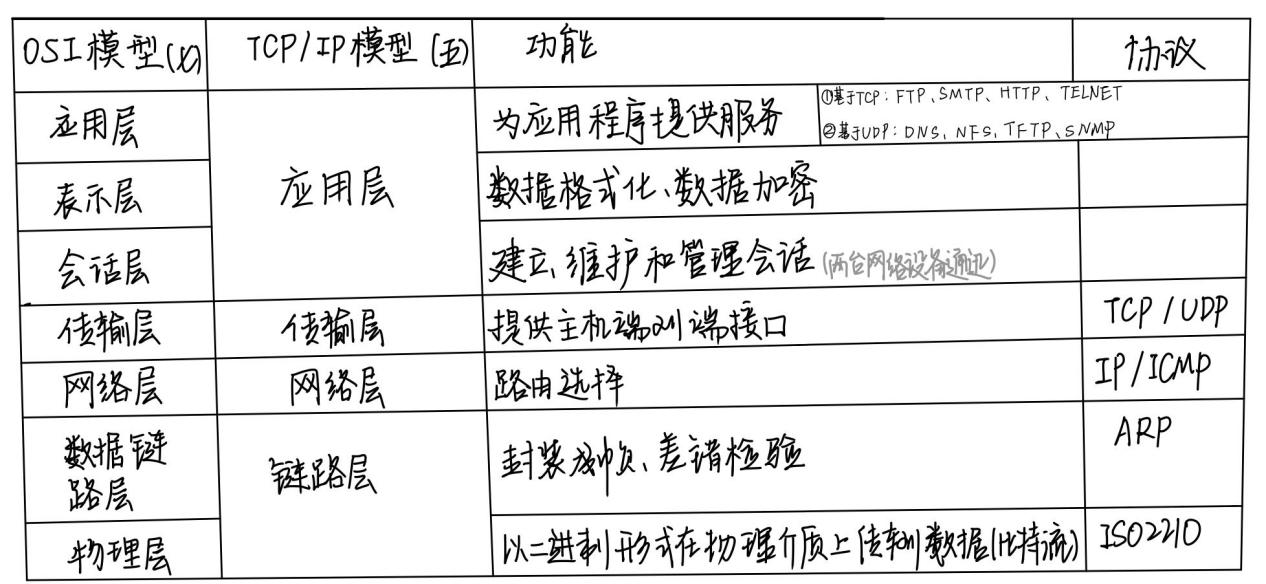
CSDN:111辄。祝学业顺利！

2022.5.30 （计算机面试专业课资料 7个问题）

2022.6.1（计算机面试专业课资料7个问题）

2022.6.2（计算机面试专业课资料6个问题。其它问题都不太常考，先不整理了，搞下一门）

**1.OSI和TCP/IP模型各个层之间的协议和功能**



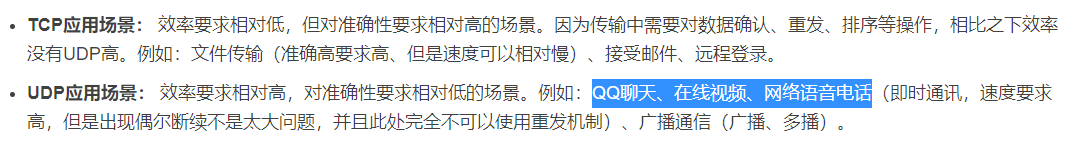
1. **计算机网络为什么要分层？优点？**
2. 各层独立。只需要独立实现本层功能，不需要知晓其他层的实现细节。
3. 灵活性佳，易于实现和维护。当某层发生变化时，只要层间接口关系保持不变，则此层以上或以下层不受影响。
4. **TCP和UDP的异同点**

