## 说一下TCP/IP[三次握手](https://so.csdn.net/so/search?q=%E4%B8%89%E6%AC%A1%E6%8F%A1%E6%89%8B&spm=1001.2101.3001.7020)，四次挥手的具体细节？

1. **报文段标识符**

* 确认ACK ：TCP协议规定只有ACK=1时有效，也规定连接建立后所有发送的报文的ACK必须为1。
* 同步SYN：在连接建立时用来同步序号。当SYN=1而ACK=0时，表明这是一个连接请求报文。对方若同意建立连接，则应在响应报文中使SYN=1和ACK=1，因此SYN置1就表示这是一个连接请求或连接接受报文。
* 终止FIN：用来释放一个连接。当 FIN = 1 时，表明此报文段的发送方的数据已经发送完毕，并要求释放连接。
* 序号seq：用于对字节流进行编号。例如序号为 301，表示第一个字节的编号为 301，如果携带的数据长度为 100 字节，那么下一个报文段的序号应为 401。
* 确认号ack：期望收到的下一个报文段的序号。例如 B 正确收到 A 发送来的一个报文段，序号为 501，携带的数据长度为 200 字节，因此 B 期望下一个报文段的序号为 701，B 发送给 A 的确认报文段中确认号就为 701。

1. **三次握手**  
   最初两端的TCP进程都处于CLOSED关闭状态，Client（A）主动打开连接，而Server（B）处于LISTEN（监听状态），等待A的连接请求并被动打开连接（由客户端执行connect触发）。

* 第一次握手：由Client发出请求连接数据包： SYN=1 ACK=0 seq=x（TCP规定SYN=1时不能携带数据，但要消耗一个序号，因此声明自己的序号是 seq=x）此时Client进入SYN-SENT（同步已发送）状态，等待Server确认；
* 第二次握手：Server收到请求报文后，如果同意建立连接，则向A发送连接确认报文，即 SYN=1 ACK=1 seq=y，ack=x+1，此时Server进入SYN-RCVD（同步收到）状态；
* 第三次握手：Client收到Server的确认（SYN+ACK）后，向Server发出确认报文段，即 ACK=1，seq=x+1， ack=y+1，TCP连接已经建立，Client进入ESTABLISHED（已建立连接）状态；  
  Server收到Client的确认后，也进入ESTABLISHED状态，完成三次握手；此时Client和Server可以开始传输数据。

1. **四次挥手**  
   数据传输结束后，通信的双方都可释放连接，A和B都处于ESTABLISHED状态。当Client没有数据再需要发送给服务端时，就需要释放客户端的连接，整个过程为：

* 第一次挥手：当Client发起终止连接请求的时候，会发送一个（FIN为1,seq=u）的没有数据的报文，这时Client停止发送数据（但仍可以接受数据） ，进入FIN\_WAIT1（终止等待1）状态，等待Server确认
* 第二次挥手：Server收到连接释放报文后会给Client一个确认报文段（ACK=1,ack=u+1,seq=v）， 进入CLOSE-WAIT（关闭等待）状态  
  Client收到Server的确认后进入FIN\_WAIT2状态，等待Server请求释放连接，Server仍可向Client发送数据。
* 第三次挥手：Server数据发送完成后，向Client发起请求连接释放报文（FIN=1,ACK=1,seq=w,ack = u+1）,Server进入LAST-ACK（最后确认）状态，等待Client确认
* 第四次挥手：Client收到连接释放报文段后，回复一个确认报文段（ACK=1,seq=u+1,ack=w+1），进入 TIME\_WAIT（时间等待） 状态，Server收到后进入CLOSED（连接关闭）状态。经过等待2MSL 时间（最大报文生存时间），Client进入CLOSED状态。

