式下，WinSock与经典的socket兼容

6.1 套接字编程接口

应用进程通过套接字接入到网络

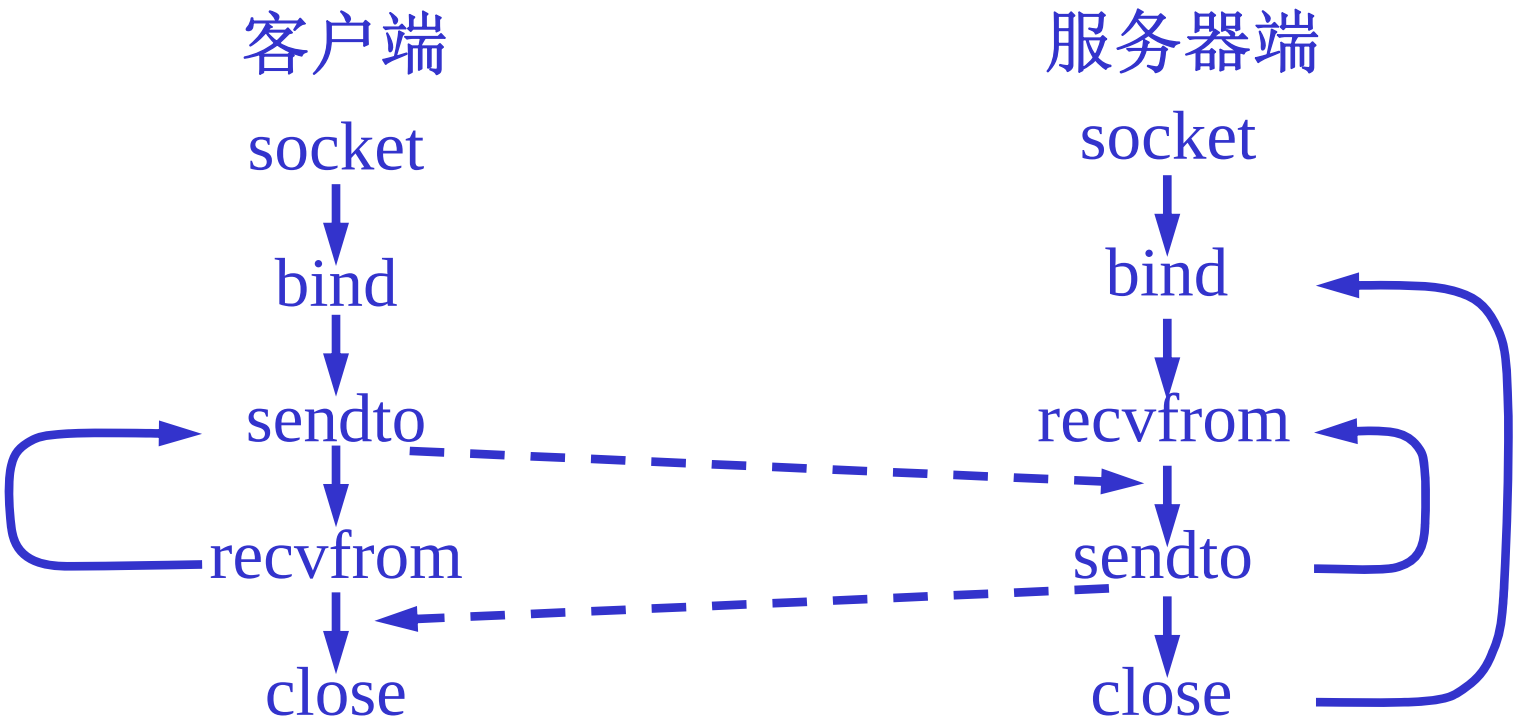


常用socket接口函数

	接口函数	功 能
	SOCKET socket(int af, int type, int protocol)	创建一个socket
	int bind(SOCKET s, struct sockaddr *name, int namelen)	将本地地址与socket绑定
TCP	int listen(SOCKET s, int backlog)	在套接字上监听连接请求
TCP	int connect(SOCKET s, struct sockaddr *name, int namelen)	与name指定的地址建立连接
TCP	SOCKET accept(SOCKET s, struct sockaddr *addr, int *addrlen)	接受与本socket的连接请求
TCP	int send(SOCKET s, char *buf, int len, int flags)	在连接的socket上发送数据
TCP	int recv(SOCKET s, char* buf, int len, int flags)	在连接的或绑定的socket上接收数据
UDP	int sendto(SOCKET s, char* buf, int len, int flags, struct sockaddr* to, int tolen);	向指定的目标地址发送数据
UDP	int recvfrom(SOCKET s, char* buf, int len, int flags, struct sockaddr *from, int* fromlen);	在socket上接收数据并记录源地址

6.1 套接字编程接口

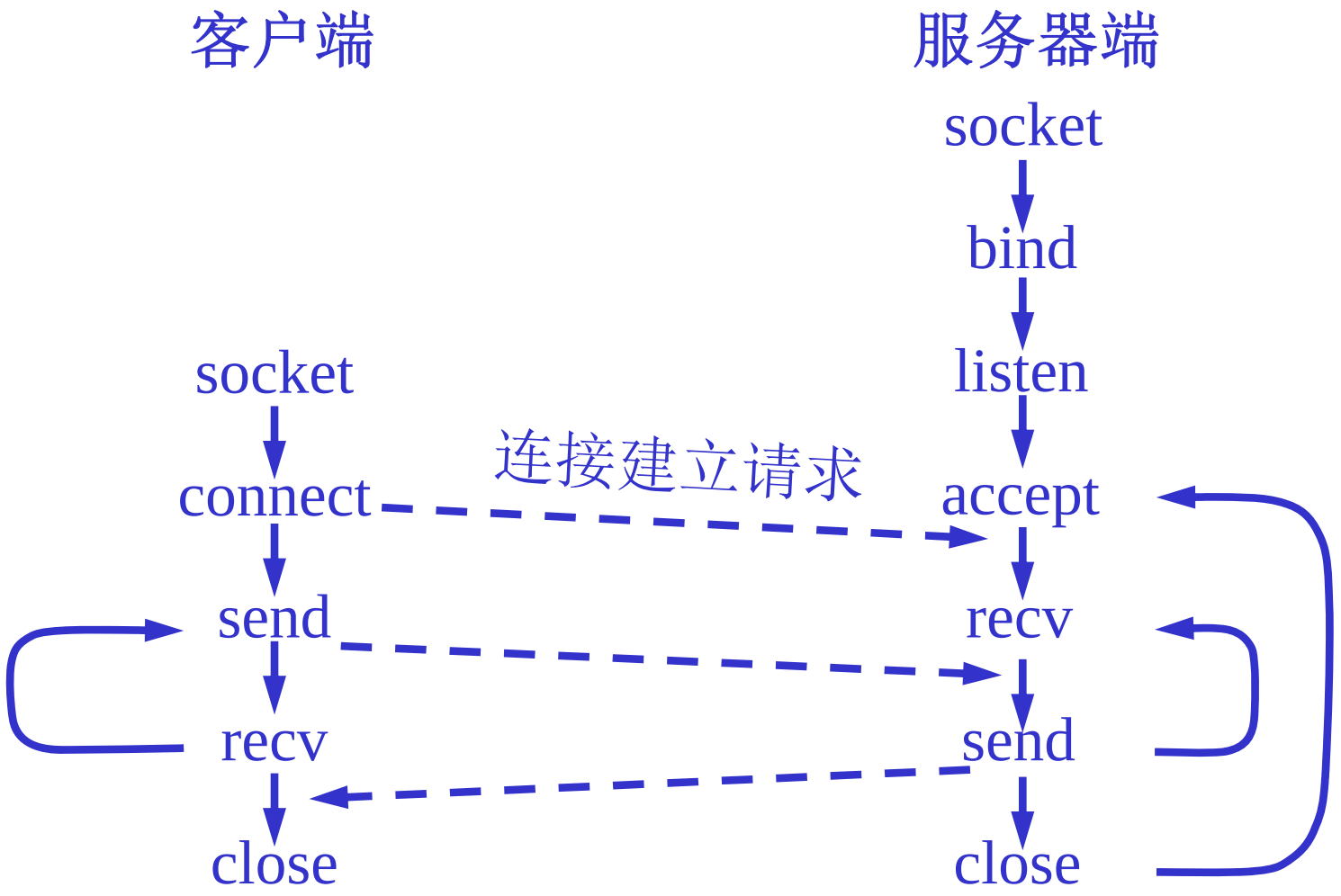
基于UDP的系统调用流程



SOCKET socket(int af, int type, int protocol);

