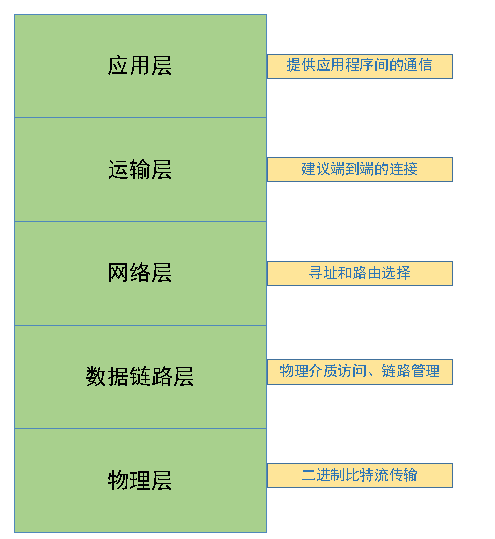
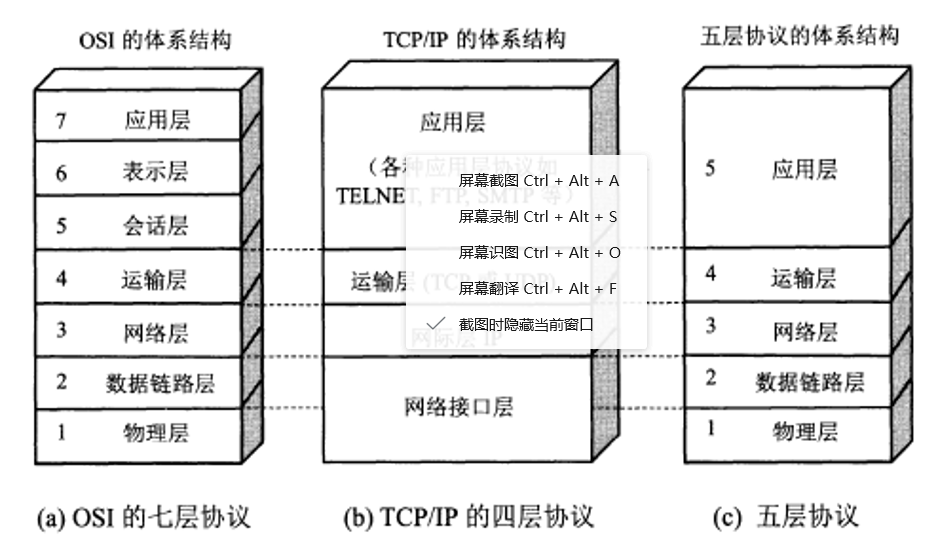
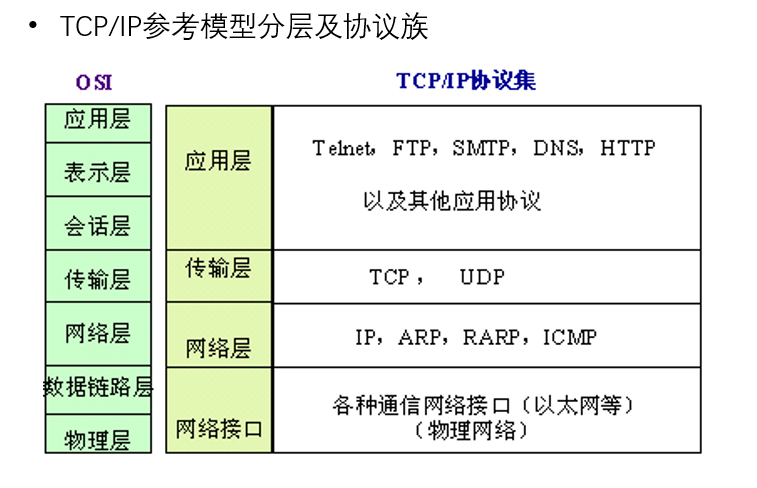
# 计算机网络各层划分和作用



分层的作用

* 通信双方功能的对等实现
* 功能模块化、清晰化
* 利于各种网络技术的层叠
* 利于异构网络的互通

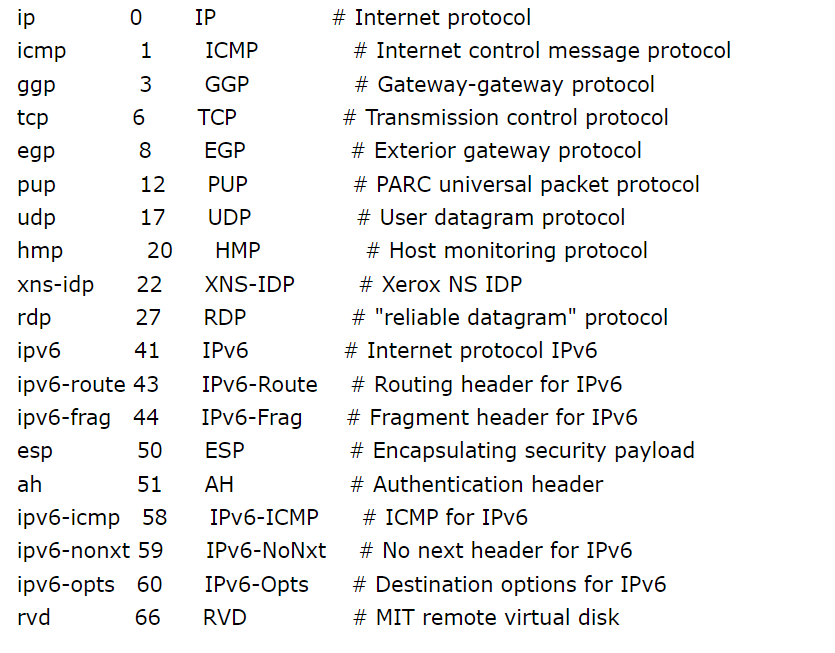
各层的协议：



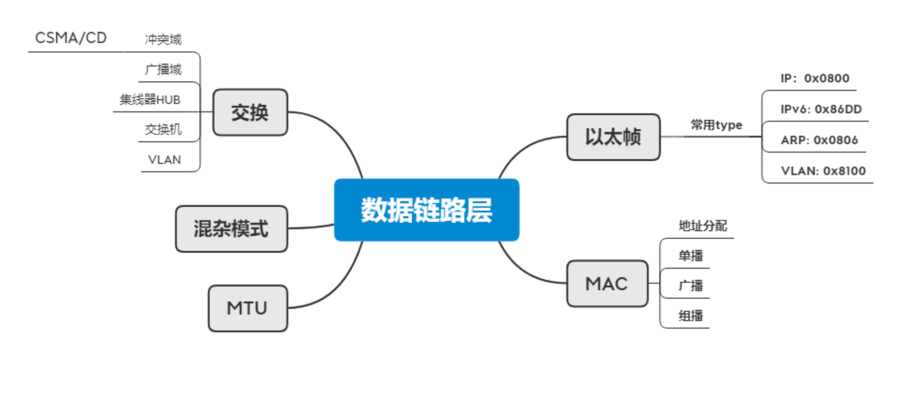
多个用户使用一个IO资源 **发送消息** 时，我们称之为“**复用**”。

多个用户使用一个IO资源 **接收消息** 时，我们称之为 “**分用**” 。

补充信息：



思维导图，常用type：



单工：永远只能接受或发送

半双工：任一时刻只能接受或发送

全双工：同时双向

CSMA/CD：发前先听，边发边听，冲突停止，强化碰撞，随机延时后重发

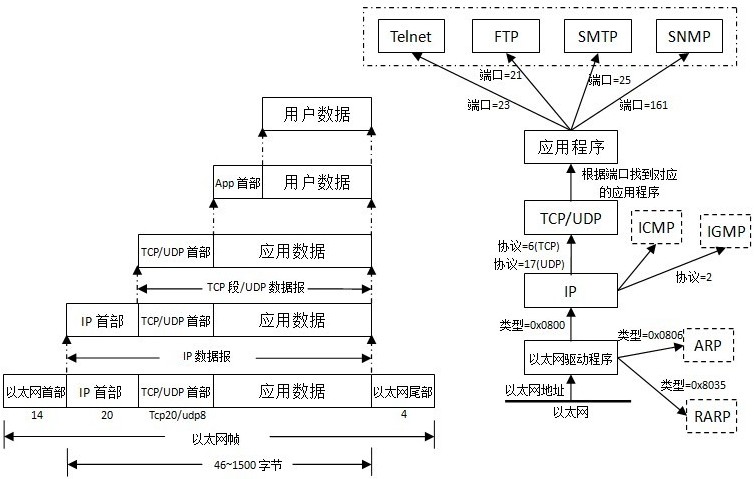
* 争用期2t：

共享网络中两个最远节点单程端到端的信号传播时延位t

* 冲突检测原则:

两个叠加后，会使线路上电压的摆动值超过正常值一倍，据此可以判断冲突的产生。

3、封装过程



3）应用层/运输层提供了端到端的服务（End to End）

网络层提供了点到点的服务(Hop by Hop)

* 交换机（switch）用于提供终端之间互相通信访问的功能。
* 路由器（Router）又称网关设备（Gateway）用于连接多个不同的网络，并提供上网功能。
* 防火墙（Firewall）是指一种将内网和公网（如Internet）分开的设备，它能允许你“同意”的人和数据进入你的网络，同时将你“不同意”的人和数据拒之门外。
* 服务器（Server）指一个管理资源并为用户提供服务的计算机。如邮件服务器，ftp服务器等
* 存储设备（Storage）是用于储存信息的设备，从某种意义上来说，可以看作类似于电脑终端设备的U盘或移动硬盘。

