数据链路层

一、概述

基本单位帧

使用点对点信道的数据链路层

1、封装成帧

数据包从应用层到网络层逐层封装之后，到达了数据链路层，数据链路层会为其在头尾添加帧头帧尾，使之在链路上以帧为单位发送数据

2、差错检测

封装好的帧通过物理层发送到传输媒体时，因为干扰出现了误码（比特0变成了比特1，比特1变成了比特0），如何让接收方知道帧出现了误码？差错检测

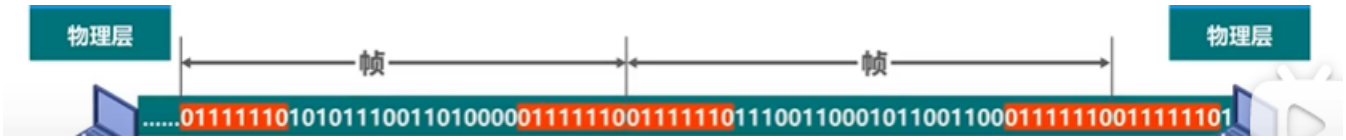
发送方会根据数据和算法计算出检错码，将其封装在帧尾，接收方就可以根据数据和帧尾的检错码判断有没有出现误码

3、可靠传输

接收方发现这个帧是个误码，会将其丢弃，因为数据链路层向上层提供的是可靠服务，可以确保接收方还可以收到这个帧的正确副本

二、封装成帧

1、帧头帧尾：控制信息、帧定界



标志：帧定界01111110（并不是所有），通过帧定界提取出一个个的帧

但是MAC帧不需要帧定界，因为物理层会在MAC帧前面封装8个字节的前导码，前同步码使接收方的时钟同步，后面1个字节的帧开始定界符标志着MAC帧，而且每一个帧间还有间隔时间，所以MAC帧不需要帧定界

2、透明传输：数据链路层对上层交付的数据没有任何限制

透明传输是指不管所传数据是什么样的比特组合，都应当能够在链路上传送。当所传数据中的比特组合恰巧与某一个控制信息完全一样时，就必须采取适当的措施，使接收方不会将这样的数据误认为是某种控制信息。这样才能保证数据链路层的传输是透明的。

1）面向字符的透明传输，添加转义字符：

发送数据时，会扫描数据部分看有没有帧定界符，如果有帧定界符或转义字符（一个字节，十进制27），会在帧定界符或转义字符前加上一个转义字符，数据通过物理层发送给接收方时，接收方数据链路层在物理层交付的比特流中提取帧，遇到第一个帧定界符，会认为这是帧的开始，识别到转义字符时剔除转义字符，并把转义字符后的帧定界符视为数据，直到帧结束

2）面向比特的透明传输，添加比特：

扫描数据部分，每5个1添加1个0比特，在接收方通过物理层接收到比特流时，从帧定界符开始，每5个连续的1遇到0比特就将其剔除，这样便可以保证帧定界符的唯一性

3、帧的数据部分长度应该远大于帧头和帧尾，但是帧的数据部分长度也有限制，MTU：最大传输单元

三、差错检测

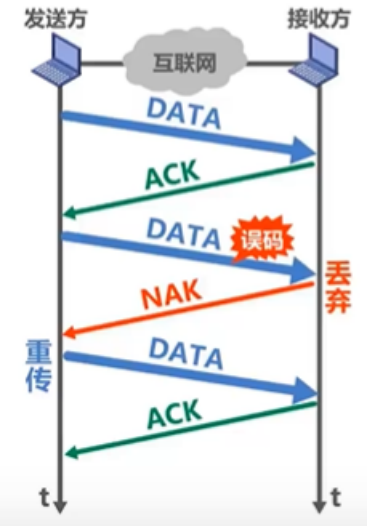
1、奇偶检验 -- 漏检率高 -- 为什么?

2、循环冗余校验CRC

发送数据前，帧尾会封装一个检错码，在发送数据的过程中产生了误码，到达接收方时，接收方便可以通过对当前数据进行特定算法和检错码进行比对，从而得知数据有了误差

四、可靠传输

1、停止 - 等待协议SW



1）发送方发送数据分组，需要接收方回复ACK确认分组，才可以将数据分组从缓存中删除，然后发送下一个分组。如果发送方发送的数据有误码，那么接收方收到之后经过校验，就会回复一个NAK否认分组，发送方就会重传该数据分组

2）超时重传：

发送方发送的数据分组丢失了，那么接收方还会一直等待不会发送ACK和NAK，发送方也无法去发送下一个数据。为了解决这个问题，出现了超时重传的概念，即发送方在发送数据时，启动超时定时器，如果超出了超时定时器设置的时间发送方还是没有接收到ACK或NAK，那么发送方启动超时重传

