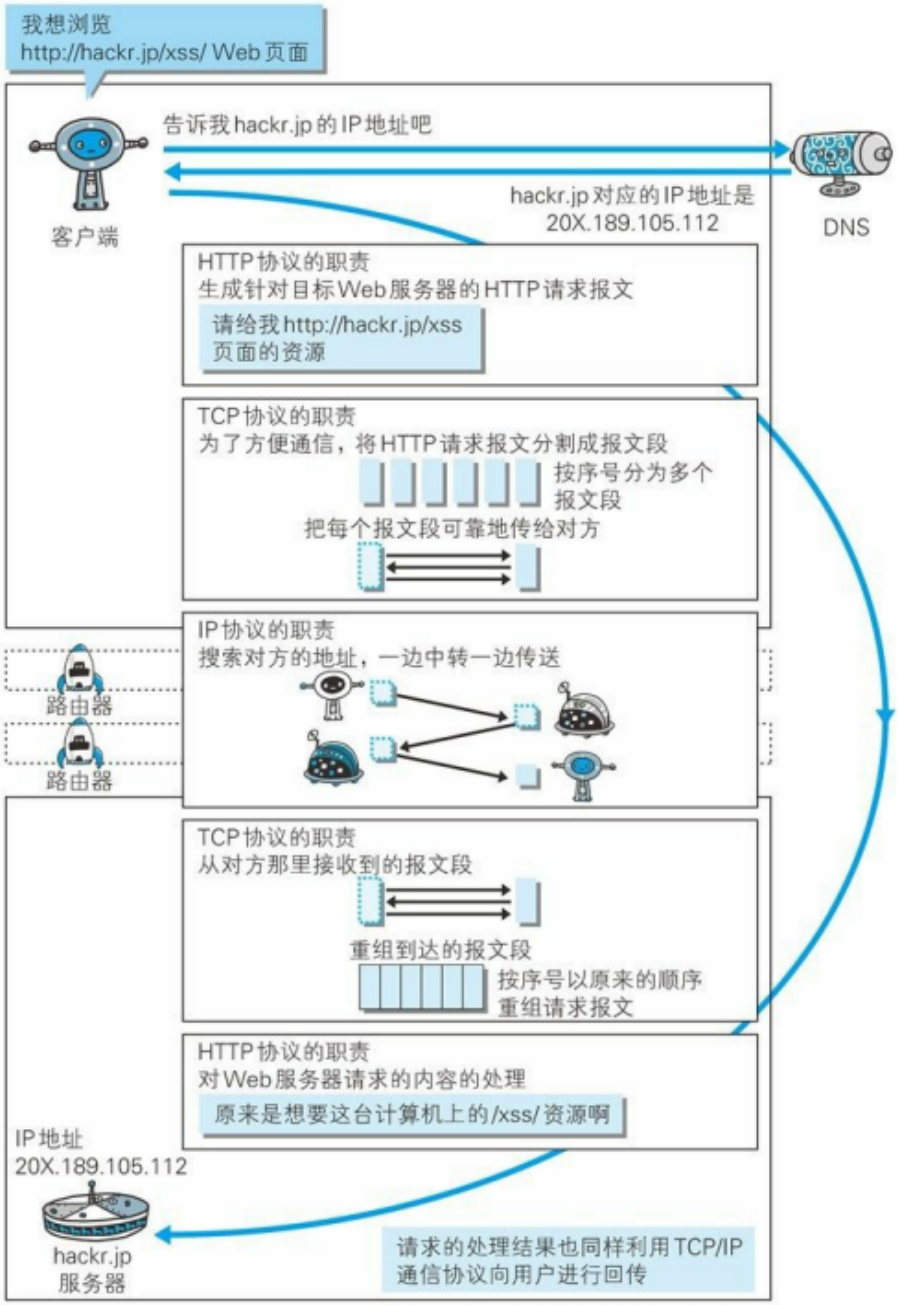
计算机网络常见面试题

浏览器输入网址之后发生了什么

1. DNS域名解析
2. HTTP协议生成请求报文
3. TCP协议将请求报文分割成报文段，进行可靠传输
4. IP协议进行分组转发
5. TCP协议重组请求报文
6. HTTP协议对请求进行处理



网络体系（OSI七层、TCP/IP四层、五层）

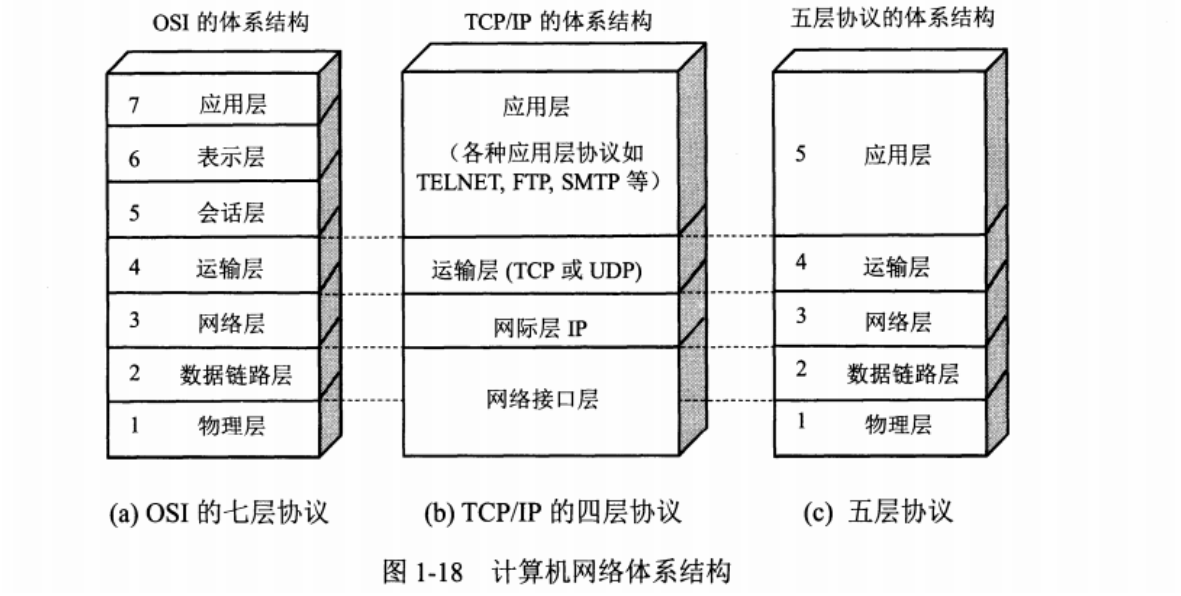


图 1-18　计算机网络体系结构

TCP/IP协议各层的作用

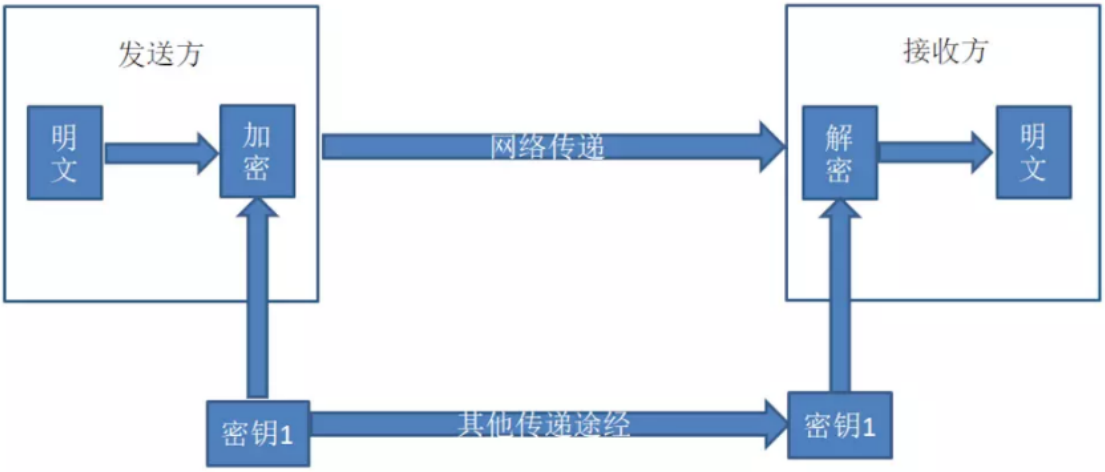
1. 应用层：应用层的任务是通过应用进程之间的交互来完成特定网络应用。应用层协议定义的是应用进程间通信和交互的规则。这里的进程是指主机中正在与运行的程序。对于不同的网络应用需要有不同的应用层协议。在互联网中的应用层协议有很多，如域名系统DNS，支持万维网应用的HTTP协议，支持电子邮件的SMTP协议，等等。
2. 传输层：传输层的任务是负责向两台主机中进程之间的通信提供通用的数据传输服务。应用进程利用该服务传送应用层报文。主要有TCP协议和UDP协议。
3. 网络层：网络层的任务是为分组交换网上的不同主机提供通信服务。在发送数据的时候，网络层把运输层产生的报文段或者用户数据报封装成分组或包进行传送。主要使用IP协议。
4. 网络接口层：操作系统中的设备驱动和计算机对应的网络接口，它们一起处理与传输媒介的物理接口细节。主要有ARP协议和RARP协议。

