计算机网络

1. 概论
2. 互联网 = 网络边缘 + 网络核心
3. 互联网的核心部分常采用网状拓扑结构，少量节点故障网络也不会瘫痪。
4. 网络分类：PAN、LAN、MAN、WAN
5. 七层：物理层、数据链路层、网络层、传输层、会话层、表示层、应用层
6. 五层：物理层、数据链路层、网络层、传输层、应用层
7. 报文（应）->报文段/用户数据报（运）->分组/数据报->帧（数）->比特（物）

**说出五层体系中各层的功能：**

**应用层**：为应用程序提供交互服务。在互联网中的应用层协议很多，如域名系统DNS、HTTP协议、SMTP协议等。

**传输层**：负责向两台主机进程之间的通信提供数据传输服务。传输层的协议主要有传输控制协议TCP和用户数据协议UDP。

**网络层**：选择合适的路由和交换结点，确保数据及时传送。主要包括IP协议。

**数据链路层**：在两个相邻节点之间传送数据时，数据链路层将网络层交下来的 IP 数据报组装成帧，在两个相邻节点间的链路上传送帧。

**物理层：**实现相邻节点间比特流的透明传输，尽可能屏蔽传输介质和物理设备的差异。

1. TCP/IP四层：网络接口层、网络层、传输层、应用层
2. 因特网边缘主机之间的两种通信方式：
3. C/S：客户服务器方式，二者角色不会改变
4. P2P：对等方式，每一方既是客户又是服务器，二者角色可以改变
5. 电路交换：有一条端到端的，利用中间节点构成的专用物理链路
6. 存储转发的两种方式：报文交换、分组交换（分组交换又分为数据报和虚电路）
7. 物理层
8. 接口特性：机械、电气、功能、过程特性
9. 调制有基带调制（数字信号-数字信号）和带通调制（数字信号-模拟信号）
10. 理想信道的最高码元传输率=2W Baud （w：带宽，Baud：波特率）
11. 信道带宽与信道容量的关系为：C=Wlog2(1+S/N) （C：容量，S：信号功率，N：噪声功率）
12. 信道复用：FDM、TDM、CDM 、STDM （频、时、码、统计时分[按需分配]）
13. 波分复用：光的频分复用
14. 码分多址CDMA（可以同时同频率，靠不同的信号波形来区分）
15. 数据链路层

PPP：广域网点对点（组成：LCP+NCP）

SLP、HDLC、MAC子层协议、LLC子层协议

ALOHA

CSMA（分为CSMA/CD（冲突检测）和CSMA/CA（冲突避免））

1. 数据链路层解决的问题：封装成帧、透明传输、差错检测。
2. 解决透明传输问题：转义符ESC、零比特插入法
3. 差错检测方法：CRC校验
4. 以太网是一种局域网，DIX V2版-世界上第一个局域网产品标准。（总线拓扑结构）（无连接，不可靠）
5. 以太网帧由（目的MAC地址）（源MAC地址）（类型）（数据）（校验）五个字段组成，数据字段范围是（46）-（1500）字节，校验字段占（4）字节，采用的是（CRC）校验。通过（类型）字段，以太网可以支持多种网络层协议。
6. 以太网采用CSMA/CD：先听后发 、边听边发、冲突停发、延迟重发
7. 网卡、HUB（集线器）、网桥、透明网桥、交换机
8. 网络层

