

第 6 章 应用层

DNS、FTP、TLNET
TFTP、HTTP协议

为什么需要应用层

- 应用层
 - 应用层概述
 - 两种工作模式
 - Client/Server
 - P2P模式

应用层为应用程序的通信提供服务。

应用层协议定义：

- 应用进程交换的报文类型，请求还是响应？
- 各种报文类型的语法、语义；
- 进程何时、如何发送报文，以及对报文进行响应的规则（同步）。

应用层功能及协议（等）：

- 域名服务：DNS；
- 文件传输：FTP；
- 电子邮件：SMTP、POP3；
- 远程登陆：TELNET；
- WWW服务：HTTP。

工作模式（Client/Server）

- 应用层
 - 应用层概述
 - 两种工作模式
 - Client/Server
 - P2P模式

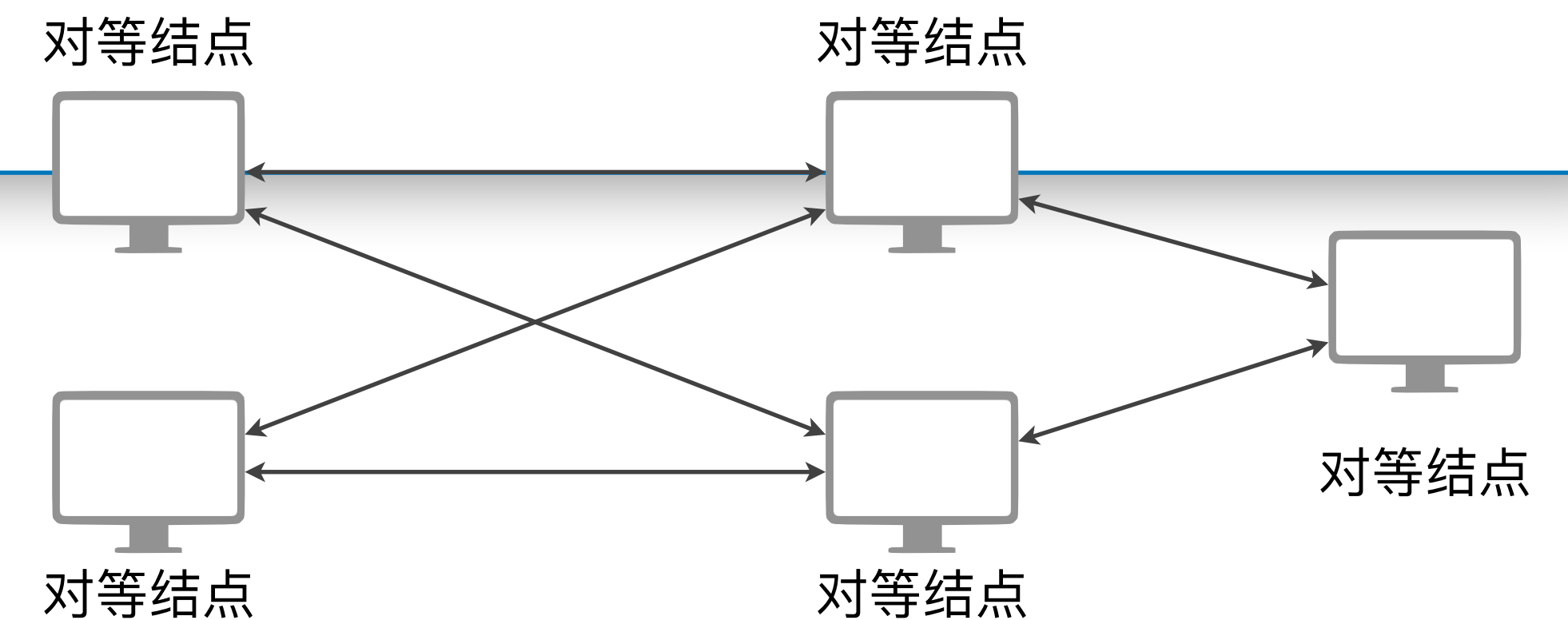
- **服务器**：提供网络服务的设备（由硬件和服务软件组成）
 - 永久提供服务；
 - 有永久性访问地址/域名。
- **客户机**：请求服务的主机。
 - 与服务器通信，使用服务器提供的服务；
 - 间歇性接入网络；
 - 可能使用动态IP地址；
 - 不与其他客户直接通信。
- **资源集中管理。**

两种模式： Client/Server模式和P2P模式。

工作模式（P2P）

- 应用层
 - 应用层概述
 - 两种工作模式
 - Client/Server
 - P2P模式

- 不存在永远在线的服务器；
- 每个主机既可提供服务，也可以请求服务；
- 任意端系统/节点之间可以直接通信；
- 节点间歇性接入网络；
- 节点可能改变IP地址；
- 可扩展性好；
- 网络健壮性好；
- 资源分散管理。



小结

- 应用层
 - 应用层概述
 - 两种工作模式
 - Client/Server
 - P2P模式

- 为什么需要应用层？
- 应用层协议的定义。
- 基本的应用层协议。
- 应用层的两种工作模式。

为什么需要域名系统

- 应用层

- 域名系统DNS

- 域名结构

- 域名服务器的概念

- 域名服务器分类

- DNS解析实例

- 服务器的可靠性

- 域名的解析过程

- DNS缓存

- 互联网上的两台主机间的通信基本前提：

- 双方需要知道对方的IP地址；
 - 双方需要知道对方通信的应用进程（端口号）。

- 现实中找某个人也是如此：

- 这个人在哪里？
 - 这个人叫什么名字？

- 点分十进制的IP地址不容易记住而使用域名，类似人们的身份证号码不易记住而使用姓名。数字适合于机器，名字适合于人类。

保密单位使用信箱作为通信地址。

- 解决办法：

- 给互联网的主机取一个唯一的名字。

域名系统概述

- 应用层
 - 域名系统DNS
- 域名结构
- 域名服务器的概念
- 域名服务器分类
- DNS解析实例
- 服务器的可靠性
- 域名的解析过程
- DNS缓存

- 互联网早期是通过**共享**一个文本文件的方式，让主机知道IP地址与名字（域名）的对关系。
- 有**新主机**加入互联网，在该文件中**新增加一条域名**与IP地址相对应的记录。
- 随着互联网规模迅速扩大，这种方式的**时效性太差**。
- **该文件现在还在使用：**
Windows: c:/Windows/System32/Drives/etc/hosts
Linux (Ubuntu) :/etc/hosts
- **文件内容格式：**
127.0.0.1 localhost
202.193.XXX.XXX ubuntu1604

DNS域名系统解决了共享文件方式中的问题

域名系统概述

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

- 互联网采用层次结构的命名树作为主机的名字。
- 使用分布式的域名系统 DNS。
- 名字到 IP 地址的解析是由若干个域名服务器程序完成的。
- 域名服务器程序在专设的结点上运行。
- 运行该程序的机器称为域名服务器。

互联网的域名结构

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

- 互联网采用了层次树状结构的命名方法。
- 任何一个连接在互联网上的主机或路由器，都有一个唯一的层次结构的**名字**，即**域名**。
- 域名的结构由**标号序列组成**，各标号之间用点隔开：
 **三级域名** . **二级域名** . **顶级域名**
- 各标号分别代表**不同级别的域名**。
- 域名只是个逻辑概念，并不代表计算机所在的**物理地点**。

互联网的域名结构

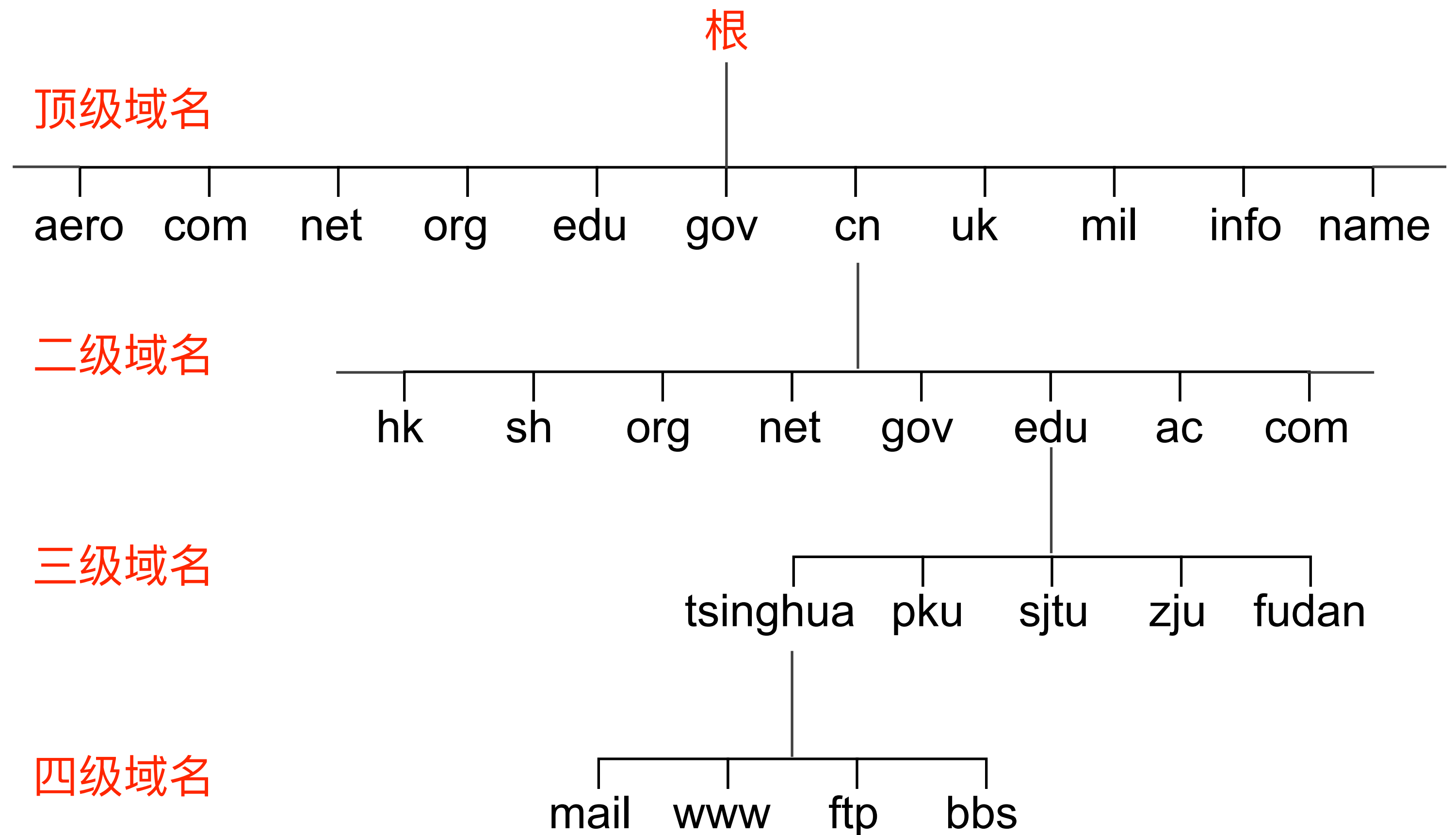
- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

- 国家顶级域名 nTLD:
 - .cn、.us、.uk等。
- 通用顶级域名 gTLD :
 - .com、.net、.org、.edu、.gov、.mil、.int等。
- 基础结构域名 (infrastructure domain):
 - 这种顶级域名只有一个，即 arpa；
 - 用于反向域名解析，因此又称为反向域名。

1990年11月28日，钱天白教授代表中国正式在国际互联网络信息中心（InterNIC）的前身DDN－NIC注册登记了我国的顶级域名CN。

域名空间

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

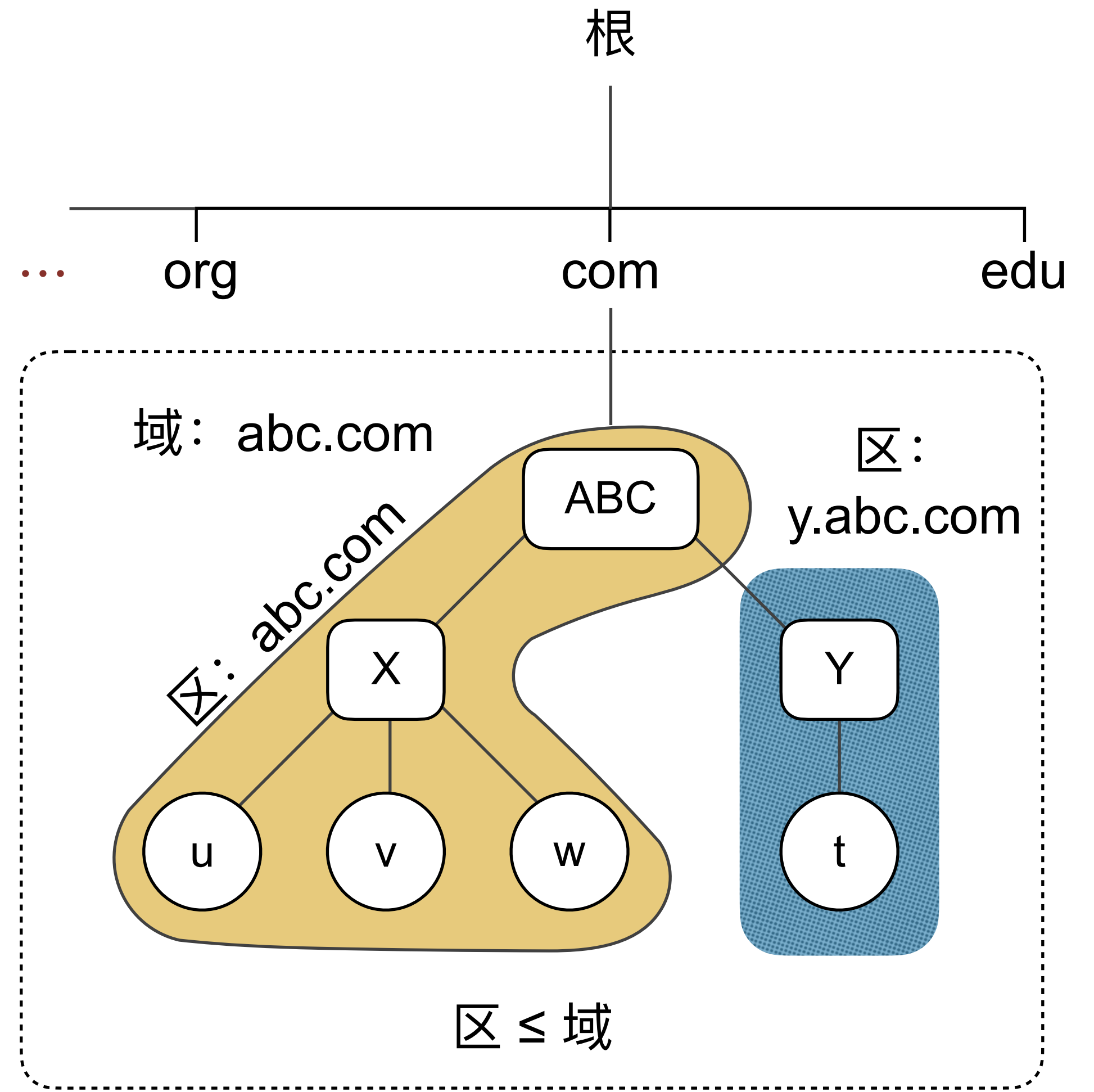
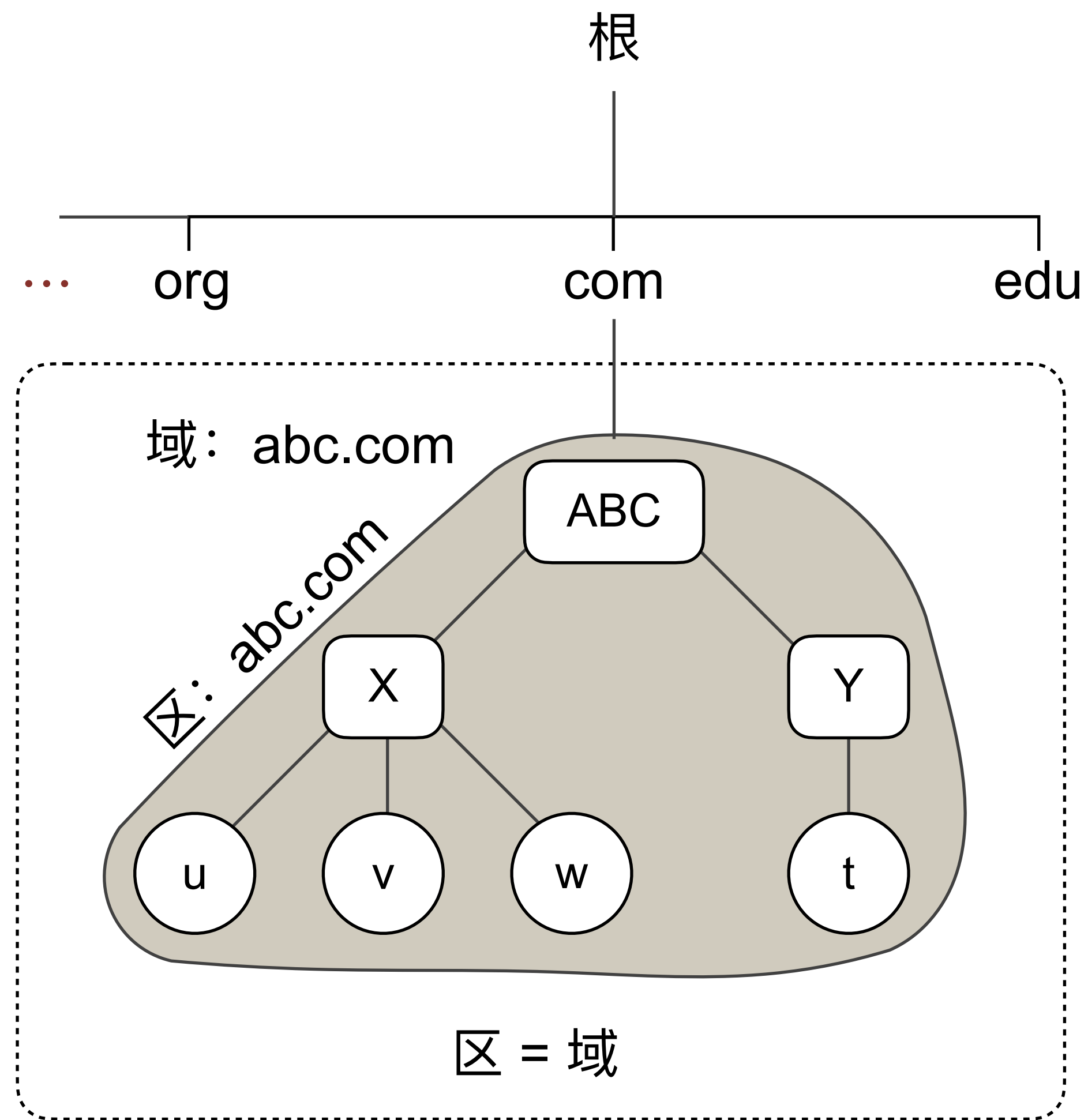


域名服务器

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

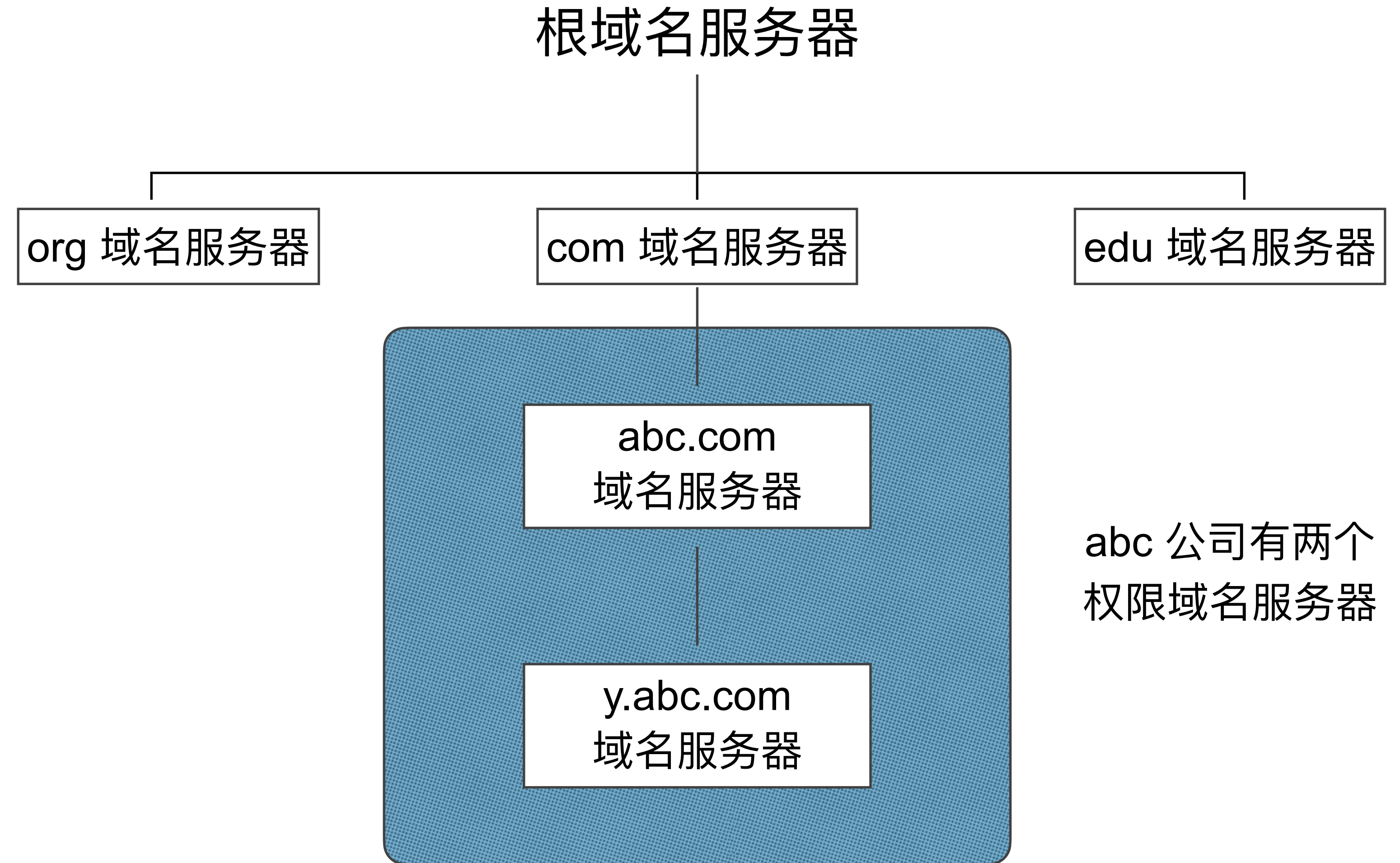
- 一个服务器所负责管辖的（或有权限的）范围叫做区 (zone)。
- 各单位根据具体情况来划分自己管辖范围的区。但在一个区中的所有节点必须是能够连通的。
- 每一个区设置相应的权限域名服务器，用来保存该区中的所有主机的域名到 IP 地址的映射。
- DNS 服务器的管辖范围不是以“域”为单位，而是以“区”为单位的。