应用层

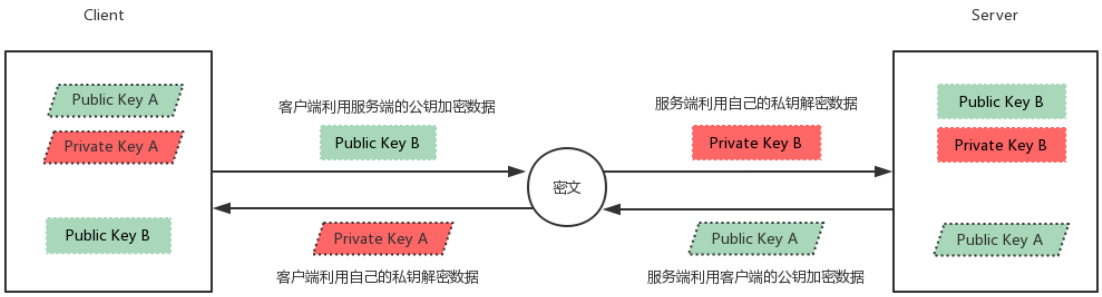
HTTP和HTTPS的区别

1. 端口 ：HTTP的URL由“http://”起始且默认使用端口80，而HTTPS的UR由“https://”起始且默认使用端口443。
2. 安全性和资源消耗： HTTP协议运行在TCP之上，所有传输的内容都是明文，客户端和服务器端都无法验证对方的身份。HTTPS是运行在SSL/TLS之上的HTTP协议，SSL/TLS 运行在TCP之上。所有传输的内容都经过加密，加密采用对称加密，但对称加密的密钥用服务器方的证书进行了非对称加密。所以说，HTTP 安全性没有 HTTPS高，但是 HTTPS 比HTTP耗费更多服务器资源。

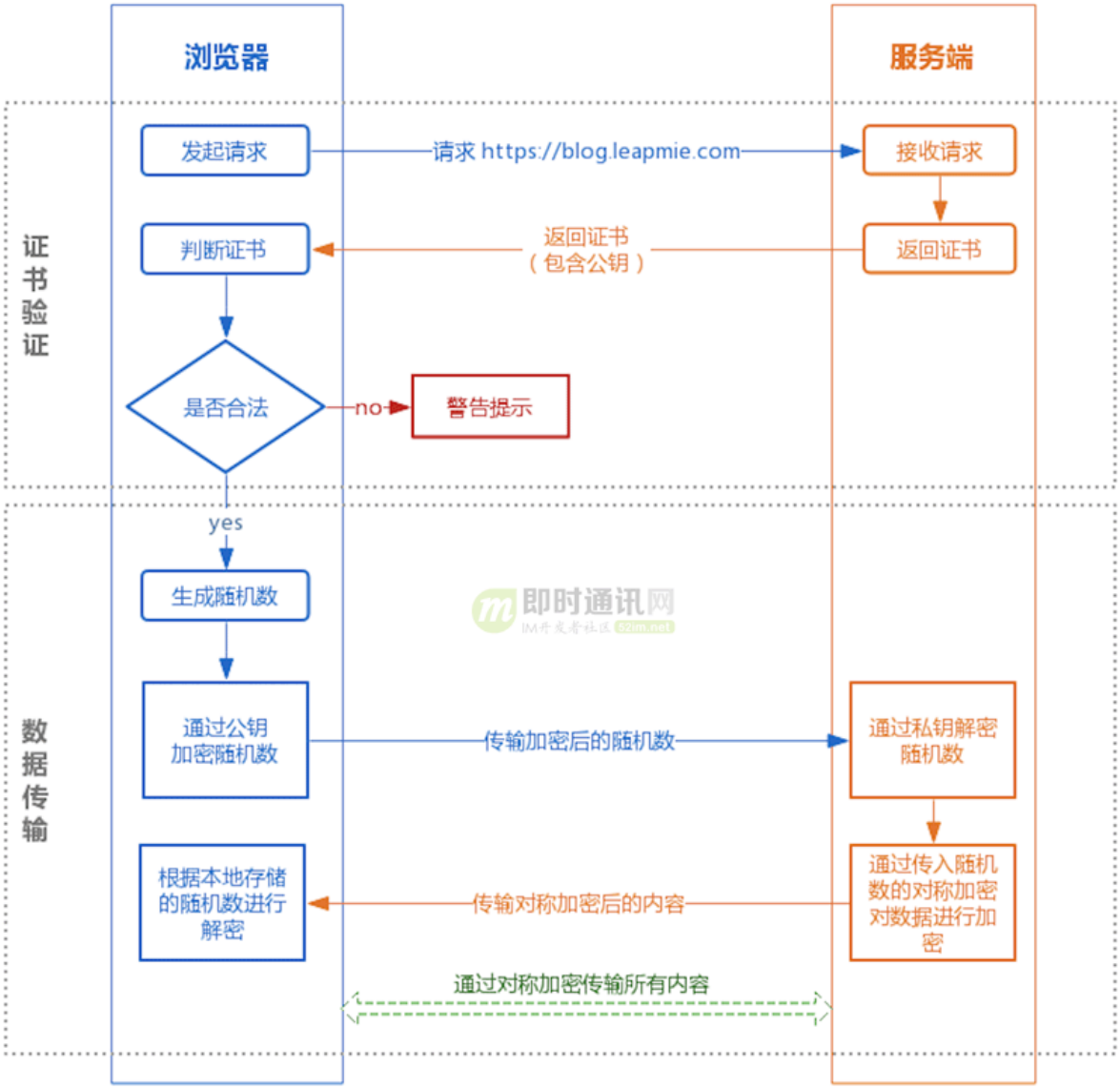
1. 对称加密：密钥只有一个，加密解密为同一个密码，且加解密速度快，典型的对称加密算法有DES、AES等；



1. 非对称加密：密钥成对出现（且根据公钥无法推知私钥，根据私钥也无法推知公钥），加密解密使用不同密钥（公钥加密需要私钥解密，私钥加密需要公钥解密），相对对称加密速度较慢，典型的非对称加密算法有RSA、DSA等。



HTTPS的加密过程



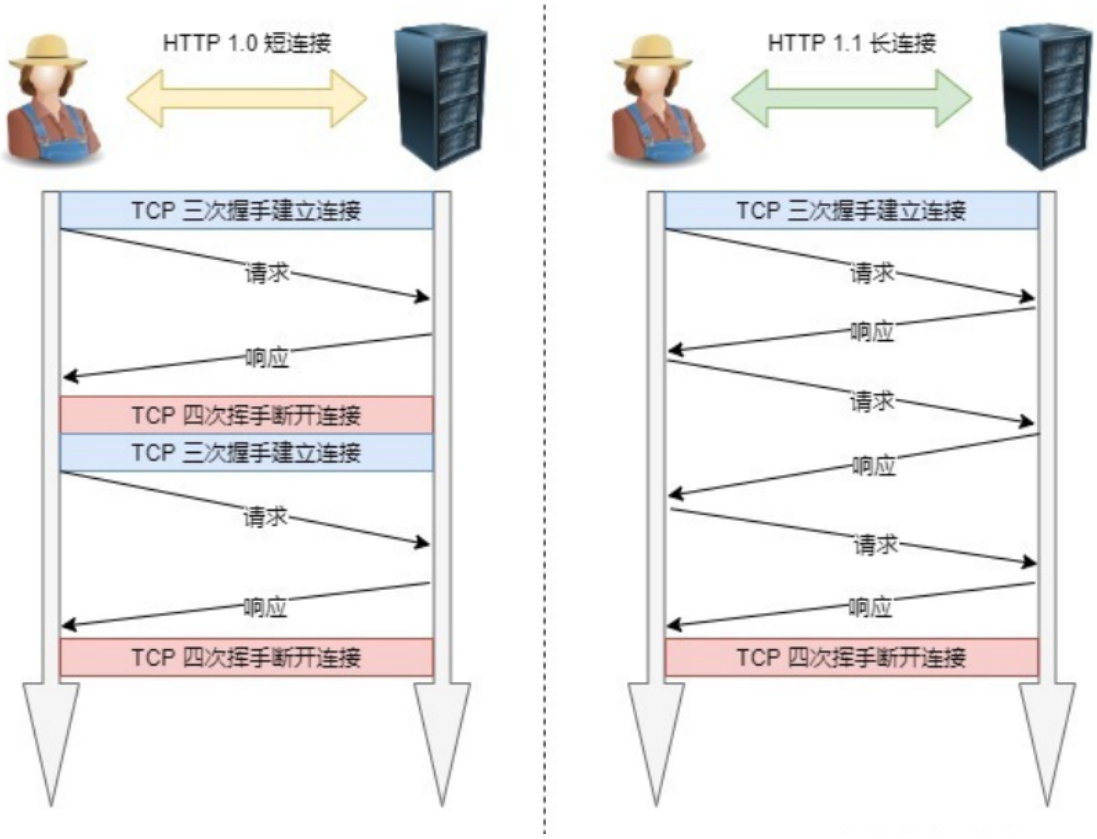
1. 证书验证，客户端发送一个证书请求个服务器端，服务器端返回证书，客户端对证书进行验证。
2. 交换密钥，使用非对称加密，客户端使用公钥进行加密，服务器端使用密钥解密。
3. 交换数据，使用对称加密的方式对数据进行加密，然后进行传输。