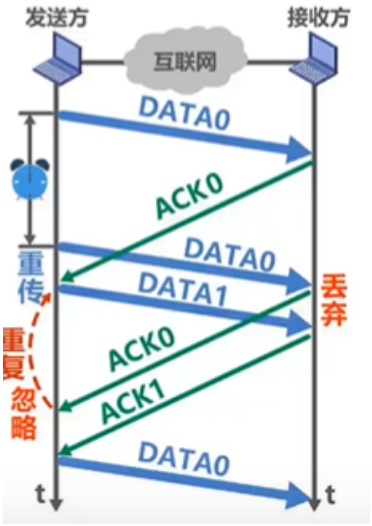
时间 略大于 发送方到接收方的往返时间

3）如果确认分组丢了呢？ -- 给发送分组编号

发送方发送DATA0，接收方的回复分组丢失，发送方在超时时间里收不到回复，就会超时重传DATA0，那么给分组编号，这样接收方就知道了本次数据分组和刚才的数据分组是一个，接收方丢弃了本次的DATA0（因为重复），为了防止发送方再次超时重传，接收方会回复一个ACK。然后发送方发送DATA1……

4）确认迟到导致的重复确认 -- 给确认分组编号

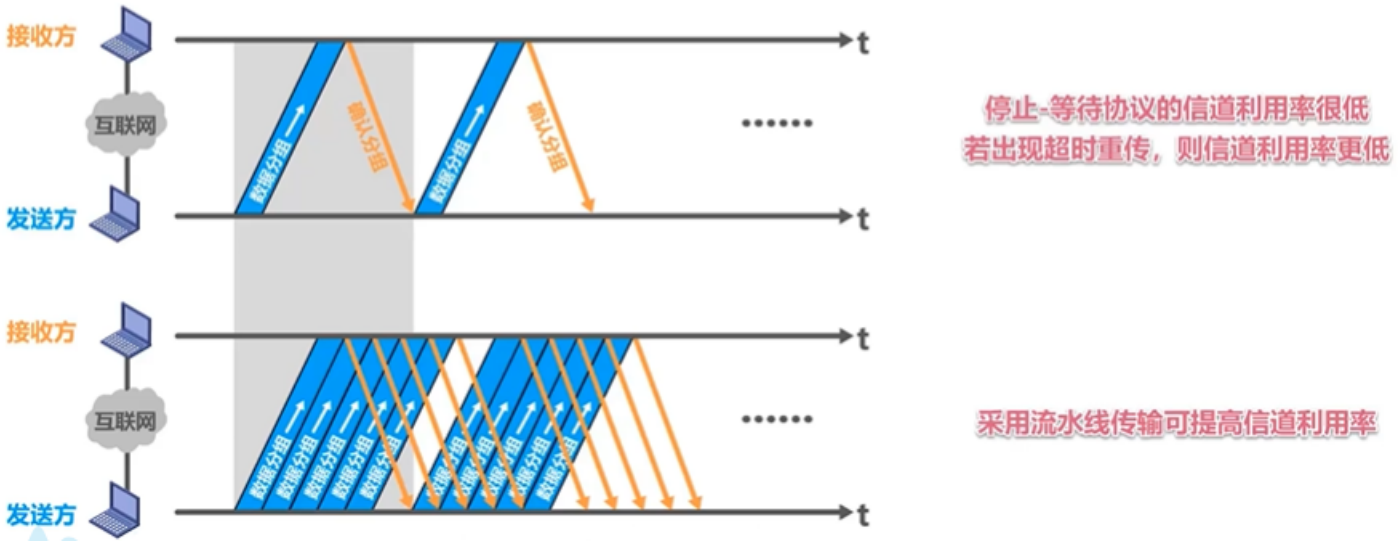
确认迟到导致发送方超时重传数据分组，确认到达后发送方再次发送同样的数据分组，接收方接收到了重复的数据分组之后，回复ACK包以免发送方再次重传



- 发送方发送DATA0，接收方发送DATA0的ACK包，但是在超时时间里，发送方并没有收到DATA0的ACK包，所以发送方会再次发送一个DATA0（超时重传）
- 在超时重传的过程中，发送方收到了之前的DATA0的ACK包，然后发送方再发送DATA1
- 接收方在接收到超时重传的DATA0时，发现这是一个重复的分组，将其丢弃，然后再发送一个ACK包，以免发送方再次重传这个数据分组
- 为了区分回复分组哪个是DATA0的，哪个是DATA1的，可以给确认分组编号
- 这样，发送方知道了有两个重复的确认分组，忽略即可
- 现在的0和之前的0不是一个数据分组

