课程 BA000007

以太网技术

ISSUE1.0

Huawei Technologies

目 录

课程说明

课程介绍

本课程是宽带网络产品工程师培训的公共课程。

本课程较全面的介绍了以太网的相关知识，包括以太网的物理层、数据链路层、VLAN、千兆以太网、生成树协议等等。

课程目标

完成本课程学习，学员能够：

- 了解以太网物理层的工作过程了解二层组播协议和生成树协议的工作过程掌握 802.1Q 帧格式和各个字段的含义
- 掌握以太网口自协商的原理。

第1章 概述

自从 1946 年第一台数字计算机问世到现在，经历了半个多世纪的时间。在这半个世纪的里程中，计算机技术的发展大体经历了三个成熟的阶段：第一个阶段是大型机时代，典型的是运行 UNIX 操作系统的大型计算机，该机器带很多终端，每个用户占用一个终端，大型机采用分时的技术为每个终端轮流服务，在用户看来自己单独享用了一个完整的计算机，这种体系结构主要用于科研机构来进行大量的数学运算。第二个阶段是客户服务器阶段，也就是所谓的 C/S 结构。最有代表性的是 NOVELL 公司的 NetWare 操作系统，这个系统分为服务器和客户机两部分，服务器软件安装在一台性能比较高的服务器上，客户机软件则安装在工作中终端上（一般是基于 DOS 操作系统的 PC 机），这些服务器和客户机通过网络连接起来，达到文件和数据库共享的目的，后来的 WINDOWS NT 也是基于这样的体系结构，但是在软件上引入了一些分布式的处理体系。第三个阶段，也就是目前所处的阶段，是网络阶段。这个阶段的特点是，计算机之间的互连越来越复杂，不但互连的速度有很大提高（达到 100M），而且在地理位置上也跨越了地域，通过高速专线把处于不同城市，甚至是不同国家的计算机网络连接起来。这样复杂的网络对网络设备提出了很高的要求。

从上面的分析可以看出，在第二和第三阶段中，必须有一种技术来把本地的许多计算机连接起来。这种技术就是所谓的局域网技术。到目前为止，存在许多种局域网技术，比如令牌环，令牌总线，以太网等等。在这些技术当中，以太网技术以其简明，高效的特点逐渐占据了主导地位。

1.1 以太网技术起源

以太网技术起源于一个实验网络，该实验网络的目的是把几台个人计算机以 3M 的速率连接起来。由于该实验网络的成功建立和突出表现，引起了 DEC，Intel，Xerox 三家公司的注意，这三家公司借助该实验网络的经验，最终在 1980 年发布了第一个以太网协议标准建议书。该建议书的核心思想是在一个 10M 带宽的共享物理介质上，把最多 1024 个计算机和其他数字设备进行连接，当然，这些设备之间的距离不能太大（最大 2.5 公里）。

之后，以太网技术在 1980 年建议书的基础上逐渐成熟和完善，并逐渐占据了局域网的主导地位。

1.2 以太网的设计目标

开始的时候，以太网设计建议书提出了以太网设计的基本目标，即所谓的功能特性。在后来的应用中对这些目标进行了不断的补充和完善，最终形成了一个成熟的体系。主要包含以下几点：

- 简明性。这是以太网技术最大的特点，正是因为简明性为将来的统治地位奠定了基础；
- 低成本。成本不要太高，一般的单位能够有能力购买需要的部件来组建网络；
- 兼容性。不应该对网络层实现施加任何限制，即以太网的所有功能都在数据链路层实现；
- 寻址灵活。应该有一种机制来确定网络中的一台计算机，全部计算机或一组计算机；
- 公平。各个终端应该公平的享有带宽；
- 高速。当时来说，10M 的速率已经是个天文数字了，所以把以太网的共享总线带宽设计为 10M；
- 分层结构。数据链路层协议不应该随物理介质的不同而变化；
- 全双工。随着以太网技术的发展，共享介质技术（即半双工）已经逐渐不能满足需求，需要效率更高的全双工技术；
- 差错控制。该技术应该能够发现传输中的错误并进行纠正，如果不能纠正，则丢弃接收到的数据；
- 速度灵活性。不应该局限在 10M 的速率上，应该能适应不同的速率；
- 优先级。网络设备应该能对一些关键性的业务提供优先可靠的传输。

这些功能目标有的是必须实现的，比如简明，低成本，寻址灵活等，有的则根据需要实现，比如优先级等，但在今天的以太网设备中，这些可选的性能目标都已经实现。

1.3 以太网基本技术

1.3.1 半双工CSMA/CD

根据以太网的最初设计目标，计算机和其他数字设备是通过一条共享的物理线路连接起来的。这样被连接的计算机和数字设备必须采用一种半双工的方式来访问该物理线路，而且还必须有一种冲突检测和避免的机制，来避免多个设备在同一时刻抢占线路的情况，这种机制就是所谓的 CSMA/CD（带碰撞检测的载波监听多路访问）。

CSMA/CD 的工作过程是这样的：终端设备不停的检测共享线路的状态，只有在空闲的时候才发送数据，如果线路不空闲则一直等待。这时候如果有另外一个设备同时也发送数据，这两个设备发送的数据必然产生碰撞，导致线路上的信号不稳定，终端设备检测到这种不稳定之后，马上停止发送自己的数据，然后再发送一连串干扰脉冲，然后等待一段时间之后再进行发送。

发送干扰脉冲的目的是为了通知其他设备，特别是跟自己在同一个时刻发送数据的设备，线路上已经产生了碰撞。检测到碰撞后等待的时间也是随机的，不过逐渐在增大。

1.3.2 以太网的物理介质

刚开始的时候，以太网是运行在同轴电缆上面的，通过复杂的连接器把计算机和终端连接到该电缆上，然后还必须经过一些相关的电信号处理，才能使用。这样的结构相对复杂，而且效率上不是很理想，只能适合于半双工通信（因为只有一条线路）。到了 1990 年，出现了基于双绞线介质的 10BAST-T 以太网，这是以太网历史上一次最重要的革命。

10BAST-T 得以实施，主要归功于多端口中继器和结构化电话布线。多端口中继器就是目前所谓的 HUB，终端设备通过双绞线连接到 HUB 上，利用 HUB 内部的一条共享总线进行互相通信。物理上这种结构是星形的，但实际上还是沿用了 CSMA/CD 的访问机制，因为 HUB 内部是通过一条内部总线把许多终端连接起来的。

10BAST-T 以太网技术使用了四对双绞线来传输数据，一对双绞线用来发送，另外一对用来接收。之所以使用一对双绞线来分别进行收发，主要是电气特性上的考虑，发送数据的时候，在一条线路上发送通常的电信号，而在另外一条线路上发送与通常电信号极性相反的信号，这样可以消除线路上的电磁干扰。

后来又出现了 100M 的以太网，即所谓的快速以太网。快速以太网在数据链路层上跟 10M 以太网没有区别，不过在物理层上提高了传输的速率，而且引入了更多的物理层介质，比如光纤，同轴电缆等。运行在两对双绞线上的 100M 以太网称为 100BAST-TX，运行在光纤上的 100M 以太网则为 100BASE-FX，还有运行在四对双绞线上的 100BAST-T4 等。

1.3.3 全双工以太网和以太网交换机

把双绞线作为以太网的传输介质不但提高了灵活性和降低了成本，而且引入了一种高效的运行模式——全双工模式。所谓全双工，就是数据的发送和接收可以同时进行，互不干扰。传统的网络设备 HUB 是不支持全双工的，因为 HUB 的内部是一条总线，数据接收和发送都是在该总线上进行，没有办法进

行全双工通信，因此，要实现全双工通信，必须引入一种新的设备，即现在的交换机。

交换机跟 HUB 的外观相同，都是一个多端口设备，每个端口可以连接终端设备和其他多端口设备。但在交换机内部就不是一条共享总线了，而是一个数字交叉网络，该数字交叉网络能把各个终端进行暂时的连接，互相独立的传输数据，而且交换机还为每个端口设置了缓冲区，可以暂时缓存终端发送过来的数据，等资源空闲之后再进行交换。正是交换机的出现，使以太网技术由原来的 10M/100M 共享结构转变为 20M/200M 独占带宽的结构，大大提高了效率，而且可以在交换机上施加一些软件策略，来实现附加的服务，比如 VLAN（虚拟局域网），优先级，冗余链路等。

1.3.4 以太网的应用

以太网设计的初衷，就是把一些计算机联系起来进行文件共享和数据库记录的传输。到目前为止，在计算机互连这个领域，以太网仍然是最活跃的技术，但已经不再局限于这个领域，在其他一些领域，以太网也大显身手，表现不俗。下面是以太网的主要应用领域：