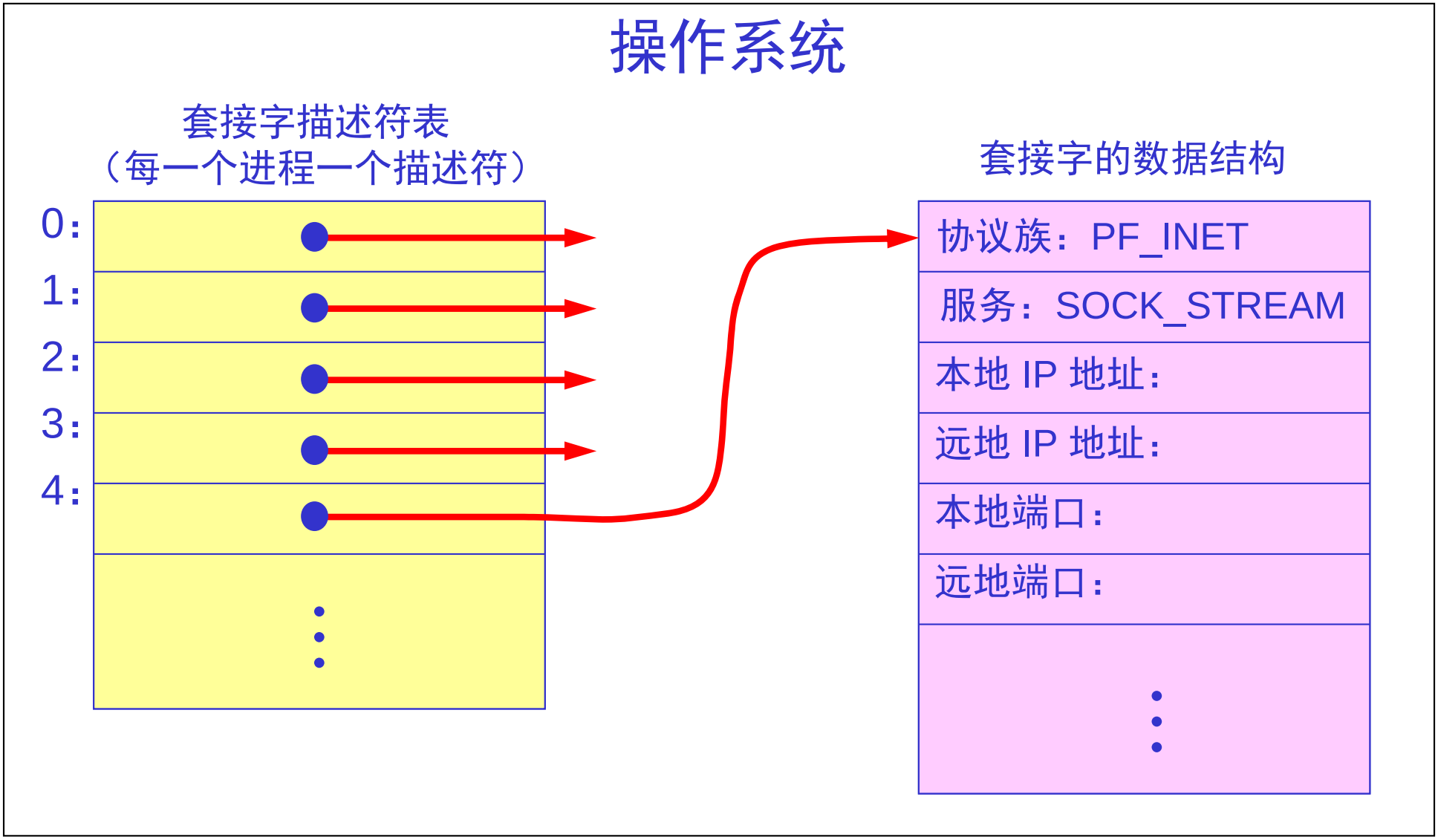
6.1 套接字编程接口

基于TCP的系统调用流程



6.1 套接字编程接口

调用socket()创建套接字



6.2 域名系统 DNS

6.2 域名系统 DNS

一、域名系统概述

- 域名系统DNS(Domain Name System)是Internet使用的命名系统
  - RFC 1034：Domain Names - Concepts and Facilities
  - RFC 1035：Domain Names – Implementation and Specification
- Internet采用层次结构的命名树作为主机的名字，并使用分布式的域名系统DNS
- 多个域名服务器上运行专门的域名服务器程序，完成域名→IP地址的解析(resolve)
- DNS基于UDP协议实现
  - 应用进程需要进行域名解析时，就调用域名解析程序(resolver)，它成为DNS的一个客户
  - 向本地域名服务器发送域名解析请求(UDP报文)，其中包含待解析的域名
  - 本地域名服务器在查找域名后，返回应答报文，其中包含对应的IP地址

6.2 域名系统 DNS

二、Internet的域名结构

- **Internet采用层次树状结构的命名方法**

> **…. 三级域名 . 二级域名 . 顶级域名**

- **顶级域名TLD(Top Level Domain)**
  - 国家顶级域名：如: **.cn** 表示中国，**.us** 表示美国，**.uk** 表示英国，等等
  - 通用顶级域名：

最早的顶级域名

域　名	含　义
.com	公司和企业
.net	网络服务机构
.org	非赢利性组织
.edu	美国专用的教育机构
.gov	美国专用的政府部门
.mil	美国专用的军事部门
.int	国际组织

6.2 域名系统 DNS

新增的通用顶级域名

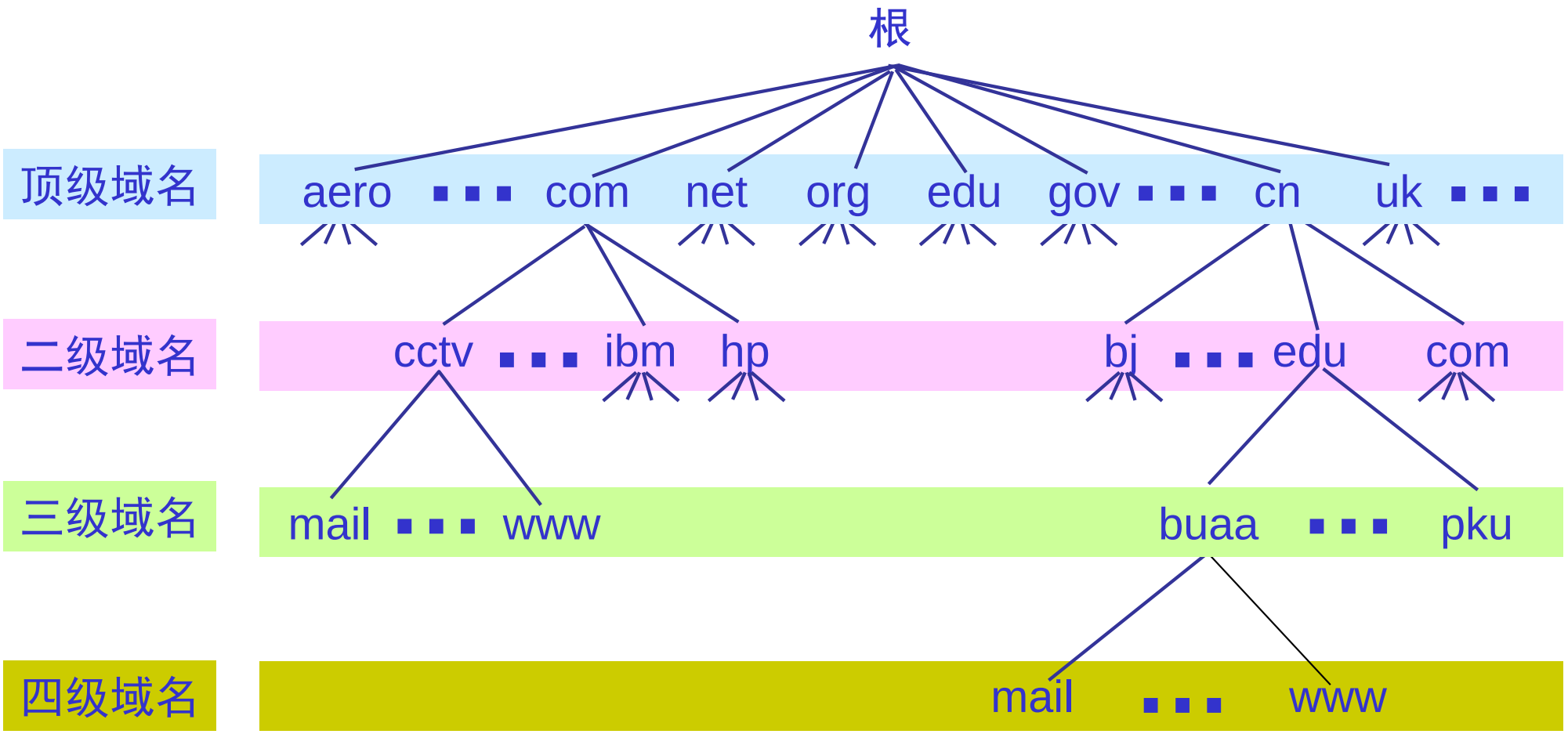
域　名	含　义
.aero	航空运输企业
.biz	公司和企业
.cat	加泰隆人的语言和文化团体
.coop	合作团体
.info	各种情况
.jobs	人力资源管理者
.mobi	移动产品与服务的用户和提供者
.museum	博物馆
.name	个人
.pro	有证书的专业人员
.travel	旅游业