为什么需要三次握手？  
为了实现可靠数据传输， TCP 协议的通信双方， 都必须维护一个序列号， 以标识发送出去的数据包中， 哪些是已经被对方收到的。 三次握手的过程即是通信双方相互告知序列号起始值， 并确认对方已经收到了序列号起始值的必经步骤。  
如果只是两次握手， 至多只有连接发起方的起始序列号能被确认， 另一方选择的序列号则得不到确认

* 为什么A在TIME-WAIT状态必须等待2MSL的时间？  
  MSL最长报文段寿命Maximum Segment Lifetime，MSL=2  
  客户端接收到服务器端的 FIN 报文后进入此状态，此时并不是直接进入 CLOSED 状态，还需要等待一个时间计时器设置的时间 2MSL。这么做有两个理由：
* 保证A发送的最后一个ACK报文段能够到达B。  
  如果 B 没收到 A 发送来的确认报文（A发送的最后一个ACK报文段可能丢失），那么就会重新发送连接释放请求报文，A 等待一段时间就是为了处理这种情况的发生。
* 防止“防止本次已失效的连接请求报文段出现在新的连接中。  
  等待一段时间是为了让本连接持续时间内所产生的所有报文都从网络中消失，使得下一个新的连接不会出现旧的连接请求报文。

# ****如果已经建立了连接，但是客户端突然出现故障了怎么办？****

TCP还设有一个保活计时器，显然，客户端如果出现故障，服务器不能一直等下去，白白浪费资源。服务器每收到一次客户端的请求后都会重新复位这个计时器，时间通常是设置为2小时，若两小时还没有收到客户端的任何数据，服务器就会发送一个探测报文段，以后每隔75秒钟发送一次。若一连发送10个探测报文仍然没反应，服务器就认为客户端出了故障，接着就关闭连接。

#### OSI七层模型

3、各层功能及协议

1）应用层

定义：计算机用户及各种应用程序和网络之间的接口

功能：

1.直接向用户提供服务接口

2.完成用户请求的各种服务

协议：文件传输FTP，电子邮件SMTP，万维网HTTP…

2）表示层

定义：处理在两个通信系统中交换信息的表示方式，如编码、数据格式转换和加密解密等

功能：

1.数据格式处理2.数据编码3.压缩和解压缩4.数据的加密和解密

3）会话层

定义：向两个实体的表示层提供建立和使用连接的方法。将不同实体之间的表示层的连接称为会话。因此会话层的任务就是组织和协调两个会话进程之间的通信，并对数据交换进行管理。

功能：

1.会话管理：建立、管理、终止会话

2.出错控制：使用校验点可使会话在通信失效时从校验点/同步点继续恢复通信，实现数据同步。适用于传输大文件

4）传输层（通信子网（数据处理）和资源子网（数据传输）的桥梁）

定义：向用户提供可靠的、端到端（（每个进程都用一个端口号唯一标识）的差错和流量控制，保证报文的正确传输。传输单位是报文段或用户数据报

功能：

1.传输连接管理：提供建立、连接和拆除传输连接的功能。传输层在网络层的基础上，提供“面向连接TCP”和“面向无连接UDP”两种服务

2.处理传输差错：（1）差错控制（2）流量控制

协议：TCP\UDP

5）网络层

定义：通过路由算法，为报文或分组通过通信子网选择最适当的路径，从源端传到目的端，为不同的网络设备提供通信服务。该层控制数据链路层与物理层之间的信息转发，建立、维持与终止网络的连接。传输单位是数据报。

一般的，数据链路层是解决统一网络内节点之间的通信，而网络层主要解决不同子网之间的通信。例如路由选择问题。

功能：

1、寻址：数据链路层中使用的物理地址（如MAC地址）仅解决网络内部的寻址问题。在不同子网之间通信时，为了识别和找到网络中的设备，每一子网中的设备都会被分配一 个唯一的地址。由于各个子网使用的物理技术可能不同，因此这个地址应当是逻辑地址（如IP地址）