虚拟化技术的关键：Hypervisor（VMM）：抽象Host的硬件资源，提供给Guest使用

虚拟化的优势：

1. 提高资源利用率
2. 减少物理机数量，降低硬件成本和维护成本
3. 部署快速灵活，便于管理
4. 提升可靠性

# 常用抓包工具：

|  |  |
| --- | --- |
| **常用抓包工具** | **特点** |
| **Wireshark** | **通用抓包工具，简单易用，开源免费，具有强大的分析能力** |
| **Tcpdump** | **Linux，抓包功能强大，命令行工具，保存的文件是二进制** |
| Windump | Windows的Tcpdump工具 |
| Tshark | Wireshark的Linux命令行工具 |
| Sniffer | 支持协议解析；网络活动监视；专家分析系统（Expert）。Sniffer Portable具有超强的专家分析能力，价格昂贵。 |
| Fidder | Http代理，它能记录所有客户端和服务器的Http和Https请求响应 |
| Charles | 设置代理抓取http包，MAC上对Charles支持更好 |
| OmniPeek | 更适用于无线网络、语音等技术 |
| Firebug | Web调试工具，集成在Internet Explorer工具栏 |
| HttpWatch | Web调试工具，网页浏览器 Mozilla Firefox下的一款开发类插件 |
| ...... |  |

Tcpdump:

* 下载到本地使用wireshark界面网络分析工具进行网络包分析
* tcpdump -r的命令去读取

如果要使用tcpdump抓取其他主机MAC地址的数据包，必须开启网卡混杂模式，所谓混杂模式，用最简单的语言就是让网卡抓取任何经过它的数据包，不管这个数据包是不是发给它或者是它发出的。一般而言，不会让普通用户设置混杂模式，因为这样可以看到别人的信息，比如telnet的用户名和密码，这样会引起一些安全上的问题，所以只有root用户可以开启混杂模式

root用户

* ifconfig 可以看到网卡是否混杂模式，如：
* ifconfig eth0 看是否有“PROMISC”标志
* ifconfig eth0 promisc #设置混杂模式
* ifconfig eth0 -promisc #取消混杂模式

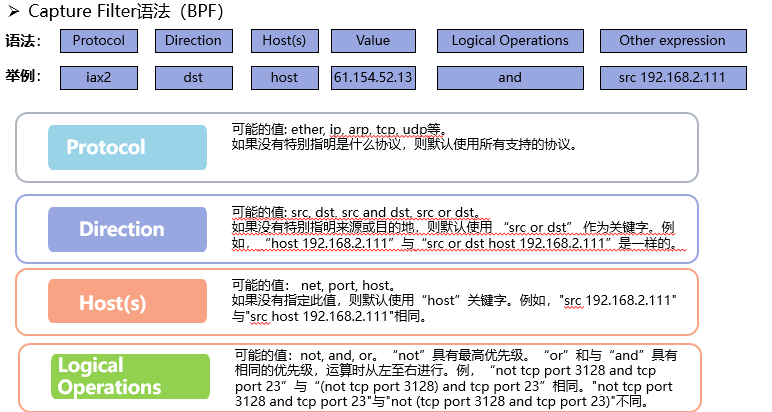
TCPDUMP命令：

* Tcpdump –help:帮助
* Man tcpdump：manual手册
* Capture Filter语法（BPF）、
* Tcpdump -nn -I eth0 tcp and host 192.168.1.163 and port 80

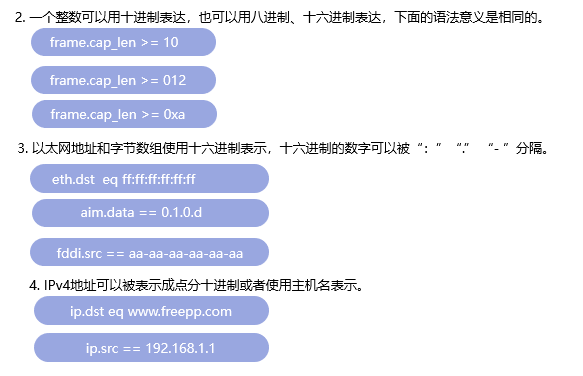
注意-nn的含义：第一个n表示以IP地址的方式显示主机名，第二个n是以端口数字的形式代替服务名

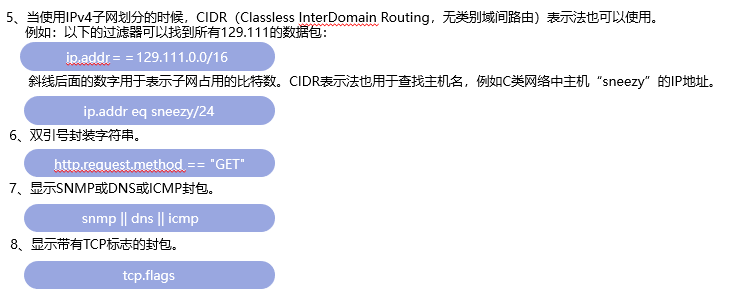
Tcpdump -nn -vs0 -I eth0 -c 100 -w 1.pcap

###-vs0 表示不限制包的大小,-c表示数量，-w表示输出到什么文件 -vv表示输出详细信息



Wireshark命令：





密钥交换算法目前常用的有RSA和Diffie-Hellman。

（1）对于密钥交换使用RSA算法，pre-master-secret由客户端生成，并使用公钥加密传输给服务器。

（2）对于密钥交换使用Diffie-Hellman算法，pre-master-secret则通过在Key Exchange阶段交换的信息，由各自计算出pre-master-secret。所以pre-master-secret没有存到硬盘，也没有在网络上传输，wireshark就无法获取session key，也就无法解密应用数据

And == &&

Or == ||

注意大小写

CIDR（Classless InterDomain Routing，无类别域间路由）表示法也可以使用

192.168.1.2/24

**物理层**

1. 透明传输，屏蔽具体传输介质和物理设备的差异
2. 向数据链路层提供服务
3. 协议数据单元：比特
4. 传输媒体：双绞线，光纤，无线信道

# 数据链路层

* 接收来自物理层的比特流形式的数据，合成以太帧，解析后数据上送网络层；
* 服务网络层，封装数据帧，向下层（物理层）访问介质并以比特流传输数据
* 链路管理和差错控制，提供可靠数据帧传输
* 协议数据单元：帧（最小64字节）

1、网卡

1）标准模式

一般情况下，我们知道网卡往往只会接受目的地址是他的数据包而不会接受目的地址不是的它的数据包，所以我们应该知道网卡只会接受我们该接收的包而不会接受其他的地址的网络数据包。

 2）混杂模式

混杂模式是指一台主机能够接受所有经过它的数据流，不论这个数据流的目的地址是不是它，它都会接受这个数据包。也就是说，混杂模式下，网卡会把所有的发往它的包全部都接收。在这种情况下，可以接收同一集线器局域网的所有数据。

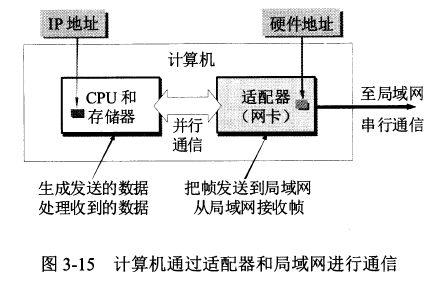
网络适配器(即有硬件，也包括软件)实现这些协议。包括操作系统中的设备驱动程序、计算机中对应的网络接口卡

MAC地址固化在适配器的ROM中

IP地址不在适配器

驱动程序是一种可以使计算机和设备通信的特殊程序，可以说相当于硬件的接口，操作系统只能通过这个接口，才能控制硬件设备的工作。网卡驱动程序就是电脑进行上网的驱动工具，和其它硬件驱动程序一样，都要安装驱动程序，没有驱动程序的硬件是运行不起的，让网卡发挥作用就要有网卡驱动。

网卡并不是独立的自治单元，因为网卡本身不带电源而是必须使用所插入的计算机的电源，并受该计算机的控制。因此网卡可看成为一个半自治的单元。当网卡收到一个有差错的帧时，它就将这个帧丢弃而不必通知它所插入的计算机。当网卡收到一个正确的帧时，它就使用中断来通知该计算机并交付给协议栈中的网络层。当计算机要发送一个IP数据包时，它就由协议栈向下交给网卡组装成帧后发送到局域网。

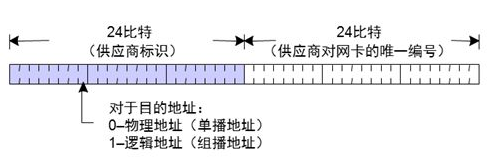


2、MAC地址

* 通过MAC地址在以太网二层链路上唯一标识网络设备，实现局域网设备间的通信
* 48位
* 前24比特是组织唯一标识符（OUI, Organizationally unique identifier）IEEE分配，我司：00:1C:54
* 后24比特由厂家自行分配

1）单播，广播，组播

* 第一字节最低位，为1，表示组播
* 第一字节最低位，为0，表示单播
* FF-FF-FF-FF-FF-FF，表示广播



冲突域和广播域：

传统集线器连接的以太网是冲突域

交换机连接的是广播域

VLAN和路由器可以隔离广播域

* Type：
  + IP： 0x0800
  + IPv6: 0x86DD
  + VLAN: 0x8100
  + ARP: 0x0806

常见的广播通信：

(1)ARP请求：建立IP地址和MAC地址的映射关系。

(2)RIP：一种路由协议。

(3)DHCP：用于自动设定IP地址的协议。

(4)NetBEUI：Windows下使用的网络协议。

(5)IPX：NovellNetware使用的网络协议。

(6)Apple Talk：苹果公司的Macintosh计算机使用的网络协议。

如果整个网络只有一个广播域，那么一旦发出广播信息，就会传遍整个网络，并且对网络中的主机带来额外的负担。因此，在设计LAN时，需要注意如何才能有效地分割广播域。

路由器不会转发广播帧

* 以太帧承载的上层数据长度范围：46~MTU（最大传输单元，一般：1500）
* 封装以太帧的长度范围：64~MTU+18（一般：1518）
* bps vs. pps
  + bps = bits per seconds
  + pps = packets per seconds
* pps = bps/(k×8＋8×8＋96)
  + k - 数据帧的长度 （bytes， 包括CRC）
  + 8×8 - 前导码的长度，
  + 96 - 每两个帧之间存在帧间隙，相当于发送96bit所需时间