区的划分方法



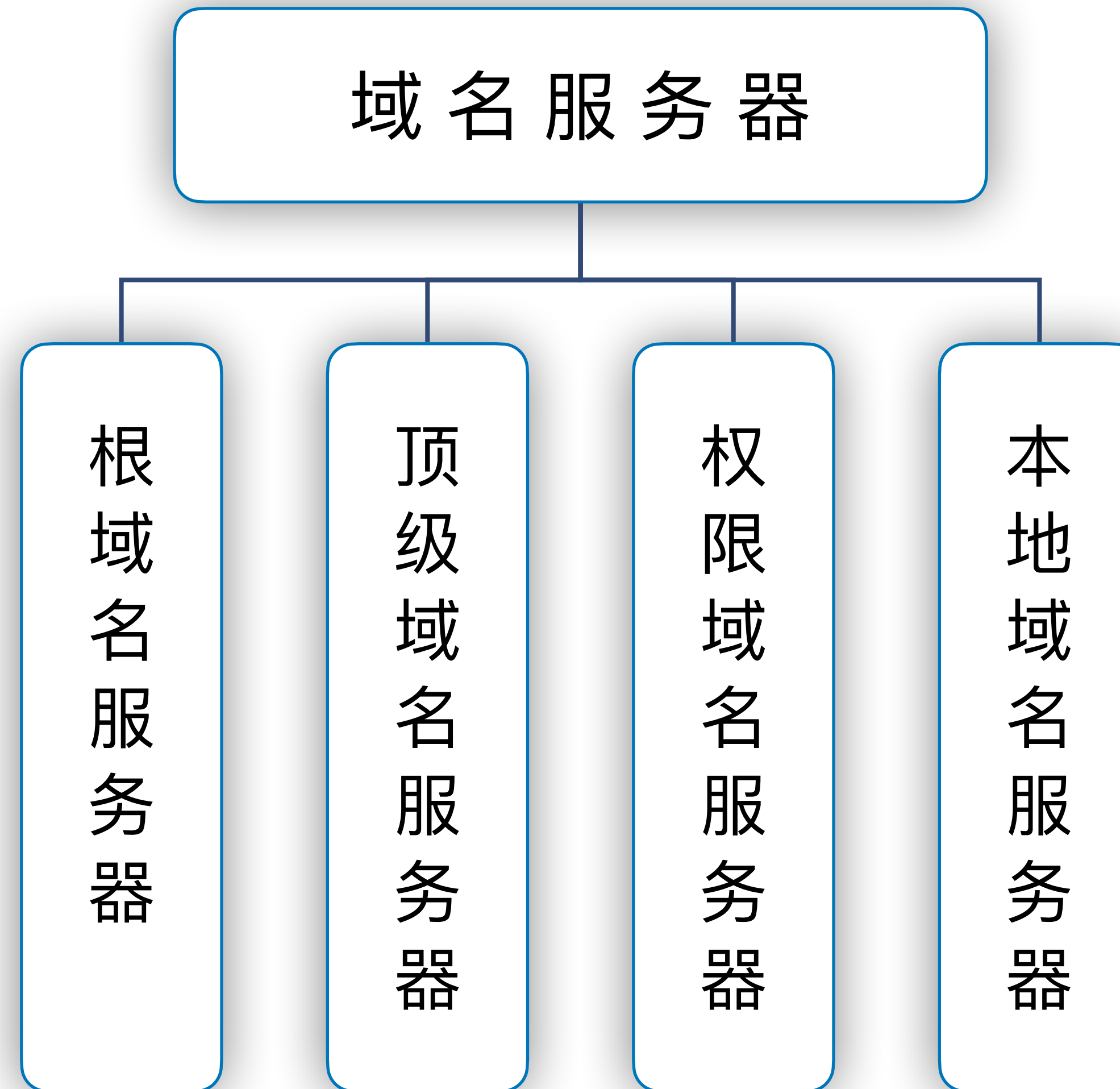
树状结构的域名服务器

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存



四种类型的域名服务器

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存



根域名服务器

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存
- 最高层次、最重要的域名服务器。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。不管是哪一个本地域名服务器，若要对互联网上任何一个域名进行解析，只要自己无法解析，首先求助于根域名服务器。
- 在互联网上共有 13 个不同 IP 地址的根域名服务器（13 套），它们的名字是用一个英文字母命名，从 a 一直到 m（前 13 个字母）。
- 13台根服务器，主根部署在美国，12个辅根有9台在美国，1台在英国，1台在瑞典，1台在日本。

为什么只有13个根域名服务器？

• 部分网络的MTU

Plateau	MTU	Comments	Reference
-----	-----	-----	-----
	65535	Official maximum MTU	RFC 791
	65535	Hyperchannel	RFC 1044
65535			
32000		Just in case	
	17914	16Mb IBM Token Ring	ref. [6]
17914			
	8166	IEEE 802.4	RFC 1042
8166			
	4464	IEEE 802.5 (4Mb max)	RFC 1042
	4352	FDDI (Revised)	RFC 1188
4352 (1%)			
	2048	Wideband Network	RFC 907
	2002	IEEE 802.5 (4Mb recommended)	RFC 1042
2002 (2%)			
	1536	Exp. Ethernet Nets	RFC 895
	1500	Ethernet Networks	RFC 894
	1500	Point-to-Point (default)	RFC 1134
	1492	IEEE 802.3	RFC 1042
1492 (3%)			
	1006	SLIP	RFC 1055
	1006	ARPANET	BBN 1822
1006			
	576	X.25 Networks	RFC 877
	544	DEC IP Portal	ref. [10]
	512	NETBIOS	RFC 1088
	508	IEEE 802/Source-Rt Bridge	RFC 1042
	508	ARCNET	RFC 1051
508 (13%)			
	296	Point-to-Point (low delay)	RFC 1144
296			
68		Official minimum MTU	RFC 791

Table 7-1: Common MTUs in the Internet

- 最初的DNS域名查询默认使用UDP协议，不希望有任何形式的分片，换句话说，要求使用唯一的UDP报文传输DNS响应。
- 当时的绝大多数的网络接口类型支持IP报文≤576 字节无需分片。IETF决定将DNS报文限制在512字节。每一个根域名服务器占用32字节，其中包括根域名的名称、IP地址、TTL(Time To Live)等参数。
- 13根域名服务器一共占用416字节，剩余的96字节用于包装DNS报文头以及其它协议参数。

13个根域名服务器

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

- A, INTERNIC.NET (美国弗吉尼亚州), IP地址: 198.41.0.4。
- B, 美国信息科学研究所 (美国加利福尼亚州), IP地址: 128.9.0.107。
- C, PSINet公司 (美国弗吉尼亚州), IP地址: 192.33.4.12。
- D, 马里兰大学 (美国马里兰州) IP地址: 128.8.10.90。
- E, 美国航空航天管理局 (美国加利福尼亚州), IP地址: 192.203.230.10。
- F, 因特网软件联盟 (美国加利福尼亚州), IP地址: 192.5.5.241。
- G, 美国国防部网络信息中心 (美国弗吉尼亚州), IP地址: 192.112.36.4。
- H, 美国陆军研究所 (美国马里兰州), IP地址: 128.63.2.53。
- I, Autonomica公司 (瑞典斯德哥尔摩), IP地址: 192.36.148.17。
- J, VeriSign公司 (美国弗吉尼亚州), IP地址: 192.58.128.30。
- K, RIPE NCC (英国伦敦), IP地址: 193.0.14.129。
- L, IANA (美国弗吉尼亚州), IP地址: 198.32.64.12。
- M, WIDE Project (日本东京), IP地址: 202.12.27.33。

顶级域名服务器

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

- 顶级域名服务器（即 TLD 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。
- 当收到 DNS 查询请求时，就给出相应的回答（可能是最后的结果，也可能是下一步应当找的域名服务器的 IP 地址）。

权限域名服务器和本地域名服务器

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

- 权限域名服务器：
 - 负责一个区的域名服务器；
 - 当一个权限域名服务器还不能给出最后的查询回答时，就会告诉发出查询请求的 DNS 客户，下一步应当找哪一个权限域名服务器。
- 本地域名服务器：
 - 当一个主机发出 DNS 查询请求时，这个查询请求报文就发送给本地域名服务器；
 - 每一个互联网服务提供者 ISP，或一个大学，甚至一个大学里的系，都可以拥有一个本地域名服务器；
 - 这种域名服务器有时也称为默认域名服务器。

DNS服务器解析实例

```
li@ubuntu1604:~$ nslookup
```

```
> server
```

```
Default server: 8.8.4.4
```

```
Address: 8.8.4.4#53
```

```
> www.tsinghua.edu.cn
```

```
Server:      8.8.4.4
```

```
Address:     8.8.4.4#53
```

```
Non-authoritative answer:
```

```
www.tsinghua.edu.cn ...
```

```
Name: www.d.tsinghua.edu.cn
```

```
Address: 166.111.4.100
```

```
>
```

查询非本
区域名



```
> server 202.112.0.13
```

```
Default server: 202.112.0.13
```

```
Address: 202.112.0.13#53
```

```
> www.tsinghua.edu.cn
```

```
Server:      202.112.0.13
```

```
Address:     202.112.0.13#53
```

```
www.tsinghua.edu.cn ....
```

```
Name: www.d.tsinghua.edu.cn ...
```

```
Address: 166.111.4.100
```

查询本
区域名



202.112.0.13为清华大学的
权威域名服务器。

提高域名服务器的可靠性

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

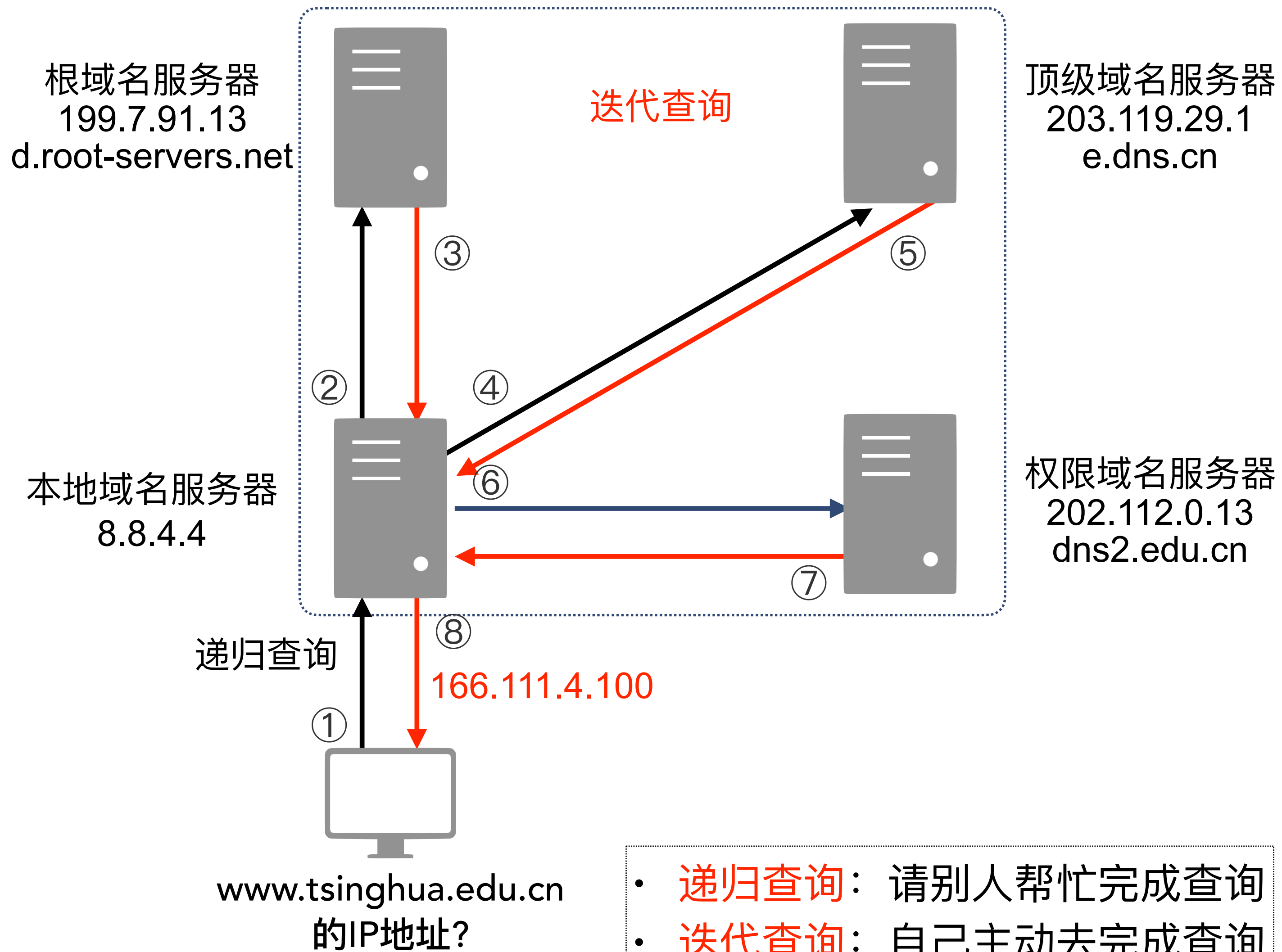
- DNS 域名服务器都把数据复制到几个域名服务器来保存，其中的一个是主域名服务器，其他的是辅助域名服务器。
- 当主域名服务器出故障时，辅助域名服务器可以保证 DNS 的查询工作不会中断。
- 主域名服务器定期把数据复制到辅助域名服务器中，而更改数据只能在主域名服务器中进行。这样就保证了数据的一致性。

域名的解析过程

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存

- 主机向本地域名服务器的查询采用递归查询。如果本地域名服务器不能解析域名的 IP 地址，本地域名服务器就以 DNS 客户的身份，向根域名服务器发出查询请求报文。
- 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：“你下一步应当向哪一个域名服务器进行查询”。
- 本地域名服务器进行后续的查询。

迭代查询



- 递归查询：请别人帮忙完成查询
- 迭代查询：自己主动去完成查询

```
li@ubuntu1604:~$ dig +trace www.tsinghua.edu.cn

; <<>> DiG 9.10.3-P4-Ubuntu <<>> +trace www.tsinghua.edu.cn
;; global options: +cmd
.           36670 IN    NS     a.root-servers.net.
.           36670 IN    NS     k.root-servers.net.
.           36670 IN    NS     d.root-servers.net.
.....
;; Received 525 bytes from 8.8.4.4#53(8.8.4.4) in 328 ms

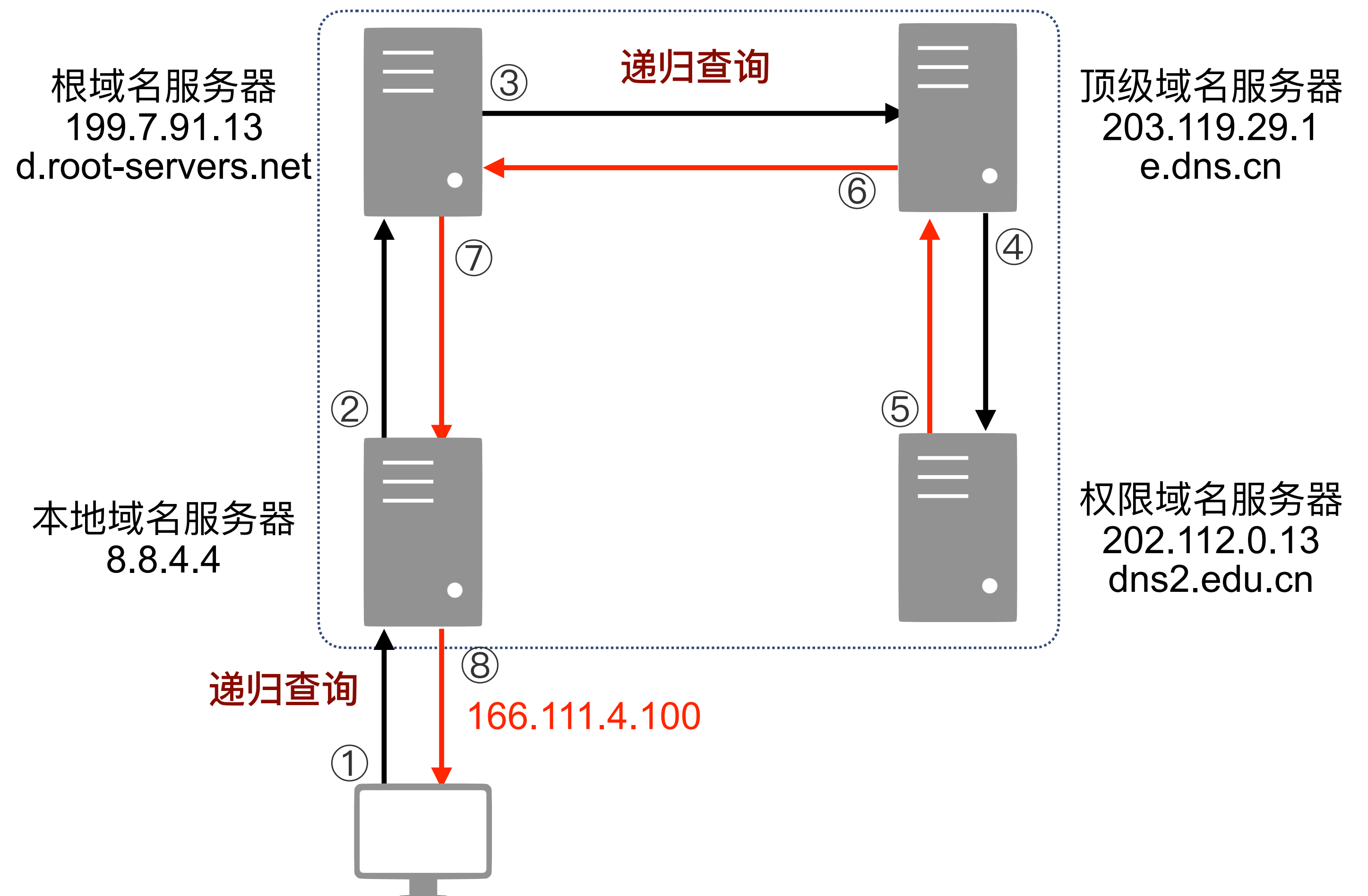
cn.         172800 IN    NS     g.dns.cn.
cn.         172800 IN    NS     ns.cernet.net.
cn.         172800 IN    NS     e.dns.cn.
.....
;; Received 710 bytes from 199.7.91.13#53(d.root-servers.net) in 54 ms

edu.cn.     172800 IN    NS     ns2.cuhk.hk.
edu.cn.     172800 IN    NS     ns2.cernet.net.
edu.cn.     172800 IN    NS     dns.edu.cn.
edu.cn.     172800 IN    NS     dns2.edu.cn.
.....
;; Received 498 bytes from 203.119.29.1#53(e.dns.cn) in 81 ms

www.tsinghua.edu.cn. 21600 IN    CNAME  www.d.tsinghua.edu.cn.
www.d.tsinghua.edu.cn. 3600 IN    A      166.111.4.100
d.tsinghua.edu.cn.  86400 IN    NS     dns.d.tsinghua.edu.cn.
;; Received 134 bytes from 202.112.0.13#53(dns2.edu.cn) in 53 ms
```

本地域名服务器采用递归查询

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存



DNS服务器的高速缓存

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存
- 每个域名服务器都维护一个高速缓存，存放最近用过的名字以及从何处获得名字映射信息的记录。
- 可大大减轻根域名服务器的负荷，使互联网上的 DNS 查询请求和回答报文的数量大为减少。
- 为保持高速缓存中的内容正确，域名服务器应为每项内容设置计时器，并处理超过合理时间的项（例如，每个项目只存放两天）。

DNS客户名字高速缓存

- 当权威域名服务器回答一个查询请求时，在响应中指明绑定有效存在的时间值。
- 增加此时间值可减少网络开销，而减少此时间值可提高域名转换的准确性。
- 解决了客户频繁向服务器请求解析不存在的域名问题。

Windows7中，ifconfig /displaydns命令



qswa.baidu.com

名称不存在。

www.baidu.com

记录名称.....: www.baidu.com

记录类型.....: 1

生存时间.....: 86400

数据长度.....: 4

部分.....: 答案

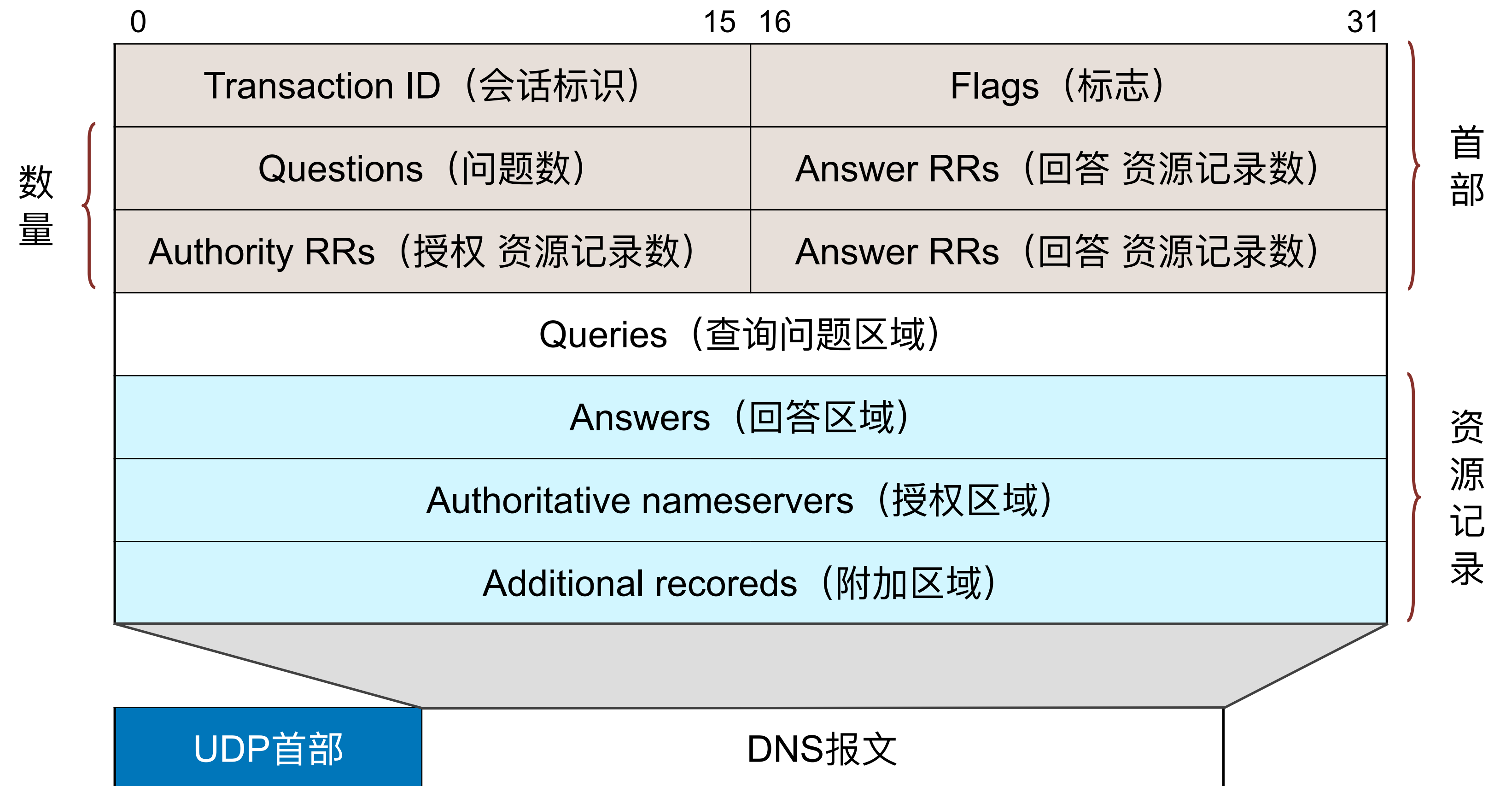
A (主机)记录: 127.0.0.1

小结

- 应用层
 - 域名系统DNS
 - 域名结构
 - 域名服务器的概念
 - 域名服务器分类
 - DNS解析实例
 - 服务器的可靠性
 - 域名的解析过程
 - DNS缓存
- 域名系统的概念。
- 域名的结构：
 - 根、顶级域名、二级域名、三级域名...
- 域名服务器：
 - 根域名服务器、顶级域名服务器；
 - 权限域名服务器、本地域名服务器。
- 域名查询：
 - 迭代查询；
 - 递归查询；
 - 名字高速缓存。

DNS报文格式

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程



DNS报文由三部分构成：首部、查询问题区域和资源记录区域

首部：Transaction ID（会话标识）

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程

- 会话标识（2字节）：
 - 是DNS报文的ID标识；
 - 对于请求报文和其对应的回答报文，这个字段是相同的，通过它可以区分DNS应答报文是哪个请求的回答。

0	15	16	31	首部
Transaction ID（会话标识）		Flags（标志）		
Questions（问题数）		Answer RRs（回答 资源记录数）		
Authority RRs（授权 资源记录数）		Answer RRs（回答 资源记录数）		

首部：Flags（标志）

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程

- 标志（4字节）：
 - QR：查询/回答标志，0为查询，1为回答；
 - opcode：0标准查询，1反向查询，2服务器状态请求。

1	4	1	1	1	1	3	4
QR	opcode	AA	TC	RD	RA	zero	rcode
Transaction ID（会话标识）				Flags（标志）			
Questions（问题数）				Answer RRs（回答 资源记录数）			
Authority RRs（授权 资源记录数）				Answer RRs（回答 资源记录数）			

首部

首部：Flags（标志）

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程
- **AA**：0为回答服务器不是该域名的权威解析服务器，1为回答服务器是该域名的权威解析服务器。
- **TC**：0为报文未截断，1为报文过长被截断（只返回了前512字节）。
- **RD**：0为不期望递归查询，1为期望进行递归查询。
- **RA**：0为回答服务器不支持递归查询，1支持递归查询。
- **zero**：保留，必须置0。
- **rcode**：返回码，0为没有错误，3为名字错误，2为服务器错误等。

首部：数量字段

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程

- 数量字段（总共8字节）：Questions、Answer RRs、Authority RRs、Additional RRs，各自表示后面的4个区域的数量：
 - Questions：表示查询问题的数量；
 - Answers RRs：表示回答资源数量；
 - Authority RRs：表示授权资源数量；
 - Additional RRs：表示附加区域资源数量。