注意：

- 域名不区分大小写
- 域名长度不超过255字符

6.2 域名系统 DNS

Internet的域名空间



6.2 域名系统 DNS

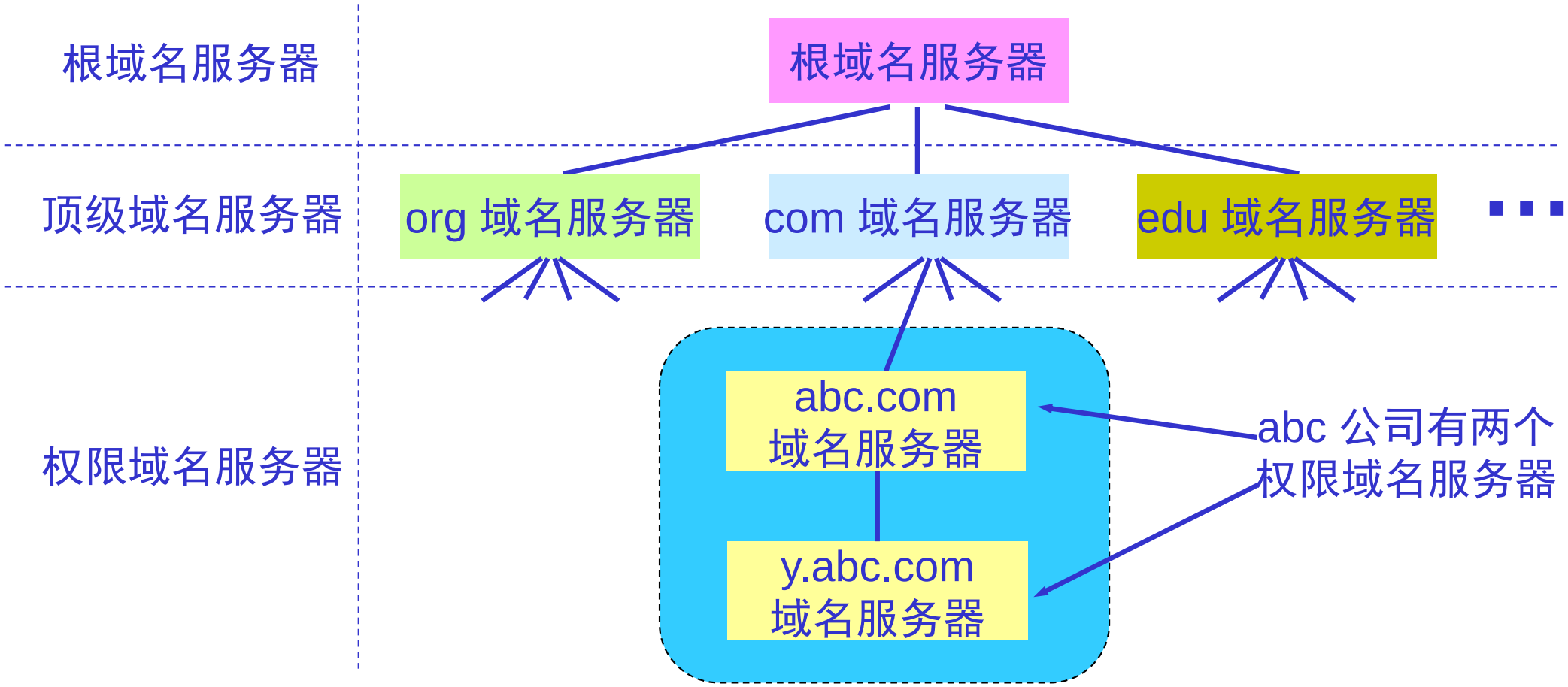
三、域名服务器

- 域名服务器分为四类
  - ① 根域名服务器
    - 根域名服务器知道所有的顶级域名服务器的域名和IP地址
    - 当本地域名服务器无法解析域名时，就求助于根域名服务器
  - ② 顶级域名服务器
    - 负责管理在该顶级域名服务器注册的所有二级域名
  - ③ 权限域名服务器
    - 负责一个区(zone)的域名服务器
  - ④ 本地域名服务器
    - 有时也称为默认域名服务器
- 域名服务器定期把数据复制到几个域名服务器来保存，其中的一个是主域名服务器，其他的是辅助域名服务器

6.2 域名系统 DNS

三、域名服务器

- 树状结构的 **DNS** 域名服务器



- 根域名服务器有**13**套装置，分布在全球**588**个地点

  (1) a.rootservers.net

  (2) b.rootservers.net

  …

  (13) m.rootservers.net

根域名服务器**l.rootservers.net**分布在世界**150**个地点

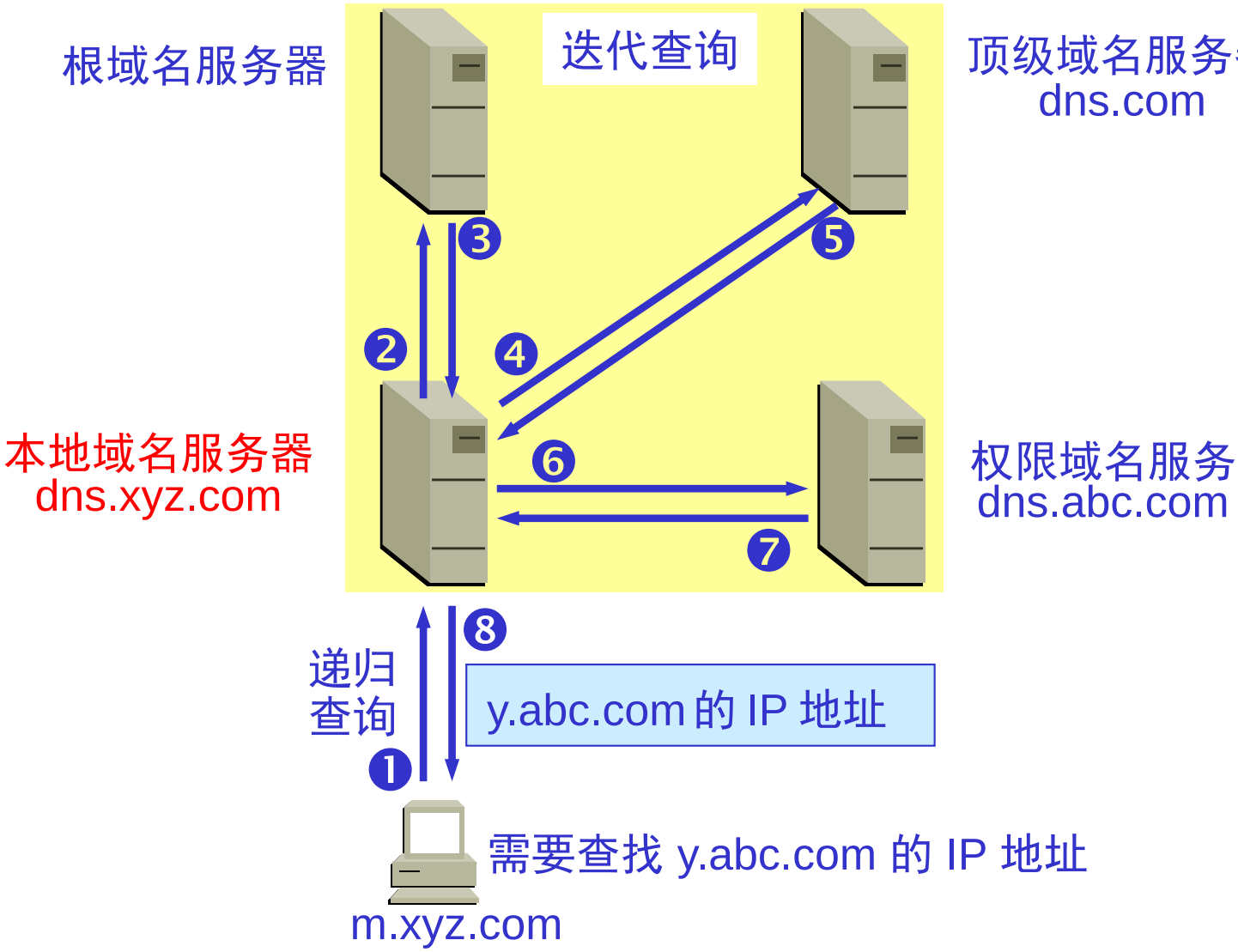
6.2 域名系统 DNS

三、域名服务器

- 主机向本地域名服务器的查询一般采用递归查询
  - 如果本地域名服务器不知道被查询域名的**IP**地址，那么它就以**DNS**客户的身份，向根域名服务器继续发出查询请求报文
- 本地域名服务器向根域名服务器的查询通常是采用迭代查询
  - 当根域名服务器收到本地域名服务器的迭代查询请求报文时
    - 要么给出所要查询的 **IP** 地址
    - 要么告诉本地域名服务器下一步应向哪一个域名服务器查询
  - 本地域名服务器进行后续的查询

6.2 域名系统 DNS

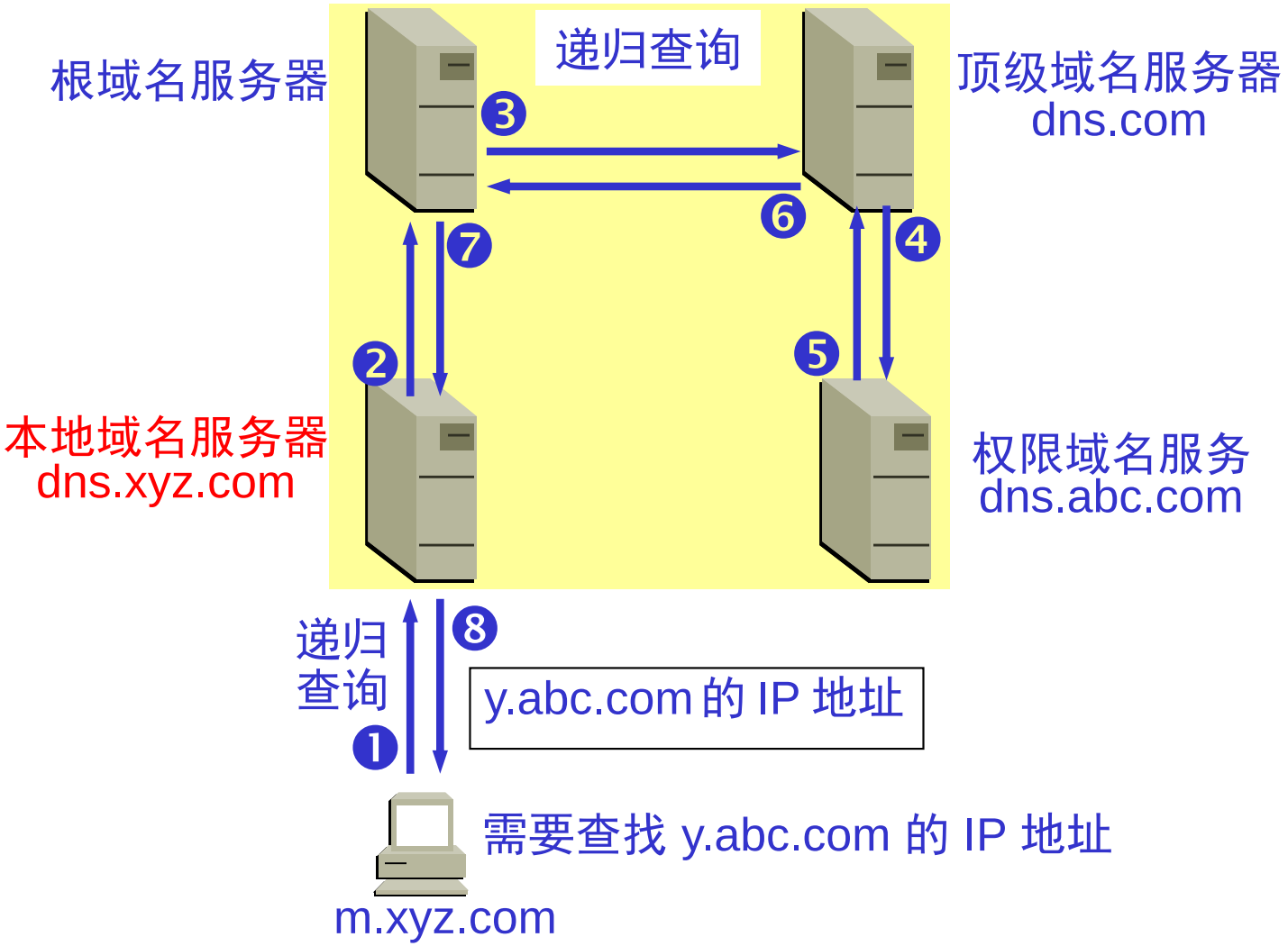
本地域名服务器采用迭代查询



- 域名服务端口号：53
- 为提高域名查询效率，域名服务器上通常设置高速缓存

6.2 域名系统 DNS

本地域名服务器采用递归查询(较少采用)