- 计算机互连：这是以太网技术的主要目标，也是最成熟的应用范围。最开始的时候，许多计算机通过同轴电缆连接起来，互相访问共享的目录，或访问在同一个物理网段上的文件服务器，各个计算机（不论是服务器还是客户机）在网络上的地位相同。随着应用的发展，这种平等的结构逐渐不适应实际的需要，因为网络上的大部分流量都是客户机跟服务器之间的，这种流量模型必然在服务器上形成瓶径。当全双工以太网和以太网交换机引入以太网之后，这种情况有所改变，取代的是把服务器连接到以太网交换机的一个高速端口（100M）上，把其他客户机连接到以太网交换机的低速端口上，这样就暂缓了瓶径的形成。现代的操作系统提供分布式服务和数据仓库服务，基于这些操作系统的服务器除了跟客户机通信之外，还要跟其他服务器交换大量的信息进行数据的同步，这样传统的 100M 快速以太网就不能满足要求了，于是 1000M 以太网应运而生。
- 高速网络设备之间互连：随着 INTERNET 的不断发展，一些传统的网络设备，比如路由器，之间的带宽已经不能满足要求，需要更高更有效率的互连技术来连接这些网络设备构成 INTERNET 的骨干，1000M 以太网成了首选的技术。传统的 100M 也可以应用在这些场合，因为这些 100M 的快速以太网链路可以经过聚合，形成快速以太网通道，速度可以达到 100M——1000M 的范围。

- 城域网中用户接入的手段：用户通过以太网技术接入城域网，实现上网，文件下载，视频点播等业务，已经变得越来越流行。之所以用以太网作为城域网的接入手段，是因为现在的计算机都支持以太网卡，这样对用户来说，不用更改任何软件和硬件配置就可以正常上网。

可以看出，以太网技术已经覆盖了网络的方方面面，从骨干网到接入网，从计算机网络到工业应用，无处不见以太网的影子。

1.4 以太网物理层及相关设备

根据 ISO 的 OSI 七层参考模型，物理层规定了两个设备之间的物理接口，以及该接口的电气特性，规程特性，机械特性等内容，以太网的物理层也不外乎这些内容，它主要的功能是提供一种物理层面的标准，各个厂家只要按照这个标准生产网络设备就可以进行互通。下面从介绍这些物理层标准开始，来分析一下以太网的物理层基础结构。

1.4.1 物理层系列标准

从以太网诞生到目前为止，成熟应用的以太网物理层标准主要有以下几种：

- 10BASE2
- 10BASE5
- 10BASE-T
- 100BASE-TX
- 100BASE-T2
- 100BASE-T4
- 100BASE-FX
- 1000BASE-SX
- 1000BASE-LX
- 1000BASE-CX
- 1000BASE-TX

在这些标准中，前面的 10，100，1000 分别代表运行速率；中间的 BASE 指传输的信号是基带方式；后边的 2，5 分别代表最大距离；TX，T2，T4，FX，SX，LX，CX 等应用于双绞线以太网和光纤以太网，含义如下：

- 100BASE-TX：运行在两对五类双绞线上的快速以太网；
- 100BASE-T4：运行在四对三类双绞线上的快速以太网；
- 100BASE-T2：运行在 2 对三类双绞线上的快速以太网；

- 100BASE-FX：运行在光纤上的快速以太网，光纤类型可以是单模也可以是多模；
- 1000BASE-SX：运行在多模光纤上的 1000M 以太网，S 指发出的光信号是长波长的形式；
- 1000BASE-LX：运行在单模光纤上的 1000M 以太网，L 指发出的光信号是短波长的形式；
- 1000BASE-CX：运行在同轴电缆上的 1000M 以太网。

在这些标准中，10BASE2，10BASE5 是同轴电缆的物理标准，现在已经基本被淘汰，10BASE-T 和 100BASE-TX 都是运行在五类双绞线上的以太网标准，所不同的是线路上信号的传输速率不同，10BASE-T 只能以 10M 的速度工作，而 100BASE-TX 则以 100M 的速度工作，其他方面没有什么两样。100BASE-T2，100BASE-T4 现在很少用，所以我们这里只选择比较有代表性的 100BASE-TX 进行叙述，其他的比如 1000M 以太网的技术在后边的章节中再进行讲述。

1.4.2 100BASE-TX物理层

100BASE-TX 是运行在两对五类双绞线上的快速以太网物理层技术，它除了规定运行的介质是五类或更高类双绞线外，还规定了设备之间的接口以及电平信号等。该标准规定设备和链路之间的接口采用 RJ-45 水晶头，电平采用 +5V 和-5V 交替的形式。

五类双绞线的 8 根线压入水晶头的 8 个线槽中，这样可以很容易的插入网络设备的网卡。

实际上，在进行数据的传输时仅仅用了五类双绞线的两对（四根）线，其中

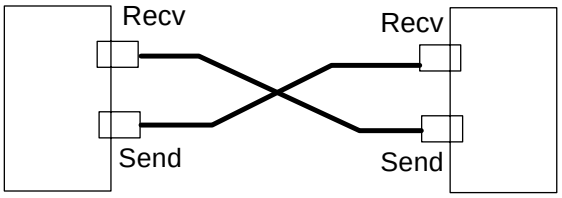


图1-1 100BASE-TX 物理层示意图

一对作为数据接收线，一对作为数据发送线，在进行数据接收和发送的时候，在一对线上传输极性相反的信号，这样可以避免互相干扰。

跟传统的同轴电缆不同的是，100BAST-TX（10BASE-T）的数据发送和数据接收使用了不同的线对，做到了分离，这样就隐含着一种全新的运做方式：

全双工方式。在这种方式下，数据可以同时接收和发送而互不干扰，这样可以大大提高效率，不过这需要中间设备的支持，现在的以太网交换机就是这样一种设备。

1.4.3 自动协商

在基于双绞线的以太网上，可以存在许多种不同的运做模式，在速度上有 10M，100M 不等，在双工模式上有全双工和半双工等，如果对每个接入网络的设备进行配置，则必然是一项很繁重的工作，而且不容易维护。于是，人们提出了自动协商技术来解决这种矛盾。

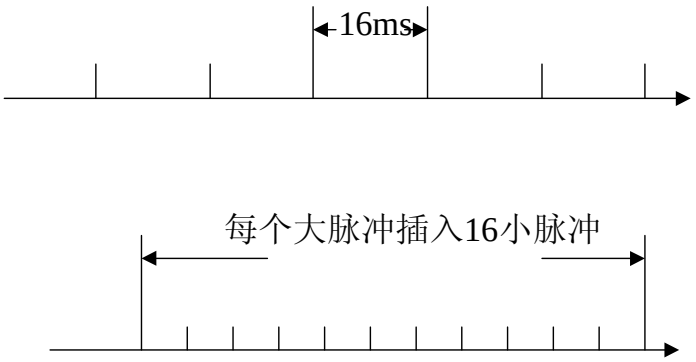


图1-2 普通链路脉冲

自动协商建立在一种底层的以太网机制上。在双绞线链路上，如果没有数据传输，链路并不是一直在空闲，而是不断的互相发送一种频率相对较低的脉冲信号（称为普通链路脉冲，NLP）（如上图所示），任何具有双绞线接口的以太网卡都应该能识别这种信号。需要注意的是，如果再插入一些（一般是 16 个）更小的脉冲（这些脉冲称为快速链路脉冲，FLP），两端设备应该也能识别。于是，我们可以使用这些快速链路脉冲来进行少量的数据传输，来达到自动协商的目的。

在设备的网卡中有一个配置寄存器，该寄存器内部保留了该网卡能够支持的工作模式，比如该网卡可以支持 100M 和 10M 模式下运行，则把相应的寄存器内容置位。在网卡加电后，如果允许自动协商，则网卡就把自己的配置寄存器内容读出来，编码后发送出去（如下图所示）。

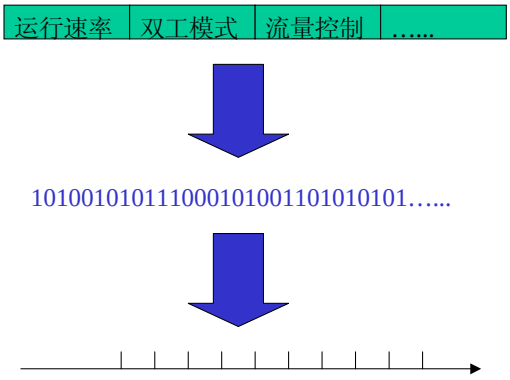


图1-3 编码发送

发送的同时，可以接收对端发送过来的自动协商数据。接收到对方发送的自动协商数据后，跟自己的配置寄存器比较，选择自己支持的且一般情况下最优的组合投入运行。比如自己支持全双工模式和 100M 的速率，对端也支持该配置，则选择的运行模式就是 100M 全双工，如果对端只支持全双工模式和 10M 的能力，则运行模式就定为全双工 10M 模式。如果两端支持的能力集合不相交，则协商不通过，两端设备不能通信。