HTTP1.0和HTPP1.1的区别

1. 缓存处理：在HTTP1.0中主要使用header里的If-Modified-Since,Expires来做为缓存判断的标准，HTTP1.1则引入了更多的缓存控制策略例如Entity tag，If-Unmodified-Since, If-Match, If-None-Match等更多可供选择的缓存头来控制缓存策略。
2. 带宽优化及网络连接的使用：HTTP1.0中，存在一些浪费带宽的现象，例如客户端只是需要某个对象的一部分，而服务器却将整个对象送过来了，并且不支持断点续传功能，HTTP1.1则在请求头引入了range头域，它允许只请求资源的某个部分，即返回码是206（Partial Content），这样就方便了开发者自由的选择以便于充分利用带宽和连接。
3. 错误通知的管理：在HTTP1.1中新增了24个错误状态响应码，如409（Conflict）表示请求的资源与资源的当前状态发生冲突；410（Gone）表示服务器上的某个资源被永久性的删除。
4. Host头处理：在HTTP1.0中认为每台服务器都绑定一个唯一的IP地址，因此，请求消息中的URL并没有传递主机名（hostname）。但随着虚拟主机技术的发展，在一台物理服务器上可以存在多个虚拟主机（Multi-homed Web Servers），并且它们共享一个IP地址。HTTP1.1的请求消息和响应消息都应支持Host头域，且请求消息中如果没有Host头域会报告一个错误（400 Bad Request）。
5. 长连接：HTTP 1.1支持长连接（PersistentConnection）和请求的流水线（Pipelining）处理，在一个TCP连接上可以传送多个HTTP请求和响应，减少了建立和关闭连接的消耗和延迟，在

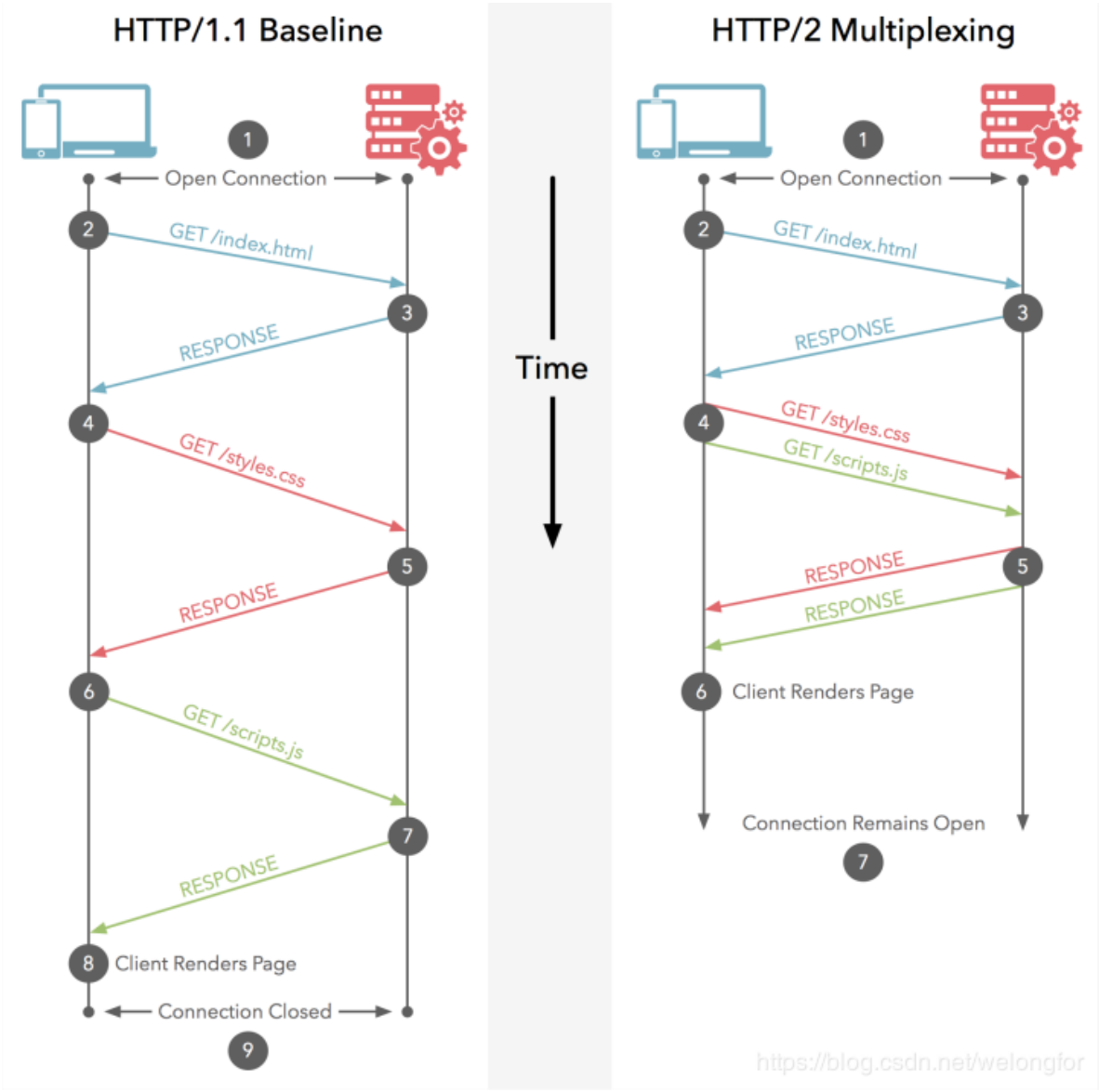
HTTP1.1中默认开启Connection： keep-alive，一定程度上弥补了HTTP1.0每次请求都要创建连接的缺点。

长连接和短连接的区别



HTTP2.0与HTTP1.x的区别

1. 新的二进制格式（Binary Format）：HTTP1.x的解析是基于文本。基于文本协议的格式解析存在天然缺陷，文本的表现形式有多样性，要做到健壮性考虑的场景必然很多，二进制则不同，只认0和1的组合。基于这种考虑HTTP2.0的协议解析决定采用二进制格式，实现方便且健壮
2. 多路复用（MultiPlexing）：即连接共享，即每一个request都是是用作连接共享机制的。一个request对应一个id，这样一个连接上可以有多个request，每个连接的request可以随机的混杂在一起，接收方可以根据request的 id将request再归属到各自不同的服务端请求里面。
3. header压缩：HTTP1.x的header带有大量信息，而且每次都要重复发送，HTTP2.0使用encoder来减少需要传输的header大小，通讯双方各自cache一份header fields表，既避免了重复header的传输，又减小了需要传输的大小。
4. 服务端推送（server push）：同SPDY一样，HTTP2.0也具有server push功能。



HTTP协议中的GET和POST方式的区别

1. 参数位置：GET方法参数位置包含在URL，POST方法参数包含在请求主体
2. 参数长度：GET方法的URL长度有限度，POST长度没有限制
3. 参数编码：GET方法参数编码是ASCII码，POST没有限制
4. TCP数据包：GET方法产生一个TCP数据包，把首部和数据一起发送，POST方法产生两个TCP数据包，先发首部，服务器响应后再发数据

HTTP的报文结构

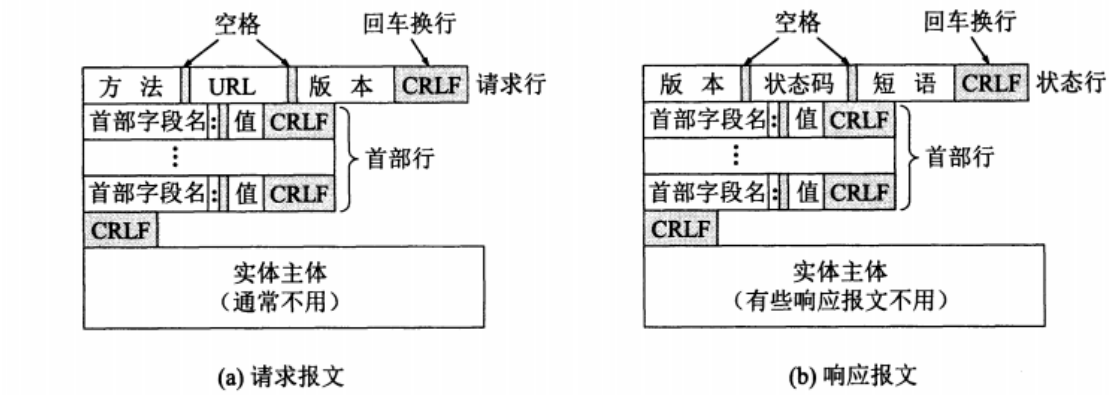


图 6-12　HTTP 的报文结构

HTTP协议有那些请求

表 6-1　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

HTTP的状态码

答：

状态码(Status-Code)都是三位数字的，分为 5 大类，原先有 33 种[RFC 2616]，后来又增加了几种[RFC 6585]。这 5 大类的状态码都是以不同的数字开头的。

1xx 表示通知信息，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，如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。

下面三种状态行在响应报文中是经常见到的。

```
HTTP/1.1 202 Accepted          {接受}
HTTP/1.1 400 Bad Request       {错误的请求}
Http/1.1 404 Not Found         {找不到}
```

HTTP的首部字段

答：

1. 通用首部字段（General Header Fields）

首部字段名	说明
Cache-Control	控制缓存的行为
Connection	逐跳首部、连接的管理
Date	创建报文的日期时间
Pragma	报文指令
Trailer	报文末端的首部一览
Transfer-Encoding	指定报文主体的传输编码方式
Upgrade	升级为其他协议
Via	代理服务器的相关信息
Warning	错误通知