配置vlan

命令行：

Sys

Display port vlan

Display vlan

Vlan 10

q

Vlan 20

Q

Display vlan

配置trunk

Interface G\_xxxx 0/0/x

Port link-type trunk

Port trunk allow-pass vlan 1 to 100

Display port vlan

Q

配置access

Interface G\_xxxx 0/0/x

Port link-type access

Port default vlan 10

q

Display port vlan

环路的危害：

* 广播风暴：

由于交换机会对广播、多播、和未知MAC的单播包进行泛洪，在存在环路的情况下，很短的时间内就会产生风暴

* MAC地址表震荡：

当交换机刚刚启动时，MAC地址表是空的，所以，所有的单播帧都会进行泛洪操作。但是如果存在环路的话，交换机在特定情况下，会从不同的接口收到相同的MAC地址，这样的话，MAC地址表将不稳定

* 二层防环：

STP生成树协议

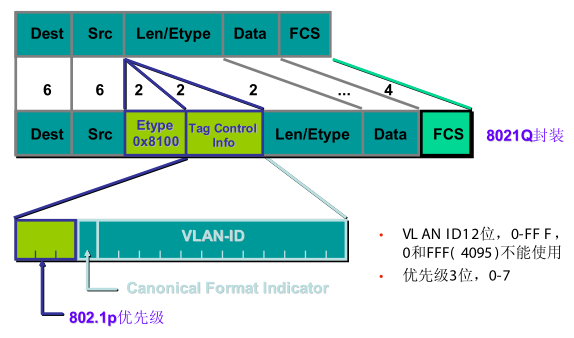
* + 使得从一台主机到所有其他主机的路径都是无环路的树状结构，从而消除兜圈子的现象
  + 原理：不改变网络的实际拓扑，通过逻辑的禁用接口，使得环路在逻辑上不存在；当线路出现故障时，将禁用的接口启用，使得网络能够发挥物理冗余路径带来的高可用性

Vlan划分方式：

* 基于端口划分：这种方法明确指定各端口属于哪个VLAN
* 基于MAC地址的划分：根据主机网卡的MAC地址进行划分
* 基于网络协议的划分：可以划分为IP/IPX/DECnet/ AppleTalk/Banyan等VLAN网络
* 基于IP地址划分：将任何属于同一IP广播组的主机认为属于同一VLAN
* 基于策略的划分：一种根据不同的情况，将多种划分VLAN的技术按照一定的安全策略进行综合运用的划分技术
* 基于用户的划分：用户识别信息，一般是计算机操作系统登录的用户

VLAN，遵循IEEE802.1Q标准，它在原来的以太网帧源MAC字段的后面加入了4个字节的VLAN Tag；

* Type：取固定值0x8100，用于以太帧中标志VLAN Tag；
* User Priority：用户优先级，用于流量的等级划分；
* Flag：该bit在以太网中固定取0；
* VLAN-ID：取值为0～4095，用于标志不同VLAN（0用于识别帧优先级，4095作为预留值）



Vlan连接方式：

* Access连接：

报文不带tag标签,一般用于和tag-unaware(不支持802.1Q封装)设备相连，或者不需要区分不同VLAN成员时使用；

* Trunk连接：

在PVID所属的VLAN不带tag标签转发,其他VLAN中的报文都必须带tag标签

用于tag-aware(支持802.1Q封装)设备相连，一般用于交换机之间的互连；

* Hybrid连接：

可根据需要设置某些VLAN报文带tag,某些报文不带tag。

trunk连接的不同在于：trunk连接只有PVID所属的VLAN不带tag,其他VLAN都必须带tag, 而Hybrid连接是可以设置多个VLAN不带tag

* 报文入方向：

交换机的决定该报文是否允许进入该端口，根据入报文的tag/untag的属性以及端口属性：

* + 报文为untag：允许报文进入该端口，并打上PVID的VLAN tag,与端口属性无关；
  + 报文为tag：在这种情况下，需要交换机来判断是否允许该报文进入端口：
    - Access端口：PVID和报文中tag标明的VLAN一致，接收并处理报文；否则丢弃。
    - Trunk/Hybrid端口：如果端口允许tag中标明的VLAN 通过，则接收并处理报文；否则丢弃。
* 报文出方向：

交换机完成对报文的转发，决定在转发出端口时是否携带tag，依据端口属性

* + Access端口：将标签剥掉，不带tag转发；
  + Trunk端口：报文所在VLAN和PVID相同，则报文不带tag；否则带tag；
  + Hybrid端口：报文所在VLAN配置为tag，则报文带tag；否则不带tag

Access 类型的端口：只能属于1 个VLAN，一般用于连接计算机的端口；

      Trunk 类型的端口：可以允许多个VLAN 通过，可以接收和发送多个VLAN 的报文，一般用于交换机之间连接的端口；

Hybrid是H3C交换机的一种端口模式，这个接口也能够允许多个VLAN帧通过并且还可以指定哪些VLAN数据帧被剥离标签，主要实现高隔离度的波分和复用

Access 端口可接受并转发的数据来源：来自PC的无VLAN信息数据包；从一个Access口入打上VLAN标记在交换机内转交给相同VLAN的access口去掉标记的无VLAN信息数据包；

           Access 端口发送出去的数据包无VLAN信息，可被PC接受或Access口接受。

 收到一个报文,判断是否有VLAN 信息：如果没有则打上端口的PVID，并进行交换转发,如果有则直接丢弃（缺省）

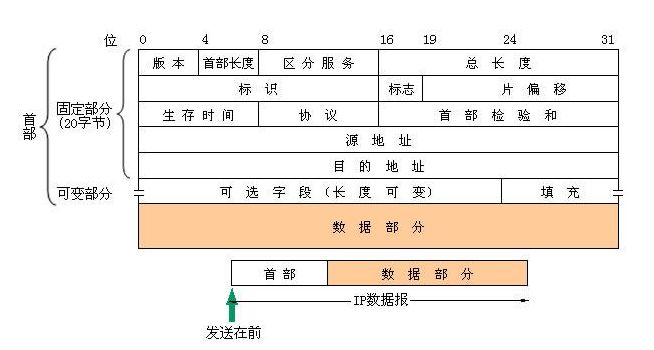
           （Access接受并转发的数据包必须是无VLAN信息的数据包）

           2. Acess 端口发报文：

           将报文的VLAN 信息剥离，直接发送出去

           （Access发送的数据报必定是无VLAN信息的数据包）

# IP



* + 地址管理
  + 路由选路

首部长度最长为60字节

版本号：0100表示IPv4，0110表示IPv6

首部长度：标识包括选项在内的IP头部字段的长度

服务类型：3bit的优先级字段和4bit TOS字段，最后一位置为0

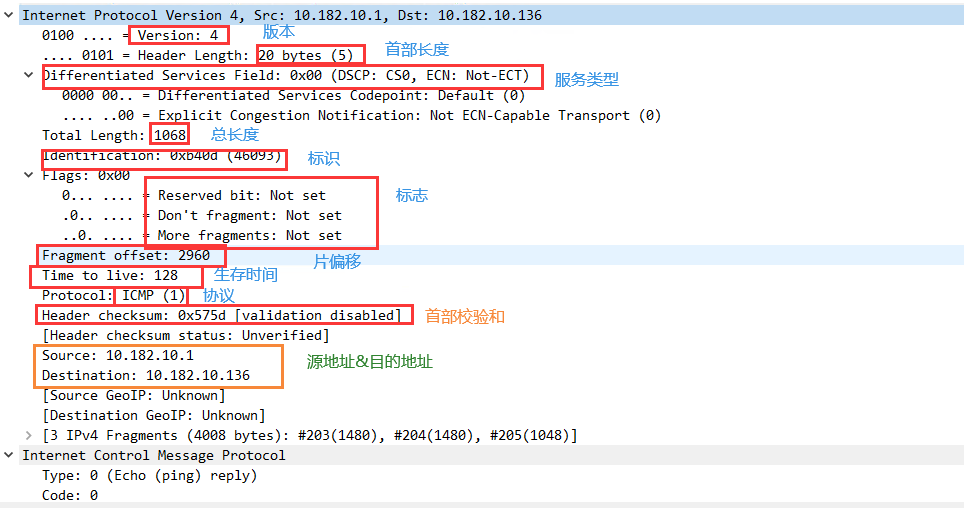
总长度：首部+数据，最长可达65535字节

标识：唯一的标识主机发送的每一份数据报。根据标识分片重组

标志：MF，DF，DF=0，才能分片，MF=0，表示最后一片

片偏移：8字节整数倍，表示距离数据段起点的位置

生存时间：TTL、跳数，8位，最大值255。每经过一个路由器-1，0，丢弃。TTL=1，本LAN传送

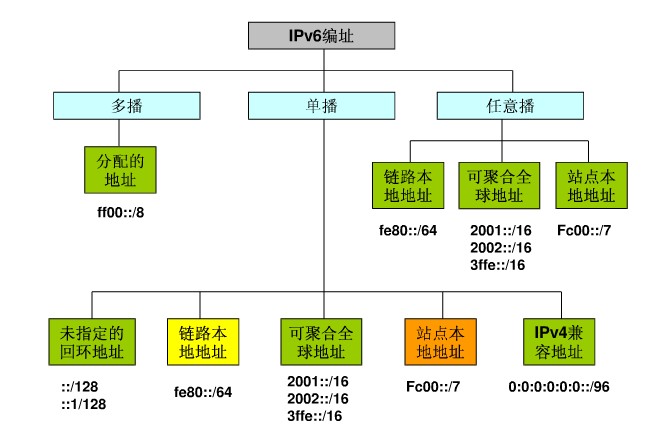


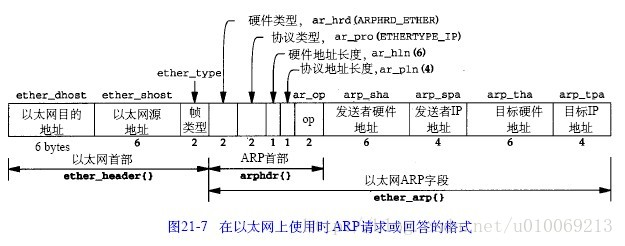
* **0.0.0.0**
* **127.0.0.1（环回地址）**
* **255.255.255.255（受限广播地址）**
* **私有地址**