一旦协商通过，网卡就把该链路置为激活状态，可以传输数据了，如果不能协商通过，则该链路不能使用，不能再进行数据传输。如果两端的设备有一端不支持自动协商，则支持自动协商的一端选择一种默认的方式工作，一般情况下是 10M 半双工模式。

📖 注意：

如果链路两端的设备有一端不支持自动协商，则支持自动协商的设备选择一种默认的工作方式，比如10M半双工模式运行。这时可能影响了效率，因为不支持自动协商的设备可能支持100M全双工。这时，我们可以禁止自动协商，并手工指定两端设备的运行模式，以增强效率。

1.4.4 集线器

当用双绞线把终端设备进行互连时，需要一个中间设备来进行集中，这个设备就是集线器，所谓的 HUB。HUB 一般是一个多端口的盒式设备，每个接口可以连接一个终端设备。这样多个设备（比如，12 个等）可以通过 HUB 连接在一起，组成一个星形的网络。需要注意的是，网络在物理上是星形结构的，但在 HUB 内部还是使用了共享总线的技术，采用 CSMA/CD 技术进行交互。

HUB 可以根据接口的特点进行区分，分为 I 类 HUB 和 II 类 HUB。这两类 HUB 在内部工作模式上没有区别，但因为提供的接口不同而使用于不同的场合。I 类 HUB 只提供一种类型的物理接口，比如只提供五类双绞线接口或只提供三类双绞线接口，或者只提供光纤接口等，而 II 类 HUB 则可以提供多种不同类型的接口，可以在一个 II 类 HUB 上集成五类双绞线接口和光纤接口等。实际中应用最多的是 I 类 HUB。

传统的 HUB 都是运行在半双工模式下的，但有一些应用需要在全双工模式下运行，于是人们开发了一种运行在全双工模式下的 HUB，即所谓全双工 HUB。这种 HUB 的内部结构还是一条总线，各个终端共享这条总线进行数据交互，所不同的是，全双工 HUB 在每个接口上预增加了一个缓冲区，如果总线繁忙，终端发送的数据可以暂存在缓冲区里面，等总线空闲之后，再进行传输。连接在全双工 HUB 上的终端设备工作在全双工模式下。

1.5 物理层总结

到此为止，以太网物理层的一些基础概念已经介绍完毕，在上面介绍的概念中，主要有下面这些需要引起重视（概述中的一些概念也在这里提起）：

- 各种物理层标准：10BASE2，10BASE5，10BASE-T，100BASE-TX，100BASE-FX ， 100BASE-T2 ， 100BASE-T4 ， 1000BASE-CX ，1000BASE-SX，1000BASE-LX，1000BASE-TX 等物理层标准的含义，100BAST-TX（10BASE-T）的物理层结构，包括接口特性等；
- 通过双绞线连接同类型的设备和不同类型的设备：在两种情况下的线序情况，以及为什么这样；
- 自动协商：自动协商的基本概念，实现机理，协商内容等，以及支持自动协商和不支持自动协商设备之间的交互情况；
- CSMA/CD 访问方式：运行在半双工模式下的设备采用这种方式访问链路，解释这种访问方式的过程以及碰撞避免机制；
- 全双工以太网：全双工以太网得以实现的基础，以及其优点等；
- HUB：I 类 HUB 和 II 类 HUB 的概念，全双工 HUB 的工作机理。

第2章 数据链路层

前面的部分讲述了物理层的一些基础概念，在这些概念的基础上，我们再来详细看一下数据链路层的概念。

2.1 数据链路层特点

按照 ISO 的 OSI 七层参考模型，互连的各个系统把各个网络功能分七个层次实现，各个层次之间相互独立，互不干扰。这样就可以实现最大限度的开放和灵活性，设备厂家只要按照层次之间的接口生产设备，就可以做到互通。因此，这个七层模型是高效权威的，而且目前大多数网络技术都是参照这个模型进行设计和开发的。

但在以太网体系结构中，七层模型中层次之间互相独立的规则就不适用了，因为开始的时候，以太网采用了一种共享介质的方式来进行数据通信，而不是传统的全双工通信，随着设备的发展，以太网中又引入了全双工模式的通信，在这样两种通信模式并存的情况下，在进行层次间的严格划分就不容易了。

在前面讲述的内容中曾经提到，针对不同的双工模式，提供不同的介质访问方法，在半双工模式下采用的是 CSMA/CD 的访问方式，而在全双工模式下则可以直接进行收发，不用预先判断链路的忙闲状态。这里需要注意的是，在以太网中，半双工和全双工是物理层的概念，而针对物理层的双工模式提供不同访问方式则是数据链路层的概念，这样就形成了以太网的一个重要特点：数据链路层和物理层是相关的。理解了这个概念，以后的学习中就相对明白了。

2.2 以太网链路层的分层结构

在上面的介绍中知道，以太网的物理层和数据链路层是相关的，针对物理层的不同工作模式（全双工和半双工），需要提供特定的数据链路层来访问。这样导致了数据链路层和物理层有很大的相关性，给设计和应用带来了一些不便。

为了避免这种不便，一些组织和厂家提出了另外一种方式，就是把数据链路层再进行分层，分为逻辑链路控制子层（LLC）和媒体访问控制子层（MAC）。这样不同的物理层对应不同的 MAC 子层，LLC 子层则可以完全独立。这样从一定程度上提高了独立性，方便了实现。

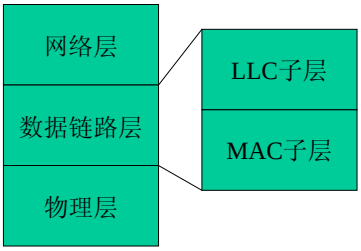


图2-4 数据链路层

2.3 MAC子层

MAC 子层是物理层相关的，也就是说，不同的物理层有不同的 MAC 子层来进行访问，比如物理层是工作在半双工模式的双绞线，则相应的 MAC 子层为半双工 MAC，如果物理层是令牌环，则有令牌环 MAC 来进行访问。在以太网中，主要存在两种 MAC：半双工 MAC 和全双工 MAC，分别针对物理层运行模式是半双工和全双工时提供访问。需要注意的，这两种 MAC 都是集成在网卡中的，网卡初始化的时候一般进行自动协商，根据自动协商的结果决定运行模式，然后根据运行模式选择相应的访问 MAC。

2.3.1 半双工MAC子层

当物理层运行在半双工模式下时，数据链路层使用半双工 MAC 进行访问。半双工 MAC 跟物理层之间至少存在六种信号进行通信，如下图所示：

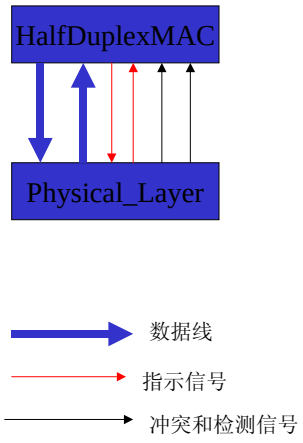


图2-5 MAC 对物理层的访问

具体工作过程是这样的：当链路层有数据要发送的时候，首先检查链路空闲信号（物理层通过该信号来报告给数据链路层链路是否空闲），如果链路空闲，则通过指示信号给物理层一个指示，告诉物理层要发送数据，然后把数据一个字节一个字节的送到数据线上（数据线是一组 8 位的信号线）。这时

候如果物理层检测到了冲突（即有另外一个终端同时发送数据），则通过冲突检测指示信号给 MAC 子层一个指示，告诉 MAC 子层线路上发生了碰撞。这时候，MAC 子层马上停止数据的发送，并发送一连串干扰信号，达到让网络上所有的设备都知道产生冲突的目的。等待一段时间后，MAC 层再次检查链路空闲信号，进行数据发送。

如果物理层接收到了数据，则通过数据接收指示来告诉 MAC 子层自己接收到了数据，然后把数据放到接收数据线上（跟发送数据线一样，也是 8 位组信号线），传给 MAC 子层。

从上面的分析中，可以看出这六种信号是：

- 数据发送线，一个 8 位组信号线；
- 数据接收线，一个 8 位组信号线；
- 链路空闲信号：一个指示位，指示链路是否空闲；
- 冲突检测信号：一个指示位，物理层使用该信号向 MAC 子层报告冲突发生；
- 发送数据指示：MAC 子层要传输数据时通过该信号告诉物理层；
- 接收数据指示：物理层接收到数据后通过该信号告诉 MAC 子层。