1. 请求首部字段（Request Header Fields）

首部字段名	说明
Accept	用户代理可处理的媒体类型
Accept-Charset	优先的字符集
Accept-Encoding	优先的内容编码
Accept-Language	优先的语言（自然语言）
Authorization	Web认证信息
Expect	期待服务器的特定行为
From	用户的电子邮箱地址
Host	请求资源所在服务器
If-Match	比较实体标记（ETag）
If-Modified-Since	比较资源的更新时间
If-None-Match	比较实体标记（与 If-Match 相反）
If-Range	资源未更新时发送实体 Byte 的范围请求
If-Unmodified-Since	比较资源的更新时间（与If-Modified-Since相反）
Max-Forwards	最大传输逐跳数
Proxy-Authorization	代理服务器要求客户端的认证信息
Range	实体的字节范围请求
Referer	对请求中 URI 的原始获取方
TE	传输编码的优先级
User-Agent	HTTP 客户端程序的信息

1. 响应首部字段（Respond Header Fields）

首部字段名	说明
Accept-Ranges	是否接受字节范围请求
Age	推算资源创建经过时间
ETag	资源的匹配信息
Location	令客户端重定向至指定URI
Proxy-Authenticate	代理服务器对客户端的认证信息
Retry-After	对再次发起请求的时机要求
Server	HTTP服务器的安装信息
Vary	代理服务器缓存的管理信息
WWW-Authenticate	服务器对客户端的认证信息

1. 实体首部字段（Entity Header Fields）

首部字段名	说明
Allow	资源可支持的HTTP方法
Content-Encoding	实体主体适用的编码方式
Content-Language	实体主体的自然语言
Content-Length	实体主体的大小（单位：字节）
Content-Location	替代对应资源的URI
Content-MD5	实体主体的报文摘要
Content-Range	实体主体的位置范围
Content-Type	实体主体的媒体类型
Expires	实体主体过期的日期时间
Last-Modified	资源的最后修改日期时间

Session和Cookie的区别

答：

1. 储存位置：Cookie是客户端会话技术，数据保存在客户端，Session是服务器端会话技术，数据保存在服务器端。
2. 存储容量：Cookie一般<=4KB，Session无限制。
3. 跨域支持：Cookie支持跨域，Session不支持。

Cookie的知识点：

1. 概念：

客户端会话技术，将数据保存在客户端

1. 使用步骤：

1. 创建Cookie对象，绑定数据

new Cookie(Cookie cookie)

1. 发送Cookie对象

response.addCookie(Cookie cookie)

1. 获取Cookie，拿到数据

Cookie[] request.getCookies()

1. 实现原理：

基于响应头的set-cookie和请求头cookie实现

1. 细节：

1. 一次可不可以发送多少个Cookie

可以，可以创建多个Cookie对象，使用response调用多次addCookie方法发送多个Cookie

1. Cookie在浏览器中保存多长时间

默认情况下，当浏览器关闭，Cookie数据被销毁

持久化存储：

setMaxAge(int seconds)

正数：将Cookie数据写到硬盘的文件中，持久化存储

负数：默认情况

0：删除Cookie

1. Cookie能不能存中文

在tomcat 8之前不能存储，tomcat 8之后可以

1. Cookie的共享问题：

1. 假设在一个tomcat服务器，部署了多个项目，那么在这些项目中Cookie能不能共享

默认情况下不能共享

setPath(String path)：设置Cookie的取值范围，默认情况下，设置当前虚拟目录，如果要共享，可以将path设置为“/”

1. 不同的tomcat服务器间Cookie的共享问题

setDomain(String path)：如果设置一级域名相同，那么多个服务器之间Cookie可以共享

1. Cookie的特点和作用

特点：

1. Cookie储存数据在客户端浏览器
2. 浏览器对于单个Cookie的大小有限制（4KB）以及对于同一域名下的总Cookie数量也有限制（20个）

作用：

1. Cookie一般用于存储少量的不太敏感的数据
2. 在不登陆的情况下，完成服务器对客户端的身份识别

如何设计API的安全性

Token授权机制、时间戳超时机制、签名机制、拒绝重复调用

HTTP攻击有哪些

跨站脚本攻击（XSS）、Dos攻击、SQL注入攻击、OS命令注入攻击、HTTP头部攻击、目录攻击、开放重定向攻击

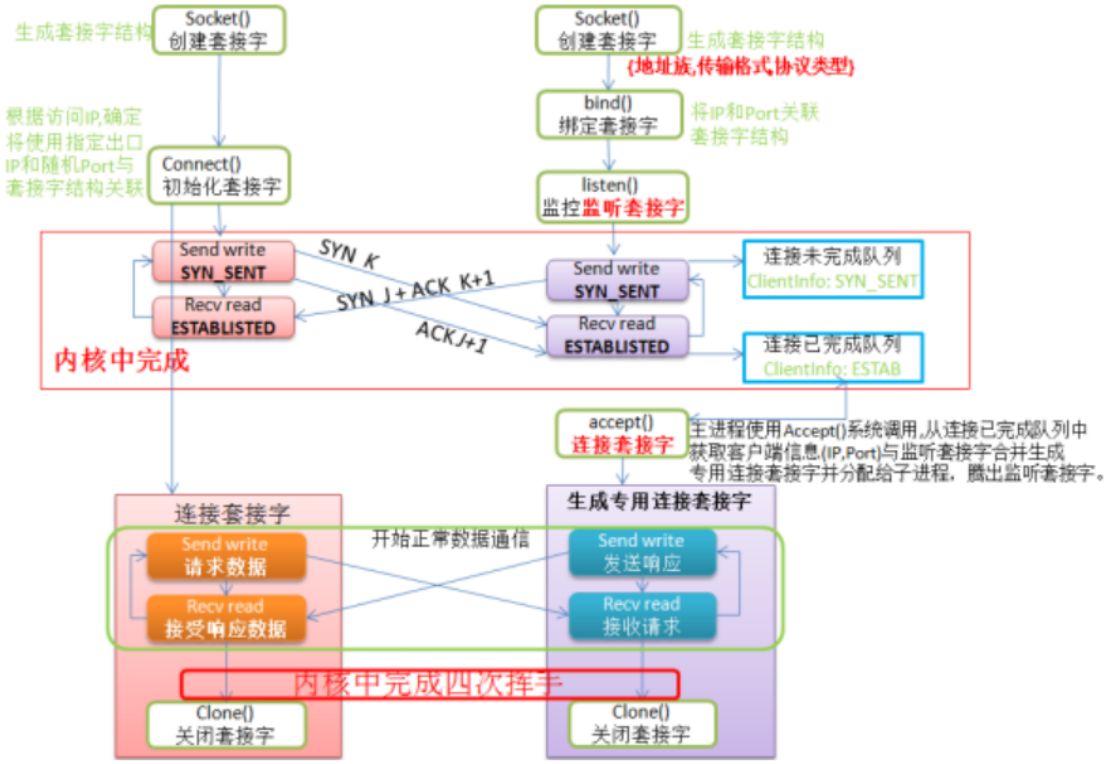
HTTP和WebSocket的联系和区别，什么情况下用WebSocket

答：HTTP1.x是半双工通信，客户端请求服务器响应，是无状态的，如果要实现实时通讯需要长轮询或者长连接的方式，这两种方式带来资源浪费。WebSocket是HTML5以后基于TCP协议应用层的一种全双工实时通讯协议。客户端和服务端可以进行信息的相互传递。它是借Http请求产生握手，在Http的头部会包含WebSocket协议的请求，所以握手之后，协议进行一个升级，转成TCP协议进行交流。
WebSocket可以应用于聊天和即时信息

WebSocket通信流程

答：

1. 建立socket通信，包括客户端和服务器端创建套接字（socket方法），服务器端绑定端口（bind方法），建立监听（listen方法），客户端创建连接（connect方法）。
2. TCP三次握手建立连接
3. 客户端服务器端收发数据（send方法、recv方法），读写数据（read方法、write方法）。
4. TCP四次挥手关闭连接
5. 关闭socket（close方法）



Socket编程主要大的步骤

1. 创建套接字（socket方法）
2. 客户端建立连接（connect方法），服务器端绑定端口（bind），建立监听（listen方法）
3. 写入数据（write方法），发送数据（send方法），接收数据（recv方法）读取数据（read方法）
4. 关闭连接（close方法）

socket常见读写错误

1. EAGAIN的错误：一般是对非阻塞端口读写产生的错误。
2. EINTR：errno=4，错误描述Interrupted system call，操作也应该继续。
3. ECONNABORTED，错误描述software caused connection abort，即“软件引起的连接中止
4. ECONNRESET 不在TCP_SYN_SENT状态收到的rst，connection reset by peer，即“对方复位连接”，这种情况一般发生在服务进程较客户进程提前终止。
5. EPIPE ，错误被描述为“broken pipe”，即“管道破裂”，这种情况一般发生在客户进程不理会（或未及时处理）Socket 错误
6. ECONNREFUSED在TCP_SYN_SENT状态收到的rst，一般说明对方没有对应的监听服务。
7. ETIMEDOUT，错误被描述为“connect time out”，即“连接超时”，这种情况一般发生在服务器主机崩溃。
8. EHOSTTUNREACH ENETUNREACH，no route to host(软错)路由上引发了一个目的地不可达的ICMP错误 其中(1)(3)，客户端会进行定时多次重试，一定次数后才返回错误。