IP

ARP：地址解析（IP地址->MAC地址）

RARP：逆地址解析

ICMP：（差错报告，不单独工作需封装在IP包中传输）

IGMP

OSPF

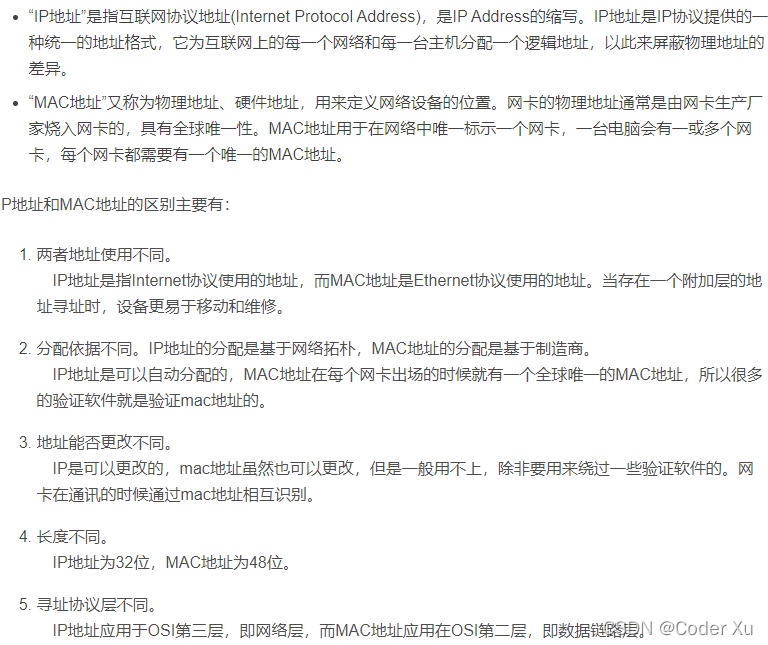
路由信息协议RIP：选择路由路径

1. IP地址共分5类，分别记作A、B、C、D、E。只有A、B、C三类地址可以分配给计算机和网络设备，D类地址为组播地址，E类地址用于实验。
2. 回送地址：127.xx.xx.xx数据直接发送给本机
3. 所有主机号为“1”的地址用于广播，主机号为0的IP地址表示网络地址
4. 本网上的本主机：用32位全为0的IP地址表示
5. 为保证IP数据报传输效率，因特网规定，主机、路由器能够接受的最短IP数据报长度为576字节
6. 子网掩码中对应于网络号和子网号的所有数据位都置为1，而对应于主机号的所有位都置为0
7. IPv4到IPv6的过渡方法：双协议栈，隧道（在IPv4区域中打通了一个IPv6隧道来传输IPv6数据分组）

**IP地址的分类**



**谈谈IP地址和MAC地址**



1. 运输层

TCP：面向连接，可靠，一对一通信，面向字节流

UDP：无连接，不可靠

1. socket：IP+端口，可以唯一确定网络中的一个连接。
2. 拥塞控制就是要防止过多数据注入到网络中，这样不会使路由器或链路过载；

**TCP和[UDP](https://so.csdn.net/so/search?q=UDP&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_46594796/article/details/_blank)区别和应用**

TCP是可靠传输，UDP是不可靠传输;

TCP面向连接，UDP无连接，即发送数据之前不需要建立连接

TCP传输数据有序，UDP不保证数据的有序性;

TCP面向字节流，把数据看成一连串字节流，UDP是面向报文的

TCP传输速度相对UDP较慢;

TCP有流量控制和拥塞控制，UDP没有;

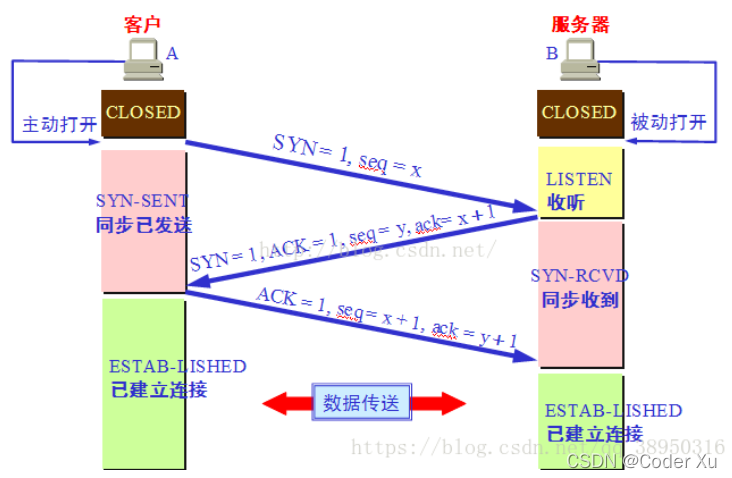
TCP是重量级协议，UDP是轻量级协议;

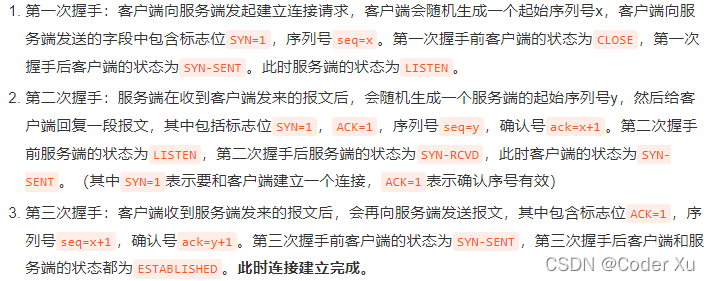
TCP首部20字节，UDP首部8字节;