🕮 提示：

MAC子层一般是在一块ASIC芯片上实现，该ASIC芯片通过引脚跟物理层连接，每个引脚为一个信号线，这些信号线组成了一些功能信号，上面提到的这六种信号就是功能信号中的一部分。

2.3.2 全双工MAC子层

全双工 MAC 子层相对半双工 MAC 子层简单，因为它不需要检测链路的空闲与忙的状态，所以就去除了上面的链路空闲信号和冲突检测信号。其工作过程如下：

当 MAC 子层有数据要发送的时候，通过数据发送指示告诉物理层，然后把数据一个字节一个字节的通过数据发送线发送出去。如果物理层检测到了数据到达，则通过接收指示信号告诉链路层，自己接收到了数据，然后通过接收数据线把数据传到 MAC 子层。

🕮 提示：

数据链路层跟物理层之间交换数据的时候（不是控制信号），是按字节进行的，这从接收数据和发送数据的信号线根数可以看出来。

2.3.3 MAC地址和数据帧的收发

除了完成物理链路的访问以外，MAC 子层还负责完成下列任务：

- 链路级的站点标识：在数据链路层识别网络上的各个站点。也就是说，在该层次保留了一个站点地址（就是所谓的 MAC 地址），来标识网络上的唯一一个站点；
- 链路级的数据传输：从上层（LLC 子层）接收数据，附加上 MAC 地址和控制信息后把数据发送到物理链路上；在这个过程中搀杂了校验等功能。

为了进行站点标识，在 MAC 子层保留了一个唯一的站点 MAC 地址，来区分该站点。MAC 地址是一个 48 比特的数字，分为下面几种类别：

- 物理 MAC 地址：这种类型的 MAC 地址唯一的标识了以太网上的一个终端（比如网卡等），实际上这样的地址是固化在硬件里面的；
- 广播 MAC 地址：这是一个通用的 MAC 地址，用来表示网络上的所有终端设备；
- 组播 MAC 地址：这是一个逻辑的 MAC 地址，来代表网络上的一组终端。它的特点是最左边一个字节的第一比特（低比特位）为 1。

上层要发送数据的时候，把数据提交给 MAC 子层，MAC 子层有自己的缓冲区，把上层提交给自己的数据进行缓存，然后增加上目的 MAC 地址和自己的 MAC 地址（源 MAC 地址），计算出数据帧的长度，形成下列格式的数据包：

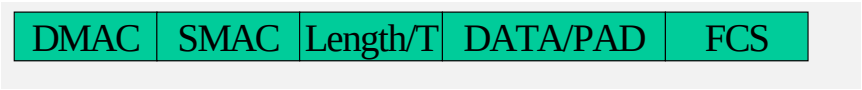


图2-6 以太网帧格式

在这个图中，DMAC 代表目的终端的 MAC 地址，SMAC 代表源 MAC 地址，而 LENGTH/TYPE 字段则根据值的不同有不同的含义：当 LENGHT/TYPE > 1500 时，代表该数据帧的类型（比如上层协议类型），当 LENGTH/TYPE < 1500 时，代表该数据帧的长度。DATA/PAD 则是具体的数据，因为以太网数据帧的最小长度必须不小于 64 字节（根据半双工模式下最大距离计算获得

的），所以如果数据长度加上帧头不足 64 字节，需要在数据部分增加填充内容。FCS 则是帧校验字段，来判断该数据帧是否出错。

上面介绍了数据的发送过程，下面说一下数据接收过程：

在计算机的网卡中维护一张接收地址列表，每当计算机网卡接收到一个数据帧之后，就把数据帧的目的 MAC 地址提取出来，跟列表中的条目进行比较，只要有一项匹配，则接收该数据帧，若无任何匹配的项目，则丢弃该数据帧。在这张接收地址列表中至少有下面两项：

1. 计算机网卡的 MAC 地址：该地址固化在网卡的 ROM 里面；

2. 广播 MAC 地址：该地址代表网络上的所有主机。

如果上层应用程序加入一个组播组，则该应用程序会通知网络层，然后网络层通知数据链路层，数据链路层根据应用程序加入的组播组形成一个组播 MAC 地址，并把该组播 MAC 地址加入接收地址列表，这样当有针对该组的数据帧的时候，MAC 子层就接收该数据帧并向上层发送。这个组播过程在以后会详细讲解。

2.4 LLC子层

在上面的介绍中提到了 MAC 子层形成的一个帧结构，其中有一个字段是 LENGTH/TYPE。这个字段的长度是 2 字节，根据取值的范围有不同的含义，在小于或等于 1500 的情况下，该值代表数据帧数据部分的长度，但当大于 1500 的时候，则代表该帧的数据部分的类型，比如该数据帧是哪个上层协议（比如 IP，IPX，NETBEUI 等）的数据单元等。

当 LENGTH/TYPE 取值大于 1500 的时候，MAC 子层可以根据 LENGTY/TYPE 的值直接把数据帧提交给上层协议，这时候就没有必要实现 LLC 子层。这种结构便是目前比较流行的 ETHERNET_II，大部分计算机都支持这种结构。注意，这种结构下数据链路层可以不实现 LLC 子层，而仅仅包含一个 MAC 子层。

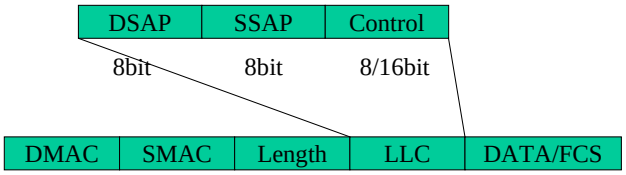
根据 LENGTH/TYPE 字段的取值，来把接收到的数据帧提交给上层协议模块，是这样进行的：每个上层协议都提供了一个回调函数，这个回调函数在数据链路层是可见的而且可以调用的，这样当数据链路层接收到一个数据帧之后，根据数据帧里的 LENGTH/TYPE 字段的取值来判断相应的协议模块，然后调用相应协议的回调函数（把数据帧的数据部分作为参数），该回调函数执行的结果就是把数据帧的数据部分挂到上层协议的接收队列中，然后给上层协议发送一个消息，告诉上层协议有一个数据包到来，然后返回，其他的事情就有上层协议来做了。

当 LENGTH/TYPE 小于或等于 1500 的情况，这种类型就是所谓的 ETHERNET_SNAP，是 802.3 委员会制定的标准，虽然目前应用不是很广泛，但是是一种很有特色的标准。

ETHERNET_SNAP 除了定义传统的链路层服务之外，还增加了一些其他有用的特性，比如定义了下面三种类型的点到点传输服务：

- 无连接的数据包传输服务：目前的以太网实现就是这种服务；
- 面向连接的可靠的数据传输服务：预先建立连接再传输数据，数据在传输过程中可靠性得到保证；
- 无连接的带确认的数据传输服务：该类型的数据传输服务不需要建立连接，但它在数据的传输中增加了确认机制，使可靠性大大增加。

这些服务都是在 LLC 子层中实现的，下面是 LLC 子层的帧格式：



IEEE802.3帧结构

图2-7 LLC 子层格式

可以看出，该数据帧的结构在 MAC 子层上是保持统一的，但 MAC 子层数据帧的 LENGTH/TYPE 字段现在已经完全成了 LENGTH，指示 MAC 数据帧数据部分的长度，然后在数据部分增加了一个 LLC 头，这个头由 DSAP（目的服务访问点），SSAP（源服务访问点）和控制字段组成。上面讲述的三种服务就是通过这三个字段完成的。下面通过一个例子来说明 SSAP 和 DSAP 的应用，假设终端系统 A 和终端系统 B 要使用面向连接的可靠的数据传输服务来进行一次数据传输，这时候会发生如下过程：

1. 终端系统 A 给终端系统 B 发送一个数据帧，请求建立一个面向连接的可靠连接；
2. 终端系统 B 接收到以后，判断自己的资源是否够用（即是否建立了太多的连接），如果够用，则返回一个确认，该确认中包含了识别该连接的 SAP 值；
3. 终端系统 A 接收到回应后，知道终端系统 B 已经在本地建立了跟自己的连接，于是终端系统 A 也开辟一个 SAP 值，来表示该连接，并发一个确认给终端系统 B，于是该连接建立；

4. 终端系统 A 的 LLC 子层把自己要传送的数据进行封装（封装成 LLC 子层的帧格式），其中 DSAP 字节填写的是终端 B 返回的 SAP，SSAP 字节填写的是自己开辟的 SAP，然后发给 MAC 子层，MAC 子层加上 MAC 地址和 LENGTH 字段之后，发送到数据链路上；

5. 终端系统 B 的 MAC 子层接收到该数据帧之后，提交给 LLC 子层，LLC 子层根据 DSAP 字段判断出该数据帧属于的连接，然后根据该连接的类型（可靠的连接还是无连接，或者带确认的无连接）进行相应的校验和确认，只有通过这些校验和确认后，才向更上层发送；