  要让某一方知道这个数据分组是重复的，接收方在收到超时重传的重复数据分组时，都会丢弃并发送ACK包，以免发送方再次重传

2、回退N协议

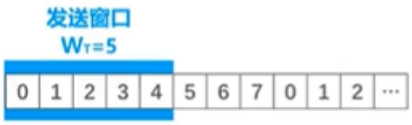


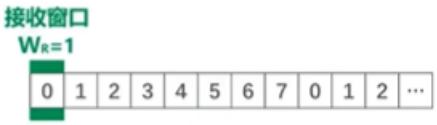
采用3个比特给分组编号，即0~7

发送窗口的尺寸：1 < W <= 2 ^ 3 - 1，（如果W = 1的话，那就变成了停止等待协议），比如本例W = 5

序号落在发送窗口里的5个连续分组可以一起发送，接收窗口只能为1

1）





正确传送：发送方一起发5个数据分组，接收方只能接受一个分组，每接受一个分组便给发送方回复，接受窗口前移一个窗口，发送窗口前移1个，这样等到接受端都接收完了，发送方的发送窗口就移动到了新的序号，然后发送方之前的数据分组就可以从删除了

2）累计确认 -- 即使有一个确认分组丢了，也不必重传数据分组

接收方不必一个一个的对数据分组进行确认，可以累计确认前n个数据分组，发送ACKn的回复包，表示前n个数据分组已经被正确接收，但是不能及时反映出差错信息

3）如果发送的数据分组有一个数据分组出现了误码 -- 回退N帧协议

如果5出现了误码，接收方根据检错码发现数据分组有误，丢弃5数据分组，然后接收端去比对窗口序号（6 7 0 1）发现不匹配，将这4个数据分组全部丢弃，并向发送方发送上一次累计确认的ACK4，每丢弃一个数据分组，就要发送1个ACK4，一共4个ACK4，发送到发送方之后，发送方发现这是重复的ACK4，于是发送方立刻重传。如果4个ACK4并没有触发发送方立即重传，超出了超时计时器的规定时

间，那么启动超时重传（在发送窗口内且已经发送的数据分组重传，即回退N帧协议），将之前的发送分组再次传输

4）发送窗口的尺寸不能超过其上限 -- 如果回复包丢失发送方超时重传，接收方不知道是新分组还是旧分组

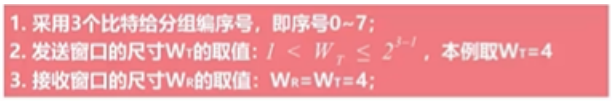
3、选择重传

1）回退N帧协议有个缺点，因为接收方只能接受一个分组，所以如果发送的数据有误码，它要丢弃所有的数据分组（哪怕除了误码之外的数据是正确的），这样大大的浪费了信道资源

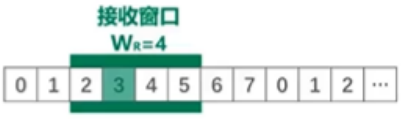
所以出现了选择重传，即接收窗口尺寸>1，这样如果数据分组有一个出现了误码，可以直接重传出现误码的分组，其他正确无误码的数据分组且在接收窗口内的收下

不能采用累计确认，而是对每一个正确分组进行逐一确认

2）比如



发送方发送0 1 2 3分组，其中2丢失，接收方只接收到了0 1 3，这时接受方发送0 1的确认分组，接收窗口向前移动2个单位，发送窗口向前移动2个单位，同时前两个数据分组从缓存中删除，接收端把0 1交付给上层



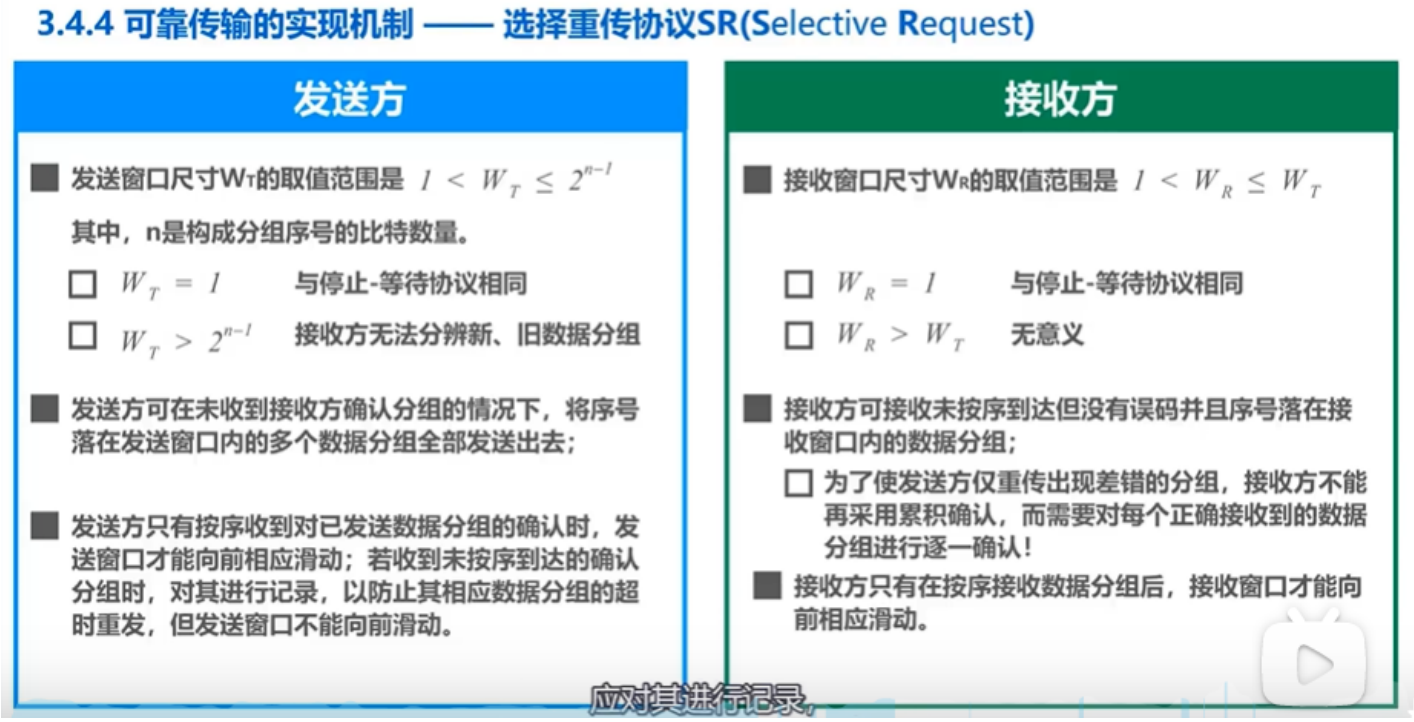
接收方再发送3号确认分组，由于发送方没有收到2号确认分组，不是按序到达的，那么发送窗口不会前移，但是会对3号数据分组进行标记，表示3号数据分组已被正确接受