数量 {	Transaction ID（会话标识）	Flags（标志）	} 首部
	Questions（问题数）	Answer RRs（回答 资源记录数）	
	Authority RRs（授权 资源记录数）	Answer RRs（回答 资源记录数）	

Queries（查询问题区域）

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程

- 查询名：需要查询的域名（如果是反向查询，则为IP地址）。
- 查询类型：每个问题有一个查询类型，取值可以为任何可用的类型值。
- 查询类：通常为1，表明是Internet数据。

Name（查询名，长度不固定）		
Type（查询类型）		Class（查询类）
0	15 16	31
Queries（查询问题区域）		
Answers（回答区域）		
Authoritative nameservers（授权区域）		
Additional recoreds（附加区域）		

资源记录

资源记录： 部分查询类型

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程

类型	助记词	说明
1	A	由域名获得IPv4地址
2	NS	查询域名服务器
5	CNAME	查询规范名称
6	SOA	开始授权
11	WKS	熟知服务
12	PTR	把IP地址转换成域名
13	HINFO	主机信息
15	MX	邮件交换
28	AAAA	由域名获得IPv6地址
252	AXFR	传送整个区的请求
255	ANY	对所有记录的请求

资源记录：回答区域

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程

Name（域名，2字节或长度不固定）		
Type（查询类型）	Class（查询类）	
Time to live（生存时间）		
Data length（资源数据长度）		
Data（资源数据，长度不固定）		

0151631

Queries（查询问题区域）		
Answers（回答区域）		
Authoritative nameservers（授权区域）		
Additional records（附加区域）		

资源记录

该区域有三个格式一样区域：回答区域，授权区域和附加区域。

DNS查询实例（服务器8.8.8.8）

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程

User Datagram Protocol, Src Port: 60010, Dst Port: domain (53) #DNS监听53号端口

Domain Name System (query) #DNS查询

Transaction ID: 0x1670 #会话标识, 与DNS回答一致

Flags: 0x0100 Standard query

0... .. = Response: Message is a query #DNS查询

.000 0... .. = Opcode: Standard query (0)

.... ..0. = Truncated: Message is not truncated

.... ..1 = Recursion desired: Do query recursively #期望递归

.... ..0.. = Z: reserved (0)

.... ..0 = Non-authenticated data: Unacceptable

Questions: 1 #问题数据量

Answer RRs: 0 #回答资源记录数

Authority RRs: 0 #授权资源记录数

Additional RRs: 0 #附加资源记录数

Queries #查询区域

www.guat.edu.cn: type A, class IN

Name: www.guat.edu.cn #查询的域名

Type: A (Host Address) (1) #由域名查询IP

Class: IN (0x0001) #为Internet数据

首部

数量

查询区域

DNS回答实例

User Datagram Protocol, Src Port: domain (53), Dst Port: 60010
Domain Name System (response)
Transaction ID: 0x1670 #会话标识, 与DNS查询一致
Flags: 0x8000 Standard query response, No error
1... .. = Response: Message is a response #DNS回答
.....
Questions: 1 #查询数量1个
Answer RRs: 1 #资源记录区域回答数量1个
Authority RRs: 0 #权威区域回答数量0个 (不是权威回答, 不是8.8.8.8管辖的区)
Additional RRs: 0 #附加区域回答数量0个

Transaction ID (会话标识)	Flags (标志)
Questions (问题数)	Answer RRs (回答 资源记录数)
Authority RRs (授权 资源记录数)	Additional RRs (附加 资源记录数)

Queries #查询区域
www.guat.edu.cn: type A, class IN
Name: www.guat.edu.cn #查询的域名
Type: A (Host Address) (1) #由域名查IP
Class: IN (0x0001) #为Internet数据

Name (查询名, 长度不固定)	
Type (查询类型)	Class (查询类)

Answers #资源记录区域 (回答区域)
www.guat.edu.cn: type A, class IN, addr 202.193.96.150
Name: www.guat.edu.cn
Type: A (Host Address) (1) #查询类型A, 由域名查询IP地址
Class: IN (0x0001) #查询类IN
Time to live: 67677 #DNS记录缓存时间
Data length: 4
Address: www.guat.edu.cn (202.193.96.150) #IP地址

Name (域名, 2字节或长度不固定)	
Type (查询类型)	Class (查询类)
Time to live (生存时间)	
Data length (资源数据长度)	
Data (资源数据, 长度不固定)	

DNS查询过程（同步）

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程

DNS客户查询过程：

- 本地查询：
 - DNS缓存中查询；
 - hosts文件中查询。
- 向本地域名服务器查询：
 - 授权直接回答；
 - DNS缓存回答；
 - 本地域名服务器向根发起迭代查询。

验证DNS查询过程

- Windows7中编辑C:\Windows\System32\drivers\etc\hosts文件。
- 清除DNS缓存：
 - C:\Users\Administrator>ipconfig /flushdns。
- ping www.baidu.com：
 - 通注意结果。
- 浏览器访问www.baidu.com：
 - 不通。

```
C:\Users\Administrator>ping www.baidu.com

正在 Ping www.baidu.com [127.0.0.1] 具有 32 字节的数据:
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128
来自 127.0.0.1 的回复: 字节=32 时间<1ms TTL=128

127.0.0.1 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
    往返行程的估计时间(以毫秒为单位):
        最短 = 0ms, 最长 = 0ms, 平均 = 0ms

C:\Users\Administrator>
```



```
hosts - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
#          38.25.63.10      x.acme.com                # x client host

# localhost name resolution is handled within DNS itself.
#          127.0.0.1        localhost
#          ::1              localhost

127.0.0.1      www.baidu.com
```


小结

- 应用层
 - DNS报文格式
 - 会话标识
 - 标志
 - 数量字段
 - 查询问题区域
 - 查询类型
 - 回答区域
 - DNS查询实例
 - DNS查询过程

- DNS报文结构（语法、语义）。
- DNS实例分析。
- DNS查询过程（同步）。

文件传送协议FTP概述

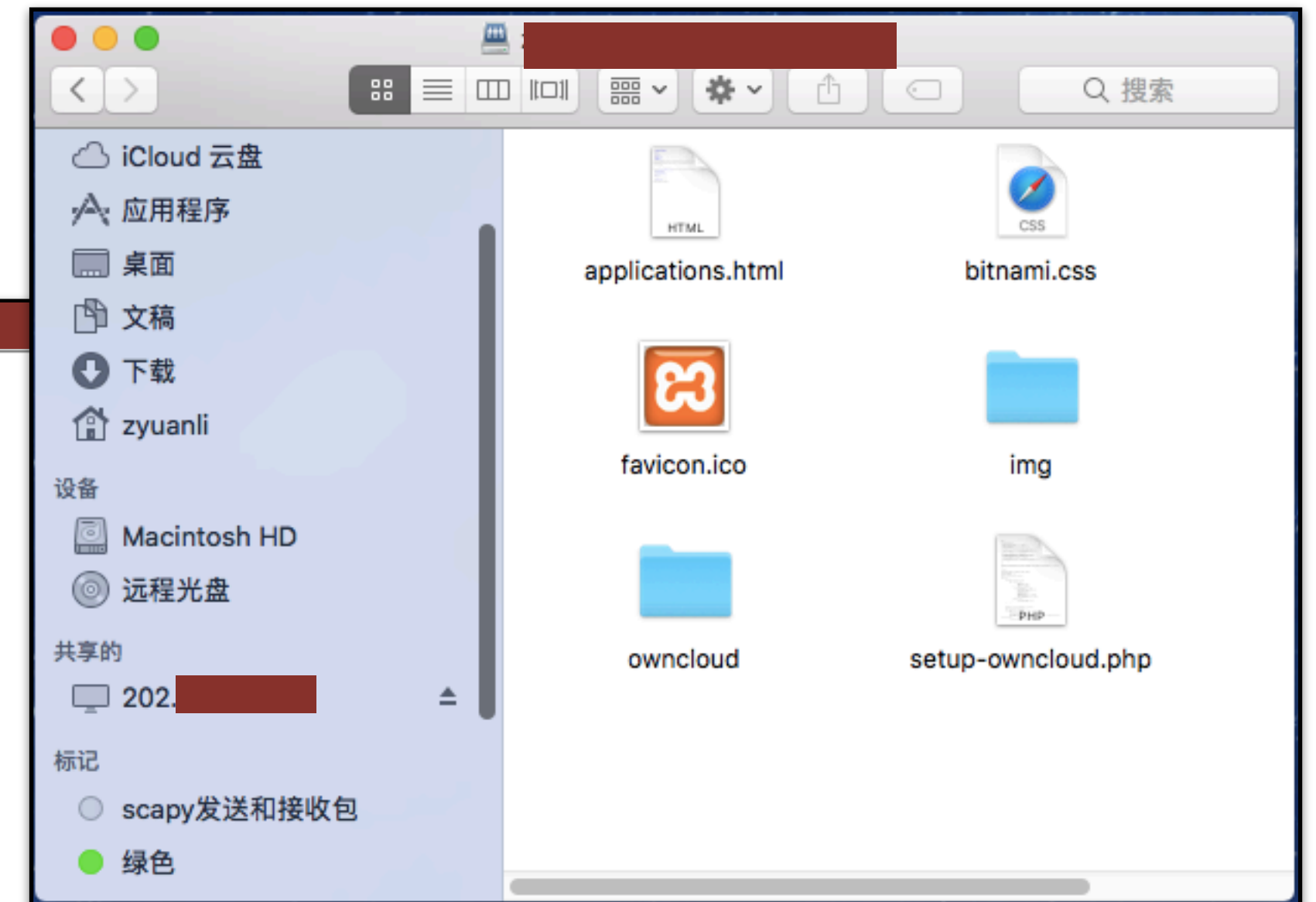
- 应用层
 - 文件传送协议
 - FTP的特点
 - FTP的两个连接
 - 工作方式
 - 主动方式
 - 被动方式
 - 传输模式与问题
 - 交互工作方式

- 文件传送协议 **FTP** (File Transfer Protocol) 是互联网上使用最广泛的协议。
- **FTP 提供交互式的访问**，允许客户指明文件的类型与格式，并允许文件具有存取权限。
- **FTP 屏蔽了计算机系统的细节**，适合在异构网络中任意**计算机之间传送文件**。
- **FTP实现的是通过网络实现异构计算机间文件的“拷贝”**。

FTP操作实例

- 应用层
 - 文件传送协议
- FTP的特点
- FTP的两个连接
- 工作方式
 - 主动方式
 - 被动方式
- 传输模式与问题
- 交互工作方式

```
Last login: Mon Nov 11 08:59:19 on ttys000
[redacted]Book-Ai:[redacted]$ ftp
ftp> open [redacted]
Connected to [redacted]
220 FTP Service
Name ([redacted]): [redacted]
331 Password required for [redacted]
Password:
230 User [redacted] logged in
ftp> ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
-rw-r--r--  1 root    root      3607 Feb 27  2017 applications.html
-rw-r--r--  1 root    root      177 Feb 27  2017 bitnami.css
drwxr-xr-x 20 root    root     4096 Dec 22  2017 dashboard
-rw-r--r--  1 root    root    30894 May 11  2007 favicon.ico
drwxr-xr-x  2 root    root     4096 Dec 22  2017 img
-rw-r--r--  1 root    root      260 Jul  9  2015 index.php
drwxr-xr-x 13 daemon  daemon   4096 Dec 22  2017 owncloud
-rw-rw-rw-  1 [redacted]  [redacted] 10838 Dec 22  2017 setup-owncloud.php
drwxr-xr-x  2 daemon  daemon   4096 Dec 22  2017 webalizer
226 Transfer complete
ftp>
```



FTP的特点

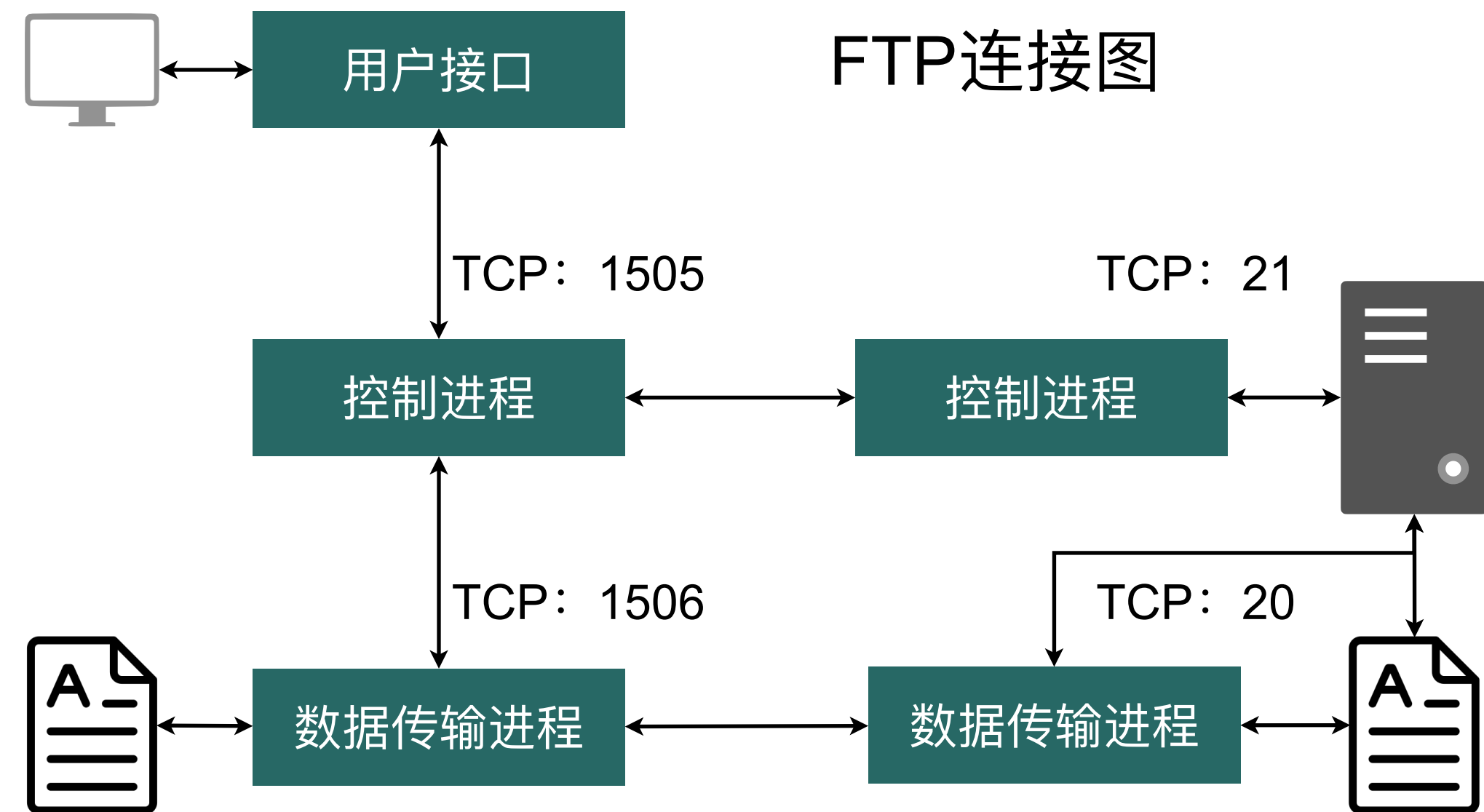
- 应用层
 - 文件传送协议
 - **FTP的特点**
 - FTP的两个连接
 - 工作方式
 - 主动方式
 - 被动方式
 - 传输模式与问题
 - 交互工作方式

- **采用C/S模式**，可为多个客户提供服务。
- 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；另外有**若干个从属进程**，负责处理单个请求。
- 消除了不同操作系统下的文件系统的不兼容性。
- 实现了**文件传送的基本服务**。
- 使用**TCP**可靠传输协议。

FTP的两个连接和工作过程

- 客户端使用任意分配的端口号 (>1023)。
- 服务器端用熟知端口号21传输控制信息。
- 服务器端用熟知端口号20传输数据。
- 服务器主程序打开21号端口，等待客户进程发送连接请求。
- 启动从属进程处理客户进程请求。
- 回到等待状态，继续接受其他客户进程请求。

主进程与从属进程的处理并发进行



使用两个不同端口号的好处：

- 使协议更加简单和更容易实现；
- 在传输文件时还可以利用控制连接。

FTP的工作方式

- 应用层
 - 文件传送协议
 - FTP的特点
 - FTP的两个连接
 - 工作方式
 - 主动方式
 - 被动方式
 - 传输模式与问题
 - 交互工作方式

- 主动方式：
 - 客户与服务器21号端口建立连接之后（端口号 $N > 1023$ ），然后监听 $N+1$ 号端口，并发送FTP命令“port $N+1$ ”到FTP服务器，服务器用20号数据端口主动向客户 $N+1$ 端口建立连接。
- 被动方式：
 - 客户用端口 N （ > 1023 ）与服务器21号端口建立连接之后，客户端打开 $N+1$ 端口，发送PASV命令给服务器，服务开启端口 P （ > 1023 ），并发送“PORT P ”告知客户，然后客户从端口 $N+1$ 和服务器端口 P 建立连接，用来进行数据传送。

主动方式 (MAC OS ftp命令)

- 应用层
 - 文件传送协议
 - FTP的特点
 - FTP的两个连接
 - 工作方式
 - 主动方式
 - 被动方式
 - 传输模式与问题
 - 交互工作方式

Internet Protocol Version 4, Src: 172.20.33.167, Dst: 202.XXX.XXX.153
Transmission Control Protocol, Src Port: 50439, Dst Port: 21, Seq: 34, Ack: 129, Len: 26
File Transfer Protocol (FTP)
PORT 172,20,33,167,197,8\r\n
Request command: PORT
Request arg: 172,20,33,167,197,8
Active IP address: 172.20.33.167
Active port: 50440

MAC OS安装telnet和ftp命令
brew install telnet
brew install inetutils
brew link --overwrite inetutils

被动模式 (MAC OS 浏览器FTP)

- 应用层
 - 文件传送协议
 - FTP的特点
 - FTP的两个连接
 - 工作方式
 - 主动方式
 - 被动方式
 - 传输模式与问题
 - 交互工作方式

Internet Protocol Version 4, Src: 172.20.33.167, Dst: 202.XXX.XXX.153
Transmission Control Protocol, Src Port: 50470, Dst Port: 21, Seq: 169,
Ack: 862, Len: 6

File Transfer Protocol (FTP)

PASV\r\n

Request command: PASV

Internet Protocol Version 4, Src: 202.XXX.XXX.153, Dst: 172.20.33.167
Transmission Control Protocol, Src Port: 21, Dst Port: 50470, Seq: 862,
Ack: 175, Len: 53

File Transfer Protocol (FTP)

227 Entering Passive Mode (202,XXX.XXX,153,179,179).\r\n

Response code: Entering Passive Mode (227)

Response arg: Entering Passive Mode (202,XXX.XXX,153,179,179).

Passive IP address: 202.XXX.XXX.153

Passive port: 46003

#179*256+179=46003

FTP文件传输模式与存在的问题

- 应用层
 - 文件传送协议
 - FTP的特点
 - FTP的两个连接
 - 工作方式
 - 主动方式
 - 被动方式
 - 传输模式与问题
 - 交互工作方式

- 文本模式：
 - ASCII模式，以文本序列传输数据。
- 二进制模式：
 - Binary模式，以二进制序列传输数据。
- FTP存在的问题：
 - FTP服务器在整个会话期间一直保持用户的状态（state），并且把特定的用户账户与控制联系起来；
 - 如果用户在远程目录树上不断跳转时，服务器必须追踪到用户在远程树上的当前位置；
 - 当客户端用户数量不断增多，FTP服务器所要维持的会话总数会迅速增长。这样大大限制FTP服务器的性能。

FTP的交互工作方式

- 应用层
 - 文件传送协议
 - FTP的特点
 - FTP的两个连接
 - 工作方式
 - 主动方式
 - 被动方式
 - 传输模式与问题
 - 交互工作方式
- FTP客户与服务之间采用交互方式（交互控制帧）。
- **FTP 交换信息的控制帧，包含控制命令和选项。**
- 大多数 FTP 控制帧是**简单的ASCII文本**，用户向服务器发出FTP命令，服务器执行用户的FTP命令，并将执行的结果返回给用户。

小结

- 应用层
 - 文件传送协议
 - FTP的特点
 - FTP的两个连接
 - 工作方式
 - 主动方式
 - 被动方式
 - 传输模式与问题
 - 交互工作方式

- **FTP两个连接：**
 - 一个用于传输数据（20号端口）；
 - 一个用于传输命令（21号端口）。
- **FTP两种工作方式：**
 - 主动方式；
 - 被动方式。
- **FTP文件传输模式：**
 - 文本模式；
 - 二进制模式。

简单文件传送协议 TFTP

- 应用层
 - TFTP协议
 - TFTP的特点
 - 类似停止等待协议
 - TFTP格式
 - TFTP工作流程
 - TFTP实验抓包分析

- TFTP (Trivial File Transfer Protocol) 是一个很小且易于实现的文件传送协议。
- TFTP 使用客户服务器方式和使用 UDP 数据报，因此 TFTP 需要有自己的差错改正措施。
- TFTP 只支持文件传输而不支持交互。
- TFTP 没有一个庞大的命令集，没有列目录的功能，也不能对用户进行身份鉴别。

TFTP 的主要特点

- 应用层
 - TFTP协议
 - **TFTP的特点**
 - 类似停止等待协议
 - TFTP格式
 - TFTP工作流程
 - TFTP实验抓包分析
- 每次传送的数据 PDU 中有 **512 字节的数据**，但最后一次可不足 512 字节。
- 数据 PDU 也称为**文件块** (block)，**每个块按序编号**，从 1 开始。
- 支持 ASCII 码或二进制传送。
- 可对文件进行读或写。
- 使用**很简单的首部**。

TFTP 的工作很像停止等待协议

- 应用层
 - TFTP协议
 - TFTP的特点
 - 类似停止等待协议
 - TFTP报文格式
 - TFTP工作流程
 - TFTP实验抓包分析

- 发送完一个文件块后就等待对方的确认，确认时应指明所确认的块编号。
- 发完数据后在规定时间内收不到确认就要重发数据 PDU。
- 发送确认 PDU 的一方若在规定时间内收不到下一个文件块，也要重发确认 PDU。

TFTP 的工作很像停止等待协议

- 应用层
 - TFTP协议
 - TFTP的特点
 - 类似停止等待协议
 - TFTP报文格式
 - TFTP工作流程
 - TFTP实验抓包分析

- 开始工作时，TFTP 客户进程发送一个读请求 PDU 或写请求 PDU 给 TFTP 服务器进程，其熟知端口号码为69。
- TFTP 服务器进程要选择一个新的端口和 TFTP 客户进程进行通信。
- 若文件长度恰好为 512 字节的整数倍，则在文件传送完毕后，还必须在最后发送一个只含首部而无数据的数据 PDU。
- 若文件长度不是 512 字节的整数倍，则最后传送数据 PDU 的数据字段一定不满 512 字节，这正好可作为文件结束的标志。

TFTP协议（语法、语义）

- 模式字段：
- netascii，表示数据是以成行的ascii码字符组成，以两个字节\r \n作为行结束符；
- octet，则将数据看做8bit一组的字节流而不作任何解释。