TCP连接只能是一对一的（端到端）；UDP支持一对一、一对多、多对一和多对多的通信方式。

TCP有拥塞控制而UDP没有

**三次握手**





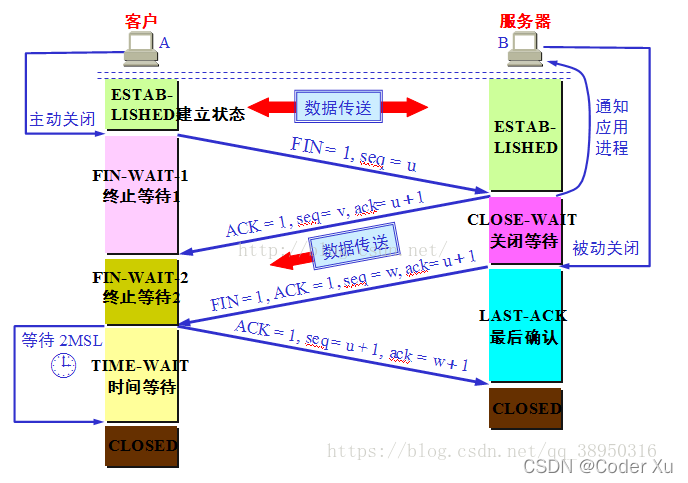
tcp都是c/s方式的，握手都是客户端主动发起的

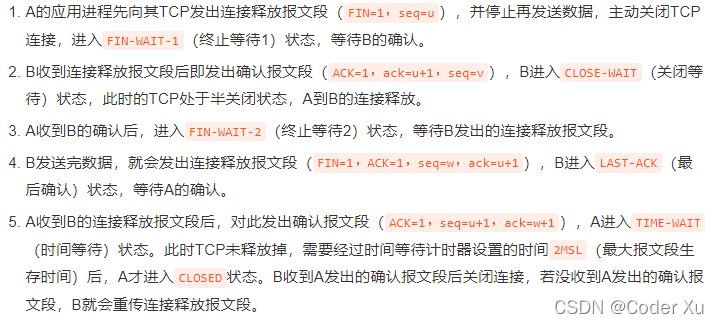
1. 客户端向服务器端发送链接请求数据包，tcp首部的标记位（同步标记SYN）是1，确认标记（ACK）是0，序号seq是x（x可能是0也可能是1）
2. 服务器端发回确认数据包，tcp首部的标记位也是1，确认标记是1，序号seq是y（S决定可能是1也可能是0），确认号ack为x+1
3. 客户端再次发送数据包，没有同步标记SYN，确认标记ACK是1，序号seq是x+1表示现在发的是第x+1字节，确认号ack是y+1

**为什么不是两次握手**

### IMG_256

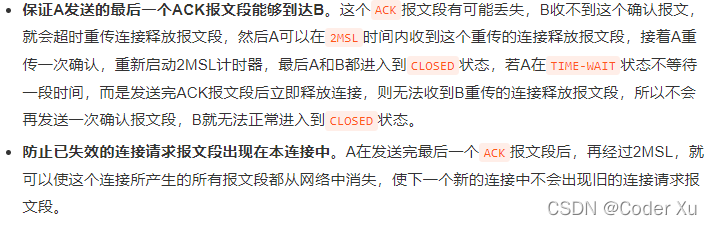
### **四次挥手**





1. 通信结束后双方都可以释放连接
2. 现在由客户端先向TCP发送连接释放报文段，该报文段首部的FIN=1，seq=u，等待服务器端的确认
3. 服务器端发回对客户端的确认，确认标记ACK为1，序号seq为v，确认号ack为u+1
4. 此时服务器仍可以向A传输数据
5. 服务器端传输完了也向客户端发送释放TCP报文段，FIN标记为1，确认标记ACK为1，序号seq为w，确认号ack为u+1
6. 客户端再次发送确认报文段，确认标记ACK为1，序号seq为u+1，确认号为w+1，之后进入TIMEWAIT状态，经过2MSL时间后，TCP真正被释放

**四次挥手中为什么等待2MSL**



### **TCP 如何保证可靠传输**

（1）数据包校验：目的是检测数据在传输过程中的任何变化，若校验出包有错，则丢弃报文段并且不给出响应，这时 TCP 发送数据端超时后会重发数据；

（2）对失序数据包重排序：既然 TCP 报文段作为 IP 数据报来传输，而 IP 数据报的到达可能会失序，因此 TCP 报文段的到达也可能会失序。TCP 将对失序数据进行重新排序，然后才交给应用层；