4 5号也同理

如果其间，2号的重传计时器超时了，这个时候启动对2号分组的超时重传

■ 发送方的发送窗口尺寸W_T必须满足：$1 < W_T \leq 2^{(n-1)}$

■ 接收方的接收窗口尺寸W_R必须满足：$1 < W_R \leq W_T$



五、MAC地址、IP地址、ARP协议

MAC地址 - 数据链路层，IP地址、ARP协议 - 网络层

1、MAC地址 - 数据链路层

1）两个主机点对点通信 - 不需要地址

2）多个主机在同一个广播信道里，想实现两个主机的通信，需要一个数据链路层的地址

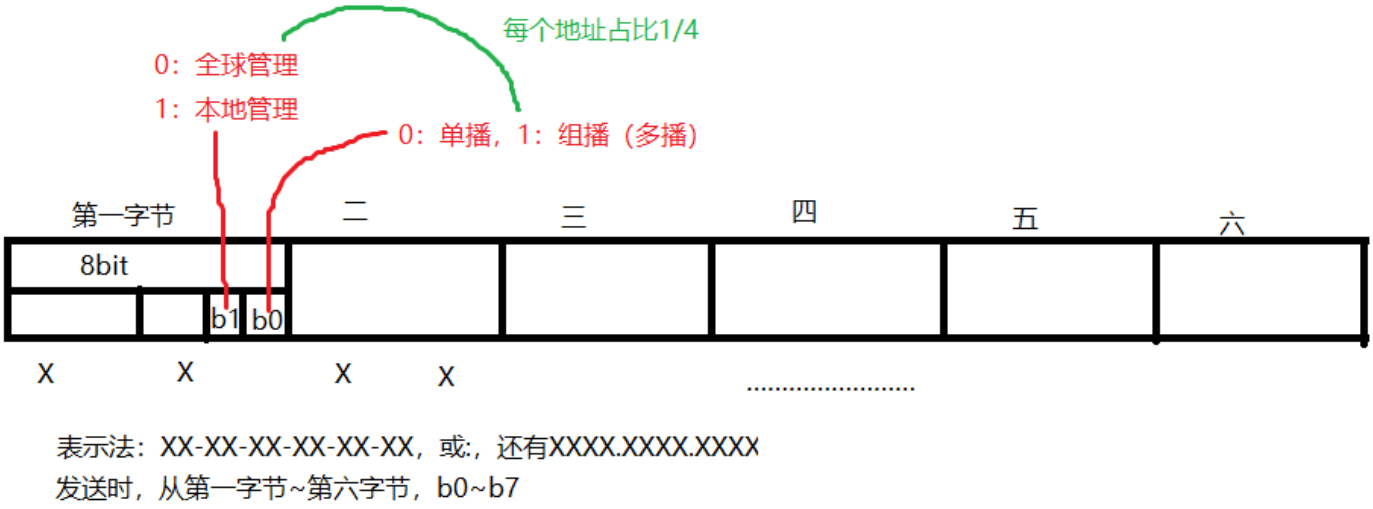
数据包到了数据链路层，会被封装成帧去发送，帧头有源地址和目的地址，这类地址是用于MAC(媒体接入控制Medis Access Control)，所以叫MAC地址。MAC地址固化在了硬件里，所以也成为硬件地址。也叫物理地址，但它不在物理层。