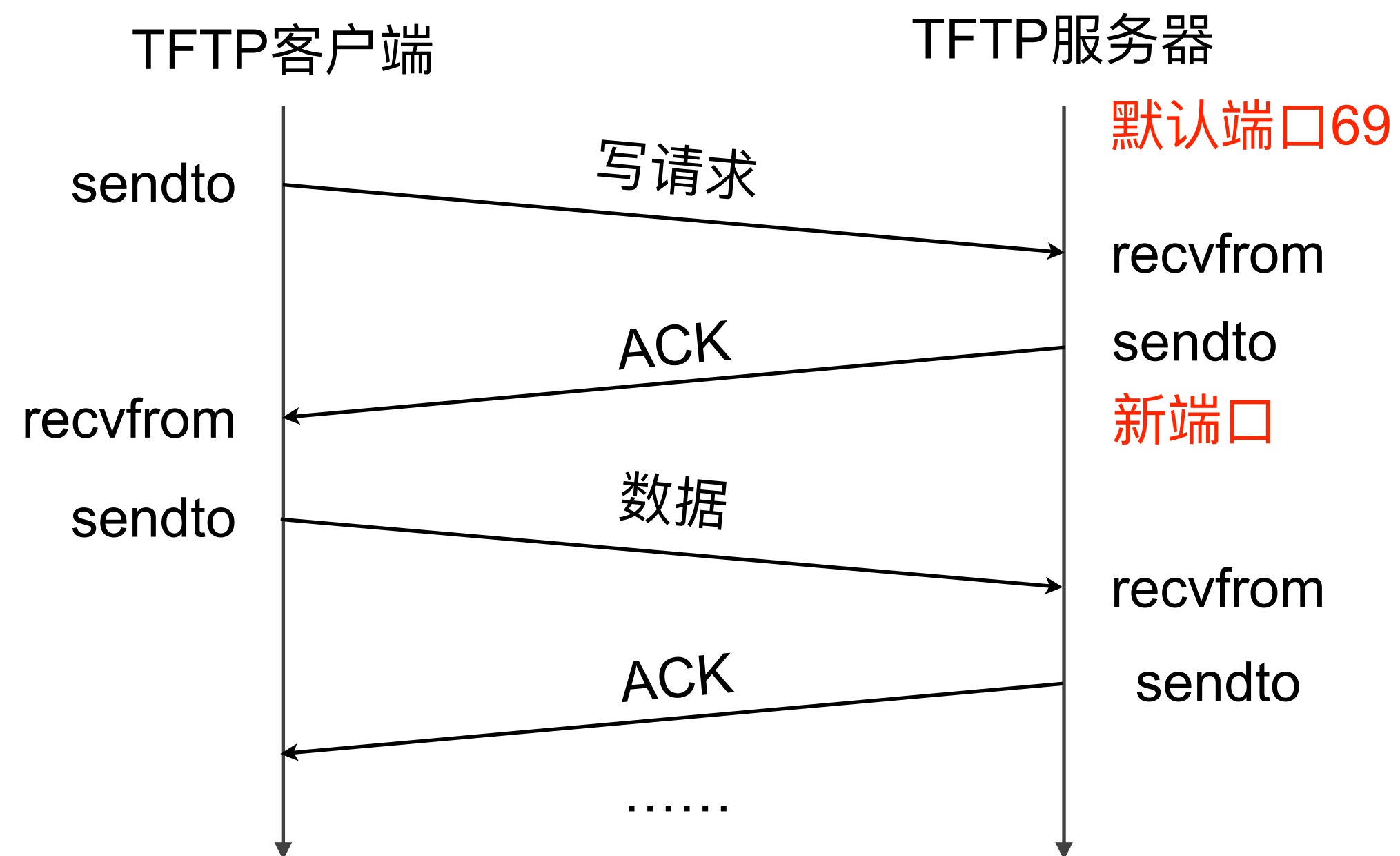
差错码	
Error code	Error message
0	未定义
1	文件未找到
2	访问非法
3	磁盘满或超过分配的配额
4	非法的TFTP操作
5	未知的传输ID
6	文件已经存在
7	没有类似的用户

IP首部	UDP首部	操作码 1=RRQ 2=WRQ	文件名	0	模式	0
20字节	8字节	2字节	N字节	1	N字节	
		操作码 3=data	块编号	数据		
		2字节	2字节	0~512字节		
		操作码 4=ACK	块编号			
		2字节	2字节			
		操作码 5=error	块编号	差错信息	0	
		2字节	2字节	N字节	1	

TFTP工作流程（同步）

- 应用层
 - TFTP协议
 - TFTP的特点
 - 类似停止等待协议
 - TFTP报文格式
 - **TFTP工作流程**
 - TFTP实验抓包分析

- 初始化连接阶段，客户**发送RRQ或WRQ**，服务器的端口号为69。
- 余下的阶段，TFTP将传输标记TID传送给UDP作为源和目的端口。
- 在网络管理中，TFTP和FTP常用于**备份网络配置文件**。



实验抓包结果

tftp						Express
No.	Source	Destination	Protocol	Length	Info	
10	192.168.1.20	192.168.1.10	TFTP	60	Write Request, File: r1-config, Transfer type: octet	
13	192.168.1.10	192.168.1.20	TFTP	46	Acknowledgement, Block: 0	
14	192.168.1.20	192.168.1.10	TFTP	558	Data Packet, Block: 1	
15	192.168.1.10	192.168.1.20	TFTP	46	Acknowledgement, Block: 1	
16	192.168.1.20	192.168.1.10	TFTP	430	Data Packet, Block: 2 (last)	
17	192.168.1.10	192.168.1.20	TFTP	46	Acknowledgement, Block: 2	

包序号	源IP	目的IP	操作码	动作
10	192.168.1.20	192.168.1.10	2	客户向服务器请求写操作
13	192.168.1.10	192.168.1.20	4	服务器确认第0块
14	192.168.1.20	192.168.1.10	3	客户向服务器写第1块512字节
15	192.168.1.10	192.168.1.20	4	服务器确认第1块
16	192.168.1.20	192.168.1.10	3	客户向服务器写第2块384字节
17	192.168.1.10	192.168.1.20	4	服务器确认第2块

小结

- 应用层
 - TFTP协议
 - TFTP的特点
 - 类似停止等待协议
 - TFTP报文格式
 - TFTP工作流程
 - TFTP实验抓包分析

- TFTP的特点。
- 类似停止等待协议。
- TFTP格式。
- TFTP工作流程。
- TFTP实验抓包分析。

远程终端协议TELNET

- 应用层
 - 远程终端协议
 - NVT的概念
 - 选项协商
 - Telnet应用场景
 - Telnet的安全性
 - SSH

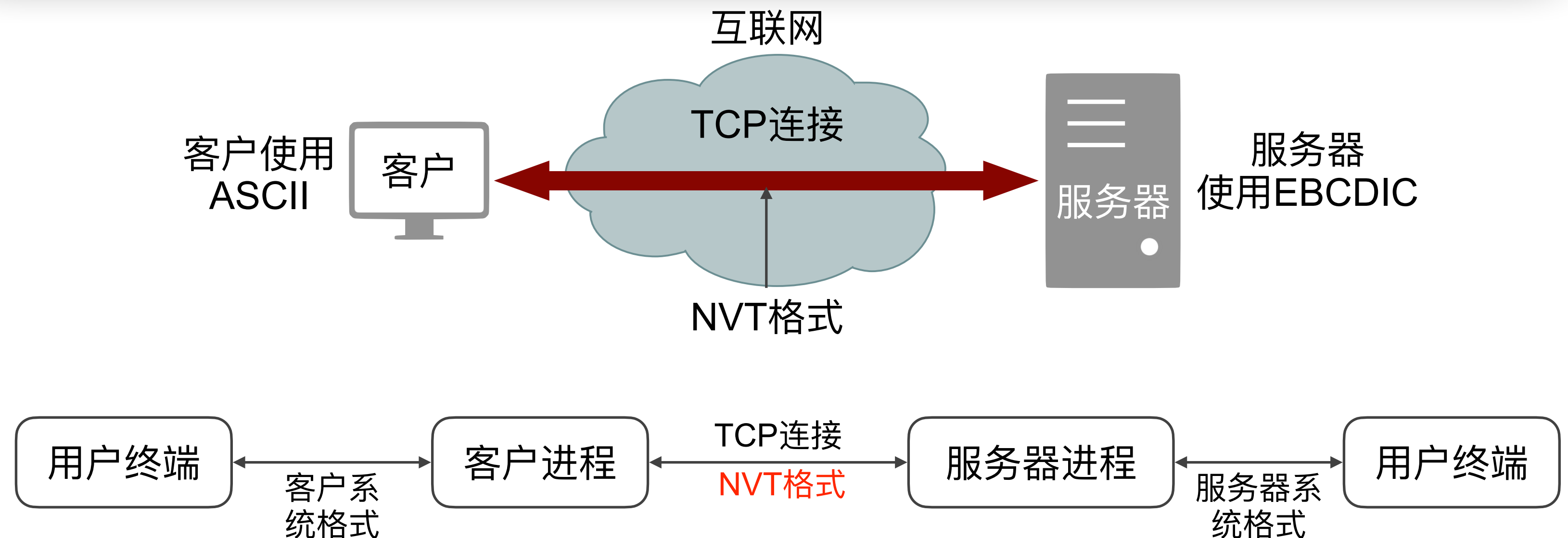
- TELNET 是一个简单的远程终端协议，是互联网正式标准：
 - 用户用 TELNET 就可在其所在地通过 TCP 连接注册（即登录）到远程的另一个主机上（使用主机名或 IP 地址）；
 - TELNET 将用户的击键传到远地主机，并且将远程主机的输出通过 TCP 连接返回到用户屏幕。这种服务是透明的，因为用户感觉到好像键盘和显示器是直接连在远得程主机上；
 - TELNET 使用客户/服务器方式。本地系统运行 TELNET 客户进程，远程主机则运行 TELNET 服务器进程；
 - 服务器中的主进程等待新的请求，并产生从属进程来处理每一个连接。

```
Mac-mini:~ $ telnet towel.blinkenlights.nl
```


网络虚拟终端 NVT 格式

- 应用层
 - 远程终端协议
 - **NVT的概念**
 - 选项协商
 - Telnet应用场景
 - Telnet的安全性
 - SSH

- 异构计算机字符编码方式不同，相互传送字符时，**转换成NVT格式**的字符：
 - 客户软件把用户的击键和命令转换成 NVT 格式，并送交服务器；
 - 服务器软件把收到的数据和命令，从 NVT 格式转换成远地系统所需的格式；
 - 向用户返回数据时，服务器把远地系统的格式转换为 NVT 格式，本地客户再从 NVT 格式转换到本地系统所需的格式。



选项协商

- 应用层
 - 远程终端协议
 - NVT的概念
 - 选项协商
 - Telnet应用场景
 - Telnet的安全性
 - SSH

- NVT使不同的系统互操作，但双方相互并不知道对方可以提供哪些功能。
- 采用双方交互一组选项的方法解决：启用某项功能（选项）时，双方先进行选项协商，使通信的双方明白哪些功能由对方提供，哪些功能无法完成，即在通信时双方可以达成一致，这就是选项协商。
- 控制命令选项协商：
 - 任一方可以在初始化时提出一个选项生效的请求，另一方可以接受，也可以拒绝这一请求。

Telnet选项协商的6种情况

- 应用层
 - 远程终端协议
 - NVT的概念
 - 选项协商
 - Telnet应用场景
 - Telnet的安全性
 - SSH

序号	选项协商格式	说 明
1	发送方 WILL X 接收方 →	发送方向接收方 “我想激活我的选项 X，你是否同意？”
	发送方 DO X 接收方 ←	接收方说 “同意”
2	发送方 WILL X 接收方 →	发送方向接收方 “我想激活我的选项 X，你是否同意？”
	发送方 DON'T X 接收方 ←	接收方说 “不同意”
3	发送方 DO X 接收方 →	发送方向接收方 “可以激活你的选项 X 吗？”
	发送方 WILL X 接收方 ←	接收方说 “同意”
4	发送方 DO X 接收方 →	发送方向接收方 “可以激活你的选项 X 吗？”
	发送方 WON'T X 接收方 ←	接收方说 “不同意”
5	发送方 WON'T X 接收方 →	发送方向接收方 “我想禁止我的选项 X，你是否同意？”
	发送方 DON'T X 接收方 ←	接收方只能说 “同意”
6	发送方 DON'T X 接收方 →	发送方向接收方 “可以禁止你的选项 X 吗？”
	发送方 WON'T X 接收方 ←	接收方只能说 “同意”

选项协商抓包

- 应用层
 - 远程终端协议
 - NVT的概念
 - 选项协商
 - Telnet应用场景
 - Telnet的安全性
 - SSH

Internet Protocol Version 4, Src: 1.1.1.1, Dst: 1.1.1.2
Transmission Control Protocol, Src Port: 23351, Dst Port: 23,...
Telnet

Do Suppress Go Ahead
Will Negotiate About Window Size
Will Remote Flow Control

Internet Protocol Version 4, Src: 1.1.1.2, Dst:1.1.1.1
Transmission Control Protocol, Src Port: 23, Dst Port: 23351, ...
Telnet

Will Echo
Will Suppress Go Ahead
Do Terminal Type
Do Negotiate About Window Size

常见的Telnet应用

- 应用层
 - 远程终端协议
 - NVT的概念
 - 选项协商
 - Telnet应用场景
 - Telnet的安全性
 - SSH

- 连接一个远程网络设备以便获取信息或进行配置。
- 连接一个大型计算机以使用其软硬件资源。
- 连接一个在线论坛以便与其他用户进行交互或通信。
- 安全性：
 - Telnet应用间的信息交换是明码交换过程。

Internet Protocol Version 4, Src: 1.1.1.2, Dst: 1.1.1.1
Transmission Control Protocol, Src Port: 23, Dst Port: 49912, ...
Telnet
Data: \r\n
Data: R1>

SSH (Secure Shell) 与Telnet

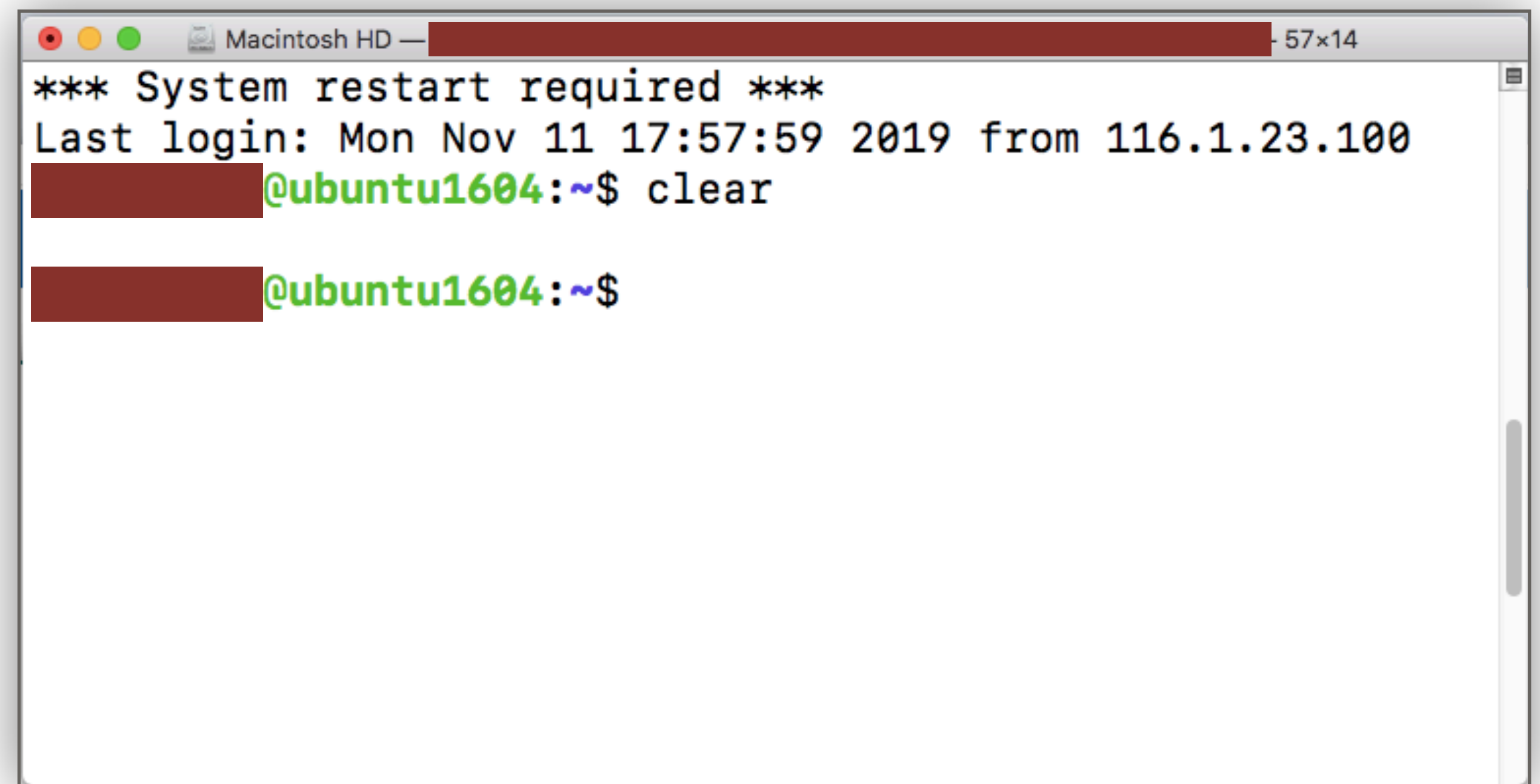
- 应用层
 - 远程终端协议
 - NVT的概念
 - 选项协商
 - Telnet应用场景
 - Telnet的安全性
 - SSH

- 相同点：
 - 远程登录其他主机；
 - 运输层采用TCP协议。
- 不同点：
 - Telnet明文传送；SSH加密传送，且支持压缩；
 - Telnet服务默认端口号为23；SSH服务默认监听22号端口；
 - SSH使用对称加密算法实现数据安全传输；
 - 对称加密算法的密钥是通过非对称加密算法（RSA）进行交换的。

SSH远程登录的实例

- 应用层
 - 远程终端协议
 - NVT的概念
 - 选项协商
 - Telnet应用场景
 - Telnet的安全性
 - **SSH**

MacBook-Air:/ li\$ ssh -p 22 -l 用户名 203.XXX.XXX.156

A terminal window titled "Macintosh HD" with a red title bar. The window shows the output of an SSH command. The text displayed is: "*** System restart required ***", "Last login: Mon Nov 11 17:57:59 2019 from 116.1.23.100", a redacted username followed by "@ubuntu1604:~\$ clear", and another redacted username followed by "@ubuntu1604:~\$".

```
*** System restart required ***
Last login: Mon Nov 11 17:57:59 2019 from 116.1.23.100
[REDACTED]@ubuntu1604:~$ clear

[REDACTED]@ubuntu1604:~$
```

小结

- 应用层
 - 远程终端协议
 - NVT的概念
 - 选项协商
 - Telnet应用场景
 - Telnet的安全性
 - SSH

- NVT的概念。
- 选项协商实例。
- Telnet的应用场景。
- Telnet的安全性。
- SSH。

WWW的问世改变了人们使用互联网的方式

- 应用层
 - 万维网概述
 - 超文本的概念
 - 工作方式
 - 万维网必须解决的几个问题
 - 统一资源定位符URL

- 1989年，蒂姆成功开发出世界上第一个Web服务器和第一个Web客户机。
- 1989年12月，蒂姆为他的发明正式定名为World Wide Web。
- 1991年5月WWW在 Internet上首次露面，立即引起轰动，获得了极大的成功被广泛应用。
- 1991年8月6日 (农历六月廿六)，世界上第一个网页诞生：
 - <http://info.cern.ch>



蒂姆·伯纳斯·李



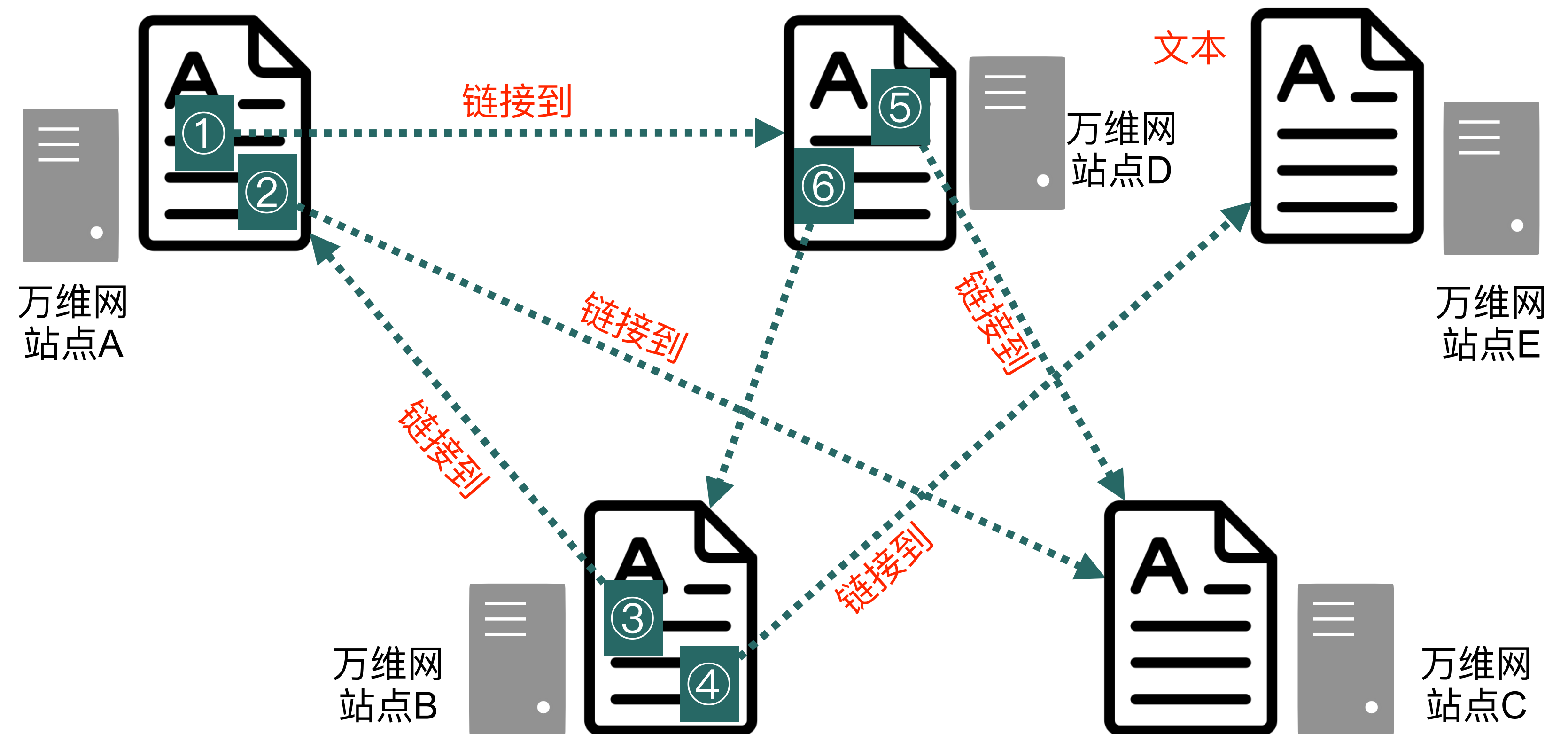
万维网概述

- 应用层
 - 万维网概述
 - 超文本的概念
 - 工作方式
 - 万维网必须解决的几个问题
 - 统一资源定位符 URL

- 万维网 WWW (World Wide Web) 并非某种特殊的计算机网络，它是无数个网络站点和网页的集合，是一个大规模的、联机式的信息储藏所。它们在一起构成了因特网最主要的部分（因特网也包括电子邮件、Usenet以及新闻组）。它实际上是多媒体集合，是由超级链接连接而成的。
- 万维网是分布式超媒体 (hypermedia) 系统，它是超文本 (hypertext) 系统的扩充。
- 一个超文本由多个信息来源链接成。利用一个链接可使用户找到另一个文档。
- 超媒体与超文本的区别：超文本文档仅包含文本信息，超媒体文档还包含其他表示方式的信息，如图形、图像、声音、动画，甚至活动视频图像。

超文本

- 应用层
 - 万维网概述
 - 超文本的概念
 - 工作方式
 - 万维网必须解决的几个问题
 - 统一资源定位符URL



万维网的工作方式

- 应用层
 - 万维网概述
 - 超文本的概念
 - 工作方式
 - 万维网必须解决的几个问题
 - 统一资源定位符 URL
- 万维网以客户/服务器方式工作。
- 浏览器是用户计算机上的万维网客户程序。
- 万维网文档所驻留的计算机则运行服务器程序，称万维网服务器。
- 客户程序向服务器程序发出请求，服务器程序向客户程序送回客户所要的万维网文档。
- 在一个客户程序主窗口上显示出的万维网文档称为页面 (page)。
- 常用的服务器软件：
 - Apache
 - Nginx
 - IIS
 - Tomcat

万维网必须解决的问题

- 应用层
- 万维网概述
- 超文本的概念
- 工作方式
- 万维网必须解决的几个问题
- 统一资源定位符 URL

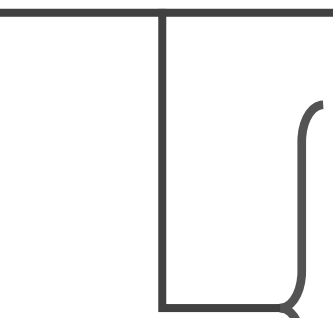
- 如何标志分布在整个互联网上的万维网文档？
 - 使用统一资源定位符 URL 来标志万维网上的各种文档；
 - 使每一个文档在整个互联网的范围内具有唯一的标识符 URL。
- 用什么协议实现万维网上各种超链的连接？
 - 超文本传送协议 HTTP (HyperText Transfer Protocol)；
 - HTTP 是一个应用层协议，它使用 TCP 连接进行可靠的传送。
- 万维网文档如何在互联网上的各种计算机上显示出来，并能清楚地标明超链？
 - 超文本标记语言 HTML 能使得万维网页面的设计者用一个超链从本页面的某处链接到互联网上的任何一个万维网页面，并且能够在自己的计算机屏幕上将这些页面显示出来。
- 用户如何查找所需的信息？
 - 各种的搜索工具（即搜索引擎）。