6. 数据传输完毕之后，终端系统 A 给终端系统 B 发送一个数据帧来告诉终端系统 B 拆除连接，于是通信结束。

这些功能都是在 LLC 子层实现的，通过这个例子，我们应该对 LLC 的功能有了一个了解，也应该把 LLC 子层和 MAC 子层的界面分清楚，当然，这些 LLC 子层的功能对网络层都是透明的。问题：就是 LLC 子层根据什么内容来把数据帧提交给网络层？在 ETHERNET_II 中，是通过 LENGTH/TYPE 字段来区分上层协议的，其实，在 LLC 子层的帧结构的数据部分中，也包含一个 TYPE 字段，该字段在上面的图中没有画出来，LLC 子层就是根据这个字段来把数据帧发送给上层协议的。

2.5 以太网数据链路层总结

到此为止，数据链路层的一些标准和协议已经解释完了，下面列出该部分的一些主要内容，这些内容是最基础的：

1. 跟物理层相关是以太网的数据链路层的最大特点；
2. 按照 802.3 标准，数据链路层分为两个子层：LLC 子层和 MAC 子层；
3. 针对不同的物理拓扑结构和双工工作模式，提供不同的 MAC 子层来进行访问；
4. 基于双绞线的半双工 MAC 子层；
5. 基于双绞线的全双工 MAC 子层；
6. 三种类型的 MAC 地址和 MAC 数据帧的收发过程；
7. LLC 子层提供的三种服务及其建立方式；
8. LLC 子层帧结构，以及帧结构中 SSAP 和 DSAP 字段的含义。

第3章 以太网交换机

在以太网物理层内容的介绍中，介绍了一种网络设备—集线器（HUB），该设备工作在物理层，同样，在数据链路层也有一种网络设备，就是本章介绍的以太网交换机。

在以太网的发展历史上曾经出现了一种称为网桥的设备，该设备的功能是连接两个物理拓扑不同的网络，比如以太网和令牌环网，网桥在这两个网络中完成地址翻译，通信中继等功能，使得网络层看来，物理上不同的两个网络是一个逻辑的网络。其实网桥也是工作在数据链路层上的，随着以太网技术的发展，连接两个异种网络的机会越来越少，于是，网桥正被以太网交换机逐渐代替。

下面从以太网交换机的体系结构出发，来分析一下以太网交换机的工作过程，性能特点，以及一些提高性能的应用，比如快速以太网通道等。

3.1 以太网交换机体系结构

从外观上看，以太网交换机跟 HUB 差不多，但端口的数目可能比 HUB 要多（一般情况下是 24 个或更多）。但在内部结构上却比 HUB 复杂得多，HUB 内部实际上是一条共享的总线，各个端口共享该总线进行 CSMA/CD 方式的通信，但以太网交换机内部可能也是一条总线，但该总线带宽要比 HUB 内部的总线高得多，足以让全部端口互相同时通信而没有阻塞。性能更高的交换机内部可能是一个交换网络，该网络完成任意端口之间的两两交换。

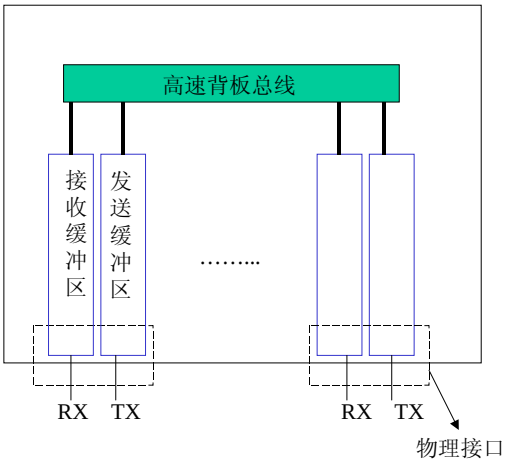


图3-8 以太网交换机的内部结构示意图

上图是一个以太网交换机的内部结构示意图，由图可以看出以太网交换机使用一条高速背板总线把各个端口连接起来。实际上，这个背板总线可能是一个高性能的数字交叉网络。注意的是，各个端口针对接收线路和发送线路，各有一个缓冲队列，当数据从终端设备发往交换机的时候，发出的数据暂存在交换机的接收队列中，然后进行下一步处理。如果交换机要把接收的数据发送给某一终端，这时候，交换机把要发送的数据发往该接收终端所在端口的发送队列，然后再发送到接收终端，如果终端忙，则一直存储在发送队列中。

其实，对每个接口的发送队列结构进行更改可以实现服务质量功能。比如，我们为每个接口设计不止一个发送队列，假设设置三个，则可以对这三个队列进行优先级划分，分成低，中，高三个优先级，然后根据数据帧的优先级字段（在以后介绍 VLAN 的时候将讲述），把数据帧放到相应的优先级队列中。在传输的时候，可以优先传输优先级高的队列，等高优先级队列内没有数据了，再传送优先级低的队列，依次类推。也可以实现一些其他的调度策略，比如 WFQ（基于优先级的加权公平队列）等调度技术。

交换机跟 HUB 的最大区别就是能做到接口到接口的转发。比如接收到一个数据帧以后，交换机会根据数据帧头中的目的 MAC 地址，发送到适当的端口，而 HUB 则不然，它把接收到的数据帧向所有端口转发。交换机之所以能做到根据 MAC 地址进行选择端口，完全依赖内部的一个重要的数据结构：CAM 表。交换机接收到一个数据帧，依靠该数据帧的目的 MAC 地址来查找 CAM 表，查找的结果是一个或一组端口，根据查找的结构，把数据包送到相应端口的发送队列。

CAM 表包含下面几项内容：

- MAC 地址
- 一个或一组端口号
- 如果交换机上划分了 VLAN，还包括 VLAN ID

交换机根据接收到的数据帧的目的 MAC 地址，来查找该表格，根据找到的端口号，把数据帧发送出去。注意，上面一个 MAC 地址可能对应多个端口，这样的 MAC 地址一般是组播 MAC 地址，其中每个端口上连接一个组播组的成员，这些内容在组播部分中将详细讲述，在这里我们要有个印象。

3.2 以太网交换机工作过程

上面提到了一个重要的数据结构 CAM 表，该表包含了交换机用来转发数据帧所涉及到的一些信息，对于交换机怎样依据这个表格进行数据帧的交换，上面部分已经讲明白了。但这又产生一个问题：这个 CAM 表是怎样生成的呢？其实，这个表格可以通过两种途径生成：

1．手工加入：通过配置命令的形式告诉交换机 MAC 地址和端口的对应；

2．交换机动态学习获得：交换机通过查看接收的每个数据帧来学习该表。

手工生成该表很简单，不过配置起来会占用大量的时间，所以通常情况下是交换机自动获得的。下面我们分析一下交换机是怎么获得这个 CAM 表的。首先提出交换机转发数据帧的基本规则：

- 交换机查 CAM 表，如果查找到结果，根据查找结果进行转发；
- 如果交换机在 CAM 表中查找不到结果，则根据配置进行处理，通常情况下是向所有的端口发送该数据帧，在发送数据帧的同时，学习到一条 CAM 表项。

开始的时候，交换机的 CAM 表是空的，当交换机接收到第一个数据帧的时候，查找 CAM 表失败，于是向所有端口转发该数据帧，在转发数据帧的同时，交换机把接收到的数据帧的源 MAC 地址和接收端口进行关联，形成一项记录，填写到 CAM 表中，这个过程就是学习的过程。

学习过程持续一段时间之后，交换机基本上把所有端口跟相应端口下终端设备的 MAC 地址都学习到了，于是进入稳定的转发状态，这时候，对于接收到的数据帧，总能在 CAM 表中查找到一个结果，于是数据帧的发送是点对点的，达到了理想的状态。

交换机还为每个 CAM 表项提供了一个定时器，该定时器从一个初始值开始递减，每当使用了一次该表项（接收到了一个数据帧，查找 CAM 表后用该项转发），定时器被重新设置。如果长时间没有使用该 CAM 表的转发项，则定时器递减到零，于是该 CAM 表项被删除。

第4章 VLAN基本概念

以太网交换机一般有十几个或几十个端口，默认情况下，连接到这些端口的计算机能够无阻隔的进行二层通信。但有些情况下，人们希望某些端口上的计算机不能被其他端口上的计算机访问，这时候采用这种默认的工作方式就不行了，我们需要引入这样一种功能：可以把交换机上任意数目的端口进行组合，这些组合的端口成为一个封闭的系统，连接到该封闭系统的计算机可以通信，但跟不在该端口集合内的其他端口上的计算机，则无法进行二层通信。这个概念便是 VLAN（虚拟局域网）。