DHCP原理

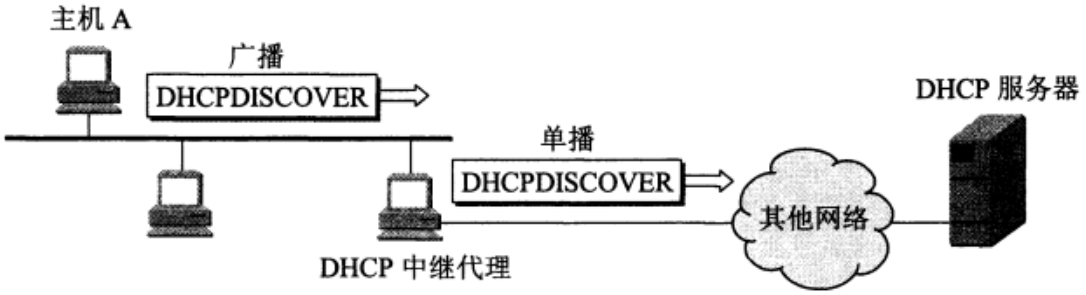


图 6-19　DHCP 中继代理以单播方式转发发现报文

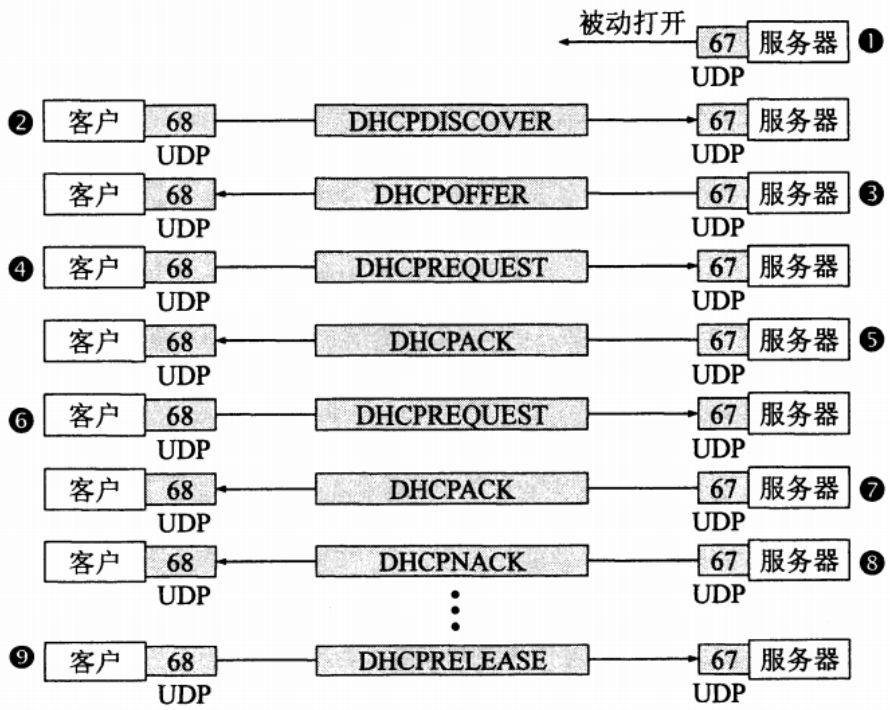


图 6-20　DHCP 协议的工作过程

❶ DHCP 服务器被动打开 UDP 端口 67，等待客户端发来的报文。

❷ DHCP 客户从 UDP 端口 68 发送 DHCP 发现报文。

❸ 凡收到 DHCP 发现报文的 DHCP 服务器都发出 DHCP 提供报文，因此 DHCP 客户

可能收到多个 DHCP 提供报文。

❹ DHCP 客户从几个 DHCP 服务器中选择其中的一个，并向所选择的 DHCP 服务器发送 DHCP 请求报文。

❺ 被选择的 DHCP 服务器发送确认报文 DHCPACK。从这时起，DHCP 客户就可以使用这个 IP 地址了。这种状态叫做**已绑定状态**，因为在 DHCP 客户端的 IP 地址和硬件地址已经完成绑定，并且可以开始使用得到的临时 IP 地址了。

DHCP 客户现在要根据服务器提供的租用期 T 设置两个计时器 T_1 和 T_2，它们的超时时间分别是 0.5T 和 0.875T。当超时时间到了就要请求更新租用期。

❻ 租用期过了一半（T_1 时间到），DHCP 发送请求报文 DHCPREQUEST 要求更新租用期。

❼ DHCP 服务器若同意，则发回确认报文 DHCPACK。DHCP 客户得到了新的租用期，重新设置计时器。

❽ DHCP 服务器若不同意，则发回否认报文 DHCPNACK。这时 DHCP 客户必须立即停止使用原来的 IP 地址，而必须重新申请 IP 地址（回到步骤❷）。

若 DHCP 服务器不响应步骤❻的请求报文 DHCPREQUEST，则在租用期过了 87.5%时（T_2 时间到），DHCP 客户必须重新发送请求报文 DHCPREQUEST（重复步骤❻），然后又继续后面的步骤。

❾ DHCP 客户可以随时提前终止服务器所提供的租用期，这时只需向 DHCP 服务器发送释放报文 DHCPRELEASE 即可。

域名系统DNS

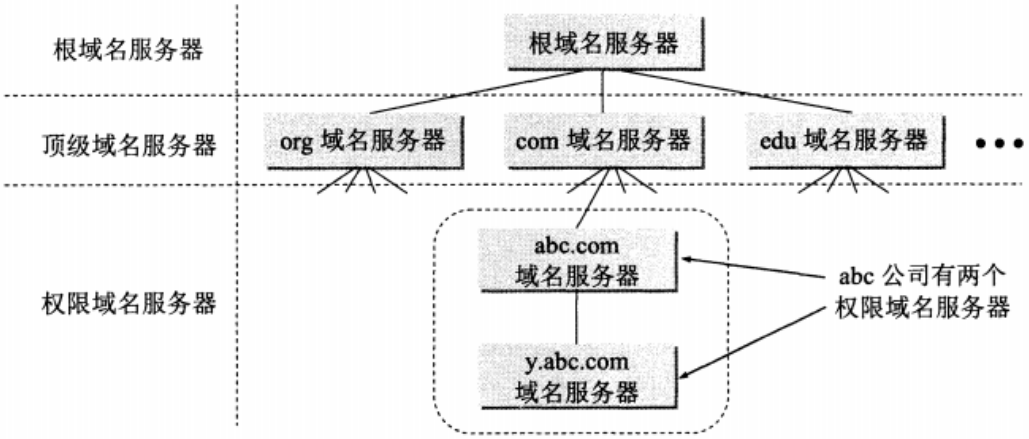


图 6-3　树状结构的 DNS 域名服务器

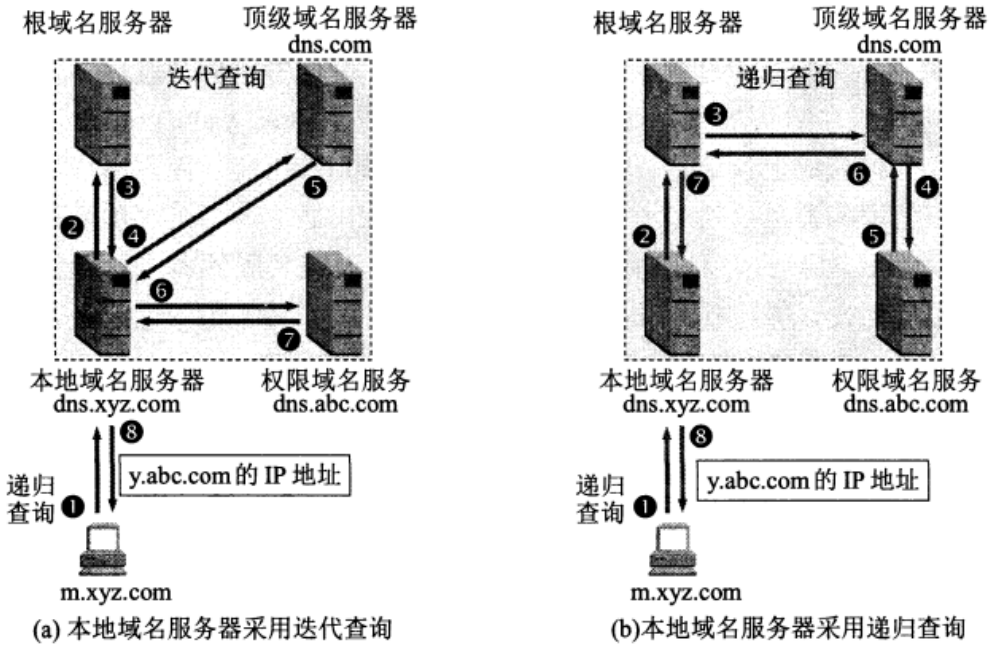
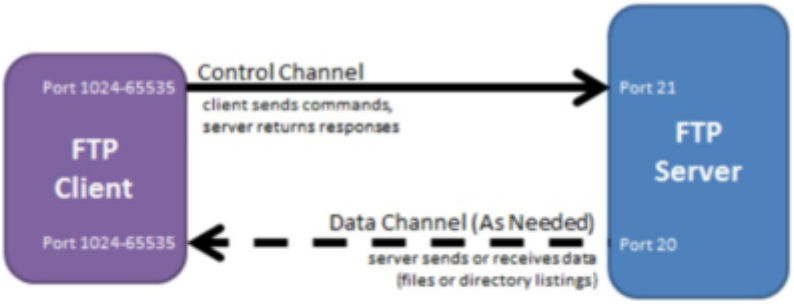