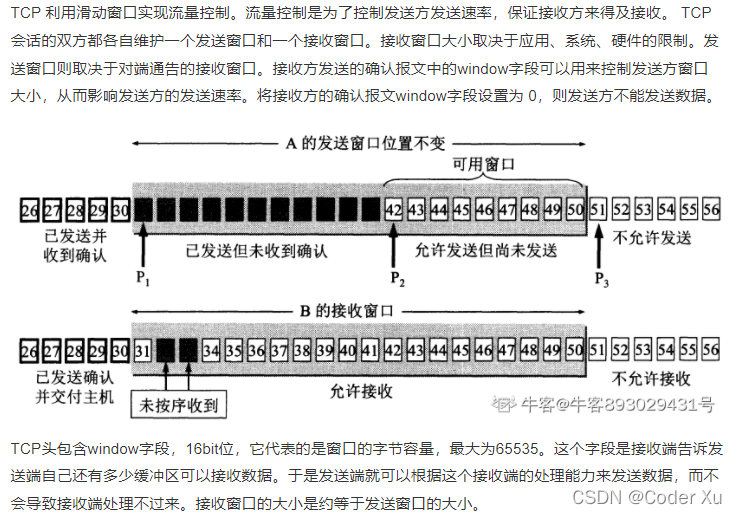
（3）丢弃重复数据：对于重复数据，能够丢弃重复数据；

（4）应答机制：当 TCP 收到发自 TCP 连接另一端的数据，它将发送一个确认。这个确认不是立即发送，通常将推迟几分之一秒；

（5）超时重发：当 TCP 发出一个段后，它启动一个定时器，等待目的端确认收到这个报文段。如果不能及时收到一个确认，将重发这个报文段；

（6）流量控制：TCP 连接的每一方都有固定大小的缓冲空间。TCP 的接收端只允许另一端发送接收端缓冲区所能接纳的数据，这可以防止较快主机致使较慢主机的缓冲区溢出，这就是流量控制。TCP 使用的流量控制协议是可变大小的滑动窗口协议。

### **TCP[流量控制](https://so.csdn.net/so/search?q=%E6%B5%81%E9%87%8F%E6%8E%A7%E5%88%B6&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_46594796/article/details/_blank)（滑动窗口机制）**



**TCP拥塞控制**

拥塞控制和流量控制不同，拥塞控制是一个全局性的过程，而流量控制指点对点通信量的控制。

拥塞控制就是为了防止过多的数据注入到网络中，这样就可以使网络中的路由器或链路不致于过载。为了进行拥塞控制，TCP 发送方要维持一个拥塞窗口(cwnd)的状态变量。拥塞控制窗口的大小取决于网络的拥塞程度，并且动态变化。发送方让自己的发送窗口取为拥塞窗口和接收方的接受窗口中较小的一个。 TCP的拥塞控制采用了四种算法，即：慢开始、拥塞避免、快重传和快恢复。在网络层也可以使路由器采用适当的分组丢弃策略（如：主动队列管理AQM），以减少网络拥塞的发生。

慢开始：

慢开始算法的思路是当主机开始发送数据时，如果立即把大量数据字节注入到网络，那么可能会引起网络阻塞，因为现在还不知道网络的符合情况。经验表明，较好的方法是先探测一下，即由小到大逐渐增大发送窗口，也就是由小到大逐渐增大拥塞窗口数值。cwnd初始值为 1，每经过一个传播轮次，cwnd 加倍。

拥塞避免：

拥塞避免算法的思路是让拥塞窗口 cwnd 缓慢增大，即每经过一个往返时间 RTT 就把发送方的 cwnd 加 1。

快重传与快恢复：

在 TCP/IP 中，快速重传和快恢复（FRR）是一种拥塞控制算法，它能快速恢复丢失的数据包。

没有 FRR，如果数据包丢失了，TCP 将会使用定时器来要求传输暂停。在暂停的这段时间内，没有新的或复制的数据包被发送。有了FRR，如果接收机接收到一个不按顺序的数据段，它会立即给发送机发送一个重复确认。如果发送机接收到三个重复确认，它会假定确认件指出的数据段丢失了，并立即重传这些丢失的数据段。有了FRR，就不会因为重传时要求的暂停被耽误。当有单独的数据包丢失时，快速重传和快恢复（FRR）能最有效地工作。当有多个数据信息包在某一段很短的时间内丢失时，它则不能很有效地工作。