4.1 VLAN的划分方式

按照上面的概念，VLAN 是交换机上的一个集合，该集合的元素就是端口。我们用一个整数来表示该集合，这样当在交换机上创建一个集合后，就面临一个问题：怎样为该集合确定其中的元素（端口）？下面是确定元素的最重要的几种方式：

4.1.1 基于端口的VLAN

最简单，也是最直接的方式就是手工指定，也就是说，一旦在交换机上创建一个 VLAN，我们可以手工指定该 VLAN 包含哪些端口。在该方式下，我们只要在交换机上进行一些简单的配置就可以了。这种方式是最直接的，也是最容易理解的。

4.1.2 基于MAC地址的VLAN

在基于端口的 VLAN 方式下，我们在交换机上进行了设置，来决定 VLAN 包含那些端口。但有些情况下，我们希望把终端系统进行分类，使它们属于指定的 VLAN。这时候，我们可以手工建立终端系统的标识跟 VLAN 之间的关系。在以太网上，MAC 地址可以唯一的标识一个终端系统，于是，我们就建立 MAC 地址跟 VLAN 之间的对应关系。

交换机仅仅根据这个 MAC 地址跟 VLAN 之间的对应关系，不能创建 VLAN 与端口的对应关系，我们于是联想到交换机内部的另外一个关系：MAC 地址与端口之间的关系（也就是 CAM 表，交换机根据该表格进行数据帧的转发）。根据这样两个关系，交换机可以创建 VLAN 与端口号之间的对应关系了，具体过程如下：

1. 交换机把手工创建的 MAC 地址跟 VLAN 号之间的对应关系下载到本地；
2. 从该对应关系中读出一行，如果该行对应的 VLAN 不存在，则创建该 VLAN，然后以 MAC 地址为索引依据，到 CAM 表中查找对应的端口号，把找到的端口号加入刚刚创建的 VLAN；
3. 若读出的该行所包含的 VLAN 已经存在，则仅仅依据 MAC 地址查询 CAM 表，把找到的端口号加入已经存在的 VLAN 里面；
4. 重复这个过程，直到该对应关系扫描完毕。

我们可以看出，上面的过程存在一个问题：交换机的 CAM 表是通过学习获得的，如果交换机在根据 MAC 地址查找相应的 CAM 表时，该 MAC 地址不存在怎么办？这并不说明该 MAC 地址对应的计算机不存在，有可能交换机还没有学习到该计算机的 MAC 地址。这种情况下，我们可以采用下面的步骤：

1. 在交换机上做配置，告诉交换机哪些端口是基于 MAC 地址的 VLAN 的。也就是说，这些交换机端口上如果连接了一个终端，这个终端的 MAC 地址出现在手工创建的 MAC 地址跟 VLAN 对应关系表里，则该端口就加入相应的 VLAN。其他没有配置的端口即使连接了满足条件的终端（也就是说，该终端的 MAC 地址出现在手工创建的那张表里），也不会被加入相应的 VLAN。这样做，主要是效率上的考虑；
2. 交换机每学习到一个 MAC 地址，就用该 MAC 地址为索引到 MAC 地址跟 VLAN 对应关系里查找相应的 VLAN 号，找到相应的 VLAN 后，创建相应的 VLAN，把学习到该 MAC 地址的端口加入刚刚创建的 VLAN。

这样的过程一直延续，最终的结果是创建了 VLAN ID 和 MAC 地址对应列表中出现的所有 VLAN。

4.2 交换机间链路

在上面的介绍中，我们可以看出，所有创建的 VLAN 都是集中在一个交换机上的，但实际中往往有这样的情况，就是一个物理的 VLAN，可能跨越了多个交换机。在一个交换机的情况下，一个 VLAN 端口接收到的数据可以根据交换机内部的一张 VLAN 和端口对应表来确定该 VLAN 所有的端口，因而一个 VLAN 的数据在同一个交换机上不可能被错误转发到另外一个 VLAN 当中。但是跨越交换机的时候就不是这样了，假设有两个 VLAN 和两个交换机，分别记做 VLANA 和 VLANB，SWITCHA 和 SWITCHB，其中 VLANA 和 VLANB 分别跨越了两个交换机，即 SWITCHA 和 SWITCHB 上既有 VLANA 的端口，也有 VLANB 的端口。在这种情况下，假如 SWITCHA 上 VLANA 的一个端口接收到了一个广播数据帧，SWITCHA 除了往自己上面所有属于 VLANA 的端口广播该数据帧以外，还必须通过 SWITCHA 和 SWITCHB 之间的一条链路

来传播该广播数据帧。如果 SWITCHB 接收到了该广播数据帧，SWITCHB 就不知道把该数据帧发往哪个 VLAN 的端口，因为该广播数据帧中不包含任何 VLAN 有关信息。

在这种情况下，我们可以在发给另外一个交换机的数据帧上附加 VLAN 信息来区分数据帧所属的 VLAN，这便是我们下面介绍的 802.1Q（简称 1Q）帧格式。如果两个交换机之间发送的以太网帧含有 VLAN 信息，那么相应的链路就称为 Trunk 链路，有时也称为 TAG 链路。

4.2.1 802.1Q帧格式

传统的以太网数据帧格式是不包含 VLAN 信息的，无法用这种传统的以太网数据帧来传送 VLAN 信息，我们要想让跨越交换机的 VLAN 能正常工作，必须重新提出一种帧格式，该帧格式与传统以太网帧格式不同的是，包含了 VLAN 信息，这便是有名的 802.1Q 帧格式。

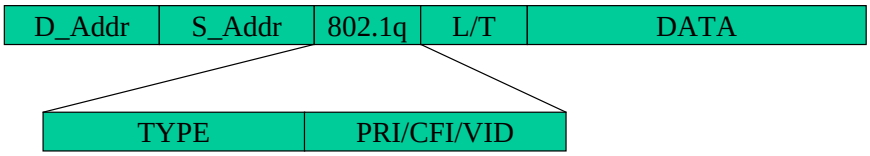


图4-9 802.1Q 帧格式

可以看出，该帧格式跟传统以太网帧格式不同的是，在传统的以太网帧格式的类型/长度字段前面，附加了一个 4 字节的额外部分，称为 1Q 标记。标记字段分为四部分：

1. TYPE：这是一个两字节长度的字段，来指出该数据帧类型，目前来说都是 0X8100。这样做的目的是跟传统的以太网数据帧兼容。当不能识别带 VLAN 标记帧的设备接收到该数据帧以后，检查类型字段，发现是一个陌生的值，于是简单丢弃即可。

2. PRI：这是一个三比特的数据字段，该字段用来表示数据帧的优先级。一共三个比特可以表示 8 种优先级，利用该字段可以提供一定的服务质量要求。一般情况下，交换机的接口提供几个发送队列，这些队列有不同的发送优先级，在把一个数据帧从该接口发送出去的时候，检查该数据帧的 PRI 字段，根据取值把该数据帧放入相应的队列中，对优先级高的帧，放到优先级高的队列中，这样可以得到优先传输服务。

3. CFI：这是一个比特的字段，该字段用在一些环形结构的物理介质网络中，比如令牌环，FDDI 等。

4. VID：这就是 1Q 数据帧的核心部分，即 VLAN ID，用来表示该数据帧所属的 VLAN，该字段是一个 12 长度的字段，这样总共可以表示 4096 个 VLAN。

当引入这种带 VLAN 标记的数据帧以后，交换机的端口就有了类别了：有些端口能够且仅仅能够识别这种带 VLAN 标记的数据帧，我们把这种端口称为 TAG 端口（标记端口），有的端口不能识别这种带标记的数据帧，我们称为非 TAG 端口，还有一些端口，不仅能识别带标记的数据帧，而且能识别不带标记的数据帧，这样的端口我们称为混合端口。连接两个交换机的端口一般是 TAG 端口或混合端口。

下面我们看一看数据包在不同类型端口之间的转发方式。

4.2.2 数据帧在不同类型端口之间的转发

我们把交换机的端口划分为 TAG 端口，非 TAG 端口和混合端口，下面我们看一下交换机在不同端口类型中转发数据帧的时候，是怎样处理的。

1. 非 TAG 端口和非 TAG 端口之间：在这种方式下转发方式最简单，交换机只根据内部的 CAM 表找到数据帧的出口，把该数据帧复制到出口的缓冲队列中即可；

2. 非 TAG 端口和 TAG 端口之间：该方式下，交换机首先判断接收到数据帧的端口所属的 VLAN 号，然后根据该 VLAN 号和物理链路类型，以及数据帧的优先级等形成一个 1Q 标记，把该标记插到接收到的数据帧的类型/长度字段前面，然后提交给 TAG 端口即可；