相同点：TCP和UDP都是传输层协议

不同点：① TCP面向连接（传输控制协议），是可靠的。在传送数据前必须建立连接， 传送结束后要释放连接。它的负载大，适用于大文件传输。

② UDP无连接，是不可靠的。在传送数据前不需要建立连接，接受UDP报文后不 需要给出任何确认。它的负载小，时延小，适用于小文件。





1. **端口和套接字**

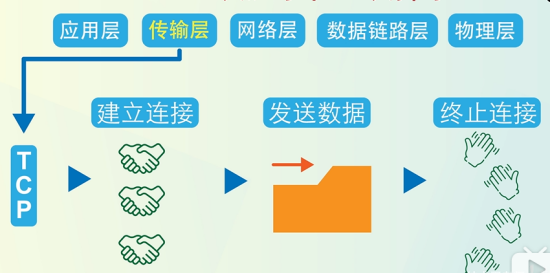
端口：用于标识主机中的应用进程

套接字：唯一标识网络中的一个主机和它上面的一个进程。套接字Socket=(主机IP地址，端 口号)

1. **协议的三要素**

语法、语义、时序

1. **TCP三次握手（建立连接）**



TCP建立连接时，需传输三个数据包，即三次握手（确认双方是否具备正常发送、接收能力）

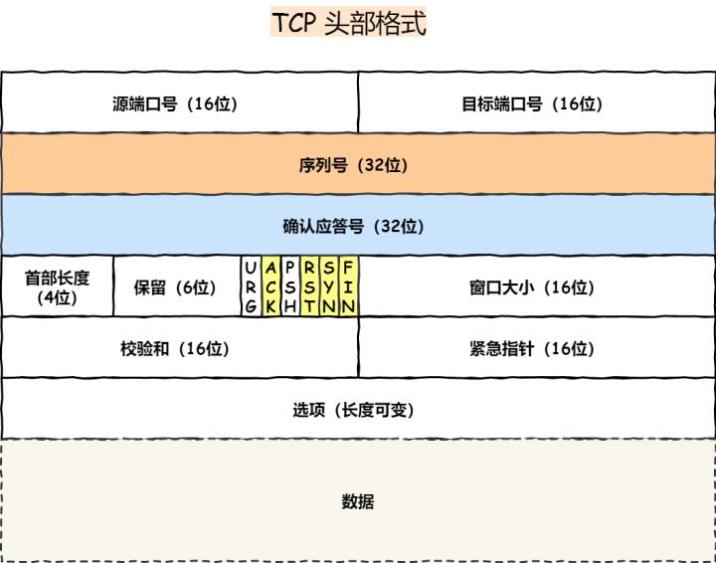
①客户端向服务器端发送一段TCP报文。其中，控制位SYN=1，表示“请求建立新连接”。（Seq=x，一般是1）

②服务器端接收到该报文后，返回一个报文。其中，控制位SYN=1，ACK=1，表示“确认客户端的报文Seq序号有效，服务器能正常连接，且同意创建新连接”。（序号Seq=y，确认号ACK=x+1）

③客户端收到服务器传来的包后，再发送第三个包。其中，SYN=0，ACK=1，表示“确认收到服务器端同意连接的信号”。（Seq=ACK收到=x+1，ACK=y+1）

Ps：SYN跟请求连接有关，ACK跟数据确认有关

第三次握手是可以携带（应用层）数据的，前两次握手是不可以携带数据的



**序列号**：标识从计算机A发送到计算机B的数据包的序号，计算机发送数据时对此标记， 解决**网络包乱序**问题。在建立连接时，由计算机生成的随机数作为其初始值，通过 SYN包传送给主机。每发送一次数据，就 累加 一次该 数据字节数 的大小。

**确认应答号**：指下一次 期望 收到的数据的序列号。发送端收到这个确认应答后可认为在这 个序号之前的数据已被正常接收。用来解决**丢包**问题。

**控制位**：①ACK：该位为1时，确认应答的字段变为有效。TCP规定除了最初建立连接时的 SYN包以外，该位必须设置为1

②SYN：该位为1时，表示希望建立连接，并在其序列号的字段进行序列号初始值 的设定

[参考链接](https://baijiahao.baidu.com/s?id=1654225744653405133&wfr=spider&for=pc)

[参考链接2](https://zhuanlan.zhihu.com/p/125715300)