1. 应用层

HTTP、HTTPS

PING属于应用层协议，是应用层直接使用网络层ICMP 的例子，它没有通过运输层的 TCP或UDP。

**HTTP和HTTPS区别**

（1）http 协议是免费使用的，而 https 协议需要到CA机构申请证书，需要缴纳费用

（2）http 是超文本传输协议，信息是明文传输，https 则是具有安全性的 ssl/tls 加密传输协议，信息是密文

（3）http 的连接很简单，是无状态的；https 协议是由SSL/TLS+HTTP协议构建的可进行加密传输、身份认证的网络协议，比 http 协议安全

（4）和 http 通信相比，https 通信会由于加减密处理消耗更多的CPU和内存资源

（5）http 和 https 使用的端口也不一样，前者是80，后者是443

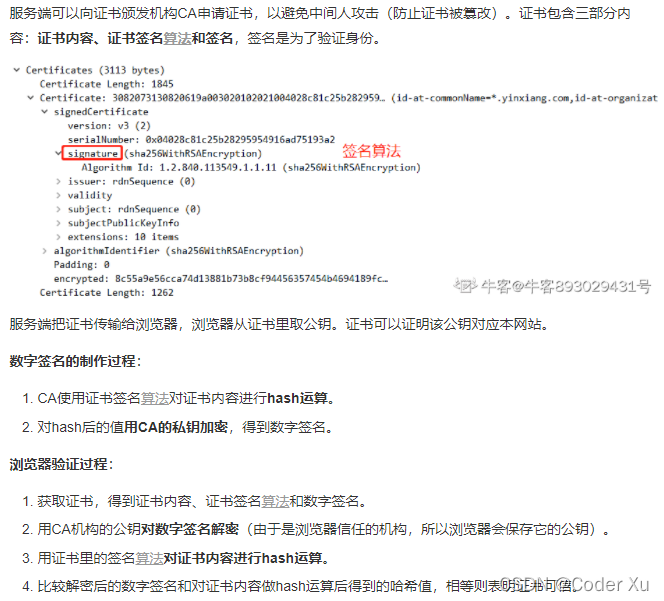
### **[对称加密](https://so.csdn.net/so/search?q=%E5%AF%B9%E7%A7%B0%E5%8A%A0%E5%AF%86&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_46594796/article/details/_blank)与非对称加密**

**对称加密**：是指加密和解密使用同一个密钥的方式，一方通过密钥将信息加密后，把密文传给另一方，另一方通过这个相同的密钥将密文解密，转换成可以理解的明文。

**非对称加密：**使用一对非对称密钥加密，即公钥和私钥，公钥可以随意发布，任何人都能获得，但私钥只有自己知道

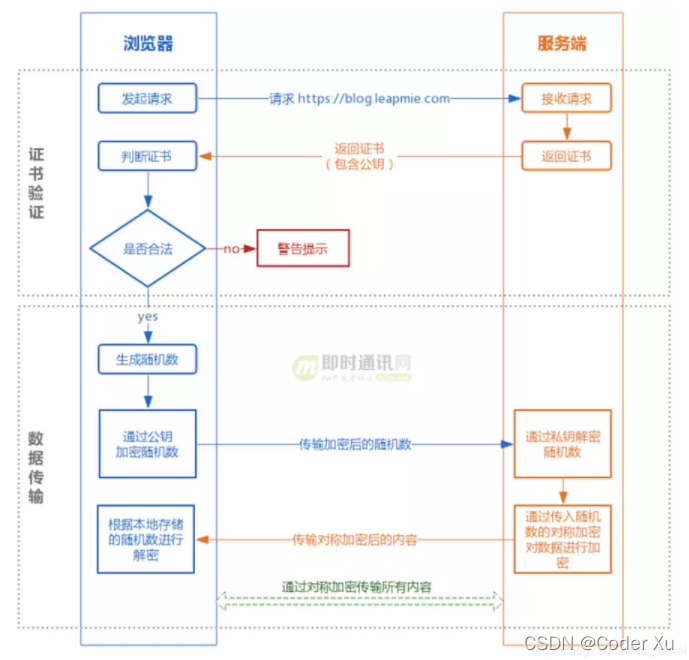
发送密文的一方使用对方的公钥进行加密处理，对方接收到加密信息后，使用自己的私钥进行解密。公开密钥与私有密钥是一对，如果用公开密钥对数据进行加密，只有用对应的私有密钥才能解密；如果用私有密钥对数据进行加密，那么只有用对应的公开密钥才能解密。因为加密和解密使用的是两个不同的密钥，所以叫作非对称加密

**什么是数字证书**



**HTTPS工作流程**

HTTPS 协议是安全的，因为 HTTPS 协议会对传输的数据进行加密，而加密过程是使用了非对称加密实现。但是 HTTPS 在内容传输使用的是对称加密，在证书验证阶段使用非对称加密  
所以HTTPS加密方式是：对称加密 + 非对称加密混合



证书验证阶段：  
1）浏览器发起 HTTPS 请求；  
2）服务端返回 HTTPS 证书；  
3）客户端验证证书是否合法，如果不合法则提示告警。

数据传输阶段：  
1）当证书验证合法后，在本地生成随机数；  
2）通过公钥加密随机数，并把加密后的随机数传输到服务端；  
3）服务端通过私钥对随机数进行解密；  
4）服务端通过客户端传入的随机数构造对称加密算法，对返回结果内容进行加密后传输。