3. TAG 端口和 TAG 端口之间：该情况跟非 TAGD 端口和非 TAG 端口之间的转发类似，交换机只把数据帧无改变的向另外一个 TAG 端口转发即可；

4. 非 TAG 端口和混合端口之间：该情况跟非 TAG 端口和 TAG 端口之间的转发过程类似；

5. 混合端口和 TAG 端口之间：该情况最为复杂，分两种情况讨论：如果接收的数据帧本身是个标记帧，则仅仅把该标记帧向 TAG 端口复制即可；如果接收到的数据帧不是标记帧，而是一般的以太网数据帧，这时候，需要给该数据帧打上一个 VLAN 标记再向 TAG 链路转发。交换机怎样确定这个 VLAN 标记呢？原来，混合端口有一个默认的 VLAN，如果接收到的数据帧不包含 TAG，则交换机根据该端口的默认 VLAN 形成一个标记，并插入到数据帧里面发送出去。所以，在配置混合端口的时候，一般情况下指定一个默认 VLAN，如果不指定，系统可能会自己指定一个 VLAN，一般是 VLAN1。

4.2.3 VLAN在交换机上的配置

有了上述 VLAN 的基础概念之后，我们看一下怎样在一个交换机上实施 VLAN，即配置 VLAN。

首先，也是最基本的一步，就是在交换机上创建 VLAN。默认情况下，交换机上只存在一个 VLAN，即 VLAN1，所有端口都属于该 VLAN。创建 VLAN 的时候，需要给出该 VLAN 对应的 VLAN 号，根据 802.1Q 帧格式，VLAN 号是有一定范围的，最大范围是 2 到 4095，但有些厂家的特殊实现可能耗费掉了一些 VLAN，故实际使用的 VLAN 可能还要少；

其次，就是把交换机的端口加入 VLAN。创建 VLAN 之后，该 VLAN 不包含任何成员，只有通过某种方式把端口加入 VLAN 当中，VLAN 才包含了成员。这项工作可以通过许多种途径实现，比如，我们上面讲到的基于端口的静态 VLAN 和基于 MAC 地址的 VLAN 等。我们以基于端口的静态 VLAN 为例，只需要把某个端口指定为所属的 VLAN 即可；

如果 VLAN 跨越了交换机，必须配置交换机之间的连接，使之能传送特定 VLAN 的数据。默认情况下，交换机的端口都是非 TAG 的，我们可以改变这种默认类型（通过配置命令），一般情况下，我们把连接两个交换机的端口配置为 TAG 端口，而其他连接终端或 HUB 的端口配置为非 TAG 端口。

完成这些配置后，交换机就可以按照需要的方式工作了。

4.3 以太网的QoS保证

目前业务越来越多，不仅仅局限于数据业务，还有一些其他相关的业务，比如 IP 电话，电视会议等。这些业务在传输的时候有一个很大的特点，就是实时性。传统的以太网数据在以太网交换机上转发都是先来先服务的，这种转发模式不适合这种实时的业务。

为了解决这个问题，我们自然而然的想到了 802.1Q 帧结构中的 PRI 字段。该字段有三个比特，可以表达 8 种数据帧的优先级，这样，我们可以对不同的业务赋予不同的优先级，交换机在转发数据的时候，按照优先级来进行转发，先为高优先级数据帧服务，然后才为低优先级数据帧服务。

在实际中是这样实现的，在交换机的每个端口上设计若干个发送队列（队列的个数越多，提供的服务等级越多），每个队列有一个对应的优先级，在发送的时候先发送高优先级的队列，再发送低优先级的队列。来一个数据帧的时候，交换机会根据 PRI 字段来插入相应的队列，这是一种典型的区分服务模型。

这样就面临一个问题，就是 PRI 字段可以表示 8 种优先级，而交换机的端口发送队列可能不到 8 个，怎样把这 8 个优先级影射到队列中？这样可以采用分段的方式，比如，有两个队列，这时候可以把 0——3 优先级影射到队列 1，而把 4——7 优先级影射到队列 2。当有三个队列的时候，可以把 0——2 影射到队列 1，3——4 影射到队列 2，而 5——7 影射到队列 3。

还有一个问题，就是终端设备一般是不能识别标记帧的，这样就不能自己给自己发送出去的帧打标记，因而不能识别 PRI 字段。这时候可以让交换机处理，通过配置命令，告诉交换机从每个端口接收到的数据帧应该赋予多少的优先级。

第5章 千兆以太网

5.1 千兆以太网基本概念

5.1.1 8B10B编码

在传统的以太网传输技术中，数据的传输是以 8 位组的形式进行的，也就是说，在数据链路层把 8 位数据提交到物理层以后，物理层经过适当的变换后发送到链路上传输，但变换的结果还是 8 比特。在光纤千兆以太网上，则不是这样。在数据链路层把 8 比特的数据提交给物理层的时候，物理层把这 8 比特的数据进行一个映射，变换成十比特发送出去。这个特点是光纤千兆以太网得以实现的基础。

实际的发送数据只有 8 位时，总共有 256 种组合，而物理层实际发送的数据有 10 位，总共有 1024 种组合。于是，我们可以把这 1024 种组合分成三部分：

1. 数据代码组，总共有256个，对应实际发送数据的256种组合；
2. 特殊代码组，是剩余的组合中选择出的一部分，用来代表控制信号；
3. 剩余的部分临时没有使用，可以供将来使用。

数据代码组和特殊代码组的选择遵循一定的规则，比如尽量做到 0 和 1 分布均匀等，这样可以优化结果。

5.1.2 有序集

在上面的讨论中，我们知道了数据代码组和特殊代码组的含义，其中数据代码组是用来表示实际传输的数据，而特殊代码组则用来传递一些面向物理层的控制信息，比如，我们讨论的自动协商就是使用特殊代码组来封装进行的。实际上，在使用特殊代码组传递控制信息的时候，为了提高容错性，往往把几个特殊代码组组合成一个整体来传送控制信息，这个整体便叫做有序集。

对于有序集，需要注意的有两点：

1. 有序集是特殊代码组的组合，数据代码组的组合不是有序集；
2. 有序集是一个整体，它把一个或几个特殊代码组组合起来，表示唯一的意思。

下面列举几个常用的有序集来加深对有序集的理解：

- IDLE 有序集：需要注意的是，在光纤千兆以太网中没有数据传输的时候，链路也不能空闲，而是传输一些不代表任何实际意义的比特位，这样可以保持两端时钟的同步，并使链路保持在激活状态。正是 IDLE 组成了这些空闲位，链路两端的设备在接受到 IDLE 有序集后，仅仅忽略，但如果一段时间内接收不到 IDLE 有序集，则认为链路故障。
- Start_Of_Packe 有序集：象名字指示的那样，这个有序集表示一个实际数据包传送的开始。在对端设备接收到这个有序集以后，它就知道，跟在后边的将是一个实际的数据包。这个有序集起定界的作用。
- End_Of_Packet 有序集：跟 Start_Of_Packet 有序集相反，它代表实际数据包的结束。
- Configuration 有序集：这个有序集跟自动协商息息相关，自动协商的数据就是封装在这个有序集后边进行传输的。也就是说，每当物理层接收到这样一个有序集的时候，它就知道，紧跟在这个有序集后边的数据就是自动协商数据。

5.1.3 数据的封装

有了上面介绍的有序集的概念之后，我们来看看数据包是怎样通过有序集来进行封装传输的。

为了讨论的方便，我们不深入讨论数据链路层和物理层是怎样交互的，我们仅仅假设数据链路层把要传输的数据（一个一个的 8 位组）提交给物理层，并给物理层一个传输数据的信号。

物理层接收到传输数据的信号后，知道数据链路层要传输数据了，于是马上向链路上发送一个 Start_Of_Packet 有序集。此前在链路上一直在传送 IDLE 有序集。紧跟 Start_Of_Packet 有序集后，就是实际的链路层数据，不过，跟链路层提交给物理层的数据不同的是，在物理层做了一个 8B 到 10B 的影射。数据链路层要传输的数据结束以后，马上给物理层发送一个信号，告诉物理层自己已经传送完，这时，物理层便往物理链路上发送一个 End_Of_Packet 有序集，结束数据的传送。此后物理层并不是一直空闲，而是不断的往链路上发送 IDLE 有序集。