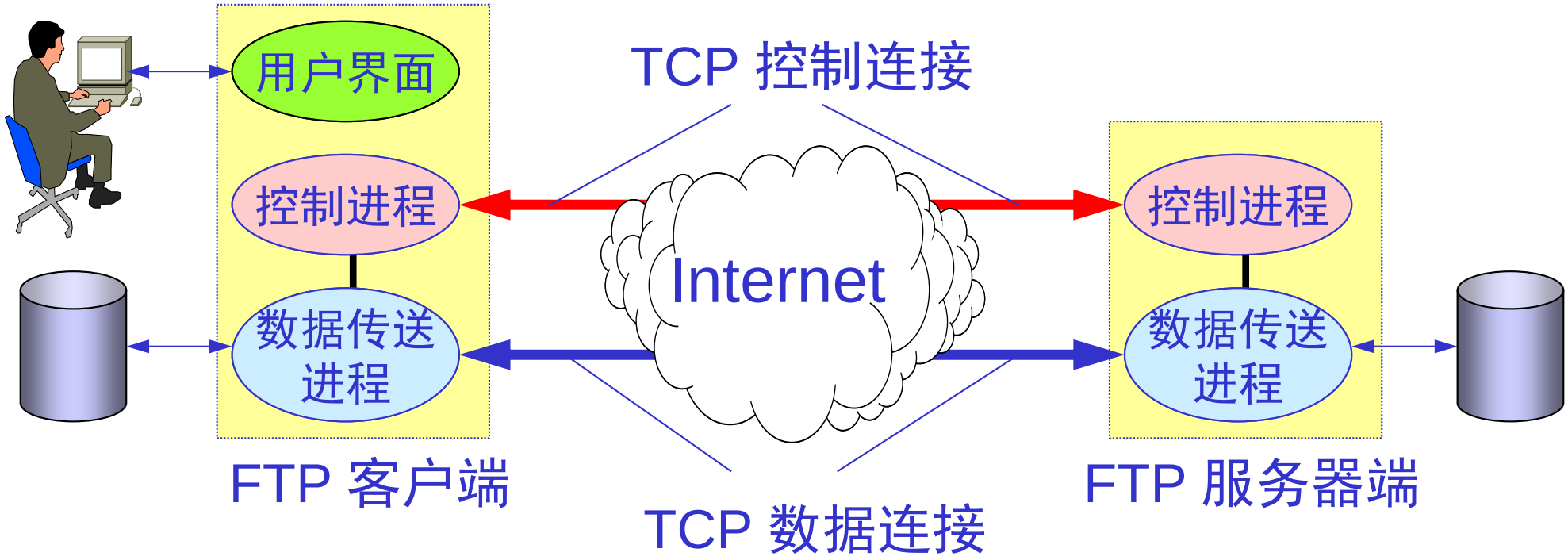


6.3 文件传送协议

6.3 文件传送协议

- **RFC 959：File Transfer Protocol (FTP)**
- **FTP**使用客户/服务器方式
  - 一个**FTP**服务器进程可同时为多个客户进程提供服务
  - **FTP**的服务器进程由两大部分组成
    - 一个主进程，负责接受新的请求
    - 若干个从属进程，负责处理单个请求
- 主进程的工作步骤
  - 打开熟知端口(端口号为**21**)，使客户进程能够连接上
  - 等待客户进程发出连接请求
  - 启动从属进程处理客户进程发来的请求
    - 从属进程对客户进程的请求处理完毕后即终止
    - 从属进程在运行期间根据需要还可能创建其他一些子进程
  - 回到等待状态，继续接受其他客户进程发来的请求
  - 主进程与从属进程的处理是并发地进行

- **FTP使用2个TCP连接：控制连接和数据连接**
- 控制连接
  - 在整个会话期间一直保持打开
  - **FTP**客户发出的传送请求通过控制连接发送给服务器端的控制进程，但控制连接不用来传送文件
- 数据连接
  - 收到**FTP**客户发送来的文件传输请求后，服务器端的控制进程创建“数据传送进程”和“数据连接”
  - 数据传送进程实际完成文件的传送，传送完毕后关闭“数据传送连接”并结束运行



6.4 万维网www

6.4 万维网WWW

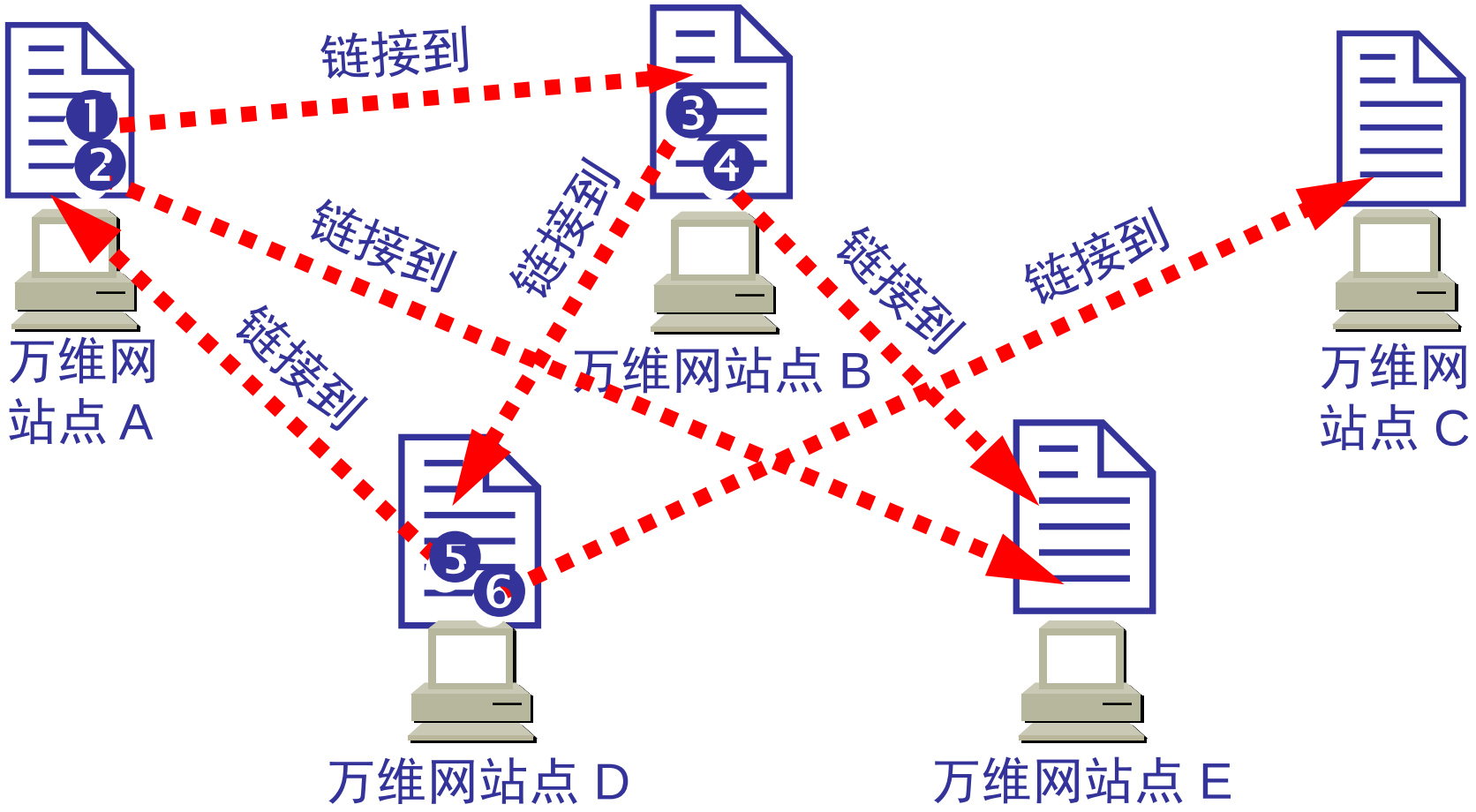
一、万维网概述

- **WWW---World Wide Web**
- **1989年，欧洲核子研究中心提出WWW**
- **1993年，第一个浏览器Mosaic**
- **1995年，Netscape Navagitor上市**
- **万维网是分布式超媒体(hypermedia)系统，它是超文本(hypertext)系统的扩充**
- **要解决的几个问题**
  - ① 怎样标识分布在整个因特网上的万维网文档？ ------**URL**
  - ② 用什么协议实现万维网上各种超链的链接？------**HTTP**
  - ③ 如何存储和表示万维网文档？------**HTML**

6.4 万维网WWW

一、万维网概述

搜索引擎即根据网页中的链接在网络中获取信息



6.4 万维网WWW

二、统一资源定位符URL(Uniform Resource Locator)