**HTTP常见状态码**



（1）200 OK：表示从客户端发送给服务器的请求被正常处理并返回成功

（2）301 Moved Permanently：永久性重定向，表示请求的资源被分配了新的URL，之后应使用更改的URL；

（3）302 Found：临时性重定向，表示请求的资源被分配了新的URL，希望本次访问使用新的URL；

（4）400 Bad Request：表示请求报文存在语法错误；

（5）403 Forbidden：服务器拒绝该次访问（访问权限问题）

（6）404 Not Found：表示服务器上无法找到请求的资源（也可以在服务器拒绝请求但不想给拒绝原因时使用）

（7）500 Inter Server Error：表示服务器在执行请求时发生了错误，也有可能是web应用存在的bug或某些临时的错误时；

（8）503 Server Unavailable：表示服务器暂时处于超负载或正在进行停机维护，无法处理请求；

**HTTP长连接和短连接，流水线是什么**

**长连接和短连接：**

（1）HTTP1.0默认使用的是短连接。浏览器和服务器每进行一次HTTP操作，就建立一次连接，任务结束就中断连接。

（2）HTTP/1.1起默认使用长连接。要使用长连接，客户端和服务器的HTTP响应头部的Connection要设置为keep-alive，才能支持长连接。

（3）HTTP长连接，指的是复用TCP连接。多个HTTP请求可以复用同一个TCP连接，这就节省了TCP连接建立和断开的消耗。

**流水线：**

默认情况下，HTTP 请求是按顺序发的，下一个请求只有在当前请求收到响应之后才会被发出，有点像停止等待协议。而在收到下一个请求之前，可能需要很长时间。流水线就是在一个 HTTP 长连接下连续发出请求，不用等待响应返回，减少延迟。

**GET和POST区别和应用**

GET产生的URL地址可以在书签收藏，而POST不可以。

GET请求会被浏览器主动缓存，而POST不会，除非手动设置。

GET请求只能进行url编码，而POST支持多种编码方式。

GET请求参数会被完整保留在浏览器历史记录里，而POST中的参数不会被保留。

GET请求在URL中传送的参数是有长度限制的（通常URL不大于2048个字符），而POST没有。

对参数的数据类型，GET只接受ASCII字符，而POST没有限制。

GET比POST更不安全，因为参数直接暴露在URL上，所以不能用来传递敏感信息。

GET参数通过URL拼接传递，POST参数通过请求体传递

GET产生一个TCP数据包；POST产生两个TCP数据包。对于GET方式的请求，浏览器会把请求头和请求体一并发送出去；而对于POST，浏览器先发送请求头，服务器响应100 continue，浏览器再发送请求体。

### **浏览器输入URL到页面返回详细过程**

1.输入网址 输入要访问的网址，即URL

2.缓存解析 浏览器获取URL后，先去缓存中查找资源，从浏览器缓存-系统缓存-路由器缓存中查看； 如果有就从缓存中显示界面，不再发送请求； 如果没有，则发送http请求；

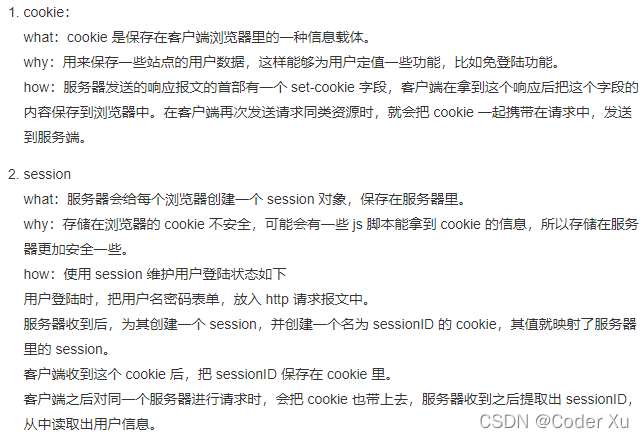
3.域名解析 发现缓存中没有资源，发送http请求； 在发送http请求之前，需要进行DNS解析(域名解析)； DNS解析：域名到IP地址的转换过程，域名的解析工作由DNS服务器完成，解析后可以获取域名相应的IP地址；

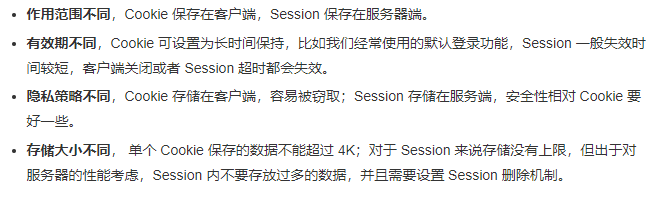
4.tcp连接 三次握手在域名解析后，浏览器向服务器发起了http请求，tcp连接； 因为tcp协议是面向连接的，所以在传输数据前必须建立连接，即三次握手；tcp连接建立后，浏览器开始向服务器发送http请求报文