A类地址的私有地址是：10.0.0.0~10.255.255.255

B类地址的私有地址是：172.16.0.0~172.31.255.255

C类地址的私有地址是：192.168.0.0~192.168.255.255





# ARP协议字段

Hardware type ：硬件类型，标识链路层协议

Protocol type： 协议类型，标识网络层协议

Hardware size ：硬件地址大小，标识MAC地址长度，这里是6个字节（48bti）

Protocol size： 协议地址大小，标识IP地址长度，这里是4个字节（32bit）

Opcode： 操作代码，标识ARP数据包类型，1表示请求，2表示回应

Sender MAC address ：发送者MAC

Sender IP address ：发送者IP

Target MAC address ：目标MAC，此处全0表示在请求

Target IP address： 目标IP

免费ARP：发送者IP和目标IP相同

* 应用场景：
  + 地址冲突检测
  + IP上线，更换IP，接口up/down
  + MAC变化，更换物理网卡，HA主备切换

Shutdown

Undo shutdown

抓取ARP报文



IPV6:







40字节

节点：所有运行IPv6应用的设备

路由器：能够转发IPv6数据包的节点

主机：无法转发IPv6数据包的节点

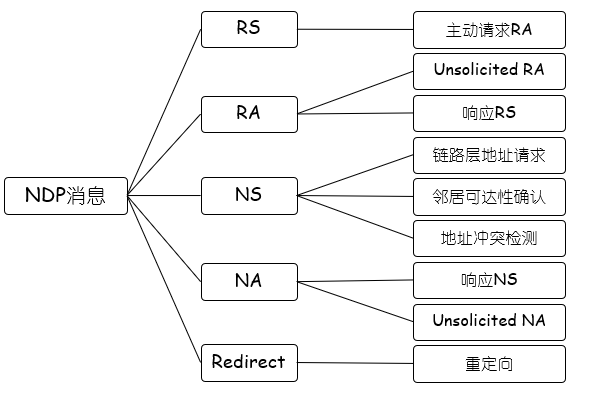
链路：由路由器分隔开的，使用相同

接口：与物理或逻辑链路相连的附件

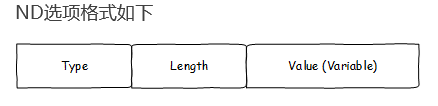
邻居：同一个链路相连的节点

|  |  |  |
| --- | --- | --- |
| 地址类型 | IPv4 | IPv6 |
| 未指定地址 | 0.0.0.0/32 | ::/128 |
| 环回地址 | 127.0.0.1/32 | ::1/128 |
| 本地链路地址 | 169.254.0.0/16 | fe80::/10 |
| 私有地址 | 10.0.0.0/8  172.16.0.0/12  192.168.0.0/16 | fc00::/7 |
| 公有地址 | 除私有外的地址 | 全球单播地址 |
| 组播地址 | 224.0.0.0/4 | FF00::/8 |

* NDP是基于ICMPv6协议的邻居发现机制。ND取代了IPv4中的ARP协议、ICMPv4路由器发现协议、以及ICMPv4重定向消息，同时提供了更丰富的其它功能，主要表现为
  + 路由器发现
  + 前缀发现
  + 参数发现
  + 地址自动配置
  + 地址解析
  + 下一跳确定
  + 邻居不可达检测
  + 重复地址检测
  + 链路层地址通告
  + 重定向



* ND消息中可包含0或多个可选项，可选项也可以多次出现在同一个ND消息中



* RFC4861中定义了5中可选项
  + SLLA：type=1，源链路层地址
  + TLLA：type=2，目标链路层地址
  + Prefix Information：type=3
  + Redirect Header：type=4
  + MTU：type=5
* RS & Solicited RA
  + 节点想要主动更新路由信息或自动配置全球地址时（如接口刚启用），需要让Router马上发出RA消息，可以通过发送RS来实现
  + 路由器收到有效的RS时，发送RA消息来回应。通过RA消息，可实现无状态地址自动配置
* Unsolicited RA
  + 路由器通过周期性的RA消息来通告自己的存在，以及各种链路参数和网络参数
  + RA消息格式与Solicited RA一致
* NS消息可以用于
  + 链路层地址解析：当节点需要知道目标节点的链路层地址时，需要发送NS消息；同时也会把自己的链路层地址提供给目标节点
  + 重复地址检测：当节点进行地址配置时，需要检测该地址是否已经被链路中的其它接口使用
  + 邻居可达性检测：当节点需要检测一个已知链路层地址的邻居是否可达时，发送NS消息
* NA消息可以用于
  + 答复NS：节点收到NS，Target Address与自己一致时，发送NA进行答复
  + 未请求的NA：节点更新I链路层地址时，为了快速传播新的信息，主动广播NA来通告新地址
* Redirect重定向消息由路由器发出，用于通知主机
  + 到达目的地址的路径上存在更好的下一跳路由器
  + 目的地是邻居（ICMP Target Address与ICMP Destination Address一致时）

|  |  |  |
| --- | --- | --- |
| ND消息 | ND消息作用 | 可包含的ND选项 |
| RS | 主机发送RS，请求路由器立即通告RA | SLLA：用于把主机的链路层地址告知路由器 |
| RA | 路由器响应RS；或周期性发送RA来通告他们的存在以及各种链路参数和网络参数 | SLLA：用于把路由器的链路层地址通告给接收主机； Prefix Info：用于把链路中的前缀信息告知接收主机，并告知主机是否自动配置无状态地址； MTU：用于把链路的IPv6 Link-MTU告知接收主机 |
| NS | 邻居链路层地址解析；邻居不可达检测；地址冲突检测 | SLLA：用于把发送方的链路层地址告知接收节点 |
| NA | 响应NS；或主动发送NA来通告链路层地址 | TLLA：用于把目标地址对应的链路层地址告知接收节点 |
| Redirect | 通知主机，有更好的下一跳节点 | TLLA：用于把目标地址对应的链路层地址告知接收节点； Redirect Header：用于包含重定向报文的部分或全部 |

邻居缓存状态：

* + Neighbor Cache包含由NUD算法维护的信息，关键信息是邻居的可达状态，包含如下5种：
    - * Incomplete：地址解析正在进行中（NS已发送，还未收到相应NA），邻居的链路层地址还未确定。
      * Reachable：目前地址已经被确认为可到达的状态（如几十秒前是可达的）。发送地址解析NS后，在规定时间内收到NA确认。
      * Stale：在收到最新的有关目的可达性的信息后，已经经历了REACHABLE\_TIME指定时间。
      * Delay：当数据包发送到处于“Stale”状态的节点时，该条目就进入Delay状态。
      * Probe：由Delay状态转换而来，并且以RetransTimer重复发送单播NS消息以便验证邻居可达性



* + 自动配置
    - * 链路本地地址

链路本地前缀 + 接口ID

链路本地前缀 + 本地随机生成的接口ID

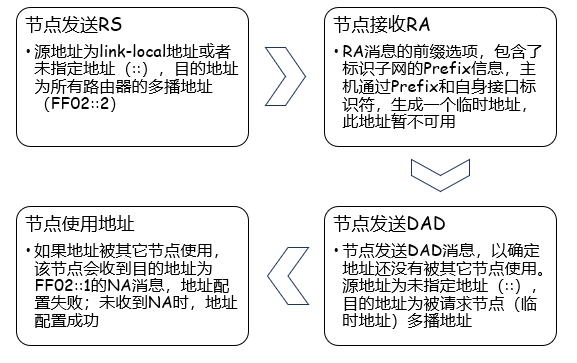
* + - * 全球地址

路由器通告前缀 + 接口ID

路由器通告前缀 + 本地随机生成的接口ID

DHCPv6分配

* + - * 需要更严密控制精确地址分配时使用，可以和自动配置同时使用
      * 手动配置
      * 大多数路由器通过手动配置地址（IPv6前缀+主机ID）
      * 地址自动配置失败时，也可手动配置



地址状态：

* + Tentative Address
    - * 临时地址。地址还未进行冲突检测，仅用于NDP
  + Preffered Address:
    - * 优先地址。任意通信中可不受限制地使用该地址。由Preffered Lifetime决定
  + Deprecated Address:
    - * 过时地址。不能用于初始化新连接，可用于已经建立的连接。由Valid Lifetime决定，Valid Lifetime必须大于等于Preffered Lifetime
  + Invalid Address
    - * 不合法地址。无法使用



# DHCP和自动配置

1. 产生的原因和解决的问题

* DHCP的优点

1. 可以集中管理网络中的IP地址，减少管理员的工作量；
2. 避免用户手工配置网络参数时造成的地址冲突；
3. 通过IP地址租期管理，提高IP地址使用效率；

* 缺点

1. IP地址随机性，用户的IP地址是随机分配的，具有不确定性，用户访问难度较大；
2. 访问权限变化，如果用户访问权限是基于IP地址划分的，那么访问权限跟随IP地址不同而变化；
3. 安全性差：外来的网络用户会对网络本身造成安全威胁；
4. 工作原理和交互流程