3）MAC地址 - 网络接口的唯一标识（而不是网络设备）

用户主机包含两个网络适配器：有线，无线，每个网络适配器都包含唯一的MAC地址，交换机和路由器有更多的网络接口，所以会有更多的MAC地址

4）可以通过MAC地址查询到厂商信息

5）

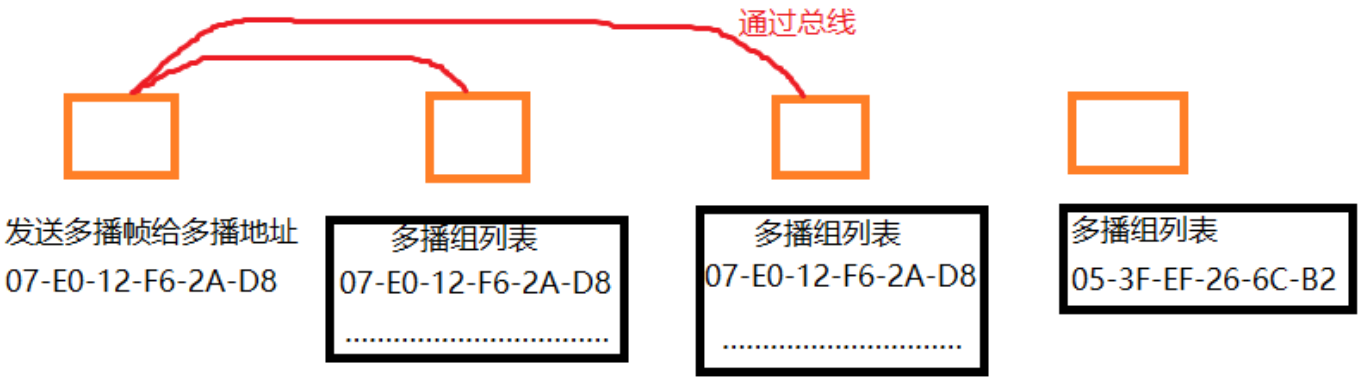


6）单播：主机B组装自己的帧（源地址和目的地址），通过总线发送出去，总线上每台机器检验自己的MAC地址和帧的目的地址是否一致，最后接收该帧，交给上层处理

广播：……（目的地址：广播地址FF-FF-FF-FF-FF-FF），总线上每台机器都会收到这个帧，发现是广播帧，最后接收该帧，交给上层处理

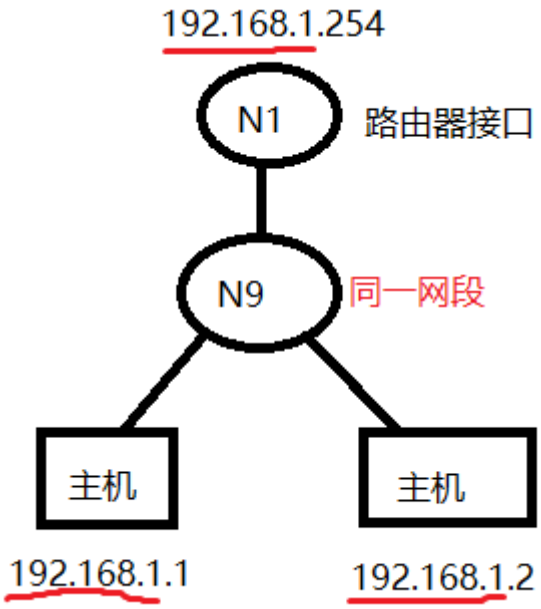
组播：

A发送多播帧给多播地址07-E0-12-F6-2A-D8，因为07的第一字节b0 = 1，所以是多播

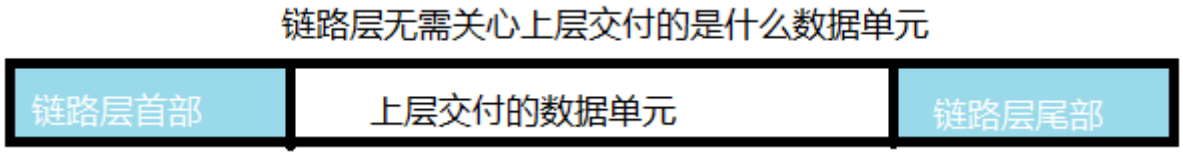


2、IP地址

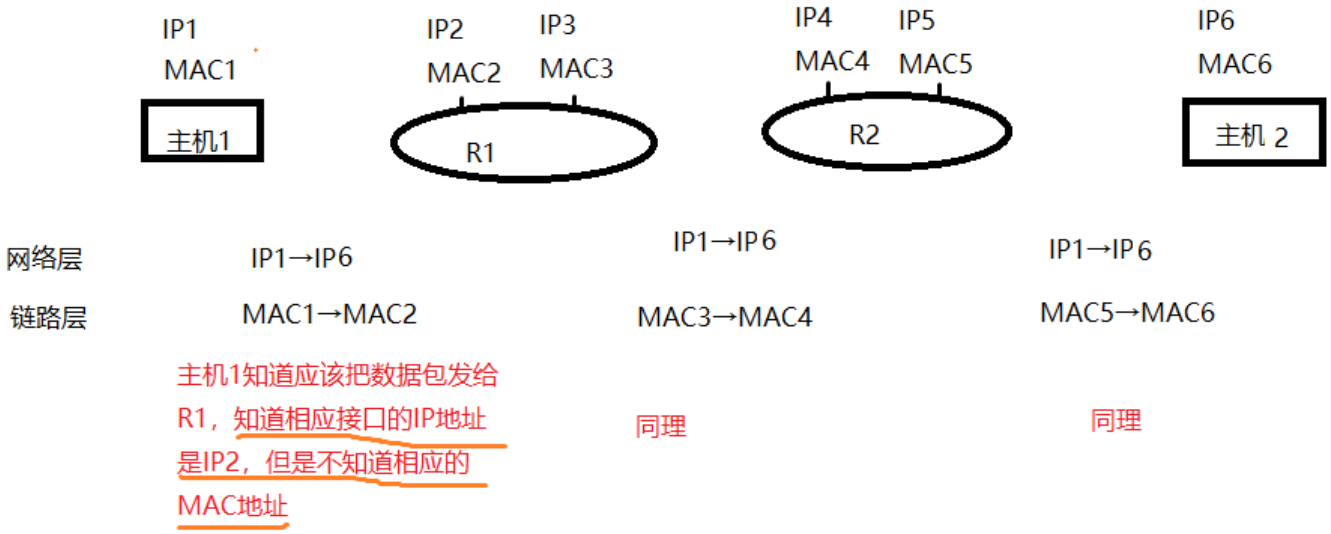