2、交换：规定不同的交换方式。常见的交换技术有：线路交换技术和存储转发技术，后者包括报文转发技术和分组转发技术。

3、路由算法：当源节点和路由节点之间存在多条路径时，本层可以根据路由算法，通过网络为数据分组选择最佳路径，并将信息从最合适的路径，由发送端传送的接受端。

4、连接服务：与数据链路层的流量控制不同的是，前者控制的是网络相邻节点间的流量，后者控制的是从源节点到目的节点间的流量。其目的在于防止阻塞，并进行差错检测（包括：流量控制、差错控制、拥塞控制）

协议：IP

6）数据链路层

定义：接受来自物理层的位流形式的数据，并封装成帧，传送到上一层；同样，也将来自上一层的数据帧，拆装为位流形式的数据转发到物理层；并且还负责处理接受端发回的确认帧的信息，以便提供可靠的数据传输。

功能：1.成帧（定义帧的开始和结束）2.差错控制（帧错+位错）3.流量控制4.访问（接入）控制

7）物理层

定义：利用传输介质为数据链路层提供物理连接，实现比特流的透明传输，屏蔽掉具体传输介质与物理设备的差异。使其上面的数据链路层不必考虑网络的具体传输介质是什么

透明传输的意义就是：不管传的是什么，所采用的设备只是起一个通道作用，把要传输的内容完好的传到对方！

##### TCP/IP四层模型（事实标准）

a)网络接口层（数据链路层）  
（1）作用：实现互连在同一种数据链路（传输介质）结点之间帧传递  
（2）传输单位：帧  
（3）功能：组帧、差错控制、流量控制和传输管理

b)网络层（网际层）  
（1）作用：实现网络中终端节点之间的通信（点对点）/实现多数据链路之间通信  
（2）传输单位：数据报  
（3）功能：1、封装数据成分组、包，路由选择2、流量控制、拥塞控制&网际互联  
（4）协议：

Ip协议：提供网络结点之间的报文传送服务 （1） IP寻址 （2） 路由控制 （3） IP分包与组包

ARP协议：实现IP地址向物理地址的映射\ RARP协议：实现物理地址向IP地址的映射

ICMP协议：探测&报告传输中产生的错误\IGMP协议：管理多播组测成员关系

c)传输层  
（1）作用：主要为两台主机上的应用提供端到端的通信，为应用层（会话）提供可靠无误的数据传输服务，为网络层提供源端与目的端的端口信息。  
（2）传输单位：报文段（TCP）/用户数据报（UDP）  
（3）协议： TCP和UDP

d)应用层  
（1）作用：负责处理特定的应用进程（主机运行的程序）间通信&交互规则  
（2）协议：HTTP协议：提供Internet网浏览服务\DNS协议：负责域名和IP地址的映射\SMTP协议：提供简单电子邮件发送服务\POP、IMAP协议：提供邮箱服务器进行远程存取邮件\FTP协议：提供应用级文件传输服务\Telent协议：提供远程登录服务（明文）\ssh协议：提供远程登录服务（加密）

**TCP的功能**

超时自动回传、滑动窗口、流量控制、拥塞控制：慢开始、拥塞避免、快重传、快恢复。

TCP协议：SMTP、FTP、POP3、TELNET、HTTP

UDP协议：DHCP（Dynamic Host Configuration Protocol，动态主机配置协议）、TFTP（Trivial File Transfer Protocol,[简单文件传输协议](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=175932409&ss_c=ssc.citiao.link)）

##### 使用Cookie的状态管理

Cookie 会根据从服务器端发送的响应报文内的一个叫做 Set-Cookie 的首部字段信息，通知客户端保存Cookie。当下次客户端再往该服务器发送请求时，客户端会自动在请求报文中加入 Cookie 值后发送出去。

