# 计算机网络

## TIPS

电路交换 这跟网线 如果被两台机器用来通信了,其他信息就无法通过这跟网线来传递了,就得等这两台机器的通信断了才能用这跟网线.

分组交换 就是把数据切成一个个小的数据包,包里面标记上要发给谁,然后通过这个网线发送,这样的话很多对机器就可以同时使用这跟网线.

中继器 物理层的路由器 网桥 数据链路层的路由器

交换机 数据链路层的路由器

路由器 ip层的延长网络的设备

以太网首部->ip包首部->tcp包首部->数据

在一个局域网(以太网)内,用mac就可以互相发数据,但是一旦要跨局域网mac就不行了,就得借助于ip了!!!

ARP(网络层协议)通过ip找mac

Mac地址是下一跳的地址老换

Ip地址是指示着我们前进的最终地址

一个网卡只有一个mac,却可以有多个ip

所以说ping命令就是icmp协议,ip也是网络层的,但是这两个协议之前肯定会有一个arp的流程,获取一跳跳的mac地址.

IGMP协议

组管理协议 就是ip多播(组播)的组管理的,也是网络层的协议.

信令就用tcp,音视频啊数据包啊就用udp

Tcp的可靠性的保证 玩的就是序列号和ack机制

这样就会有超时重传机制.

三次握手

Client 我要连你了 协商点东西吧

Server 就这些信息 看看行不

Client 可以可以 明白了 通知你下

Client xxxx

Xxx

Xxx

Client发完第三次握手后就开始发数据了.

主机地址的那些位都是1的就是广播地址.

172开头的是B类地址,192开头的是C类地址

由于链路层有路径MTU的存在,所以ip层会存在分片.

v4 4个8比特 v6 8个16比特

滑动窗口(可以不收到ack还继续发送数据)

快速重传(收到了a4的ack,但是没有收到a3的ack,说明a3包丢了,我们马上重传a3的包)

由于滑动窗口 发送端要有个缓存区 缓存那些发了但是还没收到ack的消息

窗口大小动态变化

拥塞控制 分为慢启动(一开始发送包的速度是1248指数增长)和拥塞避免(到达一定数量后就由指数变成线性)

快速恢复就是一旦发现开始丢包了,把速率直接砍一半,然后不用慢启动了,直接用拥塞避免这种线性的.

http无状态无上下文,之所以做了这么简单是为了更大的并发量,如果想保存些上下文就需要用到cookie和session.

http长连接可以避免大量的tcp连接创建断开问题.

所谓通信中的隧道就是通过加密啊或者加头啊,封装一层外面看不懂的东西,对外面就是透明的就是看不懂的字节流,那么两个节点的通信就像是有一个隧道一样.

URI就是唯一资源标识符,URL是URI的一种实现方式.

## HTTPS

非对称加密 私钥加密 公钥可以解密 公钥加密 私钥也可以解密。

一般两端需要交互，都是先把自己的公钥给对方（公钥可以给很多人），私钥只有自己有，别人拿到了你的公钥，给你发消息时就会用公钥加密发给你，加密后的消息别人截获了也没有，别人没有你的私钥也解不了。

一般都是服务器把公钥发给用户，私钥服务端自己保存，用户收到服务器的公钥后把自己的公钥用服务器的公钥加密，发给服务器，服务器用自己的私钥解开后就拿到了用户的公钥，这样两端都保存了对方的公钥，发的消息都用对方的公钥加密了，通信就是安全的了。

问题就是第一步服务器把自己的公钥发给用户的时候还是不安全的。

解决方案就是服务端把自己的公钥告诉一个第三方机构 CA机构，CA把自己的信息啥的和服务器的公钥用自己的私钥加密成一个证书发给服务器，服务器把这个证书发给客户端，因为浏览器维护了大部分CA机构的名称地址和公钥信息，就可以验证这个证书是不是合法的对的，如果是真的，就用对应的CA机构的公钥解密这个证书拿到服务器的公钥就可以继续后面的流程了。

## 三次握手与四次挥手

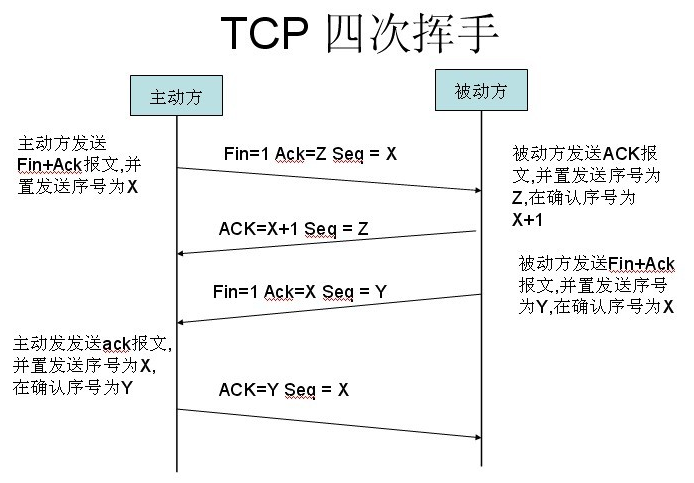


SYN叫做同步位,ACK叫做响应位,这俩大写的,每次都是1,是不会变的.

seq小写的这个才是序列号,注意client和server都会维护自己的seq序列号的,client和server也都是会给对面回复ack(小写)的时候ack=对面seq+1的.

为啥要三次握手呢,tcp连接是全双工的,也就是A给B发消息的同时B也能给A发消息.

那么A给B发完消息后,需要收到B的ack才能确定A->B的通路打通,如果只有两次握手的话client->server的通路可以确定通了,但是server->client的还没有通,因为server给client发的消息(第二次握手)没有收到ack,所以要有第三次握手.



四次挥手两边谁都可以触发,顺序并不一定是上面那样,也就是谁确定自己不给对面发消息了,就给对面发个FIN,但是对面可能还给你发,所以在四次挥手的时候会有个半关闭的概念.

四次挥手就是你给我发FIN,我给你ack,我给你发FIN,你给我回ACK.

这是因为服务端的LISTEN状态下的SOCKET当收到SYN报文的连接请求后，它可以把ACK和SYN(ACK起应答作用，而SYN起同步作用)放在一个报文里来发送。但关闭连接时，当收到对方的FIN报文通知时，它仅仅表示对方没有数据发送给你了；但未必你所有的数据都全部发送给对方了，所以你可能未必会马上会关闭SOCKET,也即你可能还需要发送一些数据给对方之后，再发送FIN报文给对方来表示你同意现在可以关闭连接了，所以它这里的ACK报文和FIN报文多数情况下都是分开发送的。