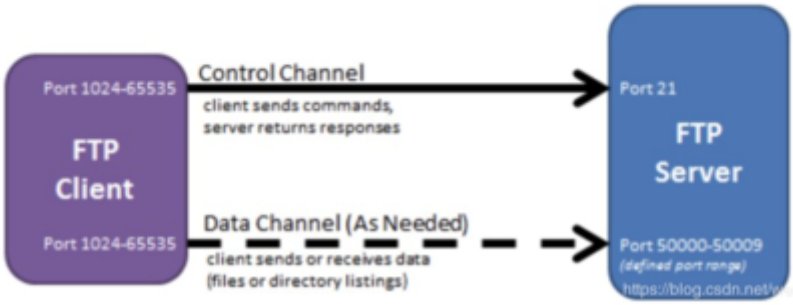
图 6-5　DNS 查询举例

FTP的两种模式

- 主动模式：服务器端主动建立数据连接，其中服务器端的端口号为 20，客户端的端口号随机，但是必须大于 1024，因为 0~1023 是熟知端口号。



- 被动模式：客户端主动建立数据连接，其中客户端的端口号由客户端自己指定，服务器端的端口号随机。



传输层

TCP三次握手和四次挥手的图

1. 三次握手

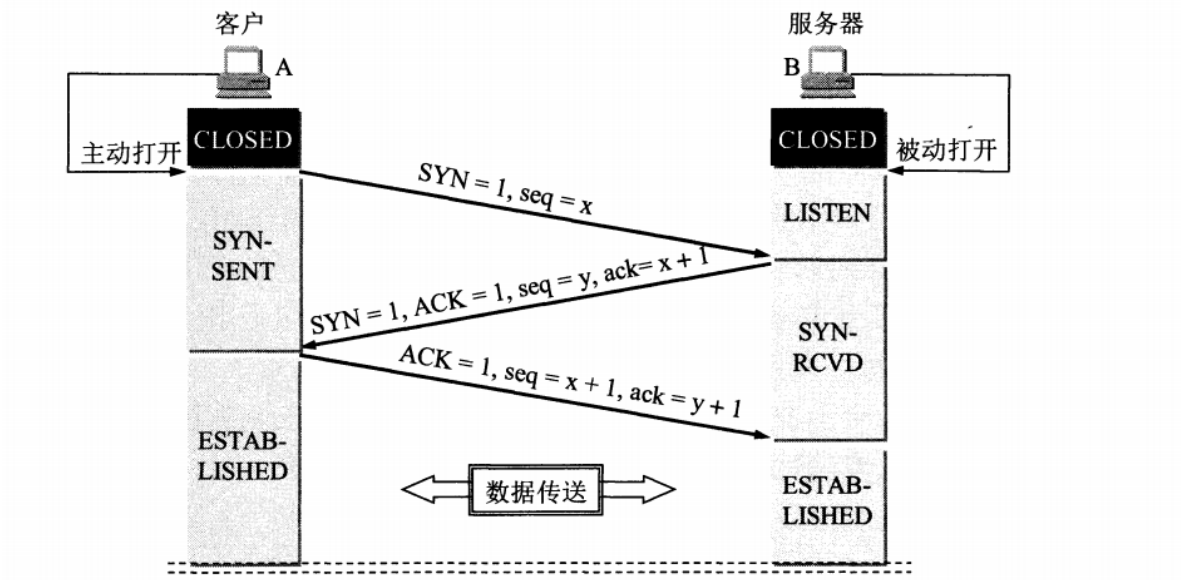


图 5-28 用三报文握手建立 TCP 连接

1. 四次挥手

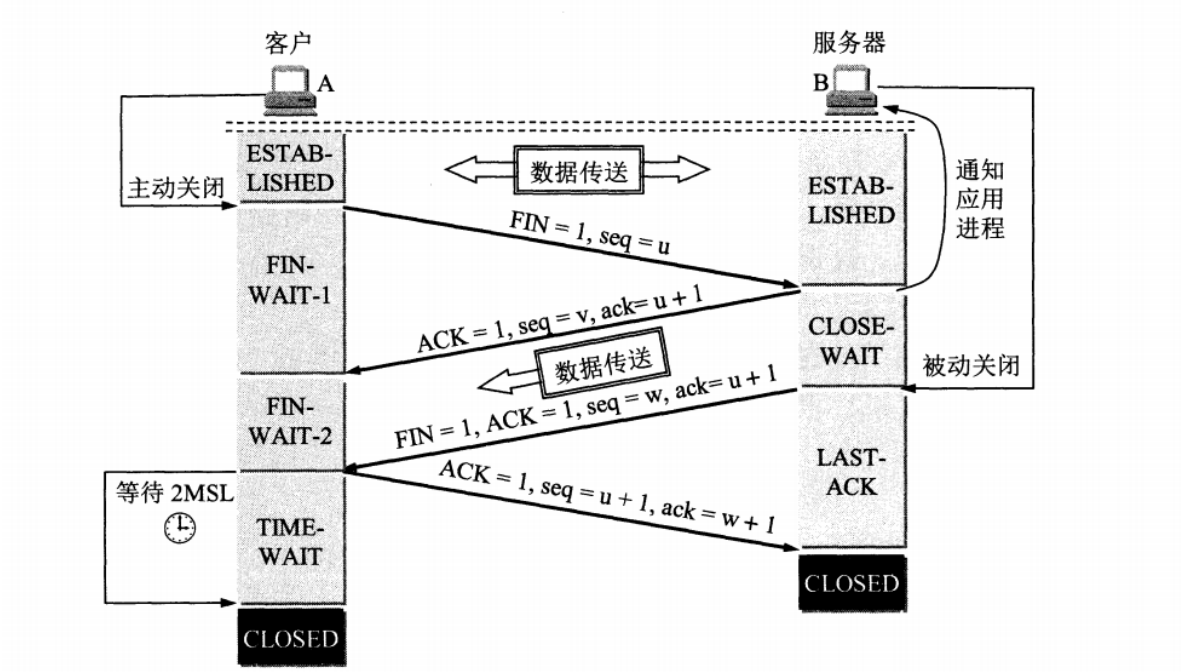


图 5-29 TCP 连接释放的过程

第三次握手可以发送数据了吗

不能，需要三次握手成功之后才开始发送数据

TCP连接唯一性如何保证

对TCP而言在三次握手时的SYN标志会使用上一个ISN值，这个值是使用32位计数器，内由0-4294967295，每一次连接容都会分配到一个ISN值，连接双方对这个值会记录共识，假如这个值不一，就说明了这个连接已超时或无效甚至是被人恶意攻击冒充连接。

为什么要有TIME_WAIT

为什么 A 在 TIME-WAIT 状态必须等待 2MSL 的时间呢？这有两个理由。

第一，为了保证 A 发送的最后一个 ACK 报文段能够到达 B。这个 ACK 报文段有可能丢失，因而使处在 LAST-ACK 状态的 B 收不到对已发送的 FIN + ACK 报文段的确认。B 会超时重传这个 FIN + ACK 报文段，而 A 就能在 2MSL 时间内收到这个重传的 FIN + ACK 报文段。接着 A 重传一次确认，重新启动 2MSL 计时器。最后，A 和 B 都正常进入到 CLOSED 状态。如果 A 在 TIME-WAIT 状态不等待一段时间，而是在发送完 ACK 报文段后立即释放连接，那么就无法收到 B 重传的 FIN + ACK 报文段，因而也不会再发送一次确认报文段。这样，B 就无法按照正常步骤进入 CLOSED 状态。

第二，防止上一节提到的“已失效的连接请求报文段”出现在本连接中。A 在发送完最后一个 ACK 报文段后，再经过时间 2MSL，就可以使本连接持续的时间内所产生的所有报文段都从网络中消失。这样就可以使下一个新的连接中不会出现这种旧的连接请求报文段。

我是服务端你是TIME_WAIT客户端，我给你发了一段数据，我的seq是100，你应该回我ack多少（假设已经建立了三次握手，现在给你发了一段数据）

建立连接：Ack号 = Seq号 + 1

传输数据：Ack号 = Seq号 + 传递的字节数 + 1

TCP和UDP的区别

答：

1. 连接：UDP是无连接的，发送数据之前无需建立连接，发送数据结束后也无需释放连接。TCP是面向连接的，在发送数据之前需要通过三次握手建立连接，发送数据结束后需要通过四次挥手释放连接。
2. 交付：UDP使用尽最大努力交付，即不保证可靠交付，主机不需要维持复杂的连接状态表。TCP提供可靠交付，即通过TCP连接传送的数据，无差错、不丢失、不重复，并且按序到达。
3. 数据：UDP是面向报文的，UDP对于应用层交下来的报文，添加首部后就向下交给IP层，既不合并，也不拆分，一次交付一个完整的报文。TCP是面向字节流的，虽然应用程序和TCP的交互是一次一个数据块（大小不等），但TCP把应用程序交下来的数据看出成一连串无结构的字节流。TCP不保证接收方应用程序所收到的数据块和发送方应用程序所发出的数据块具有对应大小关系。
4. 通信双方：UDP支持一对一、一对多、多对一、多对多的交互通信。TCP连接是点对点（一对一），每一条TCP连接只能有两个端点。
5. 拥塞：UDP没有拥塞控制，网络的拥塞不会使源主机的发送速率降低。TCP通过慢开始、拥塞避免、快重传、快恢复等算法进行拥塞控制。
6. 首部：UDP首开销小，只有8个字节。TCP首部是20个字节。