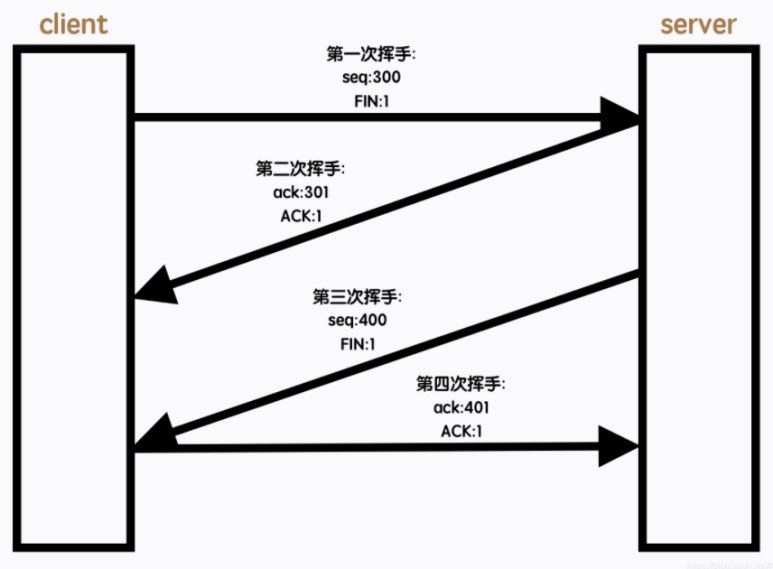
[参考链接3](https://wenku.baidu.com/view/45d47af4270c844769eae009581b6bd97f19bc88.html)

1. **为什么不两次握手？为什么不四次握手？为什么要三次握手？**

两次握手会造成消息滞留堵塞的情况下，服务器重复接受无用的连接请求SYN报文，造成重复分配资源。比如客户端发送请求连接的报文，但是中间该报文堵塞滞留了，客户端一直没等到回应，也不知道服务器端收到没有，就再发了一次。而这是第一次发的SYN请求报文也到了，服务器就会收到两次同样的请求连接报文，造成资源浪费

**为什么三次？**因为两次xxx，四次（三次握手就已经理论上最少可靠连接建立，所以不需要使用更多的通信次数），所以三次合适

1. **TCP四次挥手（断开连接）**



1. 客户端要关闭连接FIN，告知服务器我要关闭了
2. 服务器返回确认数据ACK，客户端收到，关闭从客户端到服务器的连接
3. 胆服务器到客户端的连接仍未断开，可继续传输数据。当服务器发送完所有数据后，发送FIN段来关闭此方向的连接
4. 客户端发送ACK确认关闭连接

Ps：

TCP是全双工的，双向传输。即两台设备可以同时进行双向资料传输。例如手机，在讲话时也可听到对方声音

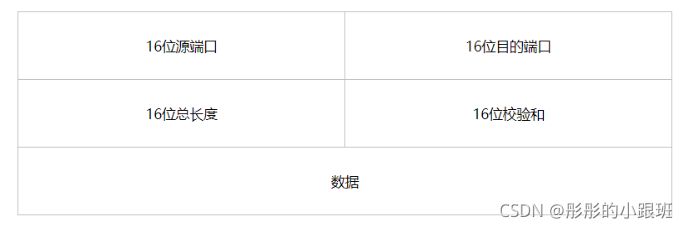
[参考链接1](https://blog.csdn.net/klk0011/article/details/114787151)

[参考链接2](https://blog.csdn.net/LOOKTOMMER/article/details/121307137)

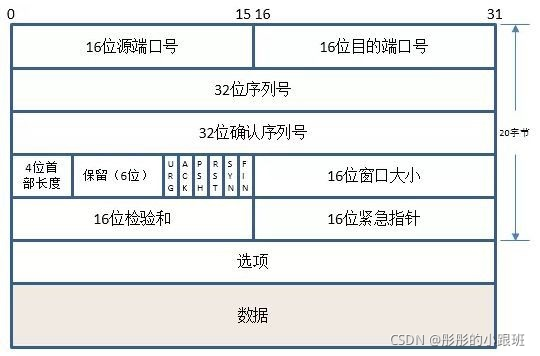
1. **TCP和UDP的头部**

UDP的结构较简单，包括16位源端口、16位目的端口，16位校验和，16位表示数据包的长度。TCP还多了32位序列号、32位确认序号及控制位等。

UDP:



TCP:



1. **CSMA/CD协议**

载波监听多点接入/碰撞检测协议。数据链路层协议，广播时使用。

**基本原理**：所有节点都共享网络传输信道，节点在发送数据之前，首先检测**信道是否空闲**，如果信道空闲则发送，否则就等待；在发送出信息后，再对**冲突**进行检测，当发现冲突时，则取消发送。

**举例**：假设有这一层楼，两旁住了几十户人，中间只有一条仅供一人同行的走道。我们看情况会怎么样：①当这些住户要经过走道出来时，首先探出头来看看走道上有没有人（这就是载波监听），如果没有，就通过走道出来；②如果走道上有人走，那么就一直盯着走道，直到走道上没人时再出来（1-坚持监听算法）；③如果有两人同时看到走道上没有人，而同时走向走道（冲突检测），则两个人发现时就马上回到自己屋里。在整个协议中最关键的是**载波监听**、**冲突检测**两部分

1. **简述下TCP建立连接的过程，TCP如何保证可靠传输？**
2. 建立可靠连接，三次握手
3. 为数据包标号，确保按序发送
4. 流量控制：控制发送方发送速率，保证接收方来得及接收
5. 拥塞控制： 当网络拥塞时，减少数据的发送
6. ARQ协议(自动重传请求)，如果发送方一段时间没有收到确认帧，重新发送

**确认** **超时重传**

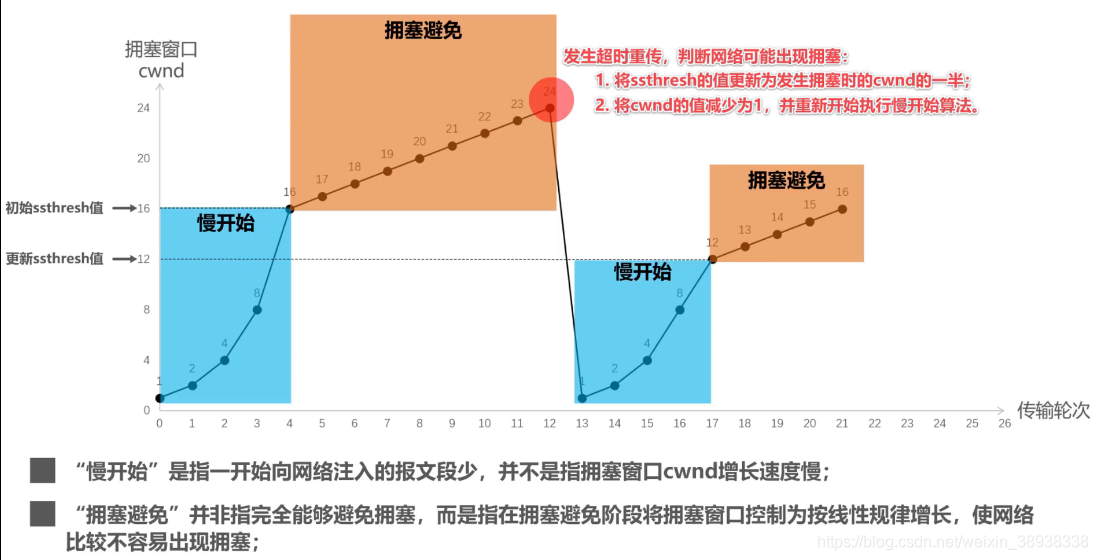
[参考链接1](https://blog.csdn.net/qq_46728644/article/details/123076640)

[参考链接2](https://blog.csdn.net/Sansipi/article/details/121459589)

1. **拥塞控制**

拥塞控制是防止过多数据进入网络中，使得路由器或链路过载。即防止输入负载超过网络吞吐量。而网络拥塞的依据是报文丢失（没有发送方收到确认报文）

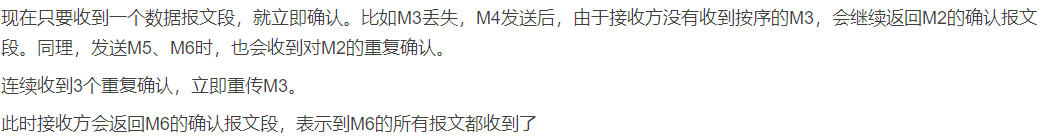
① 发送方维护用于发送的拥塞窗口crown，其大小取决于网络拥塞情况，且动态变化。发送方还会维护一个慢开始门限。当小于该门限时，使用**慢开始算法**；当大于该门限时，使用**拥塞避免算法**