- URL是对Internet上资源位置和访问方法的一种简洁表示
- URL的一般形式：

<协议>://<主机>:<端口>/<路径>

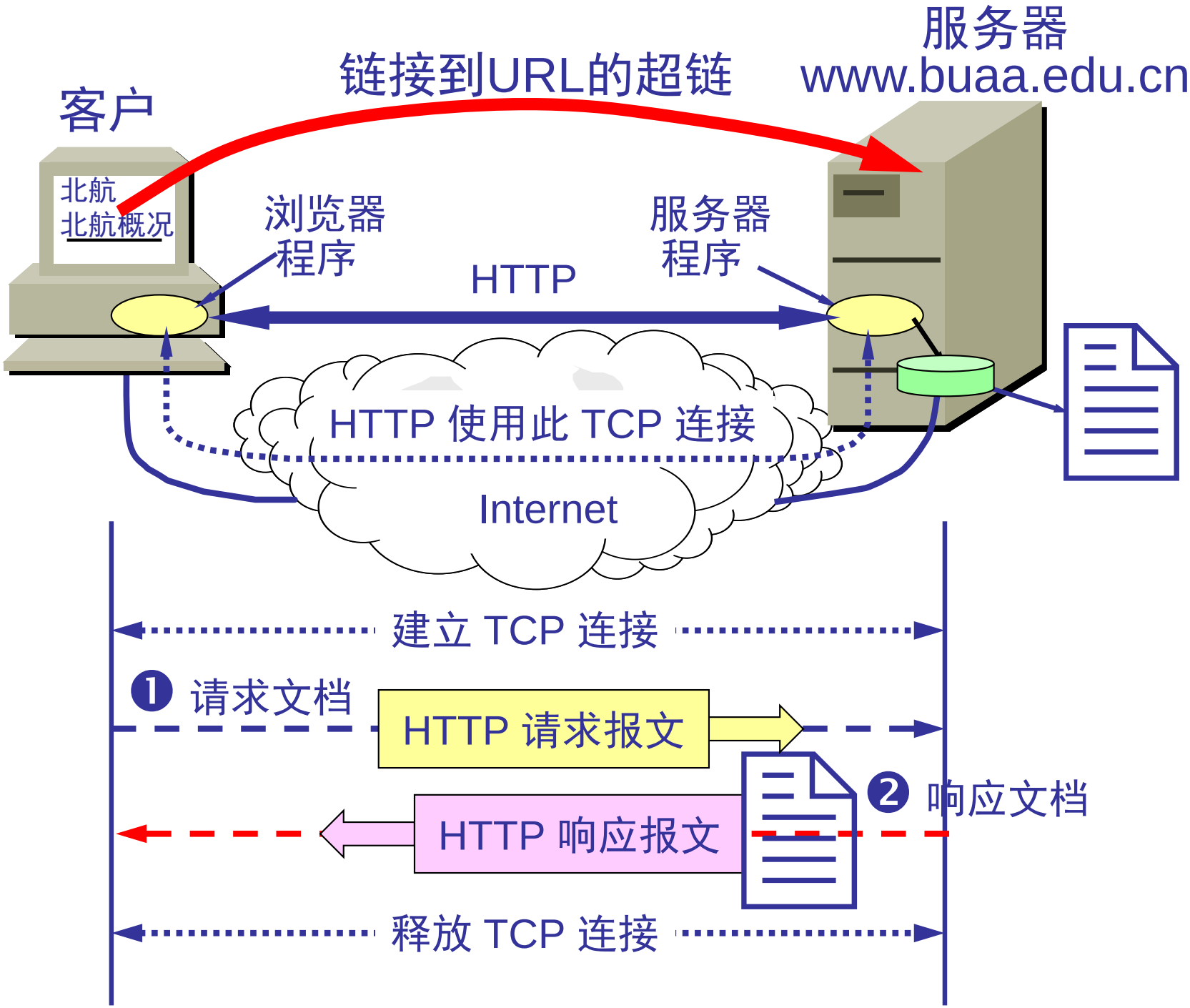
- <主机>为存放资源的主机在Internet中的域名
- <端口>和<路径>有时可省略。使用http协议时，如省略<端口>，则使用熟知端口号80；如省略<路径>，则指主页(home page)
- URL不区分大小写

6.4 万维网WWW

三、超文本传送协议HTTP

- **HTTP---HyperText Transfer Protocol**
- 基于**TCP**协议，是万维网上可靠地交换文件的重要基础
- **HTTP**是面向事务的客户/服务器协议
- **HTTP**是无状态的**(stateless)**
  - 服务器不记录客户端的访问状态
- 基本工作原理
  - **Web**服务的熟知端口号是**80**，服务器通常在该端口上监听
  - 客户端需要请求某个页面时，与服务器建立**TCP**链接，之后请求传送文件，并进行文件的传送，传送完毕后释放**TCP**连接

万维网的工作过程


服务器
www.buaa.edu.cn
链接到URL的超链
客户
北航
北航概况
浏览器
程序
服务器
程序
HTTP
HTTP 使用此 TCP 连接
Internet
建立 TCP 连接
❶ 请求文档
HTTP 请求报文
❷ 响应文档
HTTP 响应报文
释放 TCP 连接