5.收到请求 服务器收到浏览器发送的请求信息，返回响应

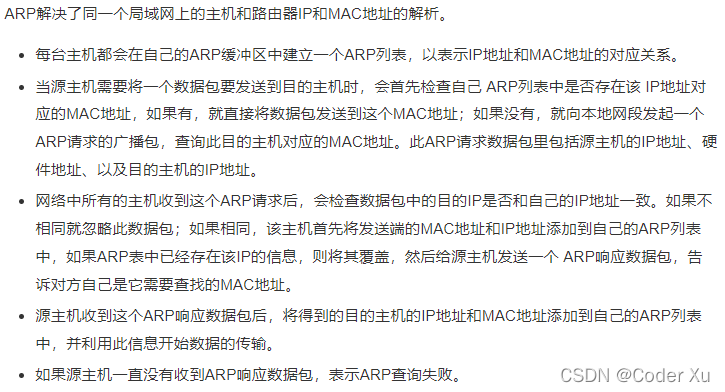
6.页面渲染 浏览器收到服务器发送的响应，显示页面内容

### **什么是Cookie和[Session](https://so.csdn.net/so/search?q=Session&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_46594796/article/details/_blank)，二者区别**





### **什么是[ARP](https://so.csdn.net/so/search?q=ARP&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_46594796/article/details/_blank)协议**



### **什么是XSS攻击**

XSS攻击原理是攻击者向有XSS漏洞的网站中输入恶意的 HTML 代码，当用户浏览该网站时，这段 HTML 代码会自动执行，从而达到攻击的目的。

例如：<script>alert("close me!")</script>

XSS 攻击类似于 SQL 注入攻击，SQL注入攻击中以SQL语句作为用户输入，从而达到查询/修改/删除数据的目的，而在xss攻击中，通过插入恶意脚本，实现对用户游览器的控制，获取用户的一些信息

### **DNS解析过程**

浏览器搜索自己浏览器的DNS缓存

若没有，则搜索本地操作系统中的DNS缓存和hosts文件

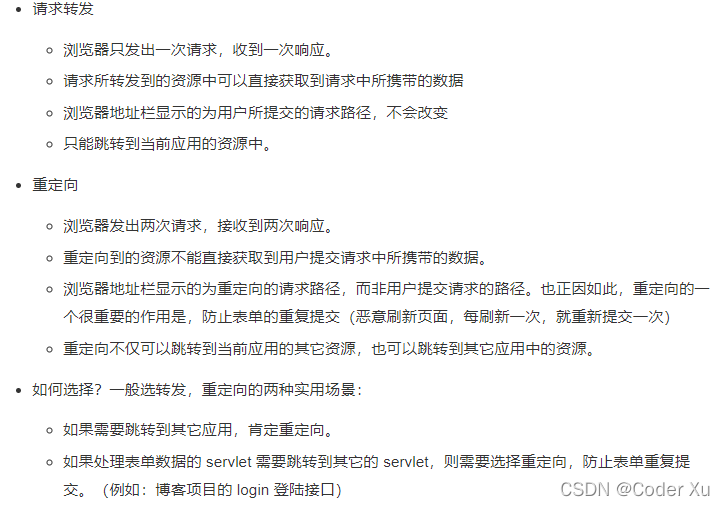
若没有，则操作系统将域名发送至本地域名服务器，本地域名服务器查询自己的DNS缓存，查找成功则返回结果，否则依次向根域名服务器、顶级域名服务器、权限域名服务器发起查询请求，最终返回IP地址给本地域名服务器

本地域名服务器将得到的IP地址返回给操作系统，同时自己也将IP地址缓存起来

操作系统将 IP 地址返回给浏览器，同时自己也将IP地址缓存起来

浏览器得到域名对应的IP地址

### **HTTP请求转发和重定向的区别**



**什么是DHCP协议**

DHCP（动态主机配置协议） 是一个局域网的网络协议。  
指的是由服务器控制一段IP地址范围，给用户提供了即插即用的联网方式，用户不需要再手动配置 IP 地址等信息  
该协议自动为用户分配TCP/IP参数信息，如：IP地址，子网掩码，网关等信息。

**什么是SQL注入**

SQL注入就是将原本的SQL语句的逻辑结构改变，使得SQL语句的执行结果和原本开发者的意图不一样；

比如说我们通过账号密码进行登录，首先需要进行查询判断：

"select \* from t\_user where name = "张三" and password = "123456"