1）标识两部分信息：主机和网络



2）



3）数据包转发过程中，源IP地址和目的IP地址始终不变；源MAC地址和目的MAC地址随着网络（链路）而改变



3、ARP协议

（可以获取目的主机或路由器接口的MAC地址）

1）B和C如何通信？

- B的ARP缓存表，如果没有，把ARP请求报文在网段里广播
- C给B发送ARP响应报文，告知自己的MAC地址
- B把C的IP地址和MAC地址存放进ARP缓存表里

2）ARP高速缓存：

IP地址 MAC地址 类型

类型分为动态和静态，动态：主机自动获取，生命周期默认为2分钟，比如主机网卡坏了换了个新的，那么IP地址不会变，但是MAC地址变了

静态：手工设置，系统重启后依然有效

3）ARP协议不可以跨网段，只能在一个网段内有效

4）没有安全校验机制

六、以太网交换机自学习和转发帧的流程

1、一些概念

工作在数据链路层（也包括物理层）

集线器（无记忆性）和交换机（有记忆性）

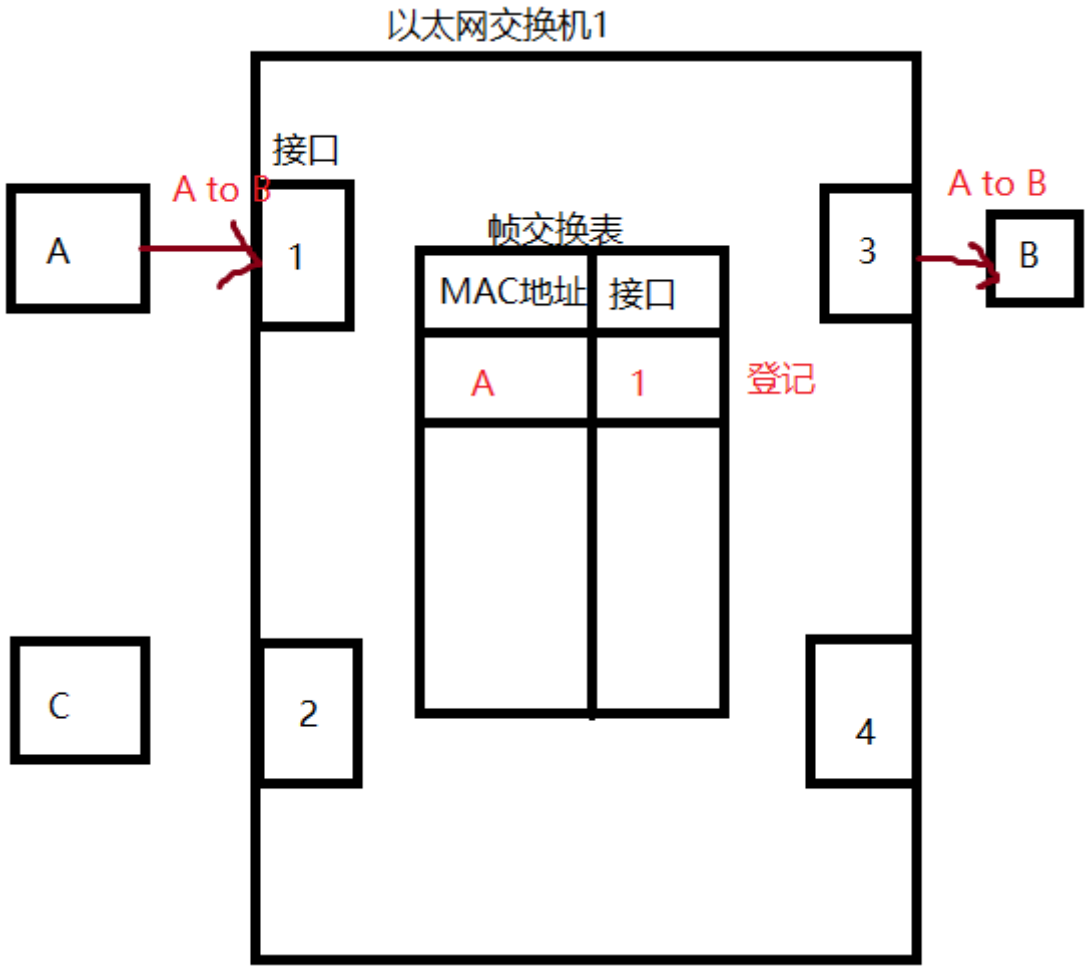
刚上电启动，帧交换表是空的，随着网络信息的转发，以太网的帧交换表会通过自学习逐渐建立