6.4 万维网WWW

很多Web服务器采用动态网页技术，即网页不是静态存储在服务器上，而是动态生成

鼠标点击超链接后发生的事件

(1) 浏览器分析超链指向页面的 URL

(2) 浏览器向DNS请求解析 www.buaa.edu.cn的IP地址

(3) 域名系统DNS解析出北航Web服务器的IP地址

(4) 浏览器与服务器建立TCP连接

(5) 浏览器发出取文件命令：

GET /chn/yxsz/index.htm

(6) 服务器给出响应，把文件index.htm发给浏览器

(7) TCP 连接释放

(8) 浏览器显示“北航概况”文件index.htm中的所有文本

HTTP/1.0每传送一个文件都需要建立一次TCP连接；HTTP/1.1进行了改进

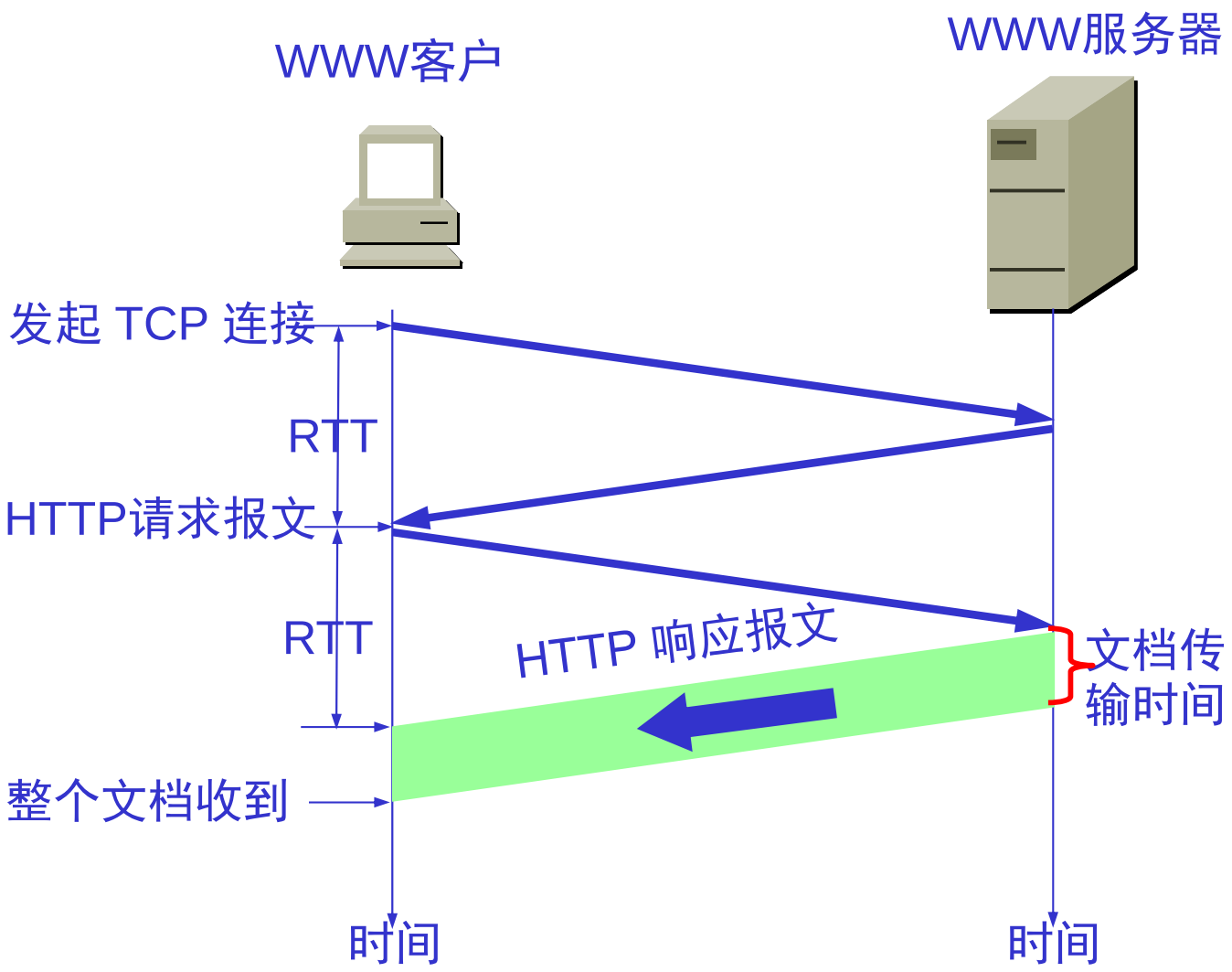
请求一个万维网文档所需的时间

- 首先建立TCP连接，需3次握手
- 在2次握手后， 第3次握手报文的数据部分可传送HTTP请求报文
- 请求文档所需时间

  文档传输时间

  +

  2倍RTT时间
- HTTP/1.0的主要缺点
  - 每请求一个文档都需要建立一次TCP连接
  - 一个Web页面常常包含数量众多的文件



6.4 万维网WWW

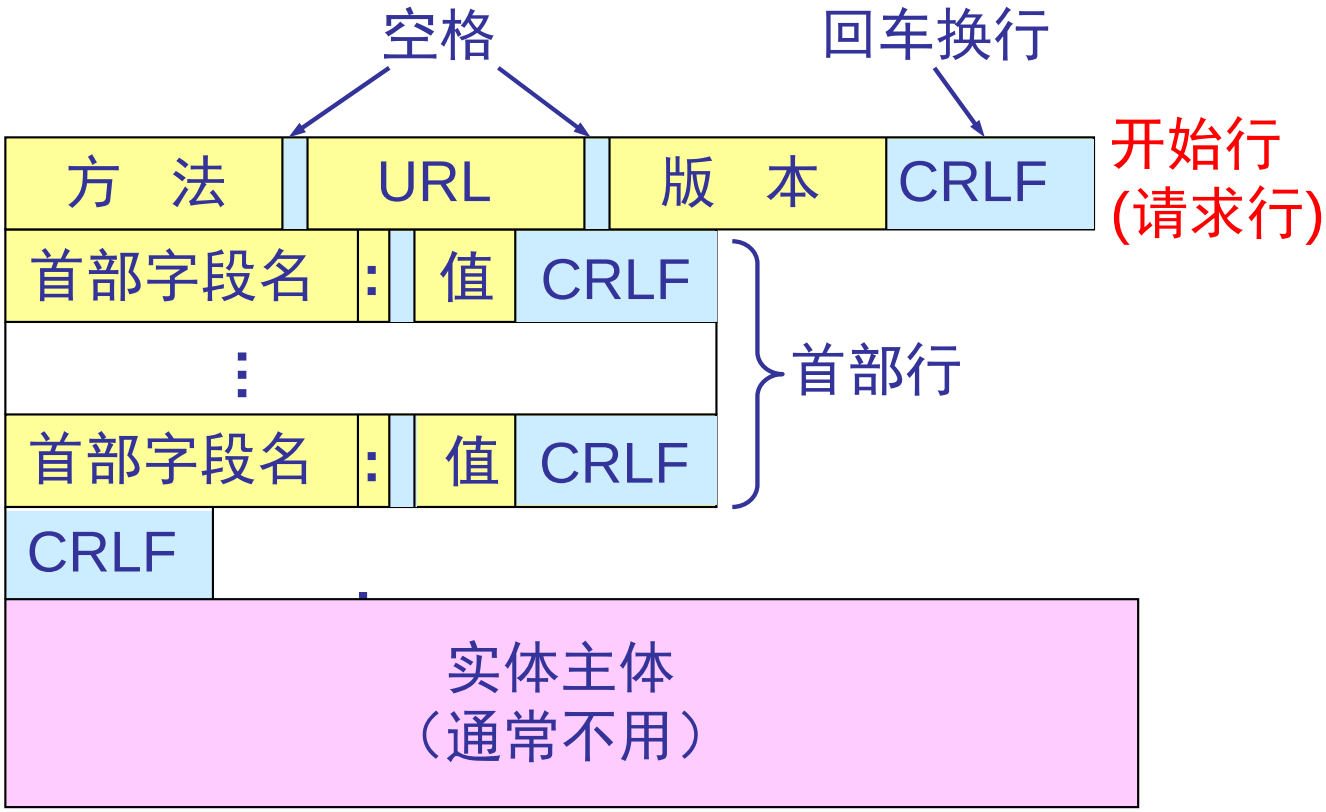
三、超文本传送协议HTTP

- **HTTP/1.1协议使用持续连接(persistent connection)**
  - 服务器发送响应后，在一段时间内保持连接，使客户与服务器可以继续传送后续的**HTTP**请求报文和响应报文
  - 这并不局限于传送同一个页面上链接的文档，而是只要这些文档都在同一个服务器上就行
  - 目前主流的浏览器都默认支持**HTTP/1.1**
- **HTTP的报文类型：两类报文**
  - 请求报文——从客户向服务器发送请求报文
  - 响应报文——从服务器到客户的回答
  - **HTTP**是面向正文的**(text-oriented)**，报文中的字段都是**ASCII** 码串，因而每个字段的长度都是不确定的

6.4 万维网WWW

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

- 方法
  - 是面向对象技术中使用的专门名词
  - 方法实际上就是一些命令
- **URL**：所请求的资源的**URL**
- 版本：**HTTP**的版本



