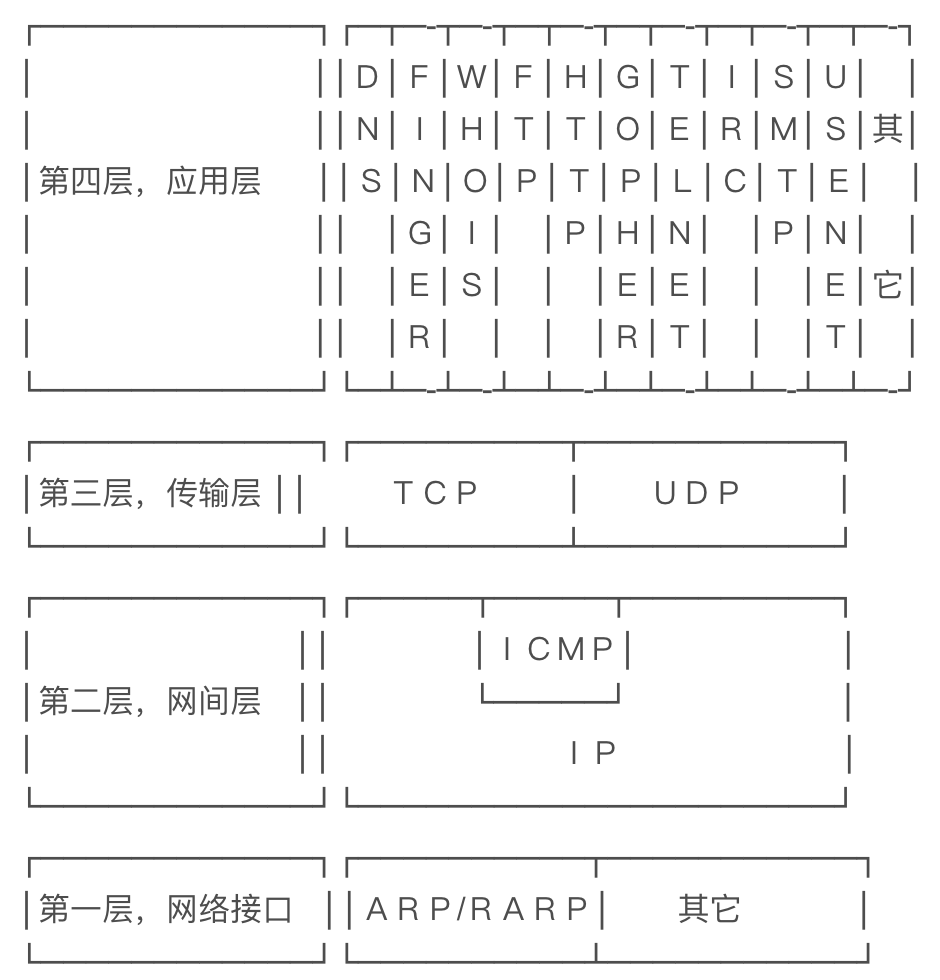
### 1、4/7层网络模型

#### 四层模型

TCP/IP分层模型，被称作因特网分层模型（Internet Layering Model）。

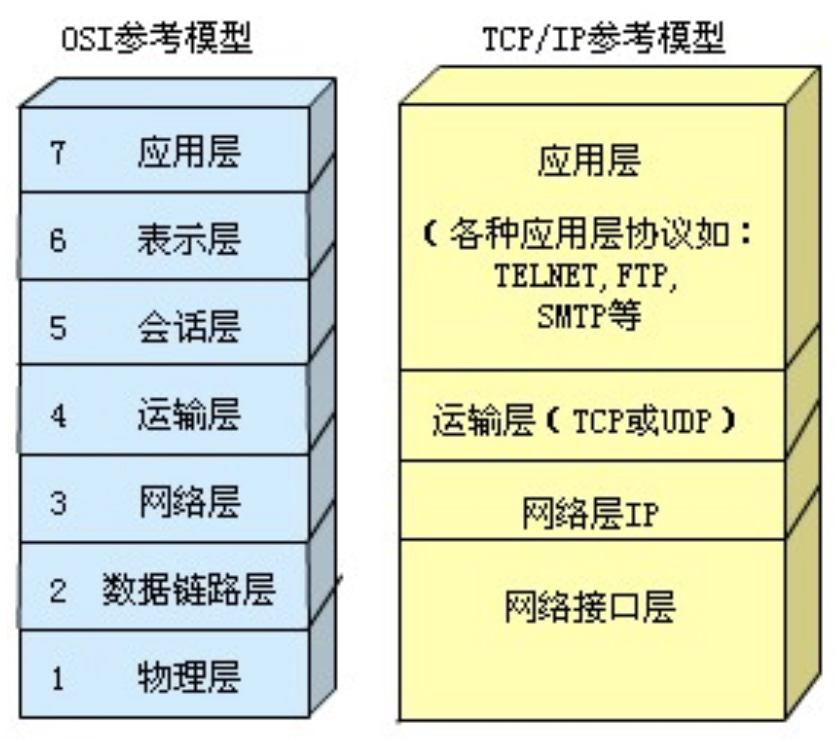


TCP/IP协议被组织成四个概念层，其中有三层对应于ISO参考模型中的相应层。ICP/IP协议簇并不包含物理层和数据链路层，因此它不能独立完成整个计算机网络系统的功能，必须与许多其他协议协同工作。

#### 七层模型

在网络历史的早期，国际化标准组织（ISO）和国际电报电话咨询委员会（CCITT）共同出版了开放系统互联的七层参考模型。一台计算机操作系统中的网络过程包括从应用请求（在协议栈的顶部）到网络介质（底部），OSI参考模型把功能分成七个层次。





### 2、TCP协议（传输层）

Transmission Control Protocol：传输控制协议，是一种面向连接的、可靠的、基于字节流的传输层通信协议。log转发：开启一个协议——tcp（三次握手和四次挥手）。