如果我们把or 1 = 1作为拼接在密码末尾传入进来，执行查询的时候sql会变成

"select \* from t\_user where name = "张三" and password = "123456" or 1 = 1

密码永远都是成立的，所以，前面无论密码正确与否都会成功登录

我们使用预编译语句就可以避免这个问题，例如：JDBC中的PrepareStatement或MyBatis中#{}

将sql预编译，传参数的时候，不会改变sql语句结构，就可以避免注入。

**URL和URI区别**

URL，统一资源定位符，URL 其实就是我们平时上网时输入的网址，它标识一个互联网资源，并指定对其进行操作或获取该资源的方法。例如 https://leetcode-cn.com/problemset/all/ 这个 URL，标识一个特定资源并表示该资源的某种形式是可以通过 HTTP 协议从相应位置获得。

URI，统一资源标识符，URL 是 URI 的一个子集，两者都定义了资源是什么，而 URL 还定义了如何能访问到该资源。URI 是一种语义上的抽象概念，可以是绝对的，也可以是相对的，而URL则必须提供足够的信息来定位，是绝对的。简单地说，只要能唯一标识资源的就是 URI，在 URI 的基础上给出其资源的访问方式的就是 URL。

通俗理解

URI表示的是一个抽象的地址，URL表示的是一个详细的地址。

抽象的地址：辽宁省大连市（这是一个抽象的地址，相当于URI）

详细的地址：辽宁省大连市高新园区软件园B2（这是一个详细的地址，相当于URL）

为什么URL是URI的子集，高新园区软件园B2（URL）属于辽宁省大连市（URI）

那么https://www.csdn.net 是一个URI（URI只指明了服务器的地址，没有具体到文件是什么类型）

那么https://www.csdn.net/image/logo.gif就是一个URL（他具体到了logo文件的位置并且logo文件是gif类型的）

**粘包与拆包**

产生原因：

（1）要发送的数据小于TCP发送缓冲区的大小，TCP将多次写入缓冲区的数据一次发送出去，将会发生粘包；

（2）接收数据端的应用层没有及时读取接收缓冲区中的数据，将发生粘包；

（3）要发送的数据大于TCP发送缓冲区剩余空间大小，将会发生拆包；

（4）待发送数据大于MSS（最大报文长度），TCP在传输前将进行拆包。即TCP报文长度-TCP头部长度>MSS。

解决策略：

（1）消息定长。发送端将每个数据包封装为固定长度（不够的可以通过补0填充），这样接收端每次接收缓冲区中读取固定长度的数据就自然而然的把每个数据包拆分开来。

（2）设置消息边界。服务端从网络流中按消息边界分离出消息内容。在包尾增加回车换行符进行分割，例如FTP协议。

（3）将消息分为消息头和消息体，消息头中包含表示消息总长度（或者消息体长度）的字段。

（4）更复杂的应用层协议。

**什么是阻塞和非阻塞，同步和异步**

（1）阻塞和非阻塞：调用者在事件没有发生的时候，一直等待事件发生，不能处理其他任务是阻塞。调用者在事件没有发生的时候，可以去处理别的事务这是非阻塞。

（2）同步和异步：调用者循环查看事件有没有发生，这种情况是同步。调用者不用自己去查看事件有没有发生，而是等待注册在时间上的回调函数自己通知自己，这是异步。

**HTTP版本区别（1.0，1.1，2.0）**

HTTP1.0：

**HTTP1.0仅提供了最基本的认证，用户名和密码都未加密（明文）**

**HTTP1.0仅支持短连接，每次发送数据都会经过TCP三次握手和四次挥手，效率低**

**HTTP1.0只使用了header的if=modified-Since和Expires作为缓存失效的标准**

**HTTP1.0不支持端点续传，每次发送数据都会发送全部数据**

**HTTP1.0认为每台计算机都只能绑定一个IP地址，不支持虚拟网络**

HTTP1.1：

HTTP1.1使用了摘要算法进行身份验证

HTTP1.1默认使用长连接：只需要建立一次连接，可以传输多次数据，传输完成之后，只需要一次切断即可。通过请求头的 Connection: keep-alive设置

HTTP1.1支持端点续传：通过请求头的Range实现

HTTP1.1使用了虚拟网络：在一台物理服务器上可以存在多个虚拟主机，共享一个IP地址

HTTP2.0：

HTTP2.0支持头部压缩：利用HPACK算法进行压缩，由于HTTP1.1头部经常出现Cookie、Accept、Sever、Range等字段可能会占用几百到几千字节，而body有时只有几十字节（“头重身轻”）

HTTP2.0使用二进制格式：HTTP2.0选择了更靠近TCP/IP的二进制格式，抛弃了ASCII码，提高了解析效率

HTTP2.0强化安全：HTTP2.0一般都跑在HTTPS上

HTTP2.0多路复用：一个连接上可以有多个请求