HTTP请求报文

6.4 万维网WWW

HTTP 请求报文的一些方法

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

6.4 万维网WWW

- **HTTP**应答报文的开始行是状态行
- 状态行包括三项内容：**HTTP**的版本、状态码、解释状态码的简单短语

- 状态码都是三位数字

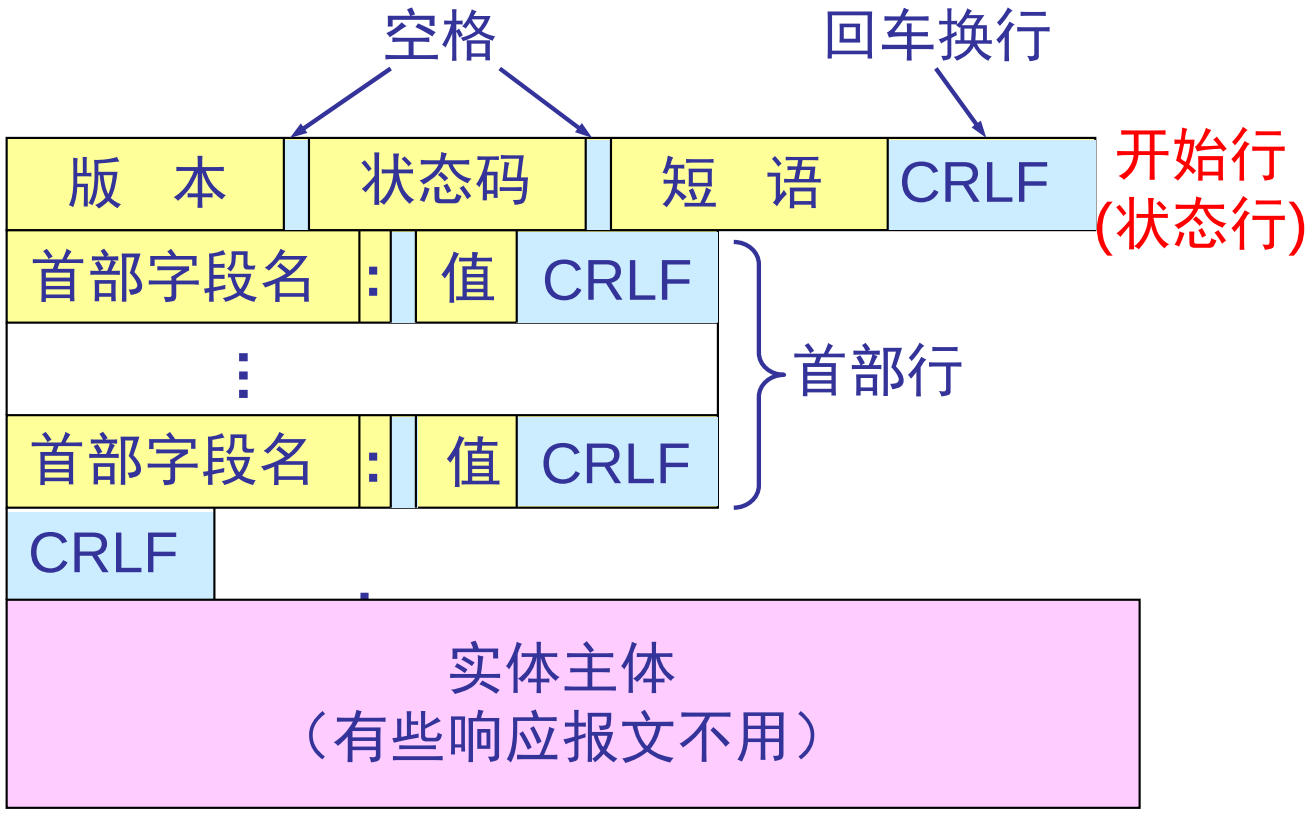
1xx：表示通知信息的，如请求收到了或正在进行处理

2xx：表示成功，如接受或知道了

3xx：表示重定向，表示要完成请求还必须采取进一步的行动

4xx：表示客户的差错，如请求中有错误的语法或不能完成

5xx：表示服务器的差错，如服务器失效无法完成请求



HTTP应答报文

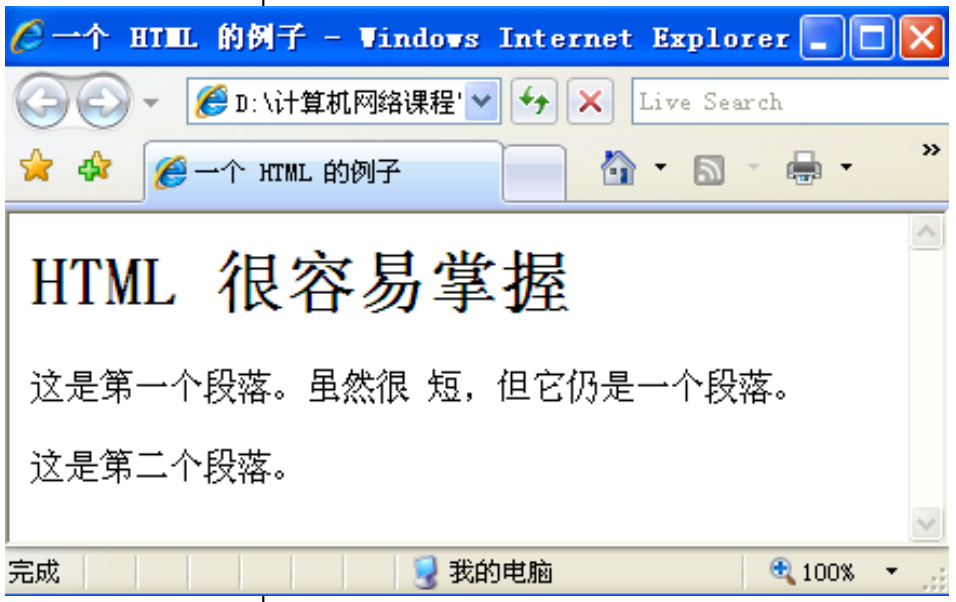
6.4 万维网WWW

四、万维网的文档

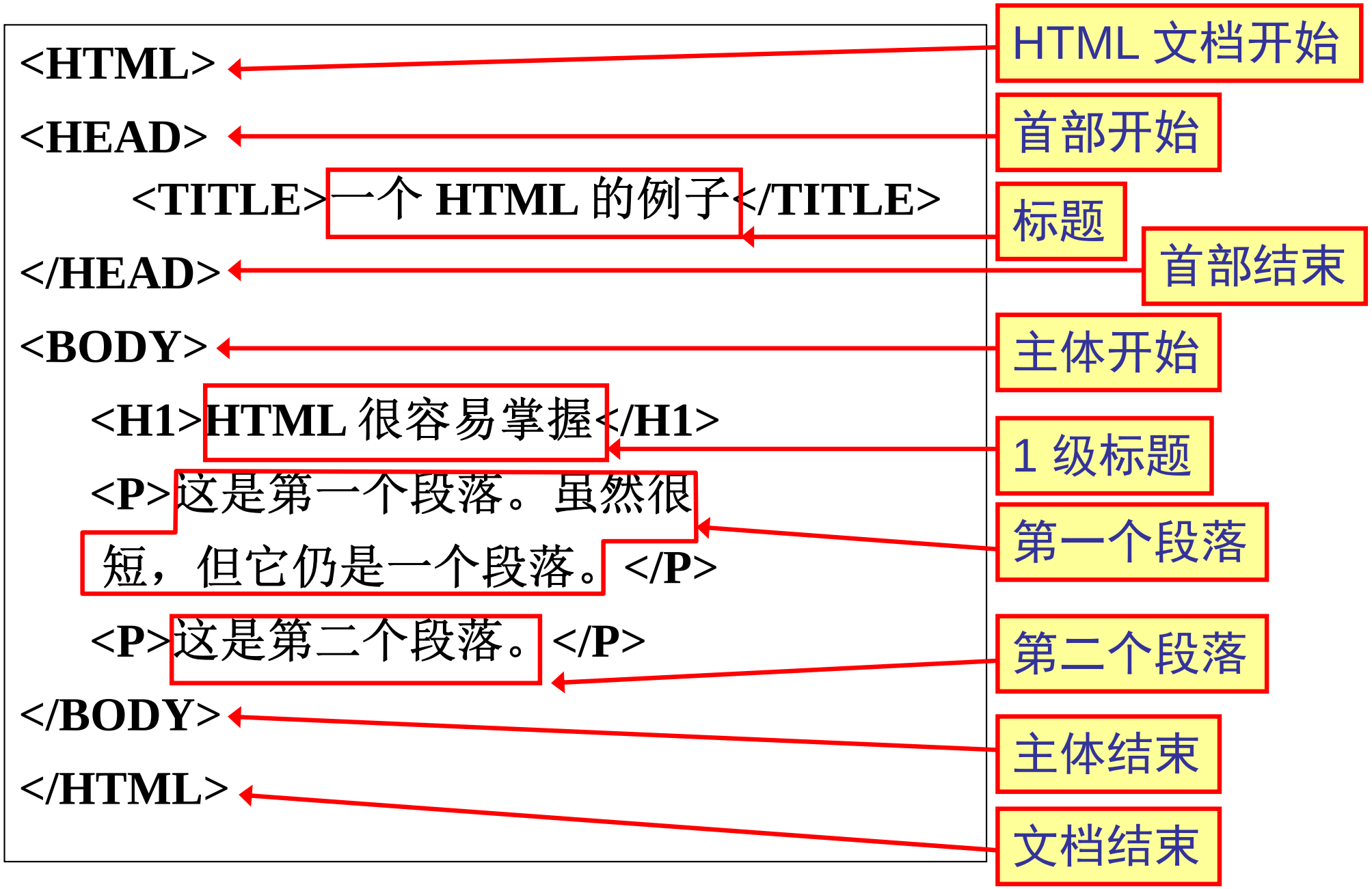
- **HTML---HyperText Markup Language**
- **RFC 1866：Hypertext Markup Language - 2.0**
- **HTML定义了许多用于排版的命令，即标签(tag)**
  - 如：<I>表示后面开始用斜体排版，</I>表示斜体结束
- **HTML把各种标签嵌入到万维网的页面中，构成HTML文档**
  - **HTML文档为文本格式**
- 浏览器从服务器读取 **HTML文档后，按照其中嵌入的各种标签，根据显示器尺寸和分辨率显示页面**
  - 仅当 **HTML** 文档是以**.html** 或 **.htm** 为后缀时，浏览器才对此文档的各种标签进行解释

HTML文档中标签的用法

```
<HTML>
<HEAD>
    <TITLE>一个 HTML 的例子</TITLE>
</HEAD>
<BODY>
    <H1>HTML 很容易掌握</H1>
    <P>这是第一个段落。虽然很
     短，但它仍是一个段落。</P>
    <P>这是第二个段落。</P>
</BODY>
</HTML>
```

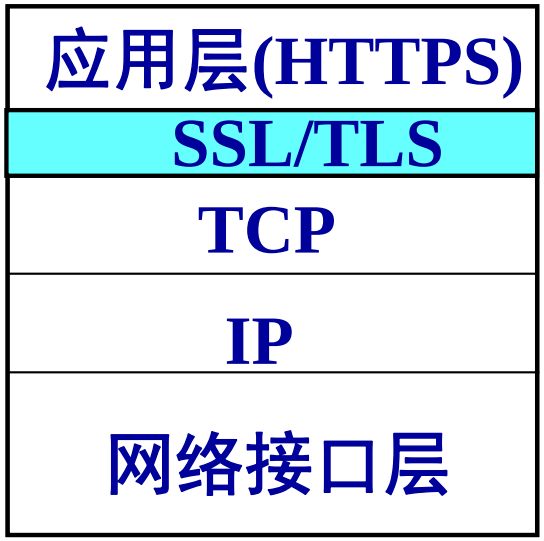


HTML文档中标签的用法

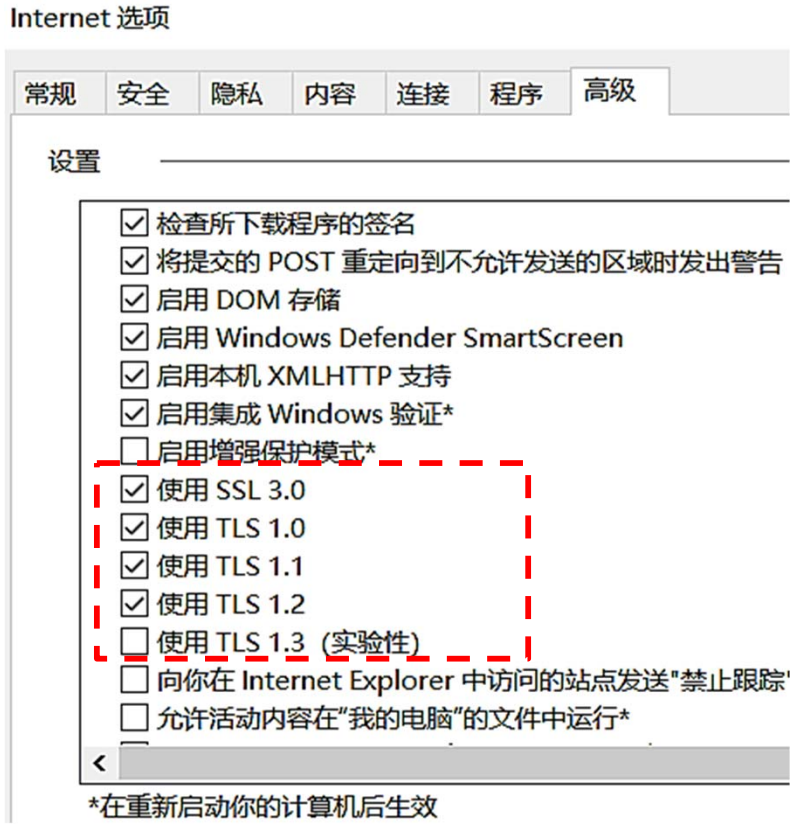


五、安全的HTTP协议：HTTPS

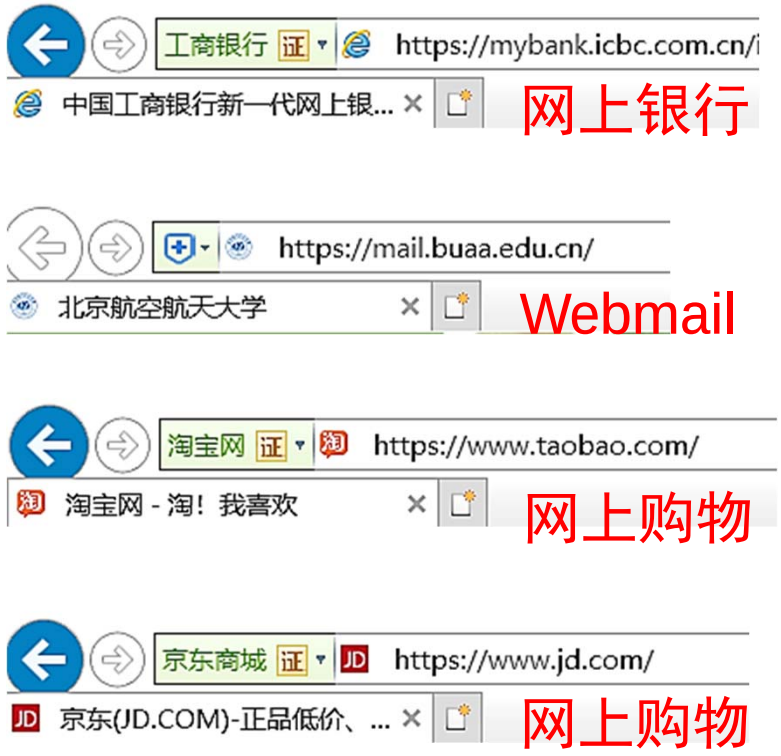
- **HTTPS：安全的HTTP协议，建立在SSL基础上**
- **SSL/TLS已被浏览器广泛支持，很多Web应用利用HTTPS协议实现安全传输**
- **HTTPS的熟知端口号为443，而不是80**