#### 特点

* 基于链接（点对点）
* 双工通信
* 可靠传输
* 拥塞控制
* 基于字节流而非报文

#### 实现细节

* 8种报文状态
* 滑动窗口机制
* KeepAlive
* Nagel算法

#### 建立链接三次握手

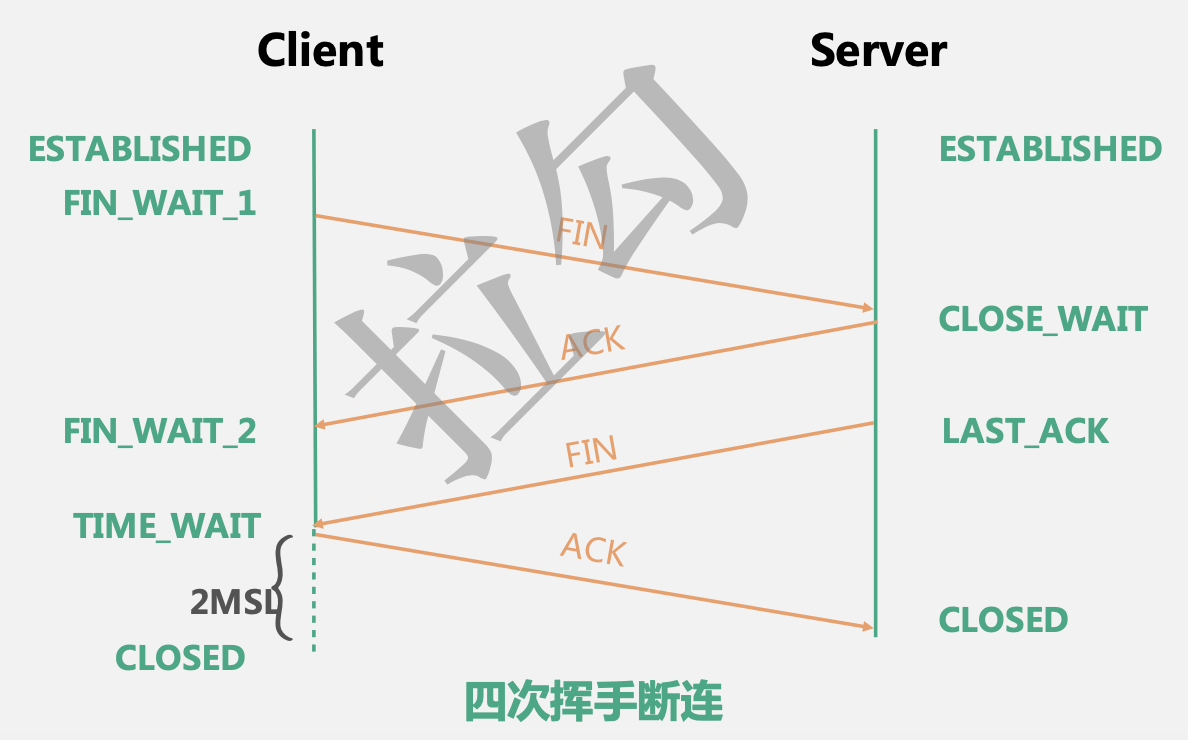
建立连接目的是为了通信双方确认开始序号，以便后续通信的有序进行，主要步骤：

* 连接开始时，（Client）发送SYN包，并包含了自己的初始序号a——连接请求；
* Server收到SYN包以后会回复一个SYN包，其中包含了对上一个a包的回应信息ACK，回应的序号为下一个希望收到包的序号，即a+1，然后还包含了自己的初始序号b——请求确认；
* Client收到回应的SYN包以后，回复一个ACK包做响应，其中包含了下一个希望收到包的序号即b+1——连接确认。



#### 关闭链接四次挥手

* 首先进行关闭的一方（即发送第一个FIN）将执行主动关闭，而另一方（收到这个FIN）执行被动关闭——请求断开连接；
* 当服务器收到这个FIN，它发回一个ACK，确认序号为收到的序号加1和SYN一样，一个FIN将占用一个序号——针对客户端的PIN的确认应答。
* 同时TCP服务器还向应用程序（即丢弃服务器）传送一个文件结束符。接着这个服务器程序就关闭它的链接，导致它的TCP端发送一个FIN——请求断开连接。
* 客户必须发回一个确认，并将确认序号设置为收到序号加1——针对服务器端的FIN的确认应答——针对服务器端的FIN的确认应答。



### 3、HTTP协议（应用层）

HyperText Transfer Protocol：超文本传输协议。用于从万维网服务器传输超文本到本地浏览器的传输协议。HTTP是一个应用层协议，由请求和响应构成，是一个标准的客户端和服务器模型。

工作原理：HTTP协议定义Web客户端如何从Web服务器请求Web页面，以及服务器如何把Web页面传送给客户端。HTTP协议采用了请求/响应模型。客户端向服务器发送一个请求报文，请求报文包含请求的方法、URL、协议版本、请求头部和请求数据。服务器以一个状态行作为响应，响应的内容包括协议的版本、成功或者错误代码、服务器信息、响应头部和响应数据。

#### 协议

HTTP请求分成3部分：请求行、请求头、空行、请求正文。

请求方法（目前大多数浏览器只支持GET和POST）：

* GET——向Web服务器请求一个文件；
* POST——向Web服务器发送数据让Web服务器进行处理；
* DELETE——从Web服务器上删除一个文件；
* HEAD——检查一个对象是否存在；
* OPTIONS——查询Web服务器的性能；
* PUT——向Web服务器发送数据并存储在Web服务器内部；
* TRACE——跟踪到服务器的路径；
* CONNECT——对通道提供支持。