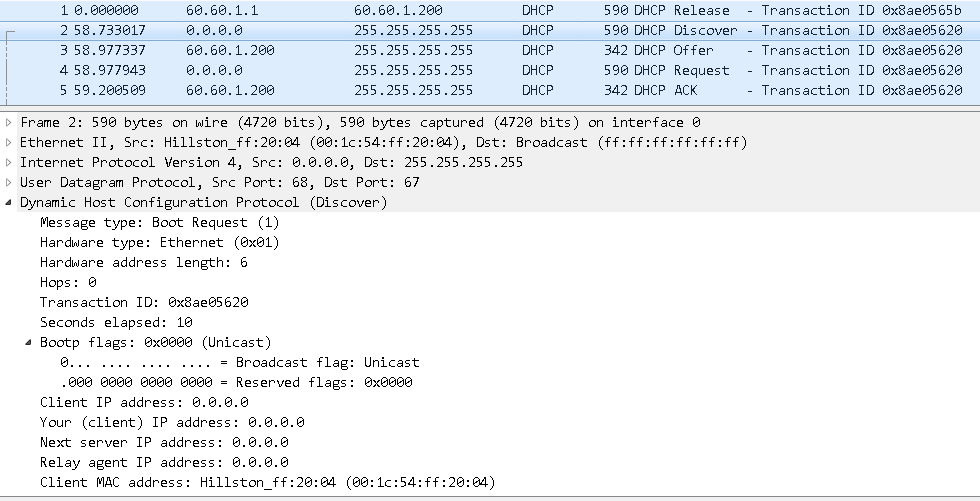
1. 源IP:0.0.0.0 ;目的IP:255.255.255.255
2. 源IP:服务器IP ;目的IP:255.255.255.255
3. 源IP:0.0.0.0 ;目的IP:255.255.255.255
4. 源IP:服务器IP ;目的IP:255.255.255.255
5. 源IP:客户机IP ;目的IP: 服务器IP
6. 源IP: 服务器IP;目的IP:客户机IP

DHCP中继：



1. 8种常见报文类型

DHCP是基于UDP的协议，客户端使用的端口号为68，服务端使用的端口号是67











1. 了解常见OPTION

|  |  |  |
| --- | --- | --- |
| **option** | **名称** | **描述** |
| 1 | Subnet Mask | 子网掩码 |
| 3 | Router Address | 路由器地址 |
| 6 | DNS | DNS server |
| 50 | Requested IP Address | 请求的IP地址 |
| 51 | Address Lease Time | 地址租约时间 |
| 53 | DHCP Message Type | DHCP 消息类型 |
| 54 | Server Identifier | 服务器标识 |
| 255 | End | 结束 |

5、8种报文类型

|  |  |
| --- | --- |
| **报文类型** | **含义** |
| DHCP DISCOVER | 客户端开始DHCP过程的第一个报文，目的是发现网络中的DHCP服务器； |
| DHCP OFFER | DHCP服务器对DHCP DISCOVER报文的响应； |
| DHCP REQUEST | （1）回应DHCP服务器的OFFER报文；  （2）在地址租约过半或者3/4租期时发送的延续租期请求； |
| DHCP ACK | DHCP服务器收到REQUEST报文并可以查找到相应的租约记录，发送ACK报文回应; |
| DHCP NAK | DHCP服务器收到REQUEST报文后没有查找到相应的租约记录，发送NAK报文回应; |
| DHCP RELEASE | 当用户不再需要使用分配IP地址时，就会"主动"向DHCP服务器发送Release报文，告知服务器用户不再需要分配IP地址，DHCP服务器会释放被绑定的租约; |
| DHCP DECLINE | DHCP客户端收到DHCP服务器回应的ACK报文后，通过地址冲突检测发现服务器分配的地址冲突或者由于其他原因导致不能使用，则发送Decline报文; |
| DHCP INFORM | DHCP客户端如果需要从DHCP服务器端获取更为详细的配置信息，则发送Inform报文向服务器进行请求，服务器收到该报文后，将根据租约进行查找，找到相应的配置信息后，发送ACK报文回应DHCP客户端 |

# IP路由基础

1. 了解路由基础概念

概念：路由器是一种连接多个网络或网段的网络设备。

当路由器从某个输入端口收到了数据包，路由器会按照数据包要去的目的IP地址，查找路由表中的路由。根据匹配到的路由出接口，将该数据包从相应的输出端口转发给下一个路由器。

**路由选择**：根据所选定的路由协议构造出**路由表**，同时经常或定期地去和相邻路由器交互路由信息而不断更新和维护路由表。

**分组转发：**根据**转发表**对数据包进行处理，将某个输入端口进入的数据包从一个合适的输出端口转发出去。

1. 学习路由表结构

**路由表**（Routing Table），又被称为路由选择信息库（RIB，Routing Info Base），RIB表中存储了所有的路由信息。只要路由器上有新生成的路由，就会被存储到RIB表中。

**转发表**（FIB，Forwarding Info Base）中存储了RIB表中最优的路由信息，是真正指导数据包转发的表。



> Destination: 目的网络地址

> Gateway: 下一跳地址

> Genmask: 子网掩码

> Metric: 优先级

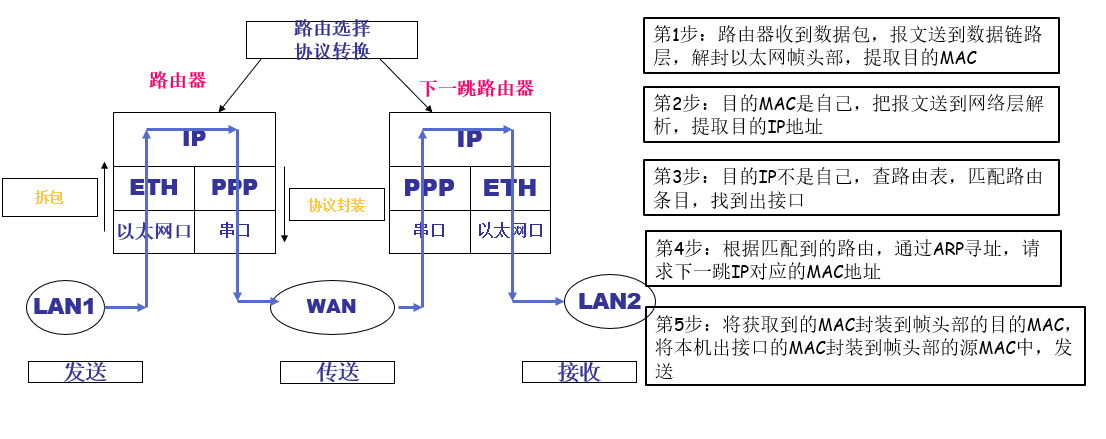
> Iface: 本地接口的名字，分组从哪个接口转发

> Flags:

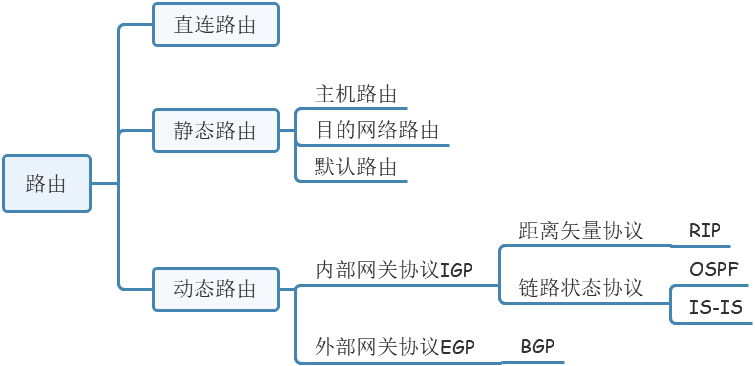
>> U: 该路由可用

>> G: 下一跳地址是路由器

>> H: 该路由是到一个主机



1. 学习路由类型

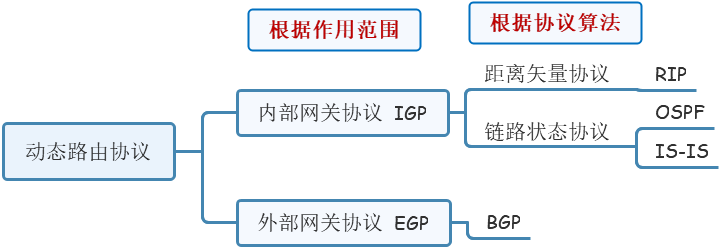


1. 直连路由：无需手工配置，只要接口配置了IP地址，同时接口的协议状态UP时，接口上配置的IP网段地址会自动出现在路由表中。直连路由下一跳为0.0.0.0
2. 静态路由：手工配置；不能自动改变；不会被通告给网络中的其他路由器；适用于小型网络
3. **主机路由:**目的地址是一个掩码长度为32位的完整主机地址。数据包匹配主机路由时，主机地址必须与目的地址完全匹配。
4. **默认路由**:又称为缺省路由，其目的网络是0.0.0.0/0，即IP地址和子网掩码全部是0。这条路由可以匹配到所有的数据包。

如果配置有默认路由，当路由器在路由表中找不到目标网络的路由时，会使用默认路由进行数据包转发。

使用默认路由可以帮助减少路由条目。

1. 目的网络路由：



|  |  |
| --- | --- |
| **静态路由** | **动态路由** |
| 由网络管理员手工指定的路由 | 路由器使用**路由协议**从其他路由器那里获悉的路由 |
| 当网络拓扑发生变化时，需要管理员手工更新静态路由 | 当网络拓扑发生变化时，路由器会自动更新路由信息 |