URL 的一般形式

- 应用层
 - 万维网概述
 - 超文本的概念
 - 工作方式
 - 万维网必须解决的几个问题
- 统一资源定位符
URL

- 由以冒号隔开的两大部分组成，并且在 URL 中的字符对大写或小写没有要求。URL 的一般形式是：
 - 例如：http://www.baidu.com:80/index.html。

<协议>://<主机>:<端口>/<路径>

 **ftp:** 文件传送协议 FTP
http: 超文本传送协议 HTTP
https: 以安全为目标的 HTTP 通道
news: USENET 新闻

- **主机**：从哪台目标主机上获取资源。
- **协议**：指明采用什么协议获取目标主机的资源。
- **端口**：从目标主机中的那个应用进程获取资源。
- **路径**：资源在主机上存放的路径。

本机文件访问与“互联网文件”访问的对比

- 应用层
 - 万维网概述
 - 超文本的概念
 - 工作方式
 - 万维网必须解决的几个问题
 - 统一资源定位符
URL

- Windows系统“完整文件”名称：

`C:\Program Files\VMware\VMware Tools\vmacthlp`

- Linux系统“完整文件”名称：

`[nic@CentOS7 ~]$ cat /etc/passwd`

- 互联网中http协议访问文件的“完整文件”名称：

`http://news.tsinghua.edu.cn:80/publish/thunews/index.html`

采用什么协议

访问哪台主机

访问哪个进程

文件路径

文件名称

URL 的部分内容省略情况

- 应用层
 - 万维网概述
 - 超文本的概念
 - 工作方式
 - 万维网必须解决的几个问题
 - 统一资源定位符 URL
- 有些浏览器为了方便用户，在输入 URL 时，可以把最前面的“http://”甚至把主机名最前面的“www”省略，然后浏览器替用户把省略的字符添上。
- 例如，用户只要键入 ctrip.com，浏览器就自动把未键入的字符补齐，变成http://www.ctrip.com。
- 如果WWW服务器监听熟知端口，则“:80”也可以省略。
- 默认访问的WWW根下的index.html等文件名也可以省略。

小结

- 应用层
 - 万维网概述
 - 超文本的概念
 - 工作方式
 - 万维网必须解决的几个问题
 - 统一资源定位符URL

- WWW概述。
- WWW必须解决的四个问题。
- URL的基本概念。

超文本传送协议 HTTP

- 应用层
 - HTTP协议
- WWW工作流程
- HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
- HTTP事务
- 持续连接

- 为了使超文本的链接能够高效率地完成，需要用 HTTP 协议来传送一切必须的信息。
- 从层次的角度看，HTTP 是面向事务的 (transaction-oriented) 应用层协议，它是万维网上能够可靠地交换文件（包括文本、声音、图像等各种多媒体文件）的重要基础。

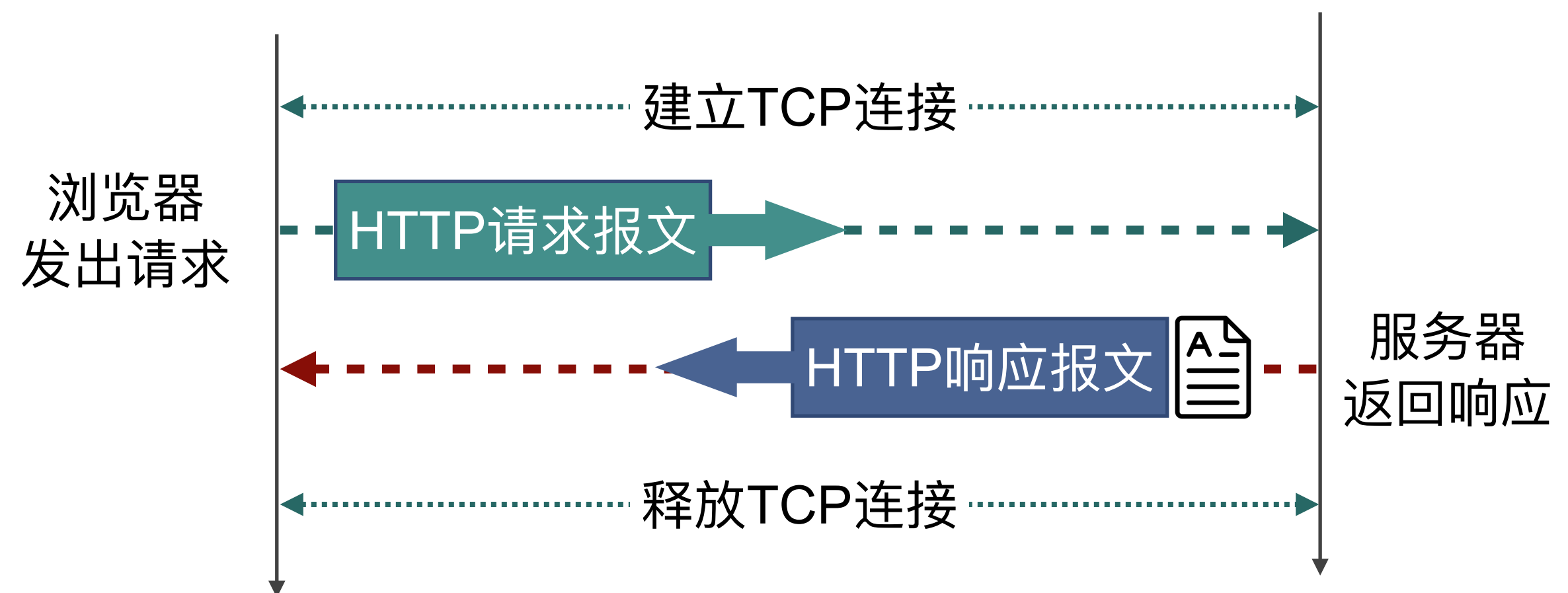
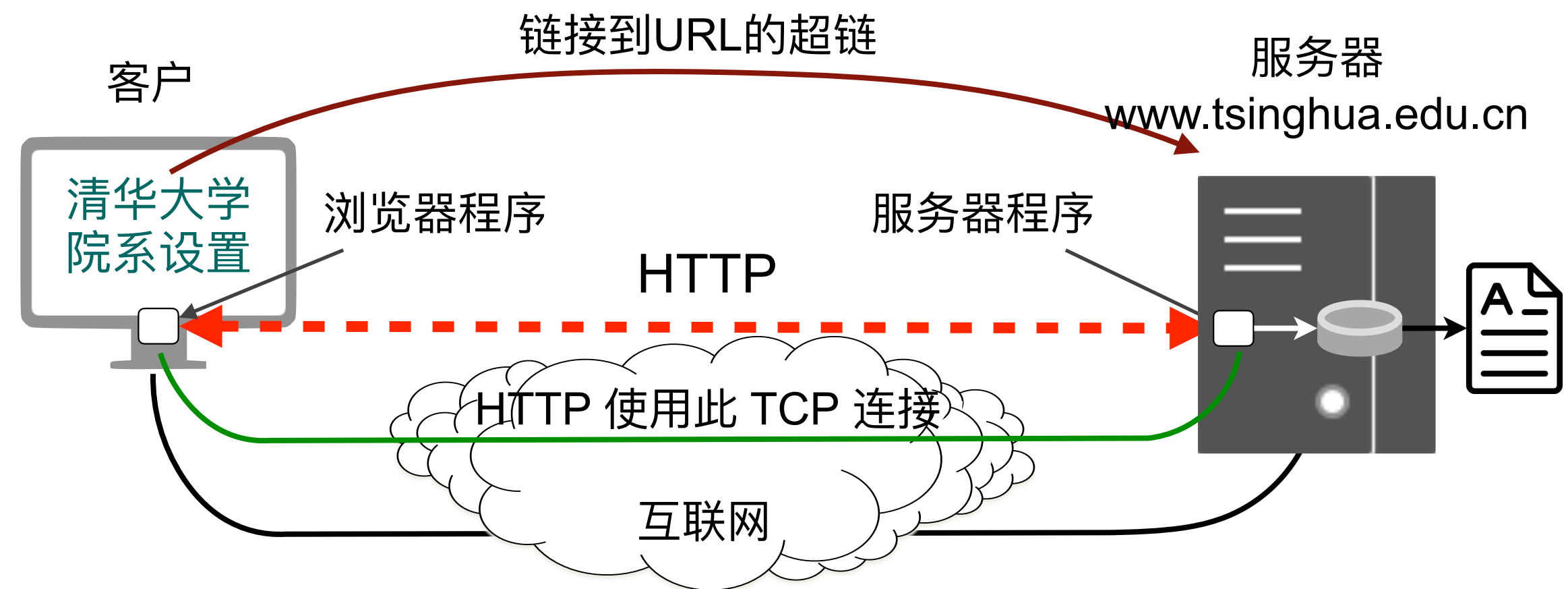
超文本传送协议 HTTP

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- HTTP协议永远都是客户端发起请求，服务器回送响应。
 - 这样就限制了使用HTTP协议，无法实现在客户端没有发起请求的时候，服务器将消息推送给客户端。
- HTTP协议是一个无状态的协议，同一个客户端的这次请求和上次请求是没有对应关系的。

WWW的工作流程

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接



WWW的工作流程

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- 每个WWW服务器运行服务器进程，该进程采用TCP协议，监听TCP 80 端口，不断检测是否有浏览器向它发出连接建立请求。
- 一旦监听到连接建立请求并建立了 TCP 连接之后，客户端浏览器就向万维网服务器发出浏览某个页面的请求，服务器接着就返回所请求的页面作为响应。最后，TCP 连接就被释放了。

用户浏览页面的两种方法：

- URL：在浏览器的地址栏中输入页面的URL；
- 超链接：在页面上鼠标点击某个链接，浏览器自动在互联网上找到链接的页面。

WWW的工作流程

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- 在浏览器和服务端之间的请求和响应的交互，必须按照规定的格式和遵循一定的规则。这些格式和规则就是超文本传送协议 HTTP。
- HTTP 规定在 HTTP 客户与 HTTP 服务器之间的每次交互，都由一个 ASCII 码串构成的请求和响应组成。
- HTTP 报文通常都使用 TCP 连接传送。

HTTP的特点

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- HTTP 使用了面向连接的 TCP 作为运输层协议，保证了数据的可靠传输。
- HTTP 协议本身也是无连接的，虽然它使用了面向连接的 TCP 向上提供的服务。
- HTTP 是面向事务的客户服务器协议。
- HTTP 1.0 协议是无状态的 (stateless)。

HTTP协议本身是无连接的

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- **无连接的含义**：限制**每次连接只处理一个请求**。服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。
- **原因**：早期每个客户端与服务器之间交换数据的**间歇性较大**（即传输具有突发性、瞬时性），并且网页浏览的联想性、发散性导致**两次传送的数据关联性很低**，大部分通道实际上会很空闲、无端占用资源。
- **现在**：网页很复杂，嵌入了很多图片，这时候每次访问图片都需要建立一次 TCP 连接就显得**很低效**。
- **解决办法**：Keep-Alive 被提出用来解决这效率低的问题。

HTTP是无状态的

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- **无状态**：指协议对于事务处理**没有记忆能力**，服务器不知道客户端是什么状态。即给服务器发送 HTTP 请求之后，服务器根据请求，发回数据。发送完，不记录任何信息。
 - HTTP请求都是独立的，Keep-Alive 没能改变这个结果；
 - 如果后续处理需要前面的信息，则必须重传。
- **优点与缺点**：优点减轻了服务器负荷。缺点每次请求可能会传输重复的内容。无状态的特点**严重阻碍了**客户端与服务器进行**动态交互的 Web 应用程序**，例如购物车程序。
- **解决办法**：Cookie和 Session。

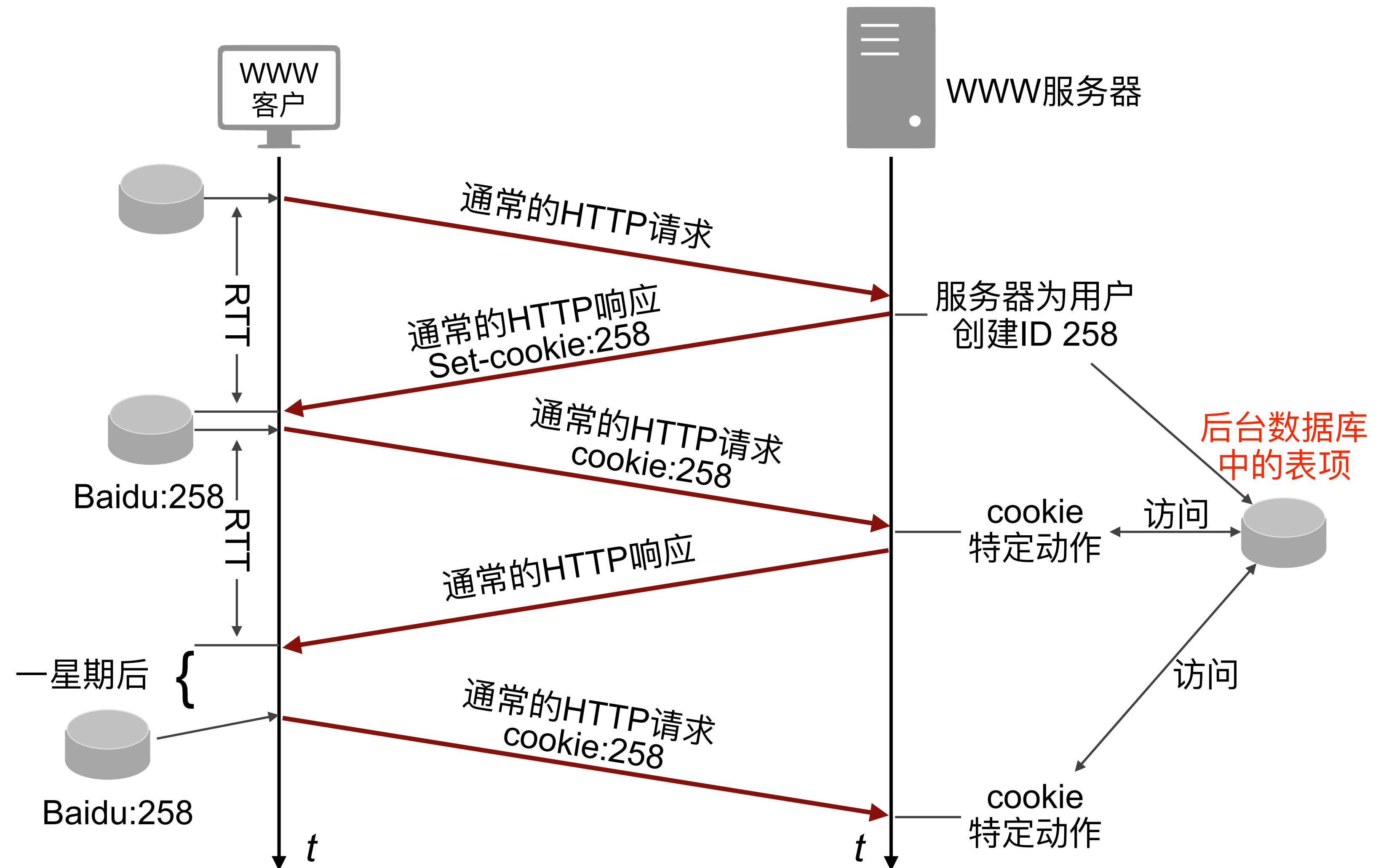
在服务器上存放用户的信息

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- 万维网站点可以使用 Cookie 来跟踪用户：
 - Cookie 表示在 HTTP 服务器和客户之间传递的状态信息。
 - 使用 Cookie 的网站服务器为用户产生一个唯一的识别码。利用此识别码，网站就能够跟踪该用户在该网站的活动。
 - Cookie会被附加在每个HTTP请求中，所以无形中增加了流量。
 - 由于在HTTP请求中的Cookie是明文传递的，所以安全性成问题。（除非用HTTPS）
 - Cookie的大小限制在4KB左右。对于复杂的存储需求来说是不够用的。

cookie

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - **cookie**
 - HTTP事务
 - 持续连接



cookie实例

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

Hypertext Transfer Protocol

HTTP/1.1 200 OK\r\n

Set-Cookie: SSID=tsinghua; domain=tsinghua.cn; path=/; max-age=800 \r\n

Content-Type: text/html\r\n

Date: Thu, 31 Oct 2019 07:36:30 GMT\r\n

Connection: keep-alive\r\n

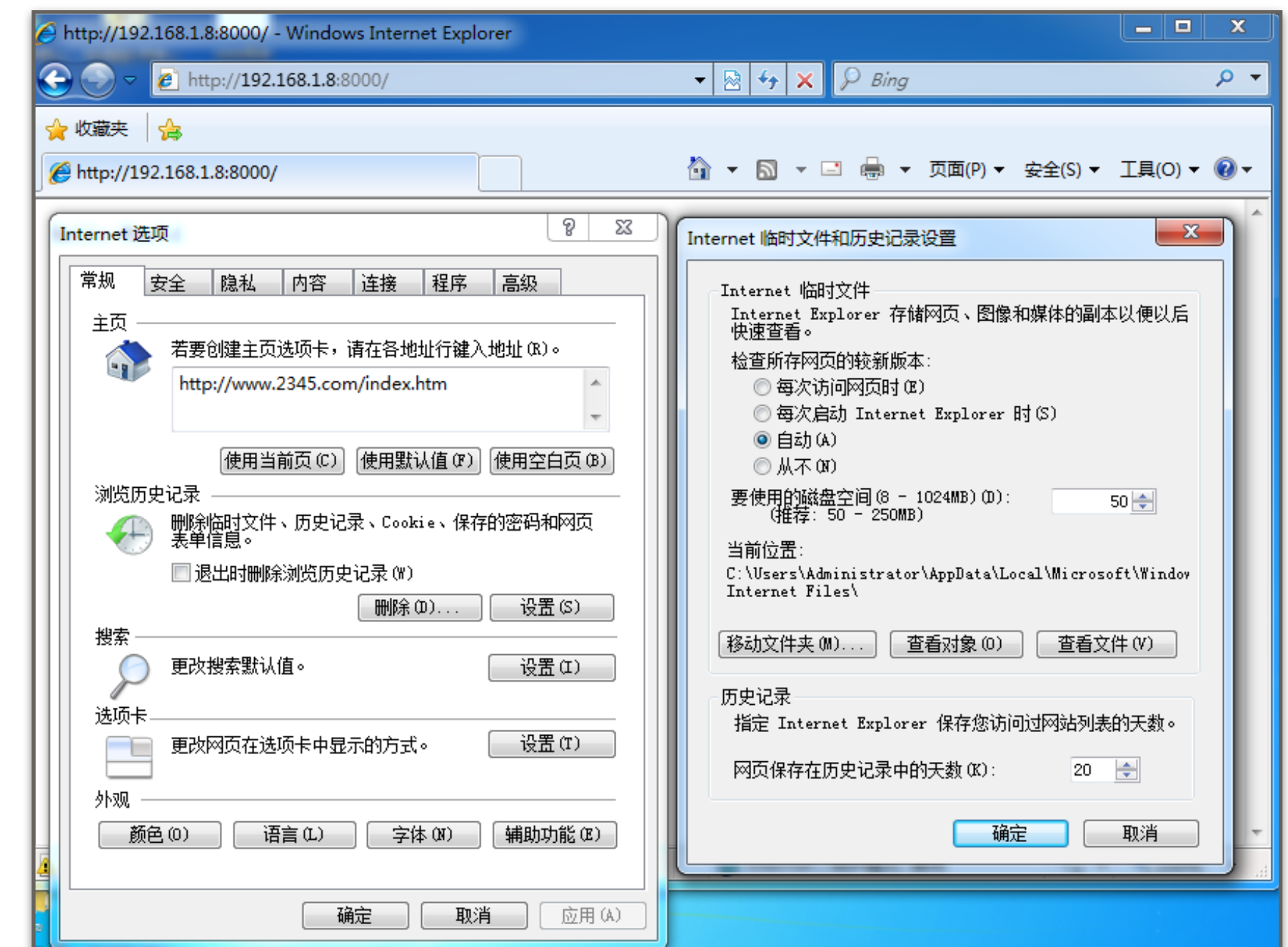
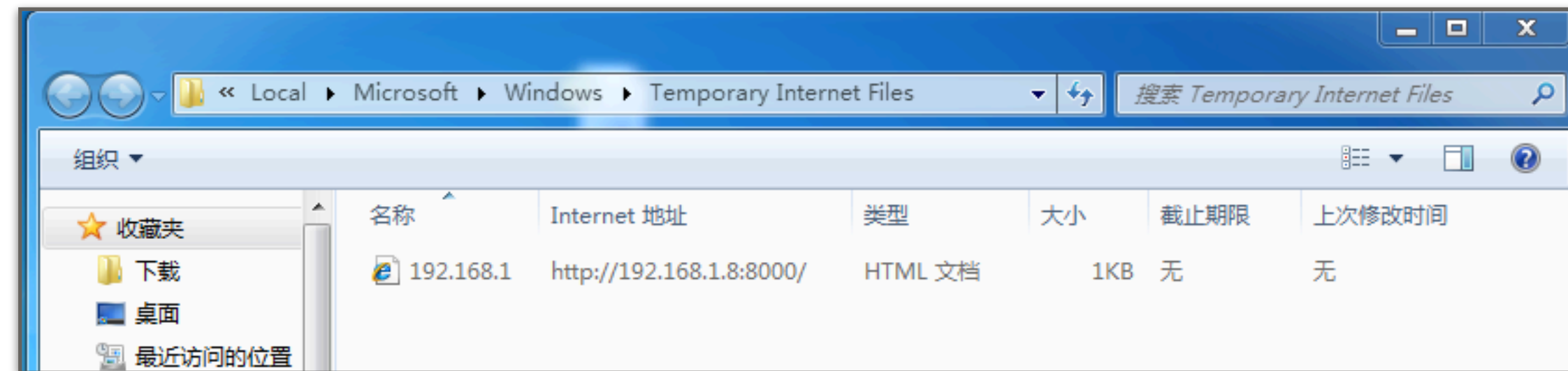
Transfer-Encoding: chunked\r\n

\r\n

Windows 7中Cookie保存位置

C:\Users\Administrator\AppData\Local\Microsoft\Windows\Temporary Internet Files

#注意设置显示隐含文件和文件夹

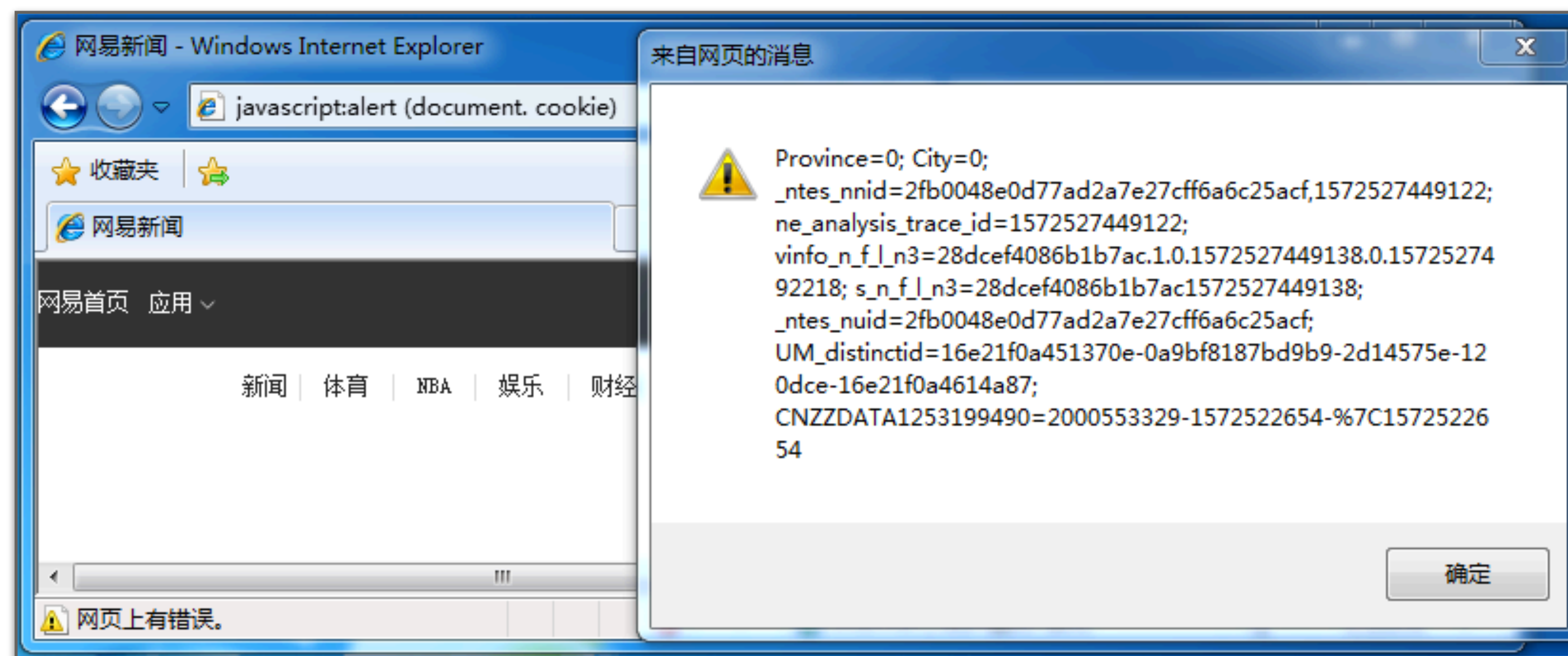


news.163.com的Cookie

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - **cookie**
 - HTTP事务
 - 持续连接