Header：HTTP请求、响应中都有Header。Header结构：“key:value”，一行即一个Header。每个Header意义均不同。

|  |  |  |
| --- | --- | --- |
| 字段 | 说明 | 示例 |
| Accept | 可接收的响应内容类型 | accept:text/plain |
| Accept- | 可接收的… |  |
| Authorization | HTTP协议中需要认证资源的认证信息 |  |
| Cache-control | 请求/回复中的，是否使用缓存机 | cache-control:no-cache |
| Connection | 客户端想要优先使用的连接类型 |  |
| Content-length | 以8进制表示的请求体的长度 | content-length:348 |
| Content-type | 请求体的MIME类型 |  |
| Host | 服务器域名和端口号，默认端口可省略 |  |
| User-agent | 浏览器的身份标识字符串 | user-agent:Mozilla/ |
| Via | 告诉服务器，这个请求是由哪个代理发出的 | Via: 1.0 fred, 1.1 itbilu.com.com (Apache/1.1) |
| Referer | 表示跳转到当前页面的之前的页面 |  |
| Origin | 发起一个针对跨域资源共享的请求 |  |

Cookies：Cookie总是保存在客户端中，按在客户端中的存储位置，可分为内存Cookie和硬盘Cookie。

Cookie是用于维持服务端会话状态的，通常是由服务端写入，在后续请求中，供服务端读取。



HTTP请求+cookie的交互流程：



#### UrlEncode

有些符号在URL中是不能直接传递的，如果要在URL中传递这些符号，那么就要使用他们的编码了。编码的格式为：%+字符的ASCII码，即一个百分号%，后面是跟对应字符的ASCII（16进制）码值。

例如：

空格的编码值是“%20”。

+ URL 中+号表示空格 %2B

空格 URL中的空格可以用+号或者编码 %20

/ 分隔目录和子目录 %2F

? 分隔实际的 URL 和参数 %3F

% 指定特殊字符 %25

# 表示书签 %23

& URL 中指定的参数间的分隔符 %26

= URL 中指定参数的值 %3D

#### 状态码

状态码被分为五大类，目前使用的HTTP协议版本是1.1，支持一下的状态码。

|  |  |  |
| --- | --- | --- |
|  | 已定义范围 | 分类 |
| 1xx | 100-101 | 信息提示 |
| 2xx | 200-206 | 成功 |
| 3xx | 300-305 | 重定向 |
| 4xx | 400-415 | 客户端错误 |
| 5xx | 500-505 | 服务器错误 |

|  |
| --- |
| 301/302 Moved Permanently（重定向）请求的URL已移走了。Response中应该包含一个Location URL，说明资源现在所处的位置。 |
| 304 Not Modified（未修改）客户的缓存资源时最新的，要客户端使用缓存。 |
| 404 Not Found 未找到资源。 |
| 501 Internal Server Error 服务器遇到一个错误，使其无法对请求提供服务。 |

#### HTTPS协议

HTTPS是身披SSL外壳的HTTP。HTTPS是一种通过计算机网络进行安全通信的传输协议，经由HTTP进行通信，利用SSL/TLS建立全信道，加密数据包。HTTPS

#### HTTP2协议

HTTP2协议是基于谷歌的SPDY协议的。

* 多路复用
* Stream
* 流量控制
* 服务端推送
* 头部压缩

### 4、UPD（传输层）

User Datagram Protocol：用户数据报协议。在IP数据报服务之上增加了最基本的服务：复用和分用以及差错检测。

UDP，非链接、非可靠传输、效率高。

UDP报头的结构如图：



TCP和UDP的区别：

|  |  |  |
| --- | --- | --- |
| 特征点 | TCP | UDP |
| 是否连接 | 面向连接 | 面向非连接 |
| 传输可靠性 | 可靠 | 会丢包，不可靠 |
| 应用场景 | 传输数据量大 | 传输量小 |
| 速度 | 慢 | 块 |

### 5、QUIC（传输层）

Quick UDP Internet Connection：低延时互联网传输协议。谷歌公司研发的一种基于UDP协议的低时延互联网传输协议。

* ORTT快速连接
* 连接迁移
* 改进拥塞控制
* 双级别流控
* 没有队头阻塞的多路复用
* 实现与升级更灵活