**路由协议概述：**

1. 路由器之间交互信息的一种语言。
2. 共享路由信息。
3. 维护路由表、提供最佳转发路径。

AS (Autonomous System) 自治系统：处于单个管理机制下的网络（如：企业、电信、移动、联通）

IGP (Interior Gateway Protocol) 内部网关协议：用于一个AS内部交换路由信息。

EGP (Exterior Gateway Protocol) 外部网关协议：用于多个AS之间交换路由信息。

DV (Distance-Vector) 距离矢量：基于距离矢量算法，路由器并不了解网络拓扑结构。类似路牌。

LS (Link-State) 链路状态：基于最短路径优先算法，路由器了解完整的网络拓扑结构。类似地图。

**路由协议操作规则**：

1.协议是在接口上运行的。

2.只能学习和发布相同协议已知的路由信息。

3.如果不同的路由协议之间需要交换路由信息，就需要进行引入(import)。

**路由器收敛：**

1. 当所有路由表包含相同网络可达性信息。
2. 网络（路由）进入一个稳定状态。
3. 网络在达到收敛前无法完全正常工作。

|  |  |
| --- | --- |
| **指标** | **备注** |
| 正确性 | 能够正确找到最优的路由，且无自环 |
| 快收敛 | 当拓扑结构发生改变之后，能够快速作出相应的路由改变 |
| 低开销 | 协议自身的开销（内存、CPU、网络带宽）最小 |
| 安全性 | 协议自身不易受攻击，有安全机制 |
| 普适性 | 适应各种拓扑结构和各种规模的网络 |

路由查看方法：

Linux pc: route –n

Windows pc: route print

StoneOS: show ip route

静态路由配置：

Linux PC: route add –net x.x.x.x/x gw x.x.x.x

Windows PC: route ADD x.x.x.x MASK x.x.x.x x.x.x.x

StoneOS: ip route x.x.x.x/x x.x.x.x

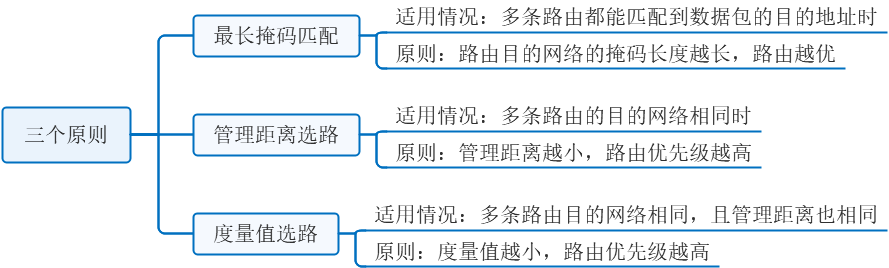
静态路由删除：

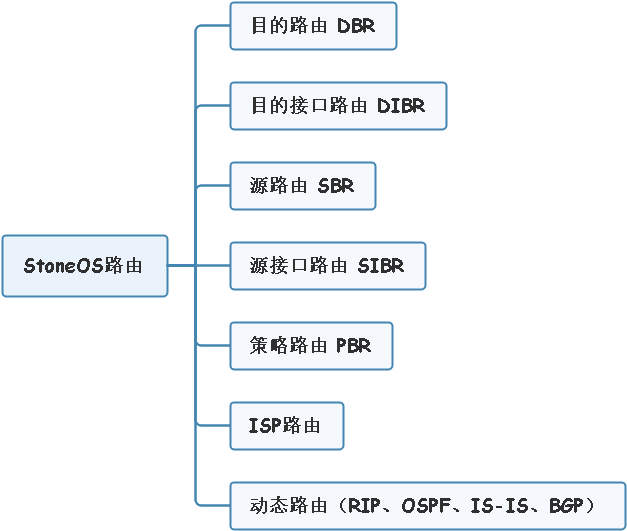
Linux PC： route del –net x.x.x.x/x gw x.x.x.x

Windows PC: route DELETE x.x.x.x

StoneOS: no ip route x.x.x.x/x x.x.x.x

1. 掌握路由选择的三个原则



****

# ICMP

* ICMP（Internet Control Message Protocol，Internet控制报文协议），用于在IP主机、路由器之间传递诊断和控制消息。
* ICMP使用IP协议进行传输
* ICMP既不是一个网络层协议，也不是一个传输层协议，而是位于两者之间



* ICMP报文可分为两大类
* 关于IP数据报传递的ICMP报文（差错报文）
* 关于信息采集和配置的ICMP报文（查询或信息类报文）
* 一般情况下，传入的信息类请求将被系统自动处理；差错类报文将传递给用户进程或传输层协议

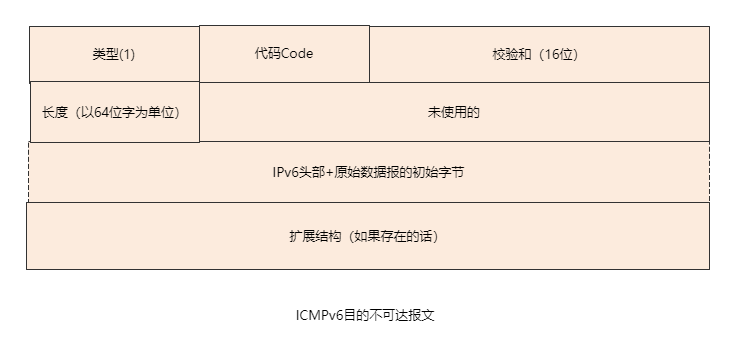
Icmpv4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 类型 | 代码 | 规范名称 | E/I | 用途 |
| 0 | 0 | 回显应答 | Inquire | 回显应答（Ping Reply），返回数据 |
| 3 | 1 | 目的主机不可达 | Error | 主机已知，但不可达 |
| 3 | 3 | 目的端口不可达 | E | 未知的/不可用的传输端口 |
| 3 | 4 | 需要分片但不允许分片 | E | 需要进行分片但被DF位禁止了，被PMTUD使用 |
| 5 | 1 | 主机重定向 | E | 指示一个可选的路由器（主机） |
| 8 | 0 | 回显请求 | I | 回显请求（Ping Request），数据可选 |
| 11 | 0 | 传输期间TTL超时 | E | 跳数限制/TTL超时 |
| 12 | 0 | 指针指示差错 | E | 字节偏移量（指针）指示第一个问题字段 |

Icmpv6

|  |  |  |  |
| --- | --- | --- | --- |
| 类型 | 代码 | 规范名称 | 描述 |
| 1 | 0 | 没有到目的地的路由 | 路由不存在 |
| 1 | 2 | 超出源地址范围 | 目的范围超出源地址范围 |
| 1 | 3 | 地址不可达 | 代码0-2不合适时使用 |
| 1 | 4 | 端口不可达 | 没有传输层实体在监听端口 |
| 2 | 0 | 数据包太大（PTB） | 需要分片 |
| 3 | 0 | 传输中超出跳数限制 | 跳数限制字段递减为0 |
| 4 | 0 | 找到错误的头部字段 | 一般的头部处理错误 |
| 4 | 1 | 无法识别的下一个头部 | 未知的下一个头部字段值 |
| 4 | 2 | 无法识别的IPv6选项 | 未知的逐条或目的地选项 |
| 128 | 0 | 回显请求 | PING请求，可能包含数据 |
| 129 | 0 | 回显应答 | PING应答，返回数据 |
| 133 | 0 | 路由器请求（RS) | IPv6 RS 和移动IPv6选项 |
| 134 | 0 | 路由器通告（RA) | IPv6 RA 和移动IPv6选项 |
| 135 | 0 | 邻居请求（NS) | IPv6 邻居发现（请求） |
| 136 | 0 | 邻居通告（NA) | IPv6邻居发现（通告） |
| 137 | 0 | 重定向报文 | 使用另一个下一跳路由器 |







ICMP重定向报文的接收者必须查看三个IP地址：

（1）导致重定向的IP地址，即之前访问的目标。可以在ICMP重定向报文ICMP数据部分的IP首部中查找

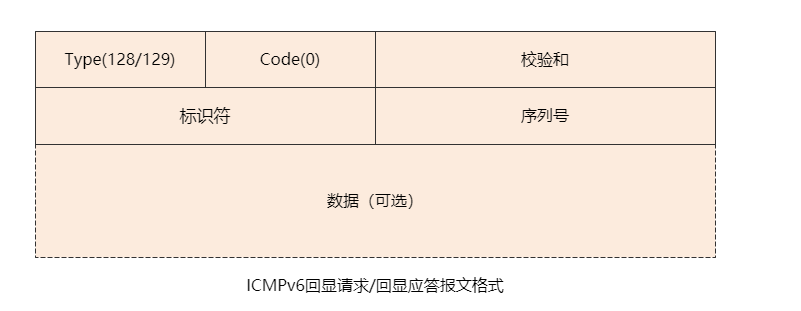
（2）发送重定向报文的路由器IP地址。ICMP重定向报文的IP数据报的源IP地址

（3）应该采用的路由器IP地址。ICMP重定向报文中ICMP首部中包含的IP地址

Ping：

* PING通常用于测试网络中另一台主机是否可达。
* PING使用ICMP Echo Request和ICMP Echo Reply来实现





* Windows ping命令使用

Usage: ping [option] target

-n count 发送回显请求数

-l size 指定echo数据大小

-i ttl 指定IP头部中的TTL

-f 数据包中设置不允许分片标志

* Linux中ping命令使用

Usage: ping [option] destination

-c count 发送回显请求数

-s packetsize 指定echo数据长度

-t ttl 指定IP头部中的TTL

Tracert：

* Tracert可以跟踪IP数据报从发送端到目的端所经过的所有路由器

Windows：Tracert

Linux：Traceroute/Tracepath

* Traceroute/Tracepath
* Traceroute程序通过向目标发送UDP数据报来实现：首先发送TTL=1的UDP数据报，然后将TTL字段每次加1，以确定路径中的每个路由器。
* 每个路由器在TTL超时丢弃UDP数据报时都返回一个“ICMP TTL超时”报文，而最终目的主机则产生一个“ICMP端口不可达”报文
* Tracert
* Tracert程序通过向目标发送ICMP Echo Request（ping）数据包来实现：首先发送TTL=1的echo request数据包，并在随后的每次发送过程中，将TTL递增1，直到目标主机响应ICMP Echo reply或TTL达到最大值，从而确定路由。
* 通过检查中间路由器发回的“ICMP TTL超时”消息来确定路由