访问news.163.com;

在地址栏中输入: javascript:alert (document. cookie)。



什么是HTTP的事务

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- 一次HTTP操作称为一个事务，其工作过程可分为四步：
 - 首先客户机与服务器需要建立连接。只要单击某个超级链接，HTTP的工作开始；
 - 建立连接后，客户机发送一个请求给服务器；
 - 服务器接到请求后，给予相应的响应信息；
 - 客户端接收服务器所返回的信息通过浏览器显示在用户的显示屏上，然后客户机与服务器断开连接。
- 以上过程中的某一步出现错误，那么产生错误的信息将返回到客户端，由显示屏输出。
- 对于用户来说，这些过程是由HTTP自己完成的，用户只要用鼠标点击，等待信息显示就可以了。

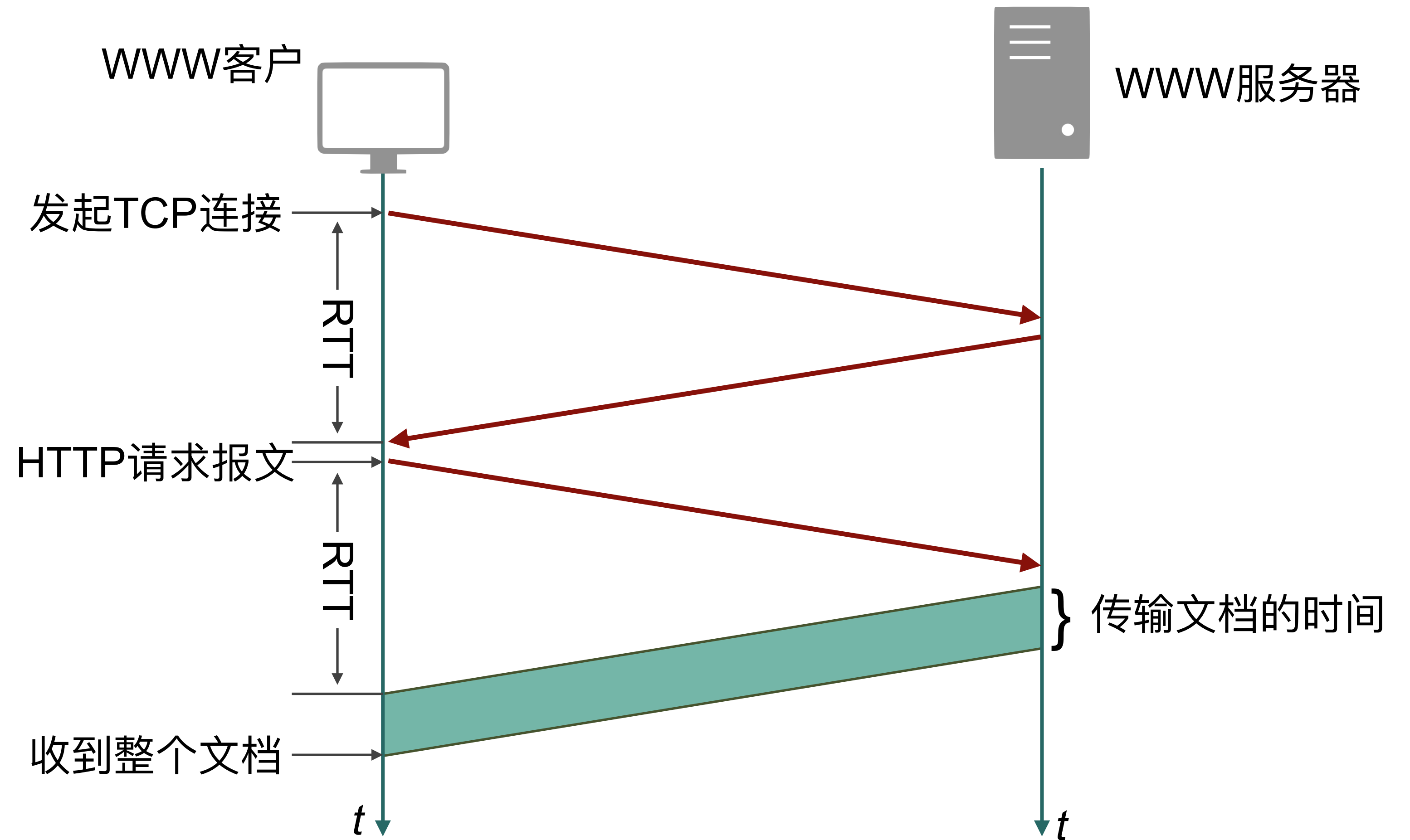
HTTP面向事务的过程

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- 浏览器访问www.baidu.com完整的HTTP事务过程：
 - 客户发出http请求；
 - 服务器响应http请求；
 - 浏览器得到html代码；
 - 浏览器解析html代码，并请求html代码中的资源（如js、css、图片等）；
 - 浏览器对页面进行渲染呈现给用户。

请求一个WWW文档所需的时间

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接



持续连接

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接
- HTTP/1.1 协议使用持续连接 (persistent connection):
 - 万维网服务器在发送响应后仍然在一段时间内保持这条连接，使同一个客户（浏览器）和该服务器可以继续在这条连接上传送后续的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；
 - 目前一些流行的浏览器的默认设置就是使用 HTTP/1.1。

持续连接的两种工作方式

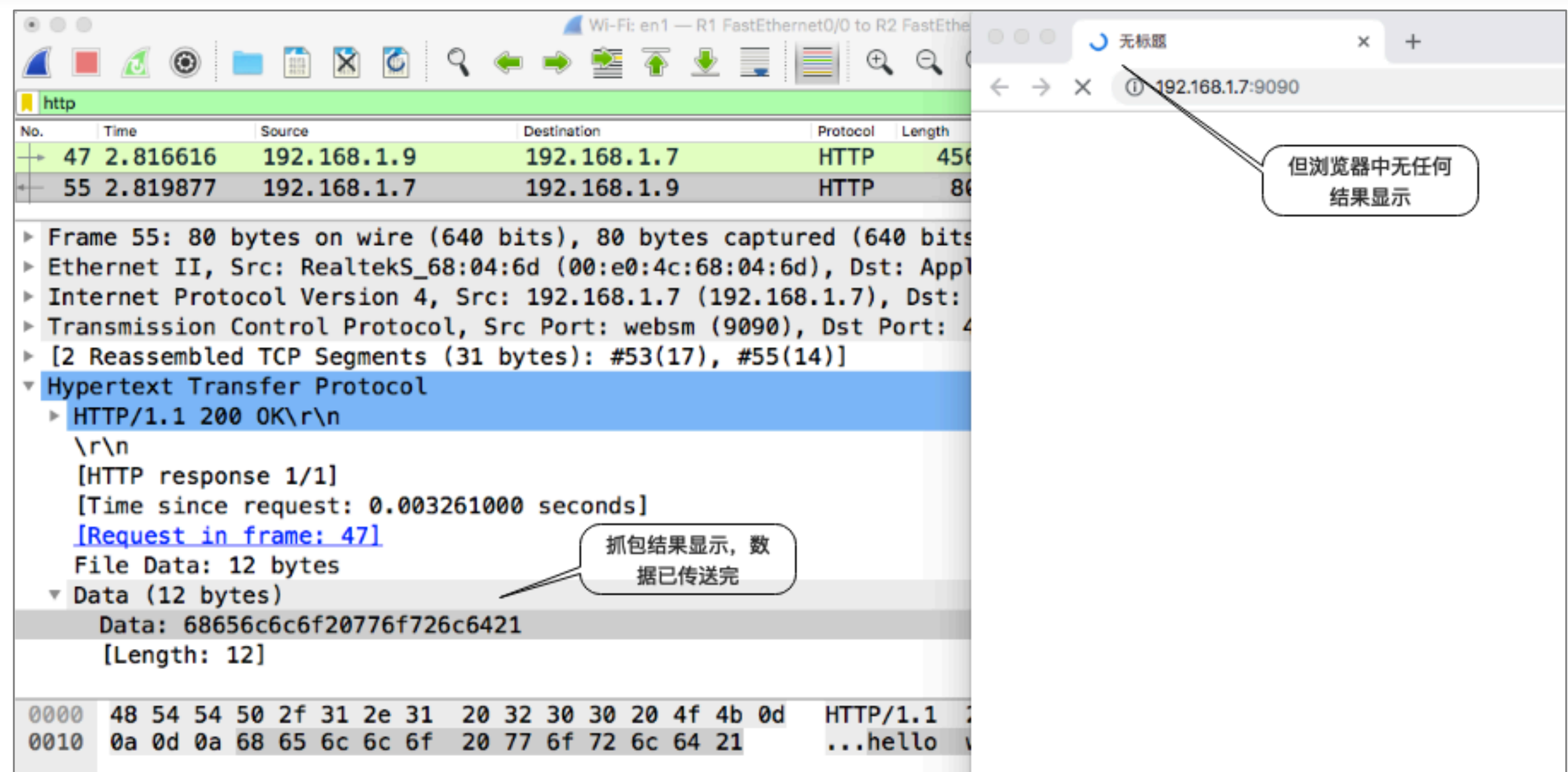
- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- 非流水线方式：
 - 客户在收到前一个响应后才能发出下一个请求。但服务器在发送完一个对象后，其 TCP 连接就处于空闲状态，浪费了服务器资源。
- 流水线方式：
 - 客户在收到 HTTP 的响应报文之前就能够接着发送新的请求报文。一个接一个的请求报文到达服务器后，服务器就可连续发回响应报文。使用流水线方式时，客户访问所有的对象只需花费一个 RTT 时间，使 TCP 连接中的空闲时间减少，提高了下载文档效率。

持续连接带来的新问题

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

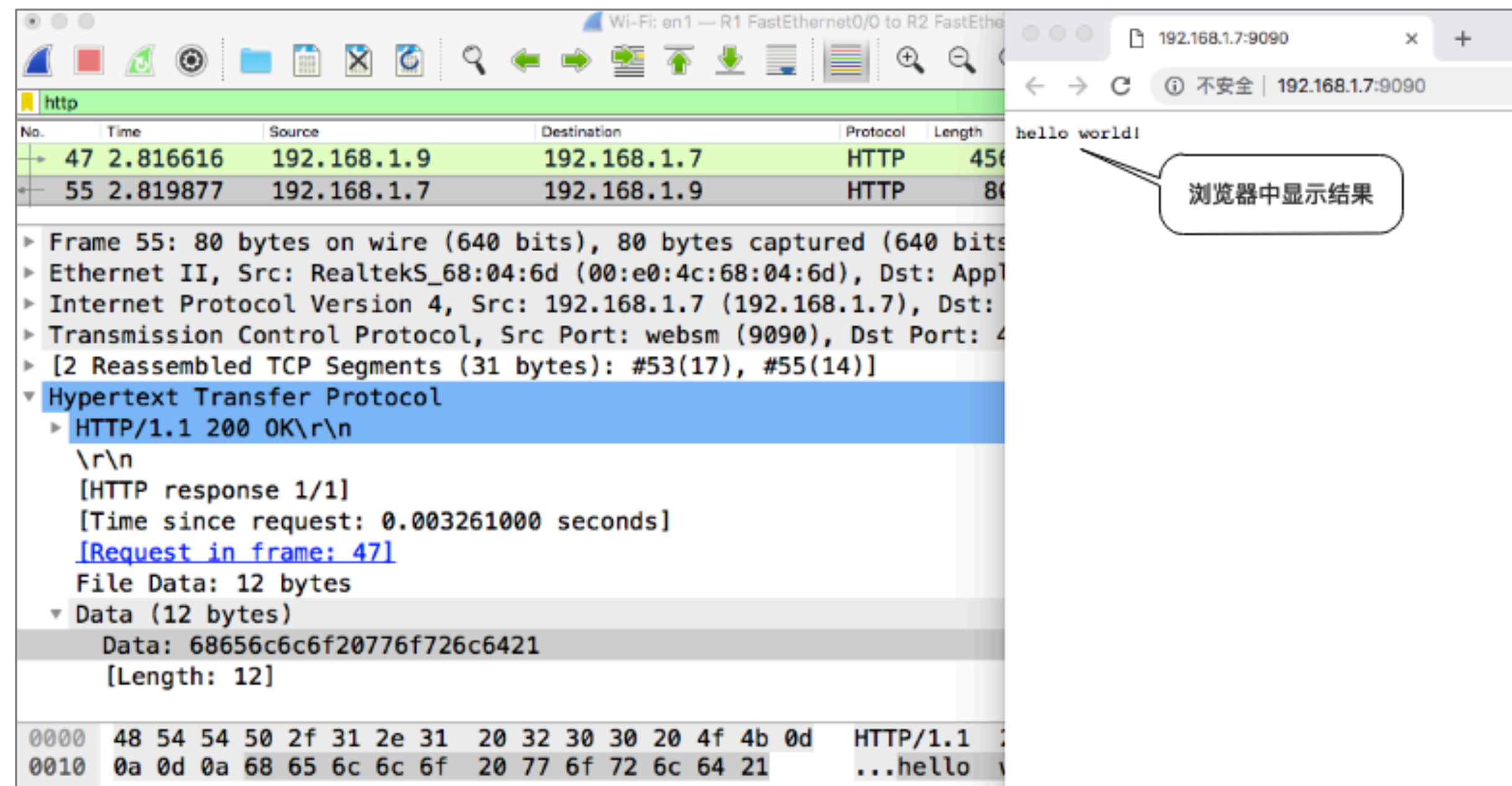
- 在持续连接情况下，客户端发出请求后，服务器发送响应，由于连接没有释放，客户端无法知道服务器数据是否传输完毕，**一直等待**。
- **当服务器进程退出之后**，客户浏览器才会**显示页面内容**。



解决办法

- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - 无连接的
 - 无状态的
 - cookie
 - HTTP事务
 - 持续连接

- 响应时采用Transfer-Encoding: chunked，解决传输数据的边界问题。
- 在HTTP响应头部中，用“Content-Length: 12395”告诉了客户端实体长度为12359字节。

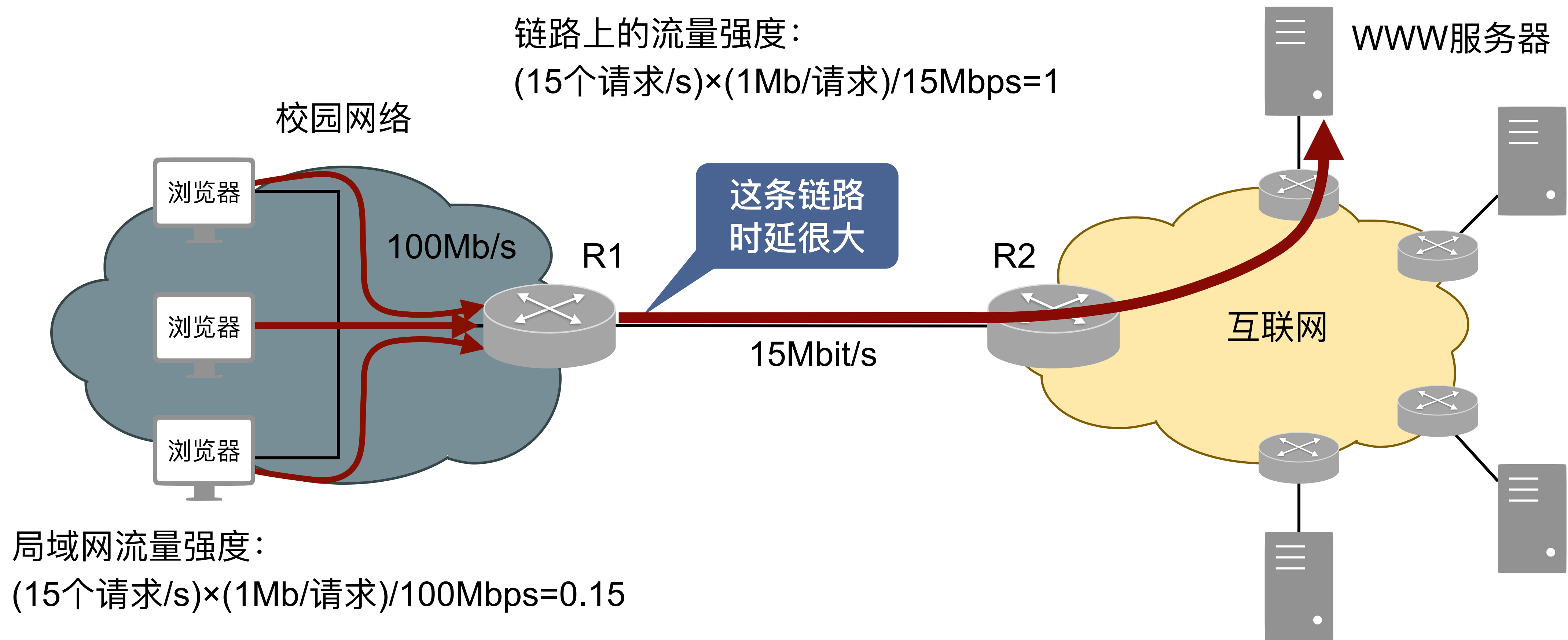


代理服务器

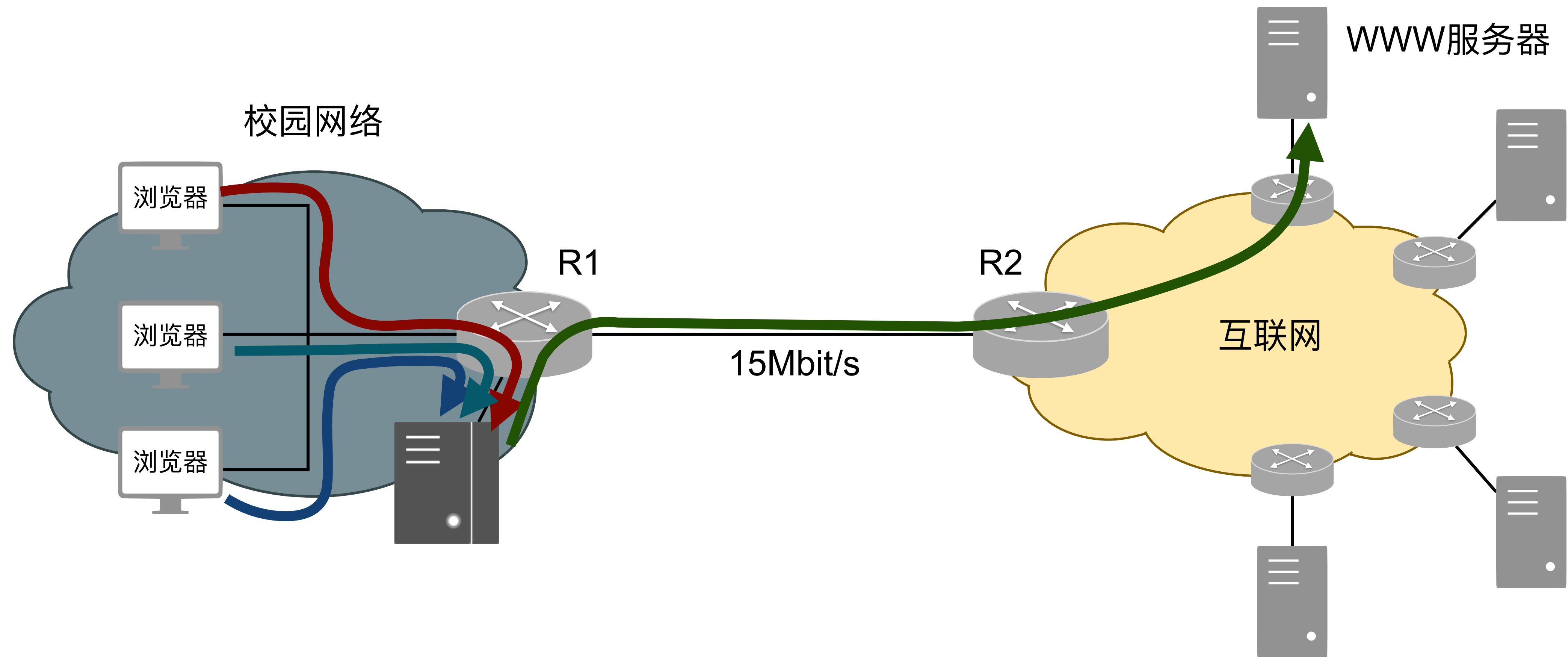
- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - HTTP事务
 - 持续连接
 - 代理服务器

- 代理服务器 (proxy server) 又称为万维网高速缓存 (Web cache), 它代表浏览器发出 HTTP 请求。
- 万维网高速缓存把最近的一些请求和响应暂存在本地磁盘中。
- 当与暂时存放的请求相同的新请求到达时, 万维网高速缓存就把暂存的响应发送出去, 而不需要按 URL 的地址再去互联网访问该资源。

没有使用高速缓存（代理服务器）的情况



使用高速缓存（代理服务器）的情况



小结

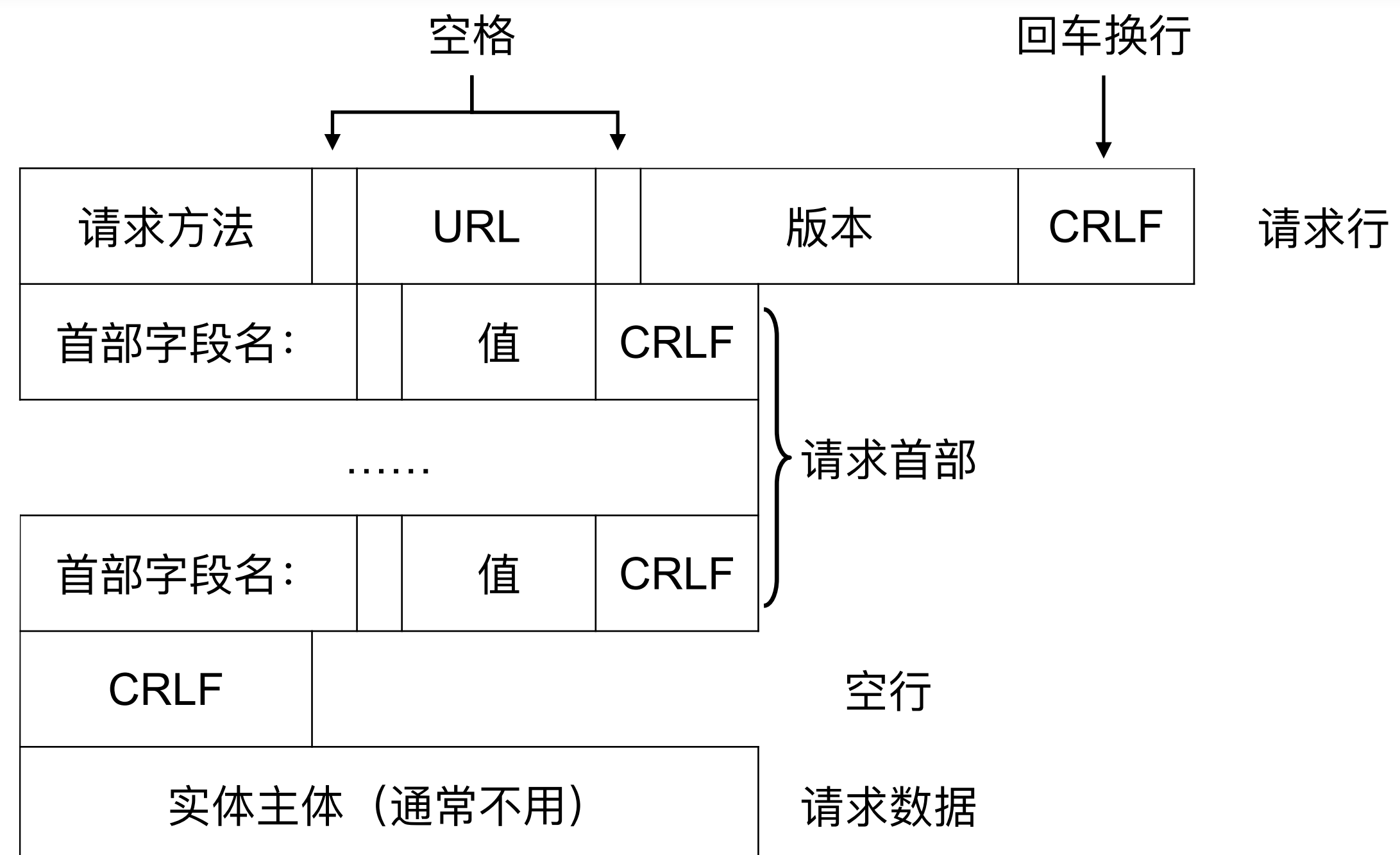
- 应用层
 - HTTP协议
 - WWW工作流程
 - HTTP的特点
 - HTTP事务
 - 持续连接
 - 代理服务器