TCP可靠连接的实现

答：

1. 滑动窗口
2. 选择确认
3. 超时重传

TCP的流量控制

TCP拥塞控制的方法

1. 慢开始

慢开始算法的思路是这样的：当主机开始发送数据时，由于并不清楚网络的负荷情况，所以如果立即把大量数据字节注入到网络，那么就有可能引起网络发生拥塞。经验证明，较好的方法是先探测一下，即**由小到大逐渐增大发送窗口**，也就是说，**由小到大逐渐增大拥塞窗口数值**。

1. 拥塞避免

拥塞避免算法的思路是让拥塞窗口 cwnd 缓慢地增大，即每经过一个往返时间 RTT 就把发送方的拥塞窗口 cwnd 加 1[①]，而不是像慢开始阶段那样加倍增长。因此在拥塞避免阶段就有“**加法增大**”AI (Additive Increase)的特点。这表明在拥塞避免阶段，拥塞窗口 cwnd **按线性规律缓慢增长**，比慢开始算法的拥塞窗口增长速率缓慢得多。

1. 快重传

1. 快恢复

TCP三次握手和四次挥手时客户端和服务端的状态

答：参考上面两个图

常见端口和对应服务

答：

(1) **服务器端使用的端口号**　　这里又分为两类，最重要的一类叫做**熟知端口号**(well-known port number)或**系统端口号**，数值为 0~1023。这些数值可在网址 www.iana.org 查到。IANA 把这些端口号指派给了 TCP/IP 最重要的一些应用程序，让所有的用户都知道。当一种新的应用程序出现后，IANA 必须为它指派一个熟知端口，否则互联网上的其他应用进程就无法和它进行通信。表 5-2 给出了一些常用的熟知端口号。

表 5-2　常用的熟知端口号

应用程序	FTP	TELNET	SMTP	DNS	TFTP	HTTP	SNMP	SNMP (trap)	HTTPS
熟知端口号	21	23	25	53	69	80	161	162	443

另一类叫做**登记端口号**，数值为 1024~49151。这类端口号是为没有熟知端口号的应用程序使用的。使用这类端口号必须在 IANA 按照规定的手续登记，以防止重复。

(2) **客户端使用的端口号**　　数值为 49152~65535。由于这类端口号仅在客户进程运行时才动态选择，因此又叫做**短暂端口号**[①]。这类端口号留给客户进程选择暂时使用。当服务器进程收到客户进程的报文时，就知道了客户进程所使用的端口号，因而可以把数据发送给客户进程。通信结束后，刚才已使用过的客户端口号就不复存在，这个端口号就可以供其他客户进程使用。

网络层

子网的划分

答：

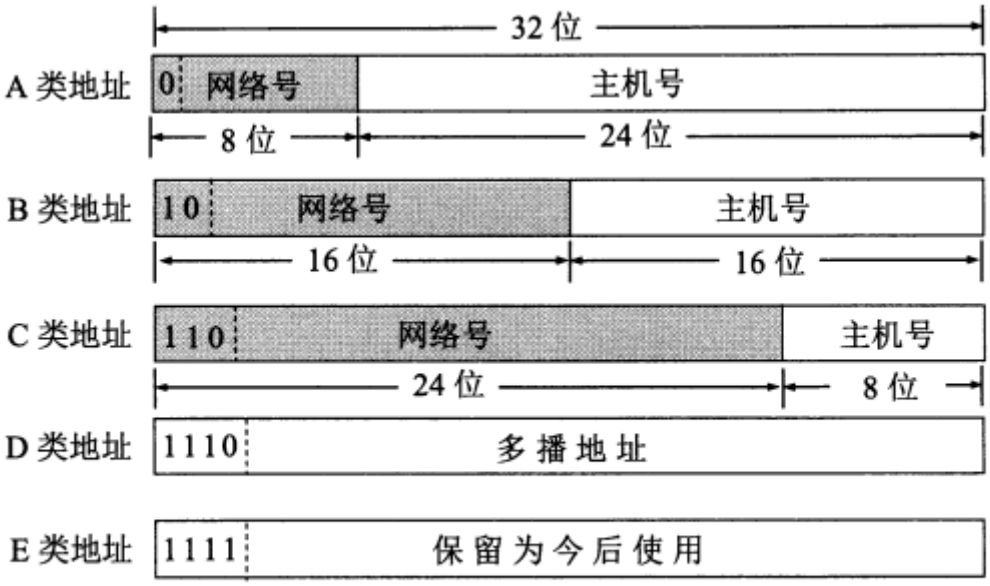


图 4-5　IP 地址中的网络号字段和主机号字段

交换机和路由器的区别

答：

1. 工作层次：交换机主要工作在数据链路层，路由器主要工作在网络层
2. 转发依据：交换机转发依据的对象是MAC地址，路由器转发依据的对象是IP地址
3. 功能：交换机功能较简单，只是将主机连接起来组件局域网，路由器可以将局域网连接起来，还能分割广播域，还能提供防火墙

如果不想要一个ip，如何释放，如何重新获取一个ip

子网掩码的最大长度

30位，子网掩码将某个IP地址划分成网络地址和主机地址两部分，在一个网段中，有2个地址是被固定占用的，一个是网段地址，一个是网段内广播地址，其他是主机可用的地址，至少一个，不然就没有意义了。也就是说，被掩码所分的网段至少要包含3个地址。

MAC表、路由表、ARP表的字段

答：

1. MAC表

2. SW1#show mac-address-table

3. Mac Address Table

4. ----------

5. Vlan Mac Address Type Ports

6. ---- ----------- -------- -----

7. 1 0003.e49e.6445 DYNAMIC Fa0/2

8. 1 0010.1148.9309 DYNAMIC Fa0/24

9. 1 00e0.b079.7815 DYNAMIC Fa0/1

1. 路由表

```
C:\Users\lvjiafa>route print
===========================================================================
接口列表
  3...40 5b d8 97 37 53 ......Microsoft Wi-Fi Direct Virtual Adapter
  6...c2 5b d8 97 37 53 ......Microsoft Wi-Fi Direct Virtual Adapter #2
  5...40 5b d8 97 37 53 ......Realtek 8822CE Wireless LAN 802.11ac PCI-E NIC
 14...40 5b d8 97 37 54 ......Bluetooth Device (Personal Area Network)
  1...........................Software Loopback Interface 1
===========================================================================

IPv4 路由表
===========================================================================
活动路由:
网络目标        网络掩码          网关       接口   跃点数
          0.0.0.0          0.0.0.0    192.168.43.1  192.168.43.233     55
        127.0.0.0        255.0.0.0         在链路上         127.0.0.1    331
        127.0.0.1  255.255.255.255         在链路上         127.0.0.1    331
  127.255.255.255  255.255.255.255         在链路上         127.0.0.1    331
      192.168.43.0    255.255.255.0         在链路上    192.168.43.233    311
    192.168.43.233  255.255.255.255         在链路上    192.168.43.233    311
    192.168.43.255  255.255.255.255         在链路上    192.168.43.233    311
        224.0.0.0        240.0.0.0         在链路上         127.0.0.1    331
        224.0.0.0        240.0.0.0         在链路上    192.168.43.233    311
  255.255.255.255  255.255.255.255         在链路上         127.0.0.1    331
  255.255.255.255  255.255.255.255         在链路上    192.168.43.233    311
===========================================================================
永久路由:
  无
```

```
IPv6 路由表
===========================================================================
活动路由:
 接口跃点数网络目标                网关
  5     71 ::/0                     fe80::4a2c:a0ff:fed2:382
  1    331 ::1/128                  在链路上
  5     71 240e:ff:b54b:aab7::/64   在链路上
  5    311 240e:ff:b54b:aab7:a5dc:54f1:9c32:f9a3/128
                                    在链路上
  5    311 240e:ff:b54b:aab7:fdb4:c2b3:db53:43d6/128
                                    在链路上
  5    311 fe80::/64                在链路上
  5    311 fe80::fdb4:c2b3:db53:43d6/128
                                    在链路上
  1    331 ff00::/8                 在链路上
  5    311 ff00::/8                 在链路上
===========================================================================
永久路由:
  无
```

1. ARP表

```
C:\Users\lvjiafa>arp -a

接口: 192.168.43.233 --- 0x5
  Internet 地址         物理地址              类型
  192.168.43.1          48-2c-a0-d2-03-82     动态
  192.168.43.255        ff-ff-ff-ff-ff-ff     静态
  224.0.0.22            01-00-5e-00-00-16     静态
  224.0.0.251           01-00-5e-00-00-fb     静态
  224.0.0.252           01-00-5e-00-00-fc     静态
  239.11.20.1           01-00-5e-0b-14-01     静态
  239.255.255.250       01-00-5e-7f-ff-fa     静态
  255.255.255.255       ff-ff-ff-ff-ff-ff     静态
```

ARP协议的作用

答：将IP地址转换为物理地址

ARP协议的工作原理

1. 首先，每个主机都会在自己的ARP缓冲区中建立一个ARP列表，以表示IP地址和MAC地址之间的对应关系。
2. 当源主机要发送数据时，首先检查ARP列表中是否有对应IP地址的目的主机的MAC地址，如果有，则直接发送数据，如果没有，就向本网段的所有主机发送ARP数据包，该数据包包括的内容有：源主机IP地址，源主机MAC地址，目的主机的IP地址。
3. 当本网络的所有主机收到该ARP数据包时，首先检查数据包中的IP地址是否是自己的IP地址，如果不是，则忽略该数据包，如果是，则首先从数据包中取出源主机的IP和MAC地址写入到ARP列表中，如果已经存在，则覆盖，然后将自己的MAC地址写入ARP响应包中，告诉源主机自己是它想要找的MAC地址。
4. 源主机收到ARP响应包后。将目的主机的IP和MAC地址写入ARP列表，并利用此信息发送数据。如果源主机一直没有收到ARP响应数据包，表示ARP查询失败。