Traceroute程序使用ICMP报文和IP首部中的TTL字段。

每个处理数据报的路由器都需要把TTL的值-1

当路由器收到一个IP数据包，如果其TTL字段为1，则路由器不转发

操作过程

主机的Traceroute程序首次发送TTL=1的UDP数据报给目的主机；

处理数据报的第一个Router将TTL字段-1，丢弃该数据报，并发回一份TTL超时的ICMP报文；源主机可以得到路径中第一个Router的地址；

Traceroute程序发送TTL=2的数据报；收到TTL超时的ICMP报文时，主机可以得到第二个Router的地址；

继续该过程，直到UDP数据报到达目的主机；目的主机收到TTL=1的数据报，处理该报文，发送ICMP端口不可达报文

Traceroute程序发送一份UDP数据报给目的主机，但它选择一个不可能的值作为UDP端口号（大于3000），使目的主机的任何一个应用程序都不可能使用该端口。该数据报到达时，使目的主机的UDP模块产生一份“端口不可达”错误的ICMP报文。

>>TTL=0时，未到达目的主机，回包类型：目的不可达（超时）

>>TTL=0时，已到达目的主机，回包类型：端口不可达。

用法: tracert [-d] [-h maximum\_hops] [-j host-list] [-w timeout]

[-R] [-S srcaddr] [-4] [-6] target\_name

选项:

-d 不将地址解析成主机名。

-h maximum\_hops 搜索目标的最大跃点数。

-j host-list 与主机列表一起的松散源路由(仅适用于 IPv4)。

-w timeout 等待每个回复的超时时间(以毫秒为单位)。

-R 跟踪往返行程路径(仅适用于 IPv6)。

-S srcaddr 要使用的源地址(仅适用于 IPv6)。

-4 强制使用 IPv4。

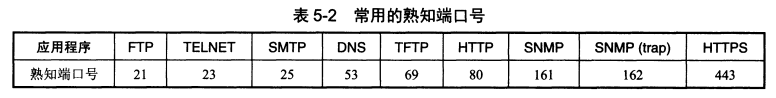
-6 强制使用 IPv6。

# 运输层：

* **物理层、链路层和网络层共同解决了主机通过异构网络互联的问题，实现了主机到主机的通信**
* **但实际上在计算机网络中进行通信的真正实体是位于通信两端主机中的进程**
* **运输层的任务是实现应用进程间的端到端的通信，向应用层提供通信服务**

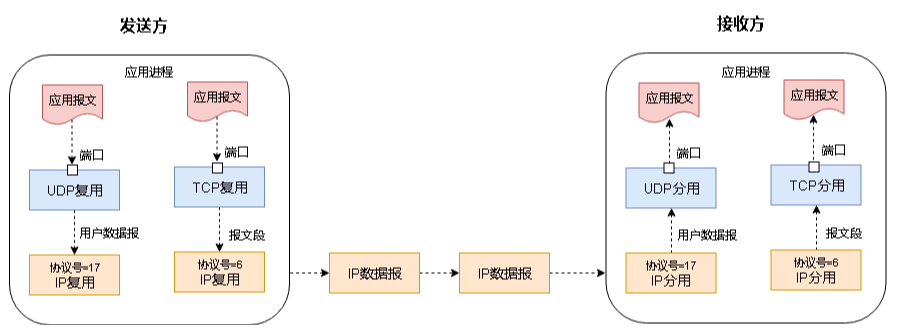
1. **运输层位于主机协议栈，是纯软件的协议**
2. **它屏蔽网络层细节（拓扑、路由选择协议）**
3. **常见协议：面向连接的TCP和无连接的UDP**

* **为了使不同的应用进程之间能够进行网络通信，就必须使用统一的方法对TCP/IP体系的应用进程进行标识**
* **TCP/IP体系使用端口号来区分应用层的不同应用进程**
  + **端口号范围：0-65535**
    - **熟知端口号：0-1023，分配给一些重要的应用协议，例如：HTTP使用80，DNS使用53**
    - **登记端口号：1024-49151，为没有熟知端口号的应用程序使用，必须在IANA按照规定的手续登记，如微软远程桌面RDP使用端口号是3389**
    - **短暂端口号：49152-65535，留给客户端进程短暂使用，通信结束后，端口号释放**



TCP： FTP, TELNET, HTTP, HTTPS, SMTP，

UDP： TFTP, DNS, SNMP

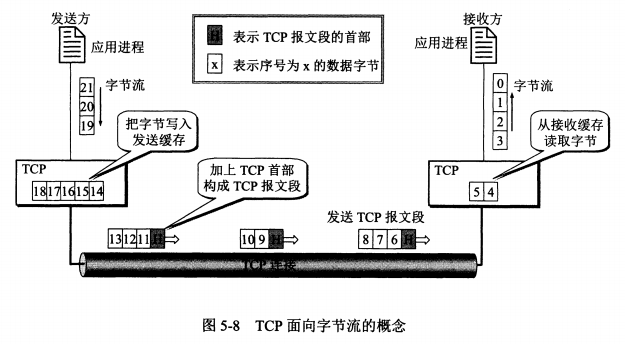


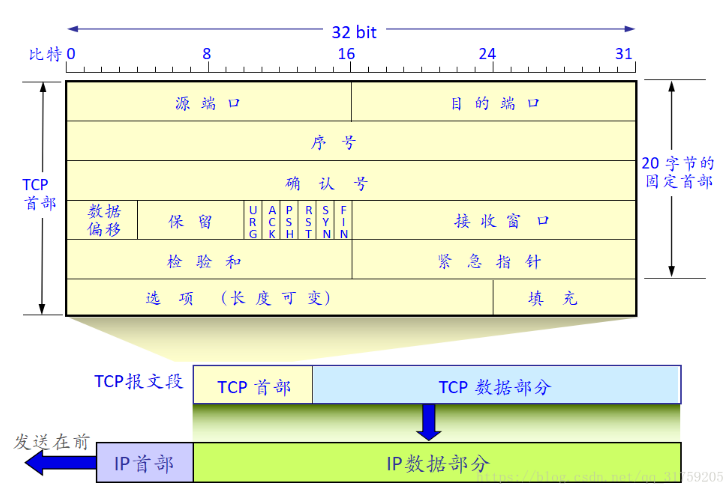
## TCP

* **TCP协议的两重身份**
  + **作为网络协议，实现有连接，可靠传输，报文按序到达。**
  + **作为一个主机软件，隔离了主机服务和网络，一个多路复用/解复用器，将诸多主机进程数据复用到IP层。**
* **TCP协议的四个要点**
  + **有连接**
  + **可靠传输**
  + **数据按序到达**
  + **端到端流量控制**

注意：TCP被设计时只保证这四点，此时它虽然也有些问题，然而很简单，但是更大的问题很快呈现出来，使之不得不考虑和IP网络相关的东西，比如公平性，效率，因此增加了拥塞控制，这样TCP就成了现在这个样子。

* **为了实现可靠传输，TCP采用了面向字节流的方式**
* **TCP在发送数据时，是从发送缓存取出一部分或全部字节并给其添加一个首部使之成为TCP报文段后进行发送**
  + **一个TCP报文段由首部和数据载荷两部分构成**
  + **TCP的全部功能都体现在它首部中各字段的作用**





* **源端口：标识发送该TCP报文段的应用进程**
* **目的端口：标识接收该TCP报文段的应用进程**
* **端口号占16 bit，范围0-65535**
* **套接字 Socket：（IP地址： 端口号）**
* TCP连接：：={socket1, socket2} = {(IP1: PORT1), (IP2, PORT2)}

**序号**

* **表示本报文段TCP数据的第一个字节的序号**
* **4字节，序号范围[0, -1]，可表示4GB数据**
* **一条TCP连接的初始序号(SYN)，初始序号随机产生**

**确认号**

* + **确认号是接收方对收到数据的确认**
  + **确认号=N，表示到序号N-1为止的所有数据都已正确收到，期望接收序号为N的数据**

**确认标志位ACK=1，确认号才有效**

* **数据偏移**
  + **即指TCP首部长度**
  + **占4 bit，并以4字节为单位**
* **保留和控制位**
  + **SYN，连接建立**
  + **FIN，连接释放**
  + **ACK，确认包，确认号有效**
  + **RST，复位连接**
  + **PSH，无需等待缓存填满尽快交付给应用层，交互式通信，希望对方立即回复，telnet/rlogin**
  + **URG，紧急报文，紧急指针有效(Control+C)**
* **窗口**
  + **指接收端的接收窗口大小，用来控制对方发送的数据量。也就是后面需要讲的滑动窗口的窗口大小。**
  + **由于TCP的头部窗口字段只有16bit，最多表示65535字节,为了表示更大的窗口，使用了可选的放大倍数。**
  + **Syn包option带上放大的windows scale**
* **检验和**
  + **占16 bit，检查范围包括TCP报文段首部和数据载荷部分**
  + **在计算校验和时，要在TCP报文段的前面加上12字节的伪首部**
* **紧急指针**
  + **紧急标志位URG=1时，紧急指针才有效**
  + **占16 bit，以字节为单位，用来指明紧急数据的长度**
* **选项**
  + **长度可变，最大40字节。TCP章节的最后会介绍几种常见选项字段**
  + **填充**
  + **由于选项的长度可变，因此使用填充来确保报文段首部总长度能被4整除**

小结

* **一个TCP连接需要四个元组来表示是同一个连接（src\_ip, src\_port, dst\_ip, dst\_port）准确说是五元组，还有一个是协议。但因为这里只是说TCP协议，所以，这里只说四元组。**
* **TCP首部的四个重要部分：**
  + **Sequence Number是包的序号，用来解决网络包乱序（reordering）问题。**
  + **Acknowledgement Number就是ACK，用于确认收到，用来解决不丢包的问题。**
  + **Window又叫Advertised-Window，也就是著名的滑动窗口（Sliding Window），用于解决流控的。**
  + **TCP Flag ，也就是包的类型，主要是用于操控TCP的状态机的。**

**TCP连接建立的目的**

* + **通信双方确认对方的存在**
  + **双方确定自己的初始序列号，并通知对方**
  + **允许双方协商一些参数（最大报文段长度、窗口大小等）**
  + **对传输实体资源进行分配**