- HTTP事务。
- HTTP的特点。
- cookie。
- 代理服务器（高速缓存）。

HTTP报文格式（请求报报文）

- 应用层
 - HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

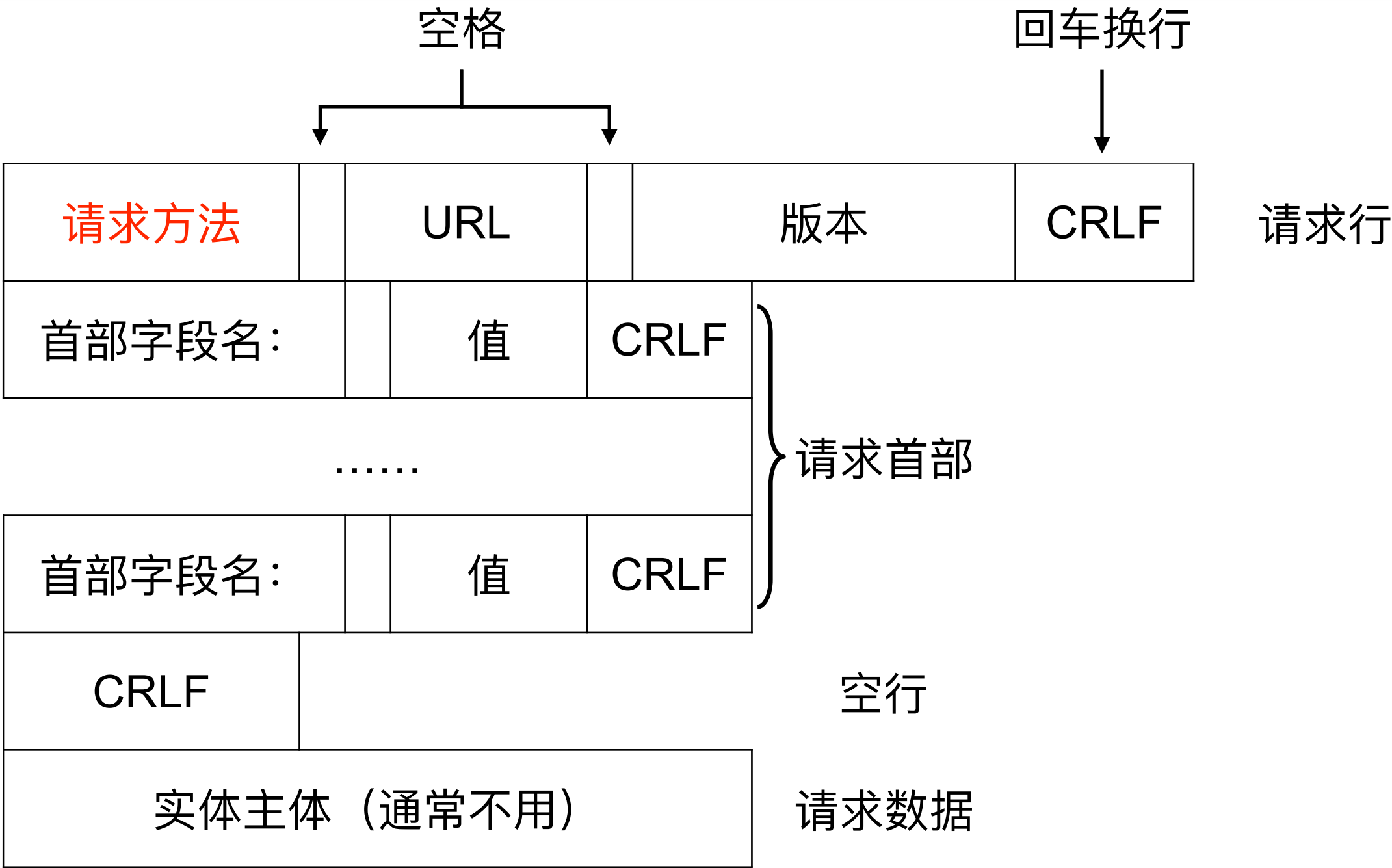
- 报文由三个部分组成：开始行、首部行和实体主体。
- 请求报文中，开始行就是请求行。



请求报文格式（请求方法）

- 应用层
 - HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

“请求方法”就是对所请求的对象进行的操作，因此这些方法实际上也就是
一些命令。因此，请求报文的类型是由它所采用的方法决定的。



请求报文的一些请求方法

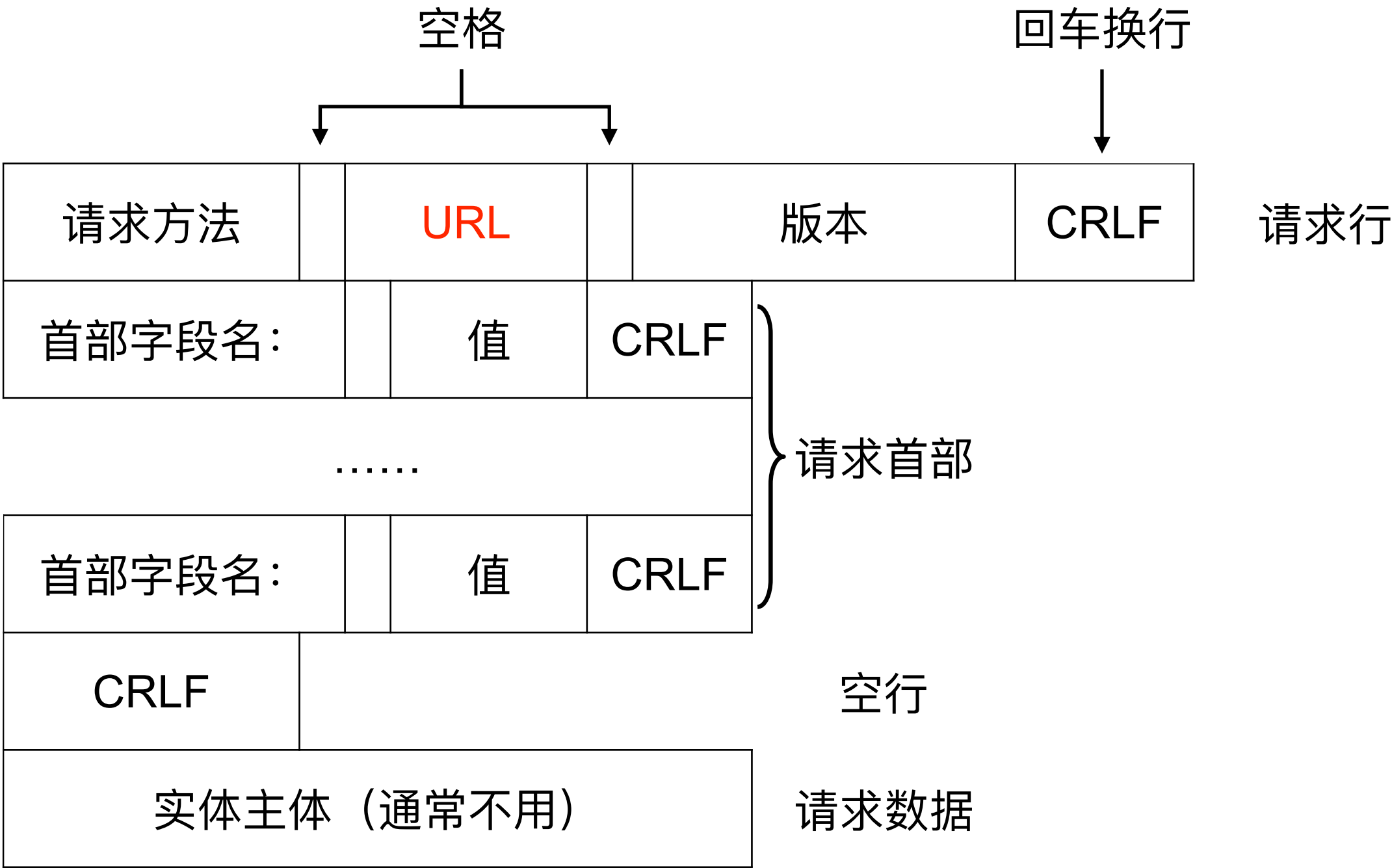
- 应用层
 - HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

- **GET**: 用来请求已被URI识别的资源, 并返回实体主体。
- **POST**: 向指定资源提交数据进行处理 (例如提交表单或者上传文件)。数据被包含在请求体中。POST请求可能会导致新的资源的建立和/或已有资源的修改。
- **HEAD**: 类似于GET请求, 只不过返回的响应中没有具体的内容, 用于获得报文首部。
- **PUT**: 从客户端向服务器传送的数据取代指定的文档的内容。
- **DELETE**: 请求服务器删除指定的页面。
- **OPTIONS**: 允许客户端查看服务器的性能。
- **TRACE**: 回显服务器收到的请求, 主要用于测试或诊断。
- **CONNECT**: HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。

请求报文格式 (URL)

- 应用层
 - HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

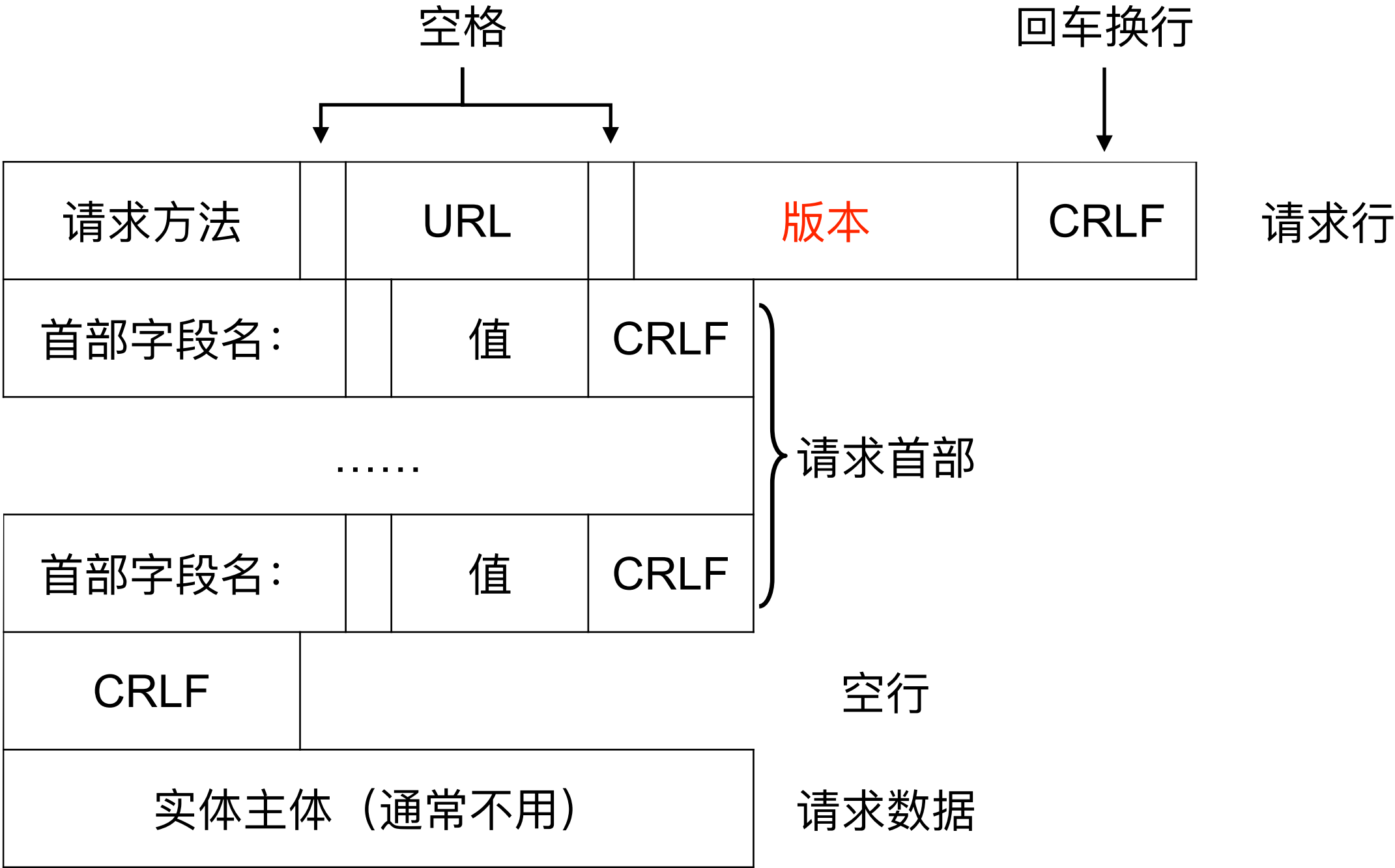
• “URL”是所请求的资源的 URL。



请求报文格式（版本）

- 应用层
 - HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

• “版本”是 HTTP 的版本。



请求报文实例

Hypertext Transfer Protocol

GET / HTTP/1.1\r\n ← 请求行

Accept: application/x-ms-application, image/jpeg, ... \r\n ← 希望接受的数据类型

Accept-Language: zh-CN\r\n ← 页面语言

User-Agent: Mozilla/4.0\r\n ← 浏览器内核和操作系统

UA-CPU: AMD64\r\n ← CPU类型

Accept-Encoding: gzip, deflate\r\n ← 声明浏览器支持的编码类型（压缩算法）

Host: 192.168.1.8:8000\r\n ← 请求服务器的IP地址和端口号

Connection: Keep-Alive\r\n ← 持续连接

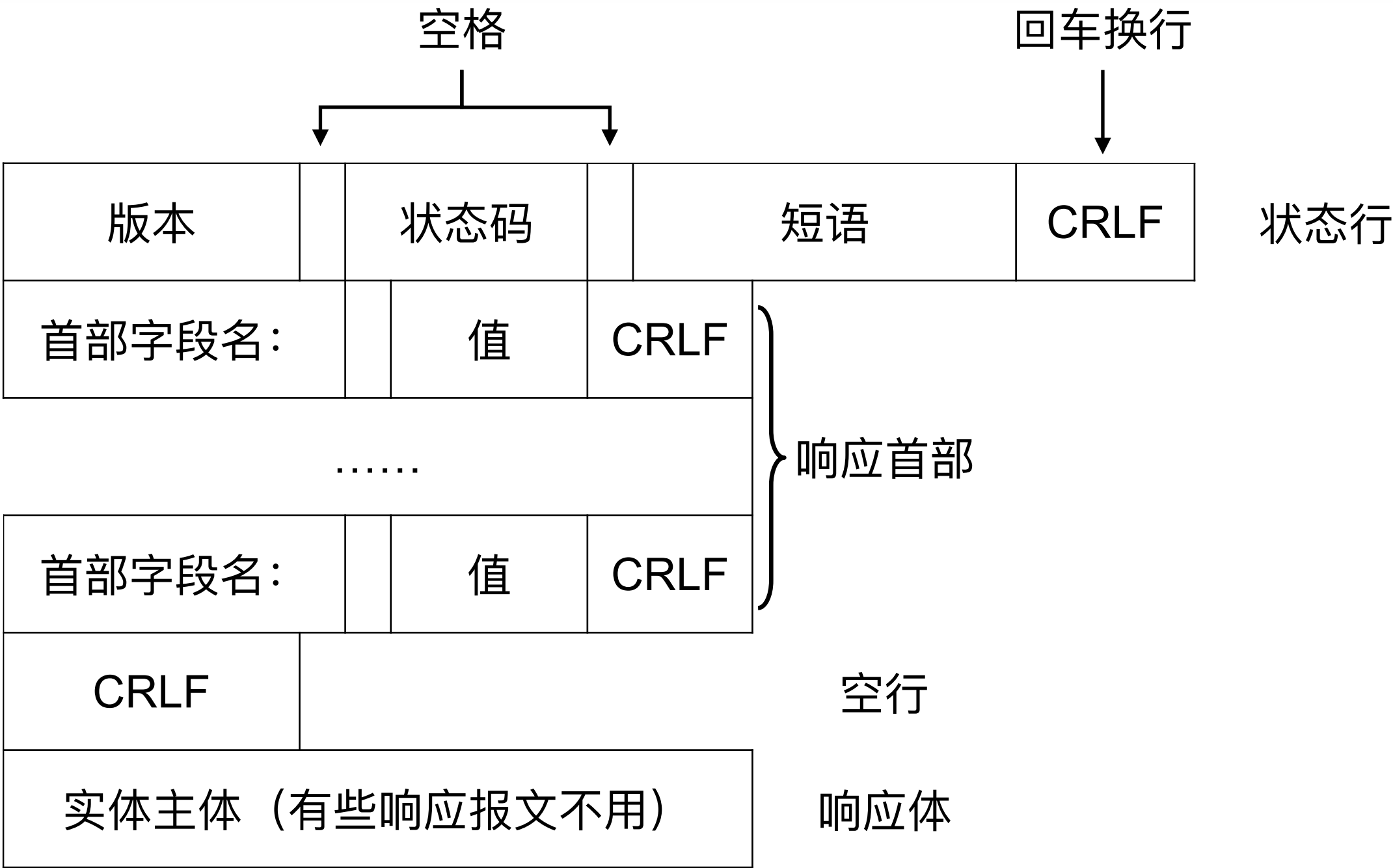
\r\n ← 回车换行

请求首部

响应报文格式

- 应用层
 - HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

- 响应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，以及解释状态码的简单短语。



响应报文格式（状态码）

- 应用层
 - HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

- 状态代码由三位数字组成，第一个数字定义了响应的类别，共分5种类别：
 - 1xx：指示信息——表示请求已接收，继续处理；
 - 2xx：成功——表示请求已被成功接收、理解和接受；
 - 3xx：重定向——要完成请求必须进行更进一步的操作；
 - 4xx：客户端错误——请求有语法错误或请求无法实现；
 - 5xx：服务器端错误——服务器未能实现合法的请求。

响应报文格式（状态码）

- 应用层
- HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

- 一些例子：
 - 200 OK：客户端请求成功；
 - 400 Bad Request：客户端请求有语法错误，不能被服务器所理解；
 - 401 Unauthorized：认证失败，请求未被接受；
 - 403 Forbidden：服务器收到请求，但是拒绝提供服务；
 - 404 Not Found：请求资源不存在，或者输入了错误的URL；
 - 500 Internal Server Error：服务器发生不可预期的错误；
 - 503 Server Unavailable：服务器当前不能处理客户端的请求，一段时间后可能恢复正常。

HTTP 响应报文实例

- 应用层
 - HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

版本		状态码		短语		CRLF	状态行
首部字段名:		值		CRLF			
.....					响应首部		
首部字段名:		值		CRLF			
CRLF		空行					
实体主体（有些响应报文不用）					响应体		

Hypertext Transfer Protocol

响应首部 { HTTP/1.1 200 OK\r\n ← 状态行
Set-Cookie: SSID=Ap4GTEq; max-age=120000 \r\n ← Cookie
Content-Type: text/html\r\n ← 文档类型
Date: Thu, 31 Oct 2019 07:28:10 GMT\r\n ← 时间日期
Connection: keep-alive\r\n ← 持续连接
Transfer-Encoding: chunked\r\n ← 分块传输编码
\r\n ← 回车换行

小结

- 应用层
 - HTTP报文格式
 - 请求报文格式
 - 请求报文实例
 - 响应报文格式
 - 响应报文实例

- HTTP协议结构（语法、语义）。
- 状态码。
- HTTP实例分析。

电子邮件

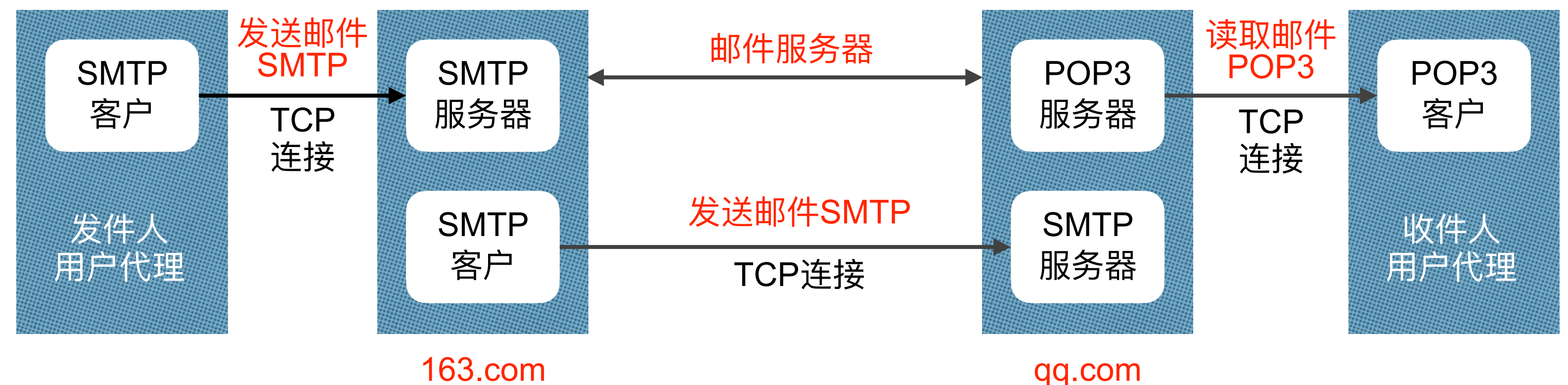
- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- 互联网上最早的应用之一。
- 两个重要标准：
 - 简单邮件传送协议；
 - 互联网报文交换格式。
- 三个主要构件：
 - 用户代理；
 - 邮件服务器；
 - SMTP和POP3协议。

电子邮件的主要构成构件

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- **用户代理 UA**：用户与电子邮件系统的接口，客户端软件。
 - 撰写、显示、处理和通信。
- **邮件服务器的功能**：
 - 发送和接收邮件；
 - 报告邮件传送的情况。
- 邮件服务器采用C/S方式工作，使用SMTP和POP3协议。



工作流程

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- 发件人调用 PC 中的用户代理撰写和编辑要发送的邮件。
- 用户代理把邮件用 SMTP 协议发给发送方邮件服务器。
- SMTP 服务器把邮件临时存放在邮件缓存队列中，等待发送。
- 发送方邮件服务器的 SMTP 客户与接收方邮件服务器的 SMTP 服务器建立 TCP 连接，然后就把邮件缓存队列中的邮件依次发送出去。
- 接收方邮件服务器将收到的邮件放入接收者邮箱。
- 接收方用户代理采用POP3（或IMAP）协议从接收方邮件服务器取回邮件。

简单邮件传输协议SMTP

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- 采用TCP协议，**监听25号端口**。
- **不使用中间服务器**：
 - 发送方和接收邮件服务器直接建立TCP连接；
 - **发送邮件不需要鉴别**；
 - SMTP采用ASCII**明文传送**。
- **三个阶段**：
 - 连接建立；
 - 邮件传送；
 - 连接释放。

电子邮件的组成

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- 电子邮件由信封 (envelope) 和内容 (content) 两部分组成。
- 电子邮件的传输程序根据邮件信封上的信息来传送邮件。
- 用户在从自己的邮箱中读取邮件时才能见到邮件的内容。
- 在邮件的信封上，最重要的就是收件人的地址。

【哔哩哔哩】账号安全中心-改密操作提醒

发件人：哔哩哔哩 <verify@service.bilibili.com>

时 间：2020年4月3日(星期五) 上午10:18

收件人：XXXX <XXXX6@qq.com>

尊敬的用户，您好：

您在2020-04-03 10:18:27修改了您的登录密码，请使用新密码登录。如有疑问，请及时联系客服人员。

祝在【哔哩哔哩】收获愉快！

.....