② **快重传算法**

不再在数据全传完再发送确认报文，而是收到一个数据报文段，就立刻发送确认。如果重复发送了三次确认，则说明有报文丢失的情况。

例如，



③ **快恢复算法**

发送方收到3个重复确认，知道丢失了个别报文段。于是不启动慢开始算法，而是执行快恢复算法。发送方会将慢开始门限和拥塞窗口值调整为当前的一般，并且开始执行拥塞避免算法。

[参考链接](https://blog.csdn.net/weixin_38938338/article/details/114867704)

1. **流量控制**

控制发送方发送帧的速率，确保接收方可以接收每个帧

数据链路层的流量控制是点到点的，传输层的数据链路是端到端的

1. 滑动窗口协议

发送方和接收方分别维护可以连续发送帧、接受帧的窗口：发送窗口和接收窗口。

发送窗口中的序号表示已经发送但是未收到确认的帧，或者可以发送的帧，每收到确认帧就往前滑动一格。

接收窗口中是可以接受帧的序号，每收到一帧就往前滑动一格。

1. 停止等待协议

当发送方和接收方的窗口大小都为1的时候，滑动窗口协议退化为停止等待协议

[参考链接](http://t.zoukankan.com/wangzheming35-p-14510819.html)

1. **流量控制和拥塞控制的区别**

流量控制和拥塞控制都是为了让接收方能够接收每个帧，减少丢包

但是，流量控制是控制发送方发送帧的速率。而拥塞控制是控制网络中数据负载的大小。前者通过滑动窗口实现，后者通过拥塞控制实现

1. **网卡**

每个网卡有唯一的MAC地址。它工作在物理层和数据链路层，是局域网和计算机的接口。

1. 它具备串行、并行转换的功能

网卡和局域网串行连接，而网卡和计算机通过I/O总线并行连接。因此它会进行串行、并行转换

1. 编码、解码功能

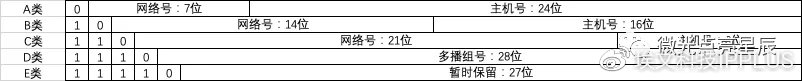
对发送数据进行编码，对接收数据进行解码

1. **MAC地址（物理地址） IP地址**

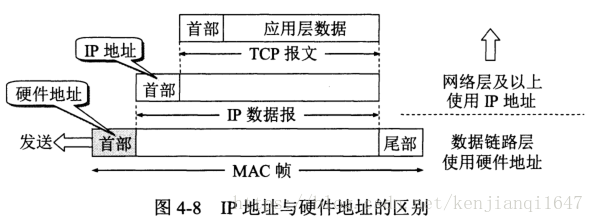
MAC地址（物理层）位于数据链路层，每个主机是唯一的。

IP地址位于网络层，包括网络号、主机号。在不同的局域网中，其IP地址不同。

（同一个物理网络上的所有主机都使用同一个网络ID，网络上的一个主机(包括网络上工作站，服务器和路由器等) 有一个主机ID与其对应）



A:大型网络，B：中型网络，C:小型



[参考地址](https://www.zhihu.com/question/49335649/answer/2053706013)

1. **公有IP和私有IP**

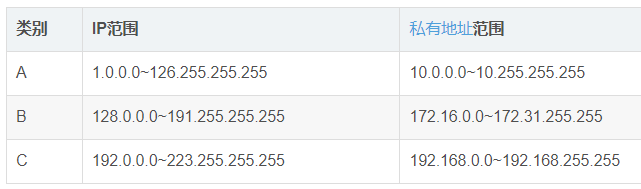
公有IP (百度查):连接以太网的地址，联通或电信分配给的（不是全网唯一的）

私有IP (ifconfig):局域网地址，不能联通外网

公网：互联网

内网（私网）：局域网

私有IP的功能：多用于公司内部，私有路由不能对外直接提供信息，内部网络不会直接收到Internet上一些攻击。但是，也无法以私有IP直接上网



[参考链接](https://blog.csdn.net/insis_mo/article/details/110187397)

1. **DNS服务器**

DNS将域名(www.baidu.com)解析为IP地址(198.xxx.xxx.xxx)