**TCP连接建立的方式**

* + **采用客户端服务器方式（C/S），主动发起连接建立的应用进程叫做客户端，被动等待连接建立的叫服务器端。**

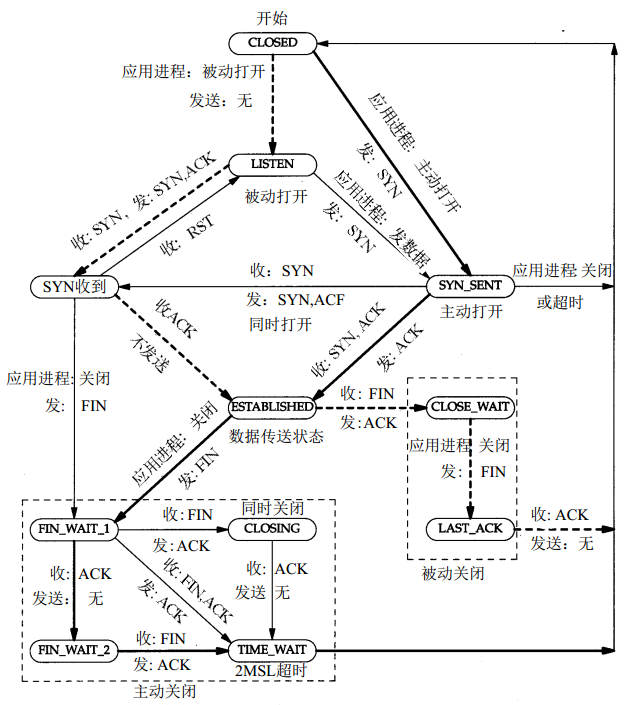
**TCP发送SYN之后会初始化2个定时器：**

TCP连接释放：

* **FIN\_WAIT\_2定时器：半关闭状态，启动fin\_wait\_2定时器，避免对端一直不发FIN，如果超过一定时限，则进入CLOSE状态。**
* **TIME-WAIT定时器：主动关闭的一端进入TIME\_WAIT的状态，启动一个时间等待定时器，测量一个连接处于TIME\_WAIT状态的时间（2MSL=4min），MSL：最长的报文段生存时间。**
* **保活定时器：检测TCP连接的另一端是否崩溃或重启。每收到一个报文，就重置保活定时器，通常=2h，发送探测报文，之后，每隔75s发送一次，共发送10个探测报文，关闭连接。(心跳包)**

**Rst关闭：**

* **主要使用场景**
  + **端口未打开(http://www.test.com:8888)**
  + **请求超时（客户端重传请求，始终收不到服务器响应，出现请求超时）**
  + **应用程序提前关闭(应用程序不接收包了)**
  + **非正常包等(在一个已关闭的连接上收到数据、已失效的连接请求)**



* 粗实线----Client
* 粗虚线----Server
* 细线----异常
* **TCP的连接建立**
  + **连接建立的目的**
  + **TCP使用“三次握手”建立连接**
  + **“三次握手”发送的TCP报文以及状态机的变化**
  + **“三次”而不是“二次握手”的必要性**
  + **TCP的连接释放**
  + **TCP通过“四次挥手”释放连接**
  + **“四次挥手”发送的TCP报文以及状态机的变化**
  + **TCP连接释放的几个定时器：**
    - **Fin-Wait-2定时器**
    - **Time-Wait定时器**
    - **保活定时器**
  + **MSL指最长报文段寿命，Time-Wait时间是2MSL，**

**它的必要性**

* + **TCP异常释放连接，RST报文的使用场景**

TCP可靠传输：

* **传输差错**：
  + **报文损坏(比特差错)/丢失/重复/乱序**
* **TCP可靠传输**：
  + **字节编号机制**
  + **数据段确认应答机制**
  + **校验和(报文损坏)**
  + **丢弃、缓存并排序(报文损坏/重复/乱序)**
  + **合理分段(TCP MSS，避免IP分片)**
  + **数据超时重传(报文损坏，丢失)**
  + **流量控制(接收方，控制发送速率，滑动窗口协议)**
  + **拥塞控制(网络，减少数据发送，慢开始/拥塞避免/快重传/快恢复算法)**

**确认应答（ACK）机制**

* + **基本上传输可靠性是靠确认号实现的，也就是说，每发送一个分段，接下来接收端必然要发送一个确认，发送端收到确认后才可以发送下一个字节。这个原则最简单不过了， “停止-等待”协议就是这个原则的字节版本。**
  + **停止等待协议的过程：发送数据**🡪**收到数据**🡪**发送确认**🡪**收到确认。**

**停止等待协议（ARQ）**

**超时时间的计算**

**通过收到接收方的ACK，才能知道接收方收到了数据。但是如果一直收不到确认，发送端要等多久？如果等太久，会造成数据延迟过大（本该重传的等待好久才重传），如果等待时间过短，可能确认还在路上，造成过度重传。**

**绝对不能随意去揣测超时的时间，而应该给出一个精确的算法去计算。毫无疑 问，一个TCP分段的回复到达的时间就是一个数据报往返的时间，因此标准定义了一个新的名词RTT。**

* + **RTT: TCP分段往返时间**
  + **RTT需要动态测量**

**用TCP的时间戳记录报文发送时间，辅助测量RTT**

**连续ARQ协议**

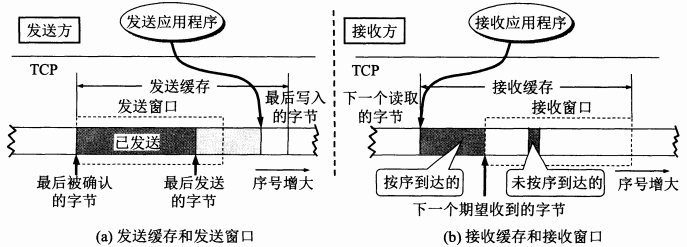
* + **流水线传输/滑动窗口**
  + **累计确认：对按序到达的最后一个分组发送确认。**
  + **捎带确认：接收方在自己有数据要发送时把确认信息顺便捎带上。**
  + **Delay ACK定时器：延迟ACK不超过0.5s，一连串MSS报文，每隔1个发送一个确认**

**连续ARQ协议**

* + **选择确认（SACK）：只重传丢失数据**
  + **选项，40字节，最多承担4个字节块的信息**

**TCP缓存和窗口的关系**

* + **发送窗口：在没有收到确认的情况下，发送方可以连续把窗口中的数据都发送出去，已发送未确认的必须暂时保留以便超时重传使用**
  + **接收窗口：接收方只能接收落在接收窗口的数据，接收方利用自己接收窗口的来限制发送方发送窗口大小**



**发送窗口根据接收方的接收窗口和网络的拥塞情况设置**

**接收窗口取决于接收方的应用、系统、硬件的限制，和自身处理能力有关**

流量控制作用于端到端的运输层

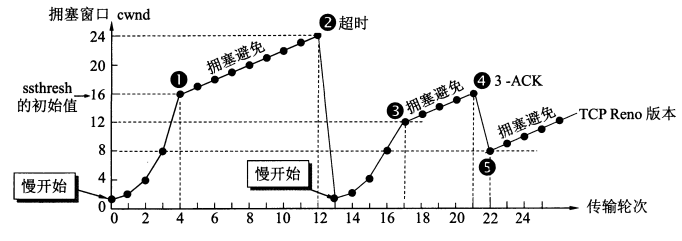
* + **流量控制就是让发送方的数据不要太快，要让接收方来得及接收**
  + **利用滑动窗口机制可以很方便地在TCP连接上实现对发送方的流量控制**

措施：

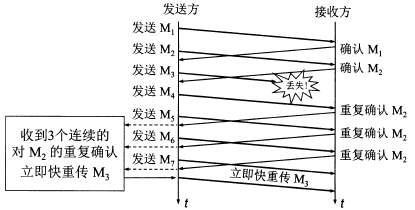
* + **持续定时器：使窗口大小信息保持不断流动，专门对付零窗口**
  + **当发送端收到零窗口通知后，启动持续计时器，若持续计时器到期，就发送一个零窗口探测报文，接收方就对零窗口探测报文进行确认，并通告自己的接收窗口，如果仍为0，发送方重新启动持续计时器。**

拥塞控制作用于点到点的网络层

* **慢开始：指数增长**
* **拥塞避免：线性增长**
* **快重传：收到3个同样的确认就立刻重传，不等到超时**
* **快恢复：cwnd不是从1重新开始**



快重传：



接收端窗口rwnd，又称通知窗口（awnd），是接收端根据目前的接收缓存大小所许诺的最新窗口值，是来自接收端的流量控制。

拥塞窗口cwnd是发送端根据自己估计的网络阻塞程度而设置的窗口值，是来自发送端的流量控制。

**TCP选项**

* **MSS**

1. 最大报文段长度(数据长度)
2. 2字节，取值范围[0, 2^16-1]，65535
3. MSS+TCP首部+IP首部=IP数据报长度(MTU)
4. SYN和SYN/ACK包携带，告知对方

* **时间戳选项**

1. TSOPT
2. 时间戳值(TSval)
3. 时间戳回显应答(TSecr)
4. RTT
5. 防止序号绕回(过滤同一个连接的重复包)

* **选择确认（SACK**

选项----SACK-Permitted/SACK

## UDP

|  |  |
| --- | --- |
| **UDP** | **TCP** |
| **无连接** | **面向连接** |
| **支持一对一/一对多/多对多的交互通信** | **支持一对一** |
| **面向报文（无分段，大小要合适）** | **面向字节流** |
| **尽最大努力交付，不可靠** | **可靠交付** |
| **无拥塞控制（恒定速率，允许丢失数据，不允许太大时延，IP电话/实时视频会议）** | **有拥塞控制/流量控制** |
| **首部开销小，仅8个字节** | **首部最小20字节，最大60字节** |