协议层次



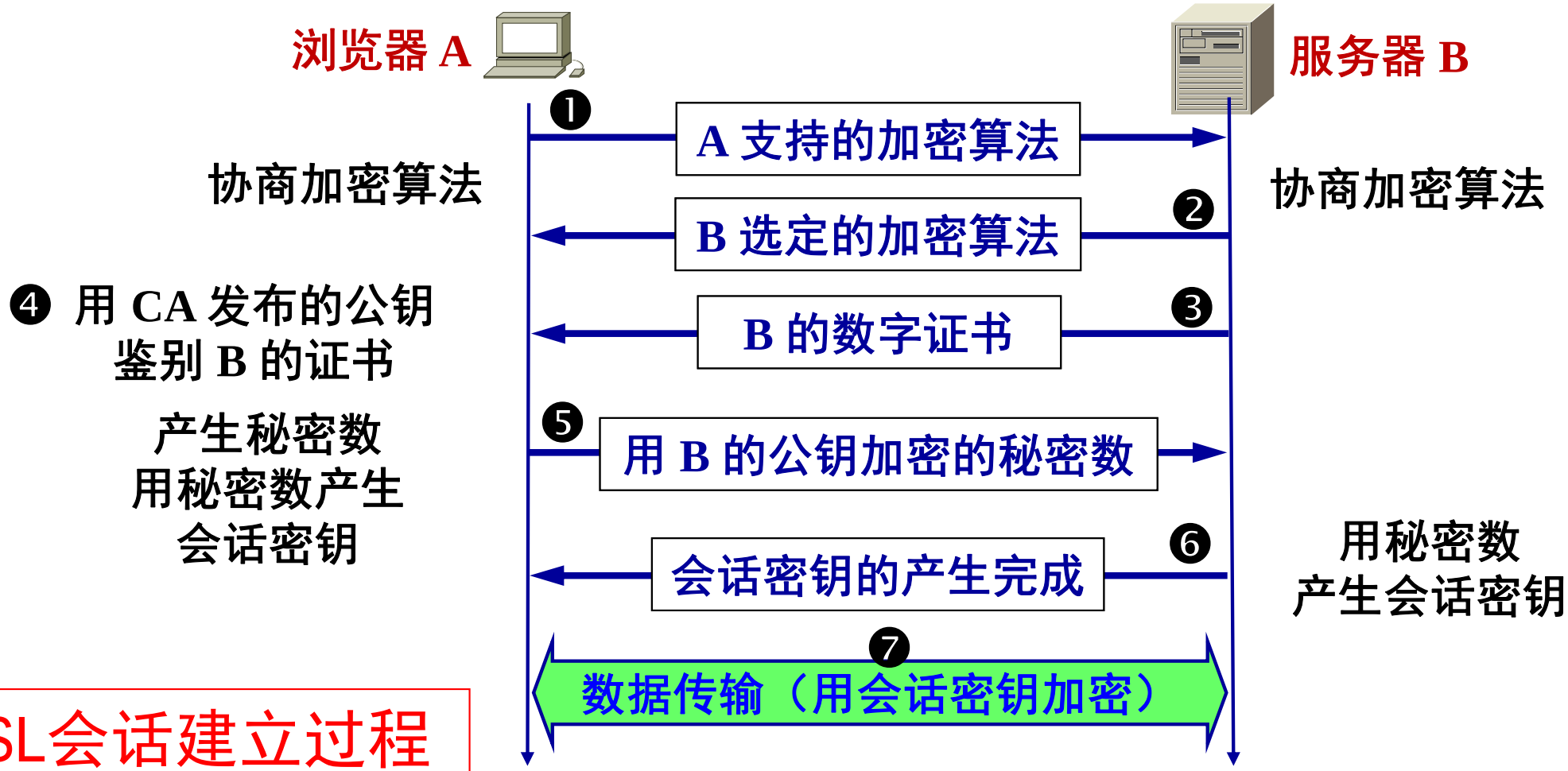
浏览器支持SSL/TLS



基于https的典型应用

五、安全的HTTP协议：HTTPS

- SSL简介
  - Secure Socket Layer，为TCP协议提供信息加密和完整性
  - Netscape于1994年开发，1996年发布 SSL 3.0，1999年IETF在 SSL 3.0 基础上推出了TLS (Transport Layer Security)



SSL会话建立过程

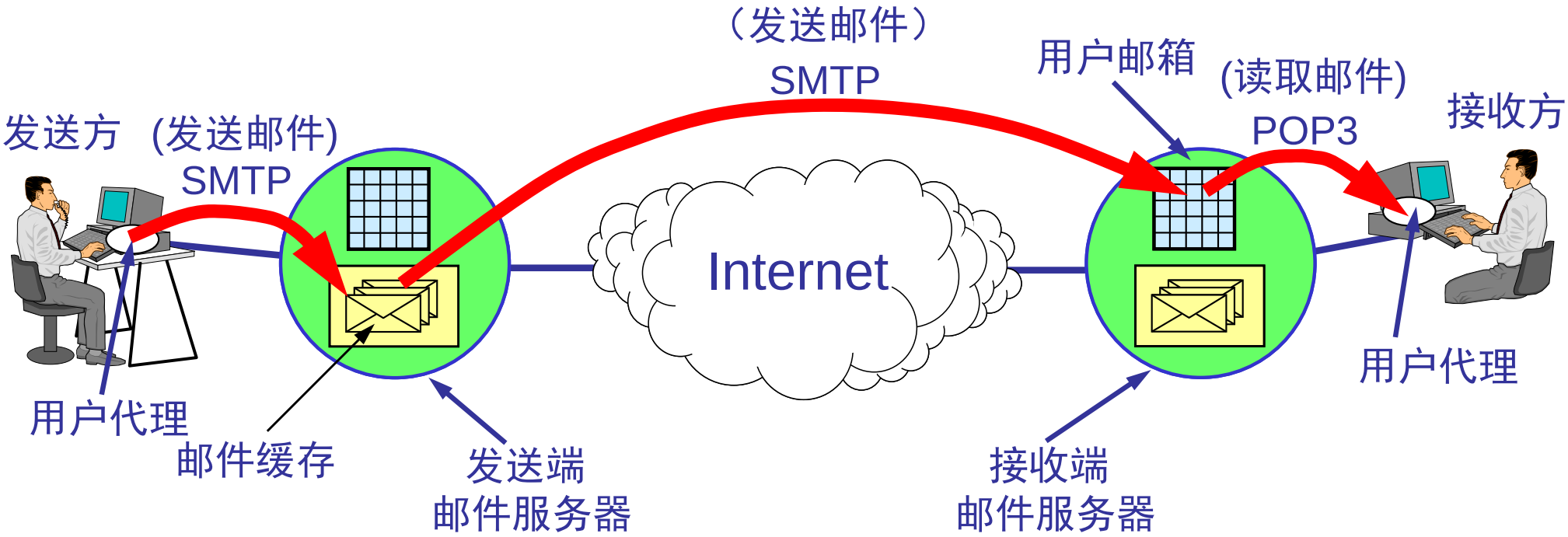
6.5 电子邮件

6.5 电子邮件

一、概述

- 发送邮件的协议：**SMTP**
  - **RFC 2821(RFC 821)：Simple Mail Transfer Protocol**
  - **RFC 2822(RFC 822)：Internet Message Format**
- 早期邮件只支持**7bit ASCII**编码，**1993**年提出了**MIME**标准
  - **RFC 2045 --- 2049**
  - 在邮件首部中说明了数据类型(如文本、声音、图像、视像等)，可在邮件中多种类型的数据
- 读取邮件的协议：**POP3**和**IMAP**
  - **POP3---Post Office Protocol**
  - **IMAP---Internet Message Access Protocol**

电子邮件的最主要的组成构件



6.5 电子邮件

- 电子邮件系统的两种实体：
  - 用户代理UA(User Agent)
    - 是用户与电子邮件系统的接口，即电子邮件客户端软件
    - 用户代理的功能：撰写、显示、处理和通信
  - 邮件服务器
    - 用于发送和接收邮件，并向发信人报告传送结果(已交付、被拒绝、丢失等）
    - 邮件服务器按照客户/服务器方式工作
    - 使用发送和读取两个不同的协议，即发送邮件的SMTP协议和客户端读取邮件的POP3协议
    - 一个邮件服务器既可以作为客户，也可以作为服务器

6.5 电子邮件

二、简单邮件传送协议 SMTP

- 简介
  - SMTP规定了两个相互通信的SMTP进程之间应如何交换信息
  - SMTP使用客户/服务器方式
    - 负责发送邮件的SMTP进程就是SMTP 客户
    - 负责接收邮件的SMTP进程就是SMTP服务器
  - SMTP定义了14条命令和21种应答信息
    - 每条命令用4个字母组成
    - 每一种应答信息一般只有一行信息，由一个3位数字的代码开始，后面附上(也可不附上)简单的文字说明
- SMTP通信的三个阶段

① 连接建立：连接在发送主机的SMTP客户和接收主机的SMTP服务器之间建立，不使用中间邮件服务器

② 邮件传送

③ 连接释放：邮件发送完毕后，SMTP释放TCP 连接

6.5 电子邮件

三、电子邮件的信息格式

- **RFC 2822(RFC 822)：Internet Message Format**
- 一个电子邮件分为信封和内容两大部分
- **RFC 822只规定了邮件内容中的首部(header)格式，而对邮件的主体(body)部分则由用户自由撰写**
- 首部中的主要字段
  - **To：后面填入一个或多个收件人的e-mail地址**
  - **Subject：邮件的主题，反映了邮件的主要内容**
  - **Cc：抄送(Carbon copy)，表示给某人发送一个邮件副本**
  - **From：发信人的电子邮件地址**
  - **Date：发信日期**
  - **Reply-to：对方回信地址**

四、MIME

- **MIME---Multi-purpose Internet Mail Extension**
- **SMTP的问题**
  - 只能传送7位ASCII码
  - 不能传送二进制数据：程序、图片、音乐、东方语言、…
- **MIME的思路**
  - 继续使用目前的[RFC 822]格式
  - 对二进制数据进行编码，将其转换为7位ASCII码
  - 邮件首部中增加字段，定义数据类型和编码规则

6.5 电子邮件

四、MIME

- **MIME新增了5种头部字段**

字段	含义
MIME-Version	MIME版本号，一般为1.0
Content-Type	报文体中数据的类型
Content-Transfer-Encoding	传输时编码格式
Content-ID	唯一的标识符
Content-Description	供人阅读的内容描述

- 传输编码规则：
  - **base64:** 又称为**Radix-64 (基数64转换)**，原始二进制数据中的每**6bit**被映射为**8bit(ASCII**字符**)**
  - **quoted-printable:** 原始二进制数据的**8bit**表示为**2**个**16**进制数，前加“**=**”