域名：

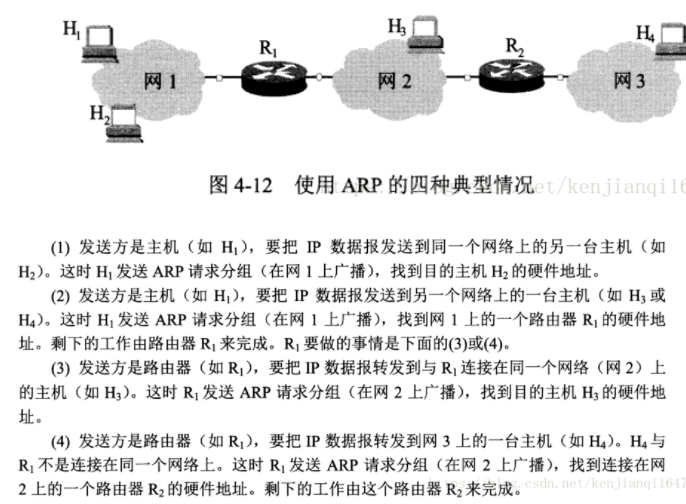


1. **DNS域名解析过程**
2. 主机向本地DNS服务器查询，查到此记录，利用此记录解析。未查到该记录，向根域名服务器查询
3. 根域名服务器告诉本地域名服务器，下一级应查询的域名服务器IP。然后递归，在有目标域名的DNS服务器上找到相应的IP地址信息（根域名->二级域名->三级域名）
4. 客户机的本地DNS服务器将递归查询结果返回给客户机，客户机利用查询到的IP访问目标主机
5. **地址解析协议ARP**

主机APR cache存放了本局域网上各主机和路由器的IP地址到MAC地址的映射表，并动态维护这个表。

在同一局域网中，主机A向主机B发送数据，首先在ARP cache中查看有无B的IP地址对应的MAC地址，并将其写进MAC帧，然后通过局域网把该MAC帧发送至该硬件地址

1. **在/不在同一局域网上数据传输**



连接在通信链路中的设备（主机或路由器）收到MAC帧时，根据MAC帧首部的硬件地址决定收下或丢弃。只有在剥掉MAC帧的首部和尾部后，把数据上传给网络层，才能在IP数据报的首部中找到源IP地址和目的IP地址

[参考链接](https://blog.csdn.net/kenjianqi1647/article/details/81636898?ops_request_misc=&request_id=&biz_id=102&utm_term=MAC%E5%9C%B0%E5%9D%80%20%E7%89%A9%E7%90%86%E5%9C%B0%E5%9D%80%20IP%E5%9C%B0%E5%9D%80&utm_medium=distribute.pc_search_result.none-task-blog-2~all~s)

1. **两个服务器之间网络已经联通，却收不到彼此的UDP报文原因**

UDP并非可靠传输，没有流量控制、拥塞控制等机制，可能发生了丢包

1. **一次HTTP请求过程（打开一个网址）**
2. DNS域名解析(域名www.baidu.com -> ip)
3. 通过IP，使用ARP地址解析协议，找到对应服务器，发起TCP三次握手
4. 建立TCP连接后，发起http请求
5. 服务器响应http，返回response
6. 浏览器解析response，请求其他资源（如js、css、图片等）
7. 浏览器渲染页面

[参考链接](https://blog.csdn.net/xupeiyan/article/details/122636831)

1. **HTTP和HTTPS协议**

HTTP协议（超文本传输协议）用于在Web浏览器和网站服务器之间传递信息。HTTP协议以明文发送内容，不提供任何形式的数据加密。如果攻击者截取了Web浏览器和网站服务器之间的传输报文，就可以直接读懂其中的信息。因此，HTTP协议不适合传输一些敏感信息，如信用卡号、密码等

HTTPS协议（安全套接字超文本传输协议）为了保证数据传输安全，在HTTP基础上加入SSL协议。SSL依靠证书验证服务器的身份，并为浏览器和服务器之间的通信加密。

[参考链接](https://www.cnblogs.com/wqhwe/p/5407468.html)

1. **HTTP常见状态码**

200 - 服务器成功返回网页

404 - 请求的页面不存在

503 - 服务器超时

[参考链接](https://blog.csdn.net/whl826661099/article/details/98606745)