2、以太网交换机的自学习和转发

（盲目泛洪和明确）

假设各主机已经知道了彼此的MAC地址，不需要进行ARP请求

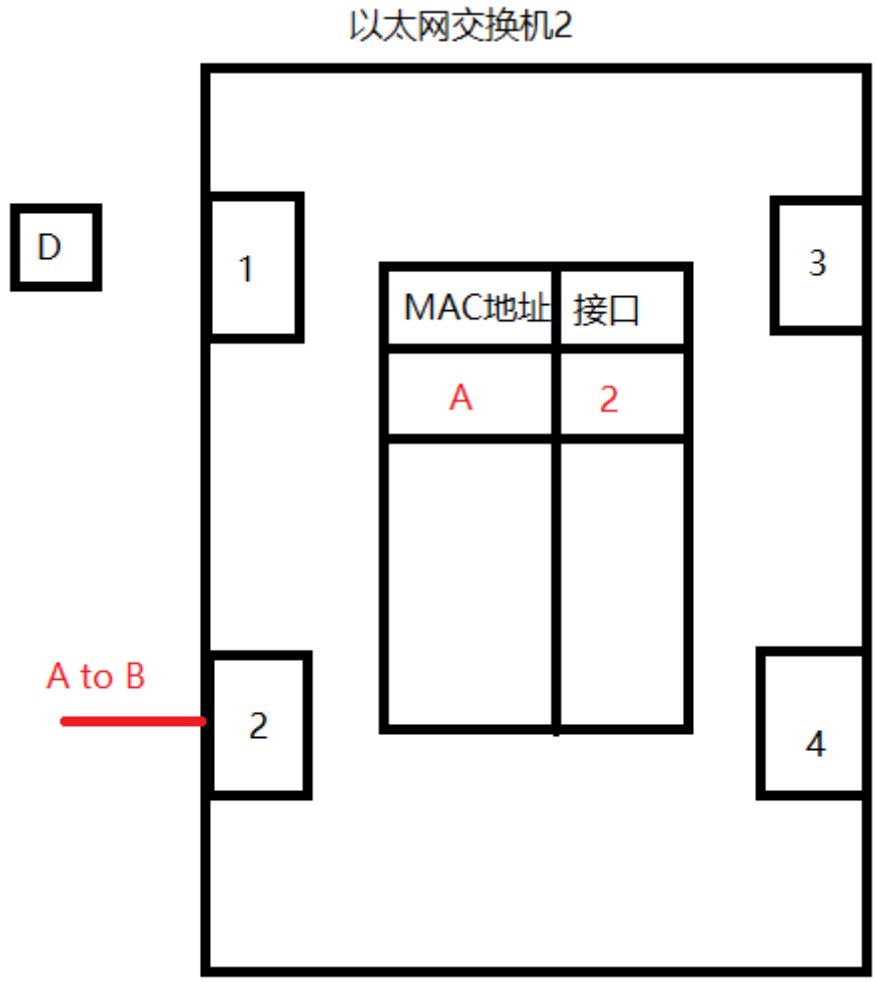
1）登记



2）转发：此时在帧交换表中寻找B的MAC地址，没有找到，就会把A to B的帧发给2 3 4接口，盲目转发（盲目泛洪）

主机B C根据帧的目的地址，判断是否要接收还是要交付上层处理

3）Ato B通过接口4（4和2相连）发送给了下一个交换机：登记、泛洪、丢弃该帧



4）转发（明确）

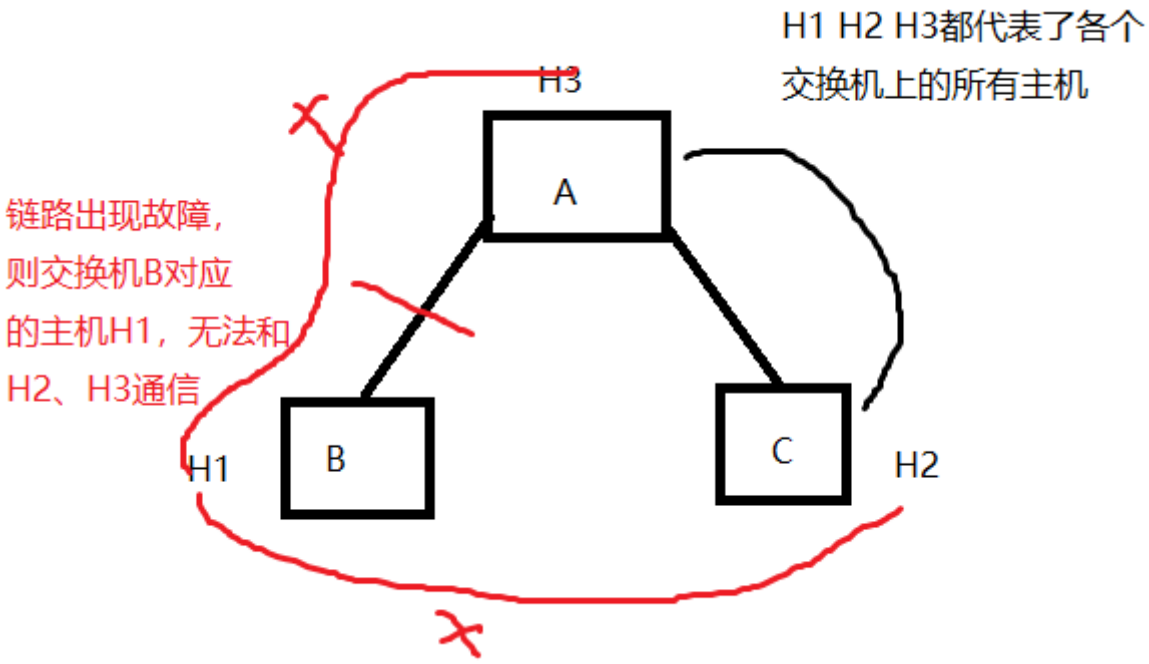
5）如果主机A和主机E通过集线器连在了同一个交换机接口，E to A?

- E to A的帧直接给了A
- E to A的帧给了1，帧交换表登记E的接口，寻找A的接口，按理说应该转发给A，但这是没有必要的，所以帧被丢弃

6）每条记录都有自己的有效时间，因为MAC地址会改变：更换网卡或者更换主机