电子邮件的组成：信封

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- Subject: 主题
- From: 发件人
- Date: 发送日期
- to: 电子邮箱地址格式: 用户名@邮件服务器域名
- CC: 抄送
- Reply-to: 回复地址

电子邮件发送举例

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- 建立TCP连接
- 发送邮件
- 释放连接

- Mac-mini:~ \$ telnet smtp.qq.com 25
- 参考网址: <https://www.cnblogs.com/cthon/p/9151467.html>

邮件读取协议POP3和IMAP

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- 两个常用的邮件读取协议：
 - POP3：邮局协议 (Post Office Protocol) 第3个版本
 - IMAP：网际报文存取协议 (Internet Message Access Protocol)

邮件读取协议POP3和IMAP

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- POP3协议：
 - 邮局协议(Post Office Protocol)第3个版本
 - 简单
 - 采用TCP协议，服务器监听110端口；
 - 仅用于客户从服务器上取邮件；
 - 需要鉴别；
 - 客户取回邮件，服务器即删除该邮件。

邮件读取协议POP3和IMAP

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- IMAP协议：
 - 网际报文存取协议(Internet Message Access Protocol);
 - 采用TCP协议，服务器监听143端口；
 - 联机协议，类似在本地操作（可创建管理文件夹）；
 - 连接IAMP仅下载邮件首部，打开邮件后传送至客户用户代理；
 - 允许收信人只读取邮件中的某一个部分；
 - 用户可以搜索邮件内容。

POP3和IMAP比较

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

操作位置	操作内容	IMAP	POP3
收件箱	阅读、标记、移动、删除邮件等	客户端与邮箱更新同步	仅在客户端内
发件箱	保存到已发送	客户端与邮箱更新同步	仅在客户端内
创建文件夹	新建自定义的文件夹	客户端与邮箱更新同步	仅在客户端内
草稿	保存草稿	客户端与邮箱更新同步	仅在客户端内
垃圾文件夹	接收并移入垃圾文件夹的邮件	支持	不支持
广告邮件	接收并移入广告邮件夹的邮	支持	不支持

SMTP与POP3、IMAP的作用范围

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- SMTP协议：
 - 用户代理与服务器之间；
 - 服务器与服务器之间。
- POP3、IMAP：
 - 用户代理与服务器之间。

通用互联网邮件扩充MIME

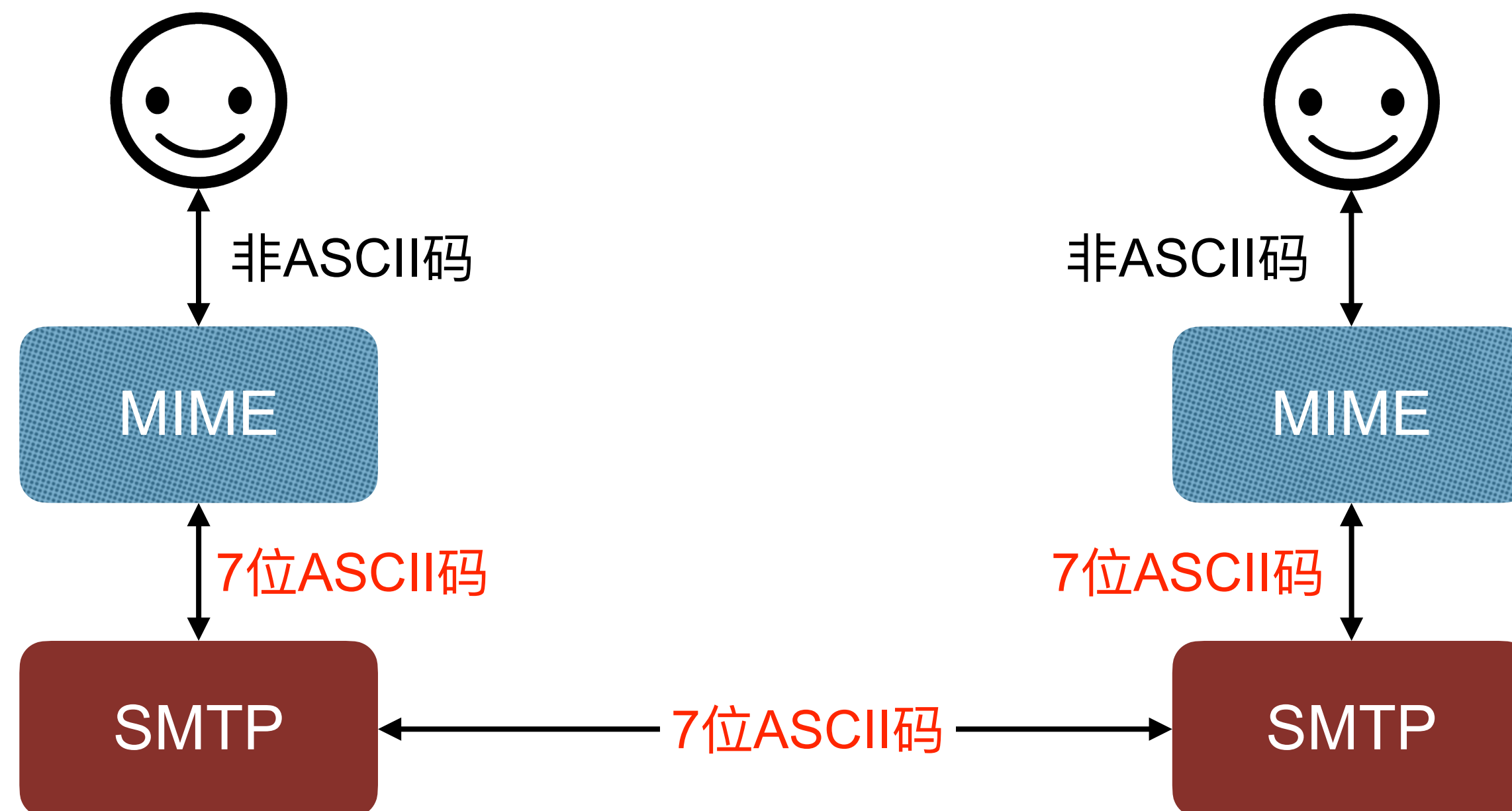
- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- SMTP协议在的问题：
 - 发送邮件不需要鉴别；
 - 不能传送二进制文件；
 - SMTP采用ASCII明文传送；
 - 拒绝超过一定长度的邮件；
 - 某些SMTP实现未完全遵循SMTP标准。

通用互联网邮件扩充MIME

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - **MIME**

- 通用互联网邮件扩充 MIME **并没有改动** SMTP 或取代它。
- MIME 的意图是继续使用目前的 [RFC 822] 格式，但**增加了邮件主体的结构**，并定义了**传送非 ASCII 码**的编码规则。



MIME三个部分

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- 5 个新的邮件首部字段，它们可包含在原有首部中。这些字段提供了有关邮件主体的信息。
- 定义了许多邮件内容的格式，对多媒体电子邮件的表示方法进行了标准化。
- 定义了传送编码，可对任何内容格式进行转换，而不会被邮件系统改变。

MIME五个新的首部

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - **MIME**

MIME 首部	<p>MIME-Version: MIME 的版本。若无此行, 则为英文文本。</p> <p>Content-Description: 这是可读字符串, 此邮件的说明。</p> <p>Content-Id: 邮件的唯一标识符。</p> <p>Content-Transfer-Encoding: 传送时邮件主体使用的编码方法。</p> <p>Content-Type: 邮件内容类型 / 子类型。</p>
邮件主体	

内容传送编码(Content-Transfer-Encoding)

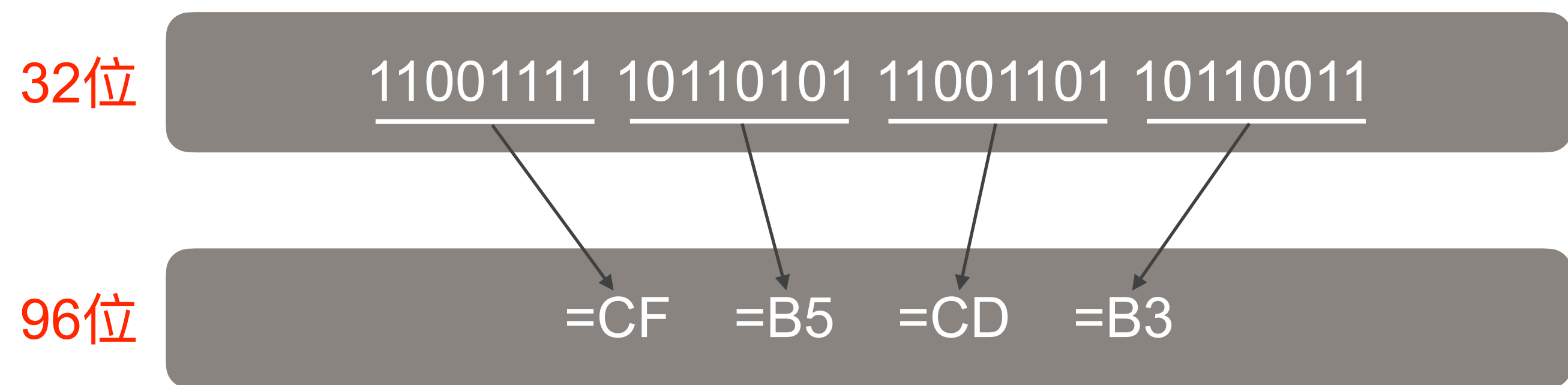
- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - **MIME**

编码方法	说明
7bit	7 位 ASCII 编码，每行不能超过 1000 个字符（包括回车和换行）。缺省编码方法。
8bit	8 位非 ASCII 编码，每行不能超过 1000 个字节（包括回车和换行）。
Binary	8 位非 ASCII 编码，任意长度的字节串。
Base64	将任意长度的字节串转换为用 7 位 ASCII 编码表示的字符串。可用于二进制和非文本数据的编码。
Quoted-printable	将任意长度的字节串转换为 ASCII 编码表示的字符串。可用于二进制和非文本数据的编码。

Quoted-printable 编码

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - **MIME**

- 适用于所传送的数据中只有少量的非 ASCII 码的情况：
 - 除“=”外，ASCII不改变；
 - 不可打印的ASCII和非ASCII：
 - 每个字节表示为16进制；
 - 前面加上“=”。



原来 1 个字节，现在需要 3 个字节，开销=200%)

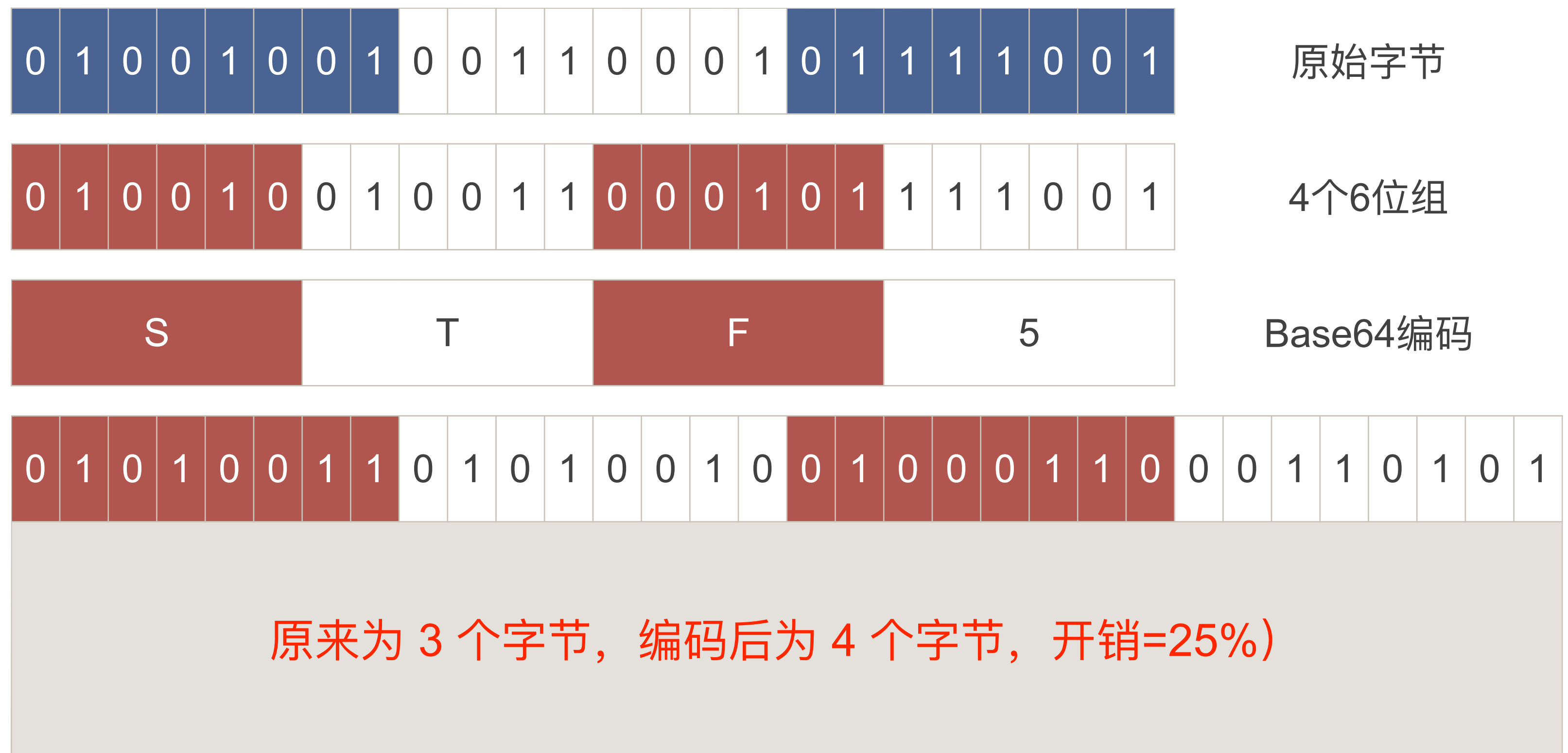
MIME: base64编码 (适合任意长度的二进制数据)

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - **MIME**

码值	字符	码值	字符	码值	字符	码值	字符
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

MIME: base64编码

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - **MIME**



MIME：内容类型

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- MIME 标准规定 Content-Type 说明必须含有两个标识符，即内容类型(type)和子类型(subtype)，中间用“/”分开。
- MIME 标准原先定义了 7 个基本内容类型和 15 种子类型。
- MIME允许发件人和收件人自己定义专用的内容类型。但为避免可能出现名字冲突，标准要求为专用的内容类型选择的名称要以字符串 X-开始。

MIME：内容类型

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - **MIME**

内容类型	子类型举例	说明
text（文本）	plain, html, xml, css	不同格式的文本
image（图像）	gif, jpeg, tiff	不同格式的静止图像
audio（音频）	basic, mpeg, mp4	可听见的声音
video（视频）	mpeg, mp4, quicktime	不同格式的影片
model（模型）	vrml	3D模型
application（应用）	octet-stream, pdf, javascript, zip	不同应用程序产生的数据
message（报文）	http, rfc822	封装的报文
multipart（多部分）	mixed, alternative, parallel, digest	多种类型的组合

MIME：内容类型举例

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - **MIME**

- From: nic@xxx.edu.cn
- To: zhnic@xxx.com
- Subject: photo
- **MIME-Version: 1.0**
- Content-Transfer-Encoding: **base64**
- Content-Type: **image/jpeg**
- • base64 encoded data ...
-
- ...base64 encoded data

小结

- 应用层
 - 概述
 - 工作流程
 - SMTP协议
 - 邮件的组成
 - POP3和IMAP
 - MIME

- 三个主要构件：
 - 用户代理；
 - 邮件服务器；
 - SMTP和POP3协议。
- 工作流程。
- MIME。

动态主机配置协议DHCP

- 应用层
 - 配置协议参数
- DHCP协议
- 中继代理
- 租用期
- DHCP工作流程

- 主机接入互联网络需配置协议参数：
 - IP 地址；
 - 子网掩码；
 - 默认路由器的 IP 地址（默认网关）；
 - 域名服务器的 IP 地址。
- 普通用户配置协议参数易出错。

动态主机配置协议DHCP

- 应用层
 - 配置协议参数
 - **DHCP协议**
 - 中继代理
 - 租用期
 - DHCP工作流程

- 动态主机配置协议 DHCP提供了**即插即用连网的机制**:
 - 这种机制允许一台计算机加入新的网络和并自动获取 IP 地址;
 - DHCP为服务器 (Web、Email等) 、且位置固定的计算机指派一个永久地址;
 - DHCP为客户端的计算机分配一个临时地址。

DHCP采用客户/服务器模式

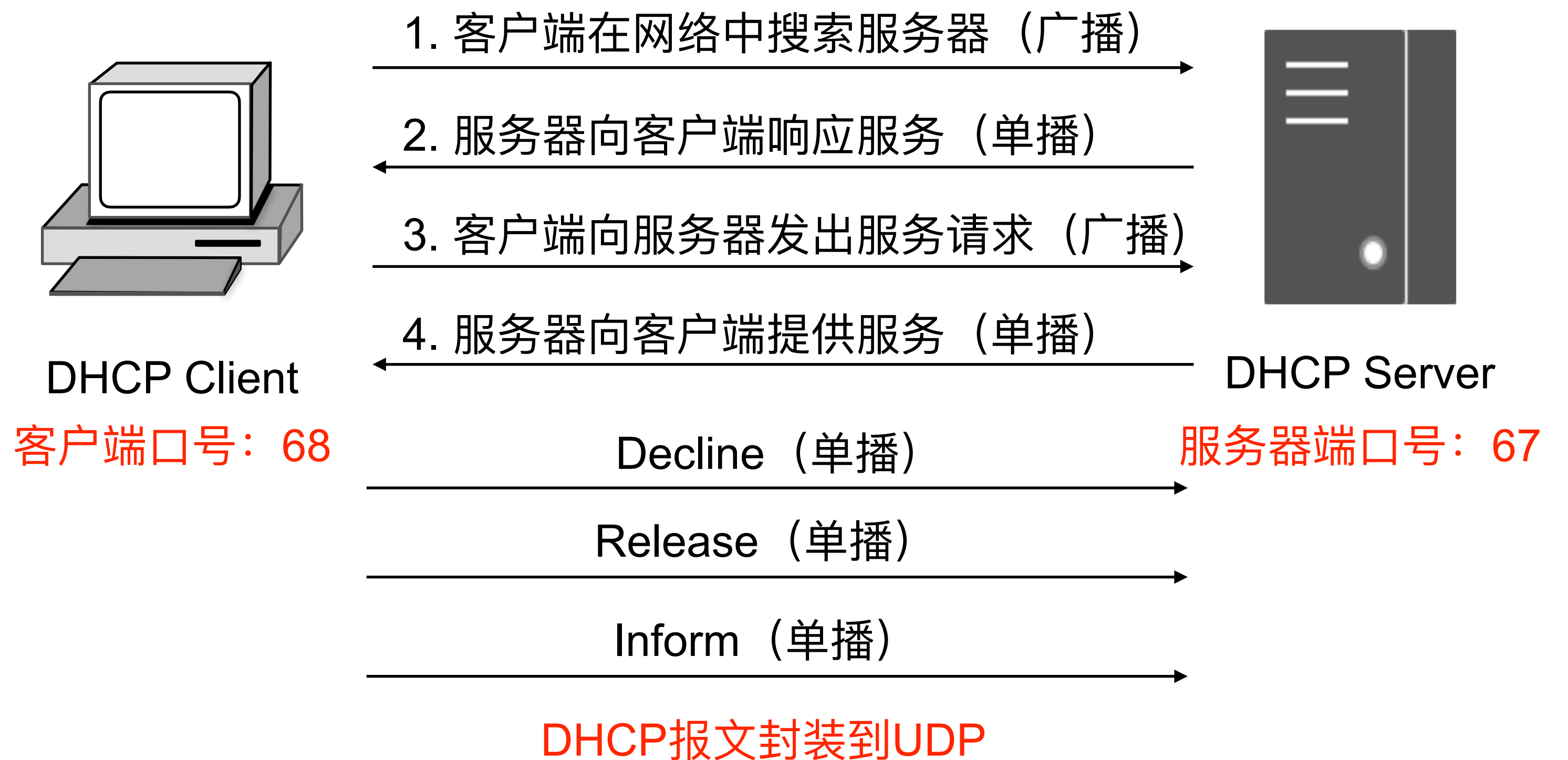
- 应用层
 - 配置协议参数
 - **DHCP协议**
 - 中继代理
 - 租用期
 - DHCP工作流程

- 需要IP地址的主机在启动时广播发送**DHCP发现报文**：
 - 该主机就成为 DHCP 客户。
- 本地网络上DHCP**服务器回答此广播报文**：
 - 服务器首先在其数据库中查找该计算机的配置信息；
 - 若找到，则返回找到的信息；
 - 否则，从服务器的IP地址池中取一个地址分配给该计算机；
 - 服务器的回答的报文称为提供报文（DHCPOFFER）。

DHCP采用客户/服务器模式

- 应用层
 - 配置协议参数
 - **DHCP协议**
 - 中继代理
 - 租用期
- DHCP工作流程

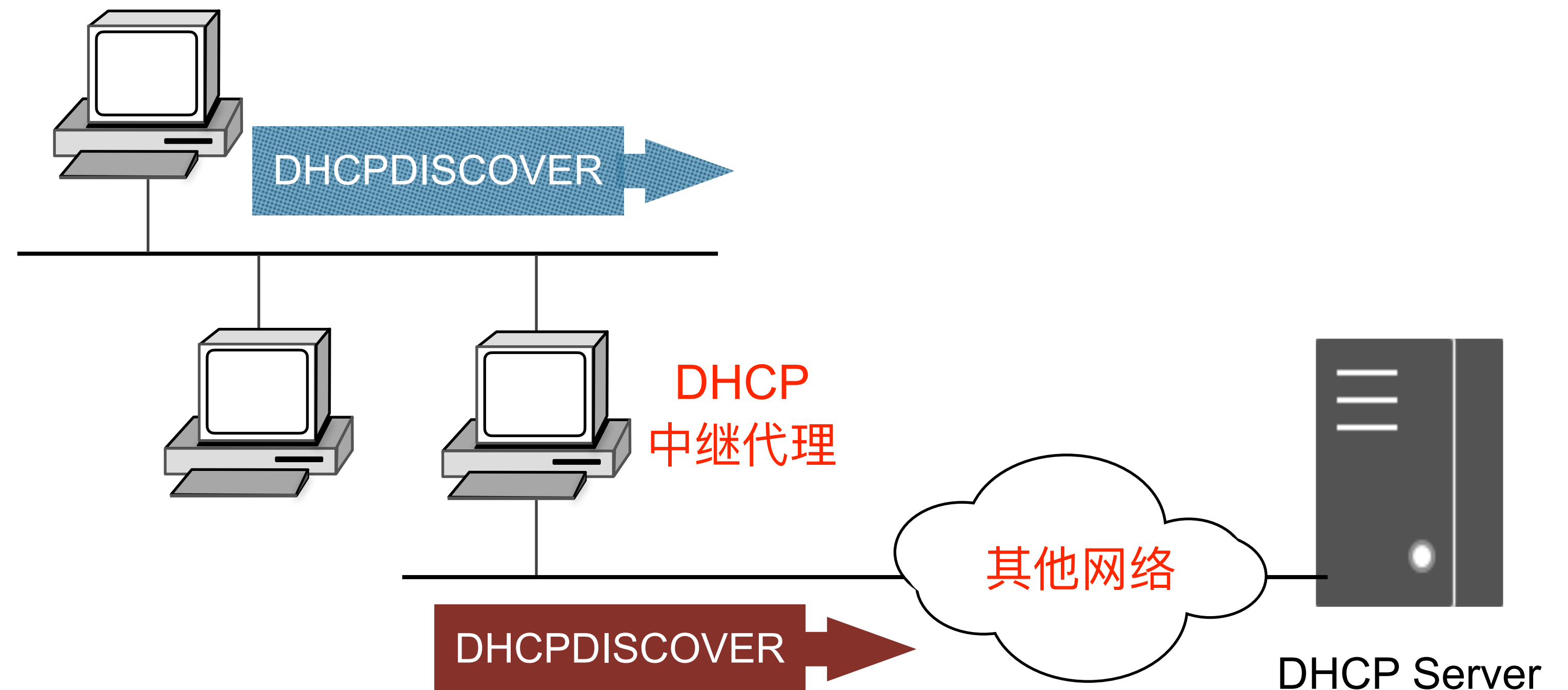
- 注意：本网络上的所有DHCP服务器都会发送回。
- DHCP客户选用收到的第一个DHCP回答来请求IP地址。



DHCP中继代理

- 应用层
 - 配置协议参数
 - DHCP协议
 - 中继代理
 - 租用期
 - DHCP工作流程

- 为减少DHCP服务器的数据，可采用中继代理。



租用期

- 应用层
 - 配置协议参数
 - DHCP协议
 - 中继代理
 - 租用期
 - DHCP工作流程

- DHCP服务器分配给DHCP客户的IP地址是临时的，DHCP客户只能在一段有限的时间内使用这个分配到的IP地址。DHCP协议称这段时间为租用期。
- DHCP客户也可在自己发送的报文中（例如，发现报文）提出对租用期的要求。
- 租用期的数值应由DHCP服务器决定（cisco默认租用期）：
 - IP Address Lease Time #租期，默认1天
 - Renewal Time Value #更新租约时间，租期的1/2
 - Rebinding Time Value #最后更新时间，租期的7/8

DHCP协议工作流程

- 应用层
 - 配置协议参数
 - DHCP协议
 - 中继代理
 - 租用期
 - **DHCP工作流程**