Cookie、session、token

session有如用户信息档案表, 存储于服务器，可以理解为一个状态列表，拥有一个唯一识别符号sessionId，通常存放于cookie中。服务器收到cookie后解析出sessionId，再去session列表中查找，才能找到相应session。依赖cookie。

cookie类似用户通行证，一个令牌，装有sessionId，存储在客户端，浏览器通常会自动添加。

token也类似临时的证书签名，一个令牌，无状态，用户信息都被加密到token中，服务器收到token后解密就可知道是哪个用户。需要开发者手动添加。

Session、cookie和token都是服务器给的。

# 跨域问题

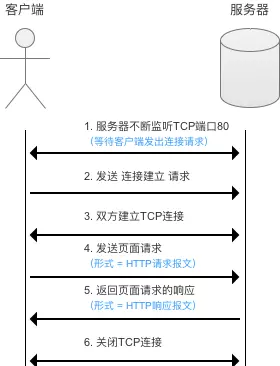
# 描述一下HTTP协议？

HTTP协议（HyperText Transfer Protocol，超文本传输协议）是用于从万维网（WWW）Web服务器 传输超文本到 本地客户浏览器 的传输协议。  
在Internet上的Web服务器上存放的都是超文本信息，客户机需要通过HTTP协议传输所要访问的超文本信息。

无状态、无连接（HTTP 1.0）

无连接指的是没有相互的连接，只有单向的连接，因为在建立一边的连接传输完数据之后就会立马断开。  
HTTP协议特点：

* 基于TCP/IP通信协议传递数据（HTML、图片文件，查询结果等）
* 属于应用层协议
* 采用客户端-服务端（请求-响应 C/S）工作方式，具体工作流程如下：



## HTTP 缓存机制？

* 浏览器第一次请求

在浏览器第一次请求数据时，此时缓存数据库中没有对应的缓存数据，需要请求服务器，服务器返回相应的数据（主体body）和缓存规则（响应头Header）后，浏览器将数据和缓存规则存储至缓存数据库中。  
HTTP缓存规则包括：Expires、Cache-Control（强制缓存规则），Etag、Last-Modified（对比缓存规则

浏览器第二次请求

* 浏览器第二次请求数据时，会根据是否需要向服务器发起请求分为 强制缓存 & 对比缓存

1. 首先执行强制缓存，服务器响应浏览器一个缓存时间（Expires/Cache-Control）
2. 如果强制缓存命中（在缓存时间内），则直接使用缓存，不需与服务器发生交互，不再执行对比缓存规则。
3. 若超出缓存时间，则执行比较缓存策略

# HTTPS

## 描述一下HTTPS？

是**以安全为目标的HTTP通道**，可理解为HTTP的加强版。实现原理是让 HTTP 先和 SSL（Secure Sockets Layer 安全套接字层，TLS（传输层安全）是更为安全的升级版 SSL）通信，再由 SSL 和 TCP 通信，也就是说 HTTPS 使用了SSL/TLS建立信道，加密数据包。

通过使用 SSL，HTTPS 具有了加密（防窃听）、认证（防伪装）和完整性保护（防篡改）。

# [Socket](https://so.csdn.net/so/search?q=Socket&spm=1001.2101.3001.7020)

## 描述一下Socket?

Socket 即 套接字，是通信的基石，是应用层 与 TCP/IP 协议族通信的中间软件抽象层，本质为一个封装了 TCP / IP协议族 的编程接口（属于传输层）。网络上的两个进程端口通过Socket实现一个双向的通信连接从而进行数据交换。

Http是一个特殊处理的socket，建立在TCP/IP上的广泛应用。

## 描述一下WebSocket？

WebSocket是HTML5一种新的协议。它借鉴了Socket这种思想，为web应用程序客户端和服务端之间（注意是客户端服务端）提供了一种全双工通信机制(full-duplex)。同时，它又是一种新的应用层协议。一开始的握手需要借助HTTP请求完成。  
WebSocket同HTTP一样也是应用层的协议，但是它是一种双向通信协议，是建立在TCP之上的。连接过程需要进行握手：