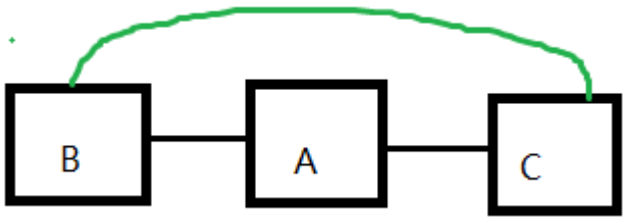
七、以太网交换机的生成树协议STP

1、无法通信



如果AB AC之间链路出现故障，则H1 H2 H3都无法互相通信

2、冗余链路



解决了问题

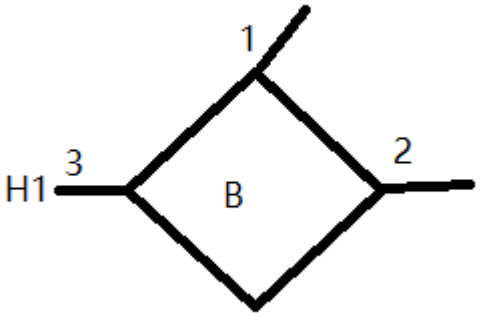
但是也带来了新的问题：转发是单向的，因为交换机有多个接口，但是还是会造成无限套娃，兜圈 - 广播风暴

3、网络环路带来的问题

1）广播风暴：大量消耗网络资源，无法转发其他帧了

2）主机收到重复的帧

3）交换机的帧交换表震荡（漂移）：比如交换机B的帧交换表，来自H1的一个帧，它会经过B的多个接口进入，而且是反复循环进入，每一次新的进入帧交换表都会删除之前的记录，不断删除，在错误记录之间反复震荡。



会在1和2之间反复震荡

4、生成树协议STP

逻辑上没有环路（不管物理上怎么连接），首次连接交换机或网络物理拓扑发送变化，都会重新生成树