广播发送ARP请求，单播发送ARP响应。

IPV6地址的基本类型

1. 单播（unicast）：单播就是传统的点对点通信。
2. 多播（multicast）：多播是一点对多点的通信，数据报发送到一组计算机中的每一个。IPV6没有广播的术语，而是将广播看作多播的一个特例。
3. 任播（anycast）：这是IPV6增加的一种类型。任播的终点是一组计算机，但是数据包只交付给其中一个，通常是距离最近的一个。

IPV6的首部

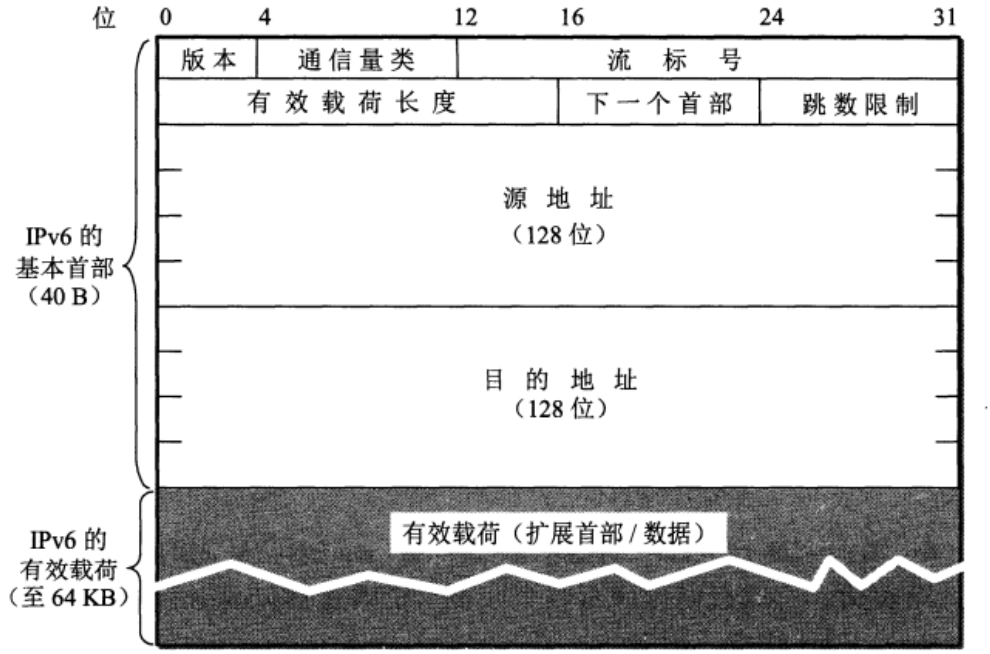


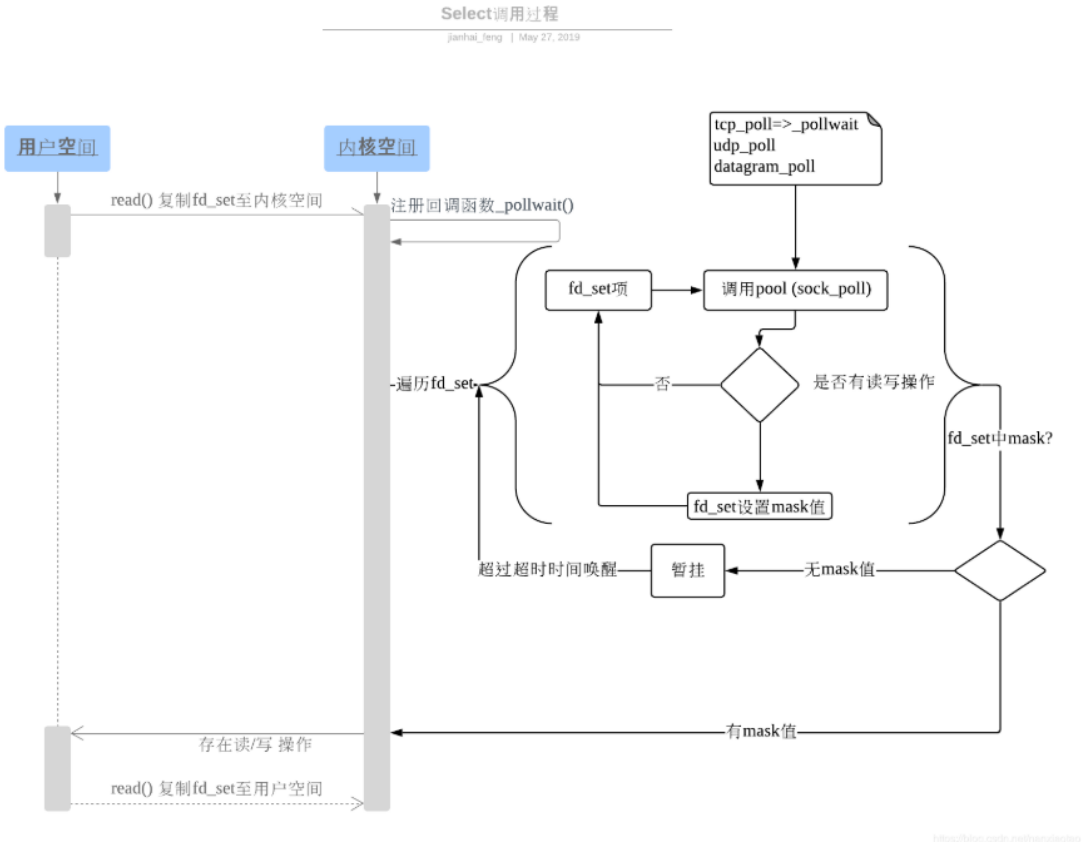
图 4-47　40 字节长的 IPv6 基本首部

路由选择协议

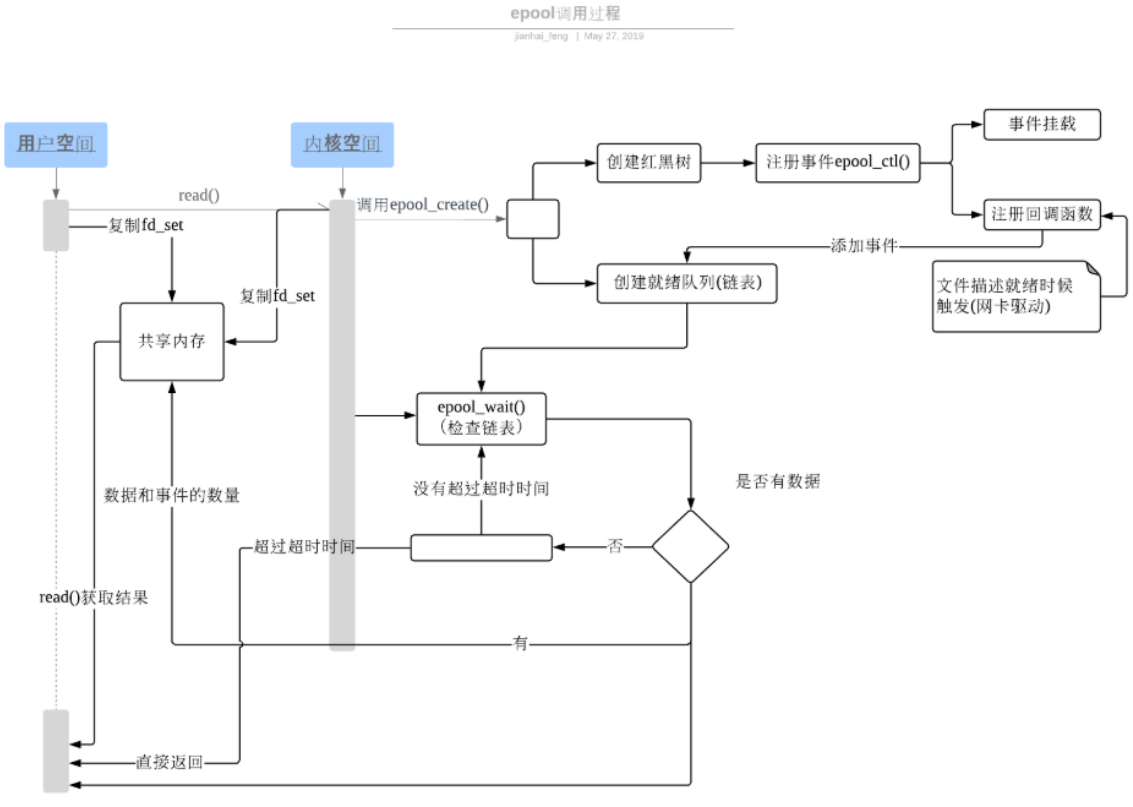
1. 内部网关协议RIP（Routing Information Protocol）：RIP是一种分布式的基于距离向量的路由选择协议。
2. 内部网关协议OPSF（Open Shortest Path First）：OSPF是使用分布式的链路状态协议。
3. 外部网关协议BGP（Border Gateway Protocol）：边界网关协议。

IO多路复用

1. select



1. epoll



epoll和select的区别