下面的图示表示了这个过程：

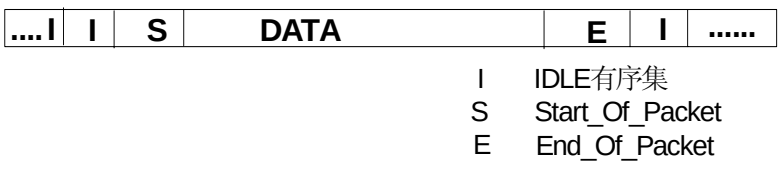


图5-10 数据的封装

上面介绍的是实际数据的封装，为了讨论的方便，我们再查看一下自动协商配置数据的封装方法。在自动协商允许的情况下，两端的系统把自己的配置寄存器内容读取出来（具体协商过程见以下的描述），封装到一个数据代码组（注意，这里是数据代码组，不是特殊代码组）里面，然后发送出去。不过，在发送的时候，并不是马上就把这个配置数据发送出去的，而是先发送一个叫做 Configuration 的有序集，配置数据紧跟在这个有序集后面。这样做的目的是让对端能正确的区分配置数据和普通数据。具体的封装图示跟上面的大体相同，不同的是把上面的 Start_Of_Packet 有序集换成了 Configuration 有序集，配置数据后边没有表示结束的有序集，因为配置数据长度是固定的，对端很容易根据长度区分。

到此为止，光纤千兆以太网的基本概念已经介绍完毕，下面来看一看自动协商协商哪些内容。

5.2 千兆以太网自动协商内容

5.2.1 双工模式

光纤千兆以太网支持半双工和全双工两种工作模式，支持半双工的目的完全是为了后向兼容。不过，目前大多数设备都支持全双工一种模式。但为了兼容和适应将来的趋势（将来会出现半双工的千兆以太网），在自动协商过程中也支持双工模式的协商。

5.2.2 流量控制

流量控制在千兆以太网中是一项重要的内容。LAN Switch 发现自己的缓冲区将要满（比如，占用了 90%）的时候，通过给连接在端口上的服务器发送一个 PAUSE 信号，告诉服务器暂停发送（否则会造成数据丢失），等 LAN Switch 把接收到的数据成功传送给客户之后，在向服务器发送一个继续发送的信号，这样一步一步便可完成任务。

这个 PAUSE 信号实际上是一种链路层信息帧，对端设备接收到以后，根据链路层头信息中的类型字段可以判定这是个控制信息帧，然后根据数据部分包含的内容进行相应的操作。比如，数据部分里含有一个停止时间的数值，假设为 Pause_Time，这个数值告诉接收该信息帧的设备在暂停 Pause_Time 时间之后再继续发送。如果在 Pause_Time 时间还没有到来的时候 LAN Switch 已经把缓冲区清空，这时可以给对端设备发送一个 Pause_Time 为零的 PAUSE 信息帧，接收设备接收到以后，马上开始发送数据

流量控制有两种工作方式：对称和不对称。所谓对称，就是参与流量控制的两端设备都有权利向对端发送 PAUSE 信息帧，达到互相控制的目的；而不对称，就是有一方处于主控地位，另外一方处于从地位，只有主控地位有权利发送 PAUSE 信息帧，而从地位的一方仅仅听从主控制方的命令。

在千兆以太网的自动协商中，对流量控制的协商主要停留在两个方面：

1. 协商要不要支持流量控制；

2. 协商主从地位。

5.2.3 运行速率

第三个要协商的内容就是传输数据的速率。在目前的情况下只有一种，即 1000M，但目前 10G 以太网正在研制之中，将来 10G 成熟之后，就会出现 1/10G 都支持的设备，这种设备跟其他设备连接的时候，就要协商运行的速率。如果连接的设备仅仅支持 1G，那么就运行在 1G 状态，如果链路两端的设备都支持 10G，则运行在 10G 速率之下。

5.3 千兆以太网自动协商过程

5.3.1 功能编码

在进行自动协商的时候，发起设备必须把自己支持的能力进行编码。通常的做法是，该设备形成一个 16 位的基页，把自己的功能编码到这个基页里面。下面是具体的基页格式：

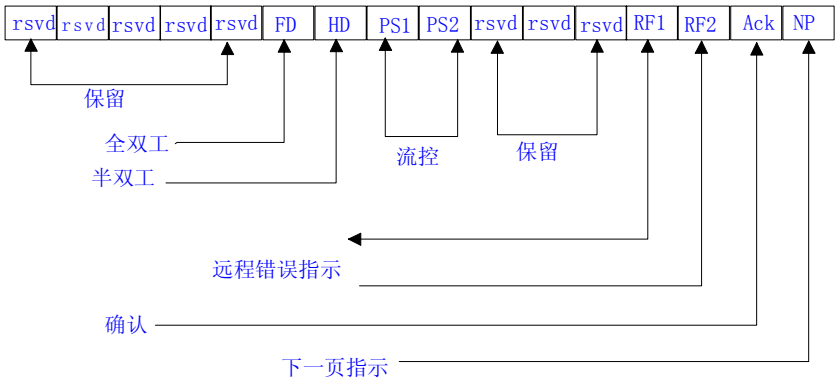


图5-11 基页格式

这个页面中各比特含义如下：

- rsvd：保留，可以供将来使用；

- FD： 全双工指示，如果设置为 1，则该设备支持全双工；
- HD： 半双工，如果设置为 1，则该设备使用半双工模式；
- PS1，PS2：流量控制指示，不同组合有不同的含义，如下：

  00：不支持流量控制；

  01：针对链路对方的不对称流量控制；

  10：对称流量控制；

  11：针对本地设备的不对称流量控制。
- RF1，RF2：远程错误位，指出远程设备的一些错误发送；
- Ack：确认，如果一台设备成功接收到了对端的基页，则发送自己的基页的时候，把该位设置为 1；
- NP：下一页指示，如果把该位设置为 1，则隐含说明该设备还要发送除基页外的其他页面，这些其他页面用来协商或配置其他功能。

自动协商的设备把自己的功能按照上面的格式编码之后，就把这个基页发送出去，具体发送过程是这样的，物理层先向链路上发送一个 Configuration 有序集，然后把配置基页发送出去。对方接收到 Configuration 有序集以后，就知道后边紧跟着的是配置基页。对端设备接收到配置基页之后，分析其中的信息，然后跟自己的配置基页比较，选择最优的配置组合。所谓最优组合，就是在这种组合下运行效率最高。比如，如果链路两端的设备都支持全双工和半双工，那么自动协商必定选择全双工模式。做出选择之后，对端设备便把自己的配置基页发送出去，并把基页中 Ack 位设置成 1，因为该设备已经接收到了配置基页。

需要注意的是，配置基页可以发送多次，直到链路两端的设备都接收到了 Ack 为 1 的配置基页。协商结束之后，链路便进入激活状态，准备传送数据了。

到此为止，关于自动协商的过程已经明确了，如果链路两端的设备都支持自动协商，按照这个过程去理解链路配置过程是有效的，出现了故障也可以根据这个过程来具体分析、排除。但目前有些设备不支持自动协商，当这些设备跟支持自动协商的设备对接的时候，便需要人工干预了。下面我们分析这样一个案例，来加深对自动协商的理解。

5.4 千兆以太网案例分析

当前有些设备不支持千兆以太网的自动协商，严格来说，这是不符合标准的。但目前来说，千兆以太网基本出现在网络的骨干位置，不是面向广大终端用户，这个缺陷便显得不重要了，因为只要人为干预一下，完全可以达到自动协商的结果。考虑这样一种配置，有一台不支持自动协商的 GSR（暂且称为

GSRA），跟一台支持自动协商的 GSR（称为 GSRB）对接。这时候，需要做如下配置：

1. 了解不支持自动协商的 GSRA 的默认工作模式，比如双工模式，是否支持流量控制等；

2. 在 GSRB 上禁止自动协商功能，然后手工配置工作模式，使之跟 GSRA 相同。

这样配置的结果跟自动协商的结果是一样的，这时候，链路检测到载波之后，马上进入激活状态，准备传送数据。

第6章 二层组播

6.1 二层组播基本概念

在前面以太网交换机的章节中，我们曾提到二层组播的概念。在本章中，我们详细介绍一下二层组播的一些基础概念。

至于为什么采用组播的形式发送数据而不采用广播的形式，其好处不言而喻，我们这里不赘述。需要说明一点的是，这里仅仅提到二层组播，还有三层组播，即所谓的 IP 组播。所谓二层组播，即数据报的转发是面向二层的，根据组播 MAC 地址来决定数据报的转发方向，而三层组播，则根据三层组播地址，即组播 IP 地址来进行数据报的转发。

6.1.1 组播MAC地址

在上面我们提到了组播 MAC 地址的概念，所谓组播 MAC 地址，是一类逻辑的 MAC 地址，该 MAC 地址代表一个组播组，所有属于该组的成员都接收以该组对应的组播 MAC 地址为目的地址的数据帧。

注意的是，组播 MAC 地址是一个逻辑的 MAC 地址，也就是说，在网络上，没有一个设备的 MAC 地址是一个组播 MAC 地址。组播 MAC 地址跟单播 MAC 地址（物理 MAC 地址）的区别是，组播 MAC 地址六个字节中，最高字节（第六字节）的最低位为 1，而单播 MAC 地址则为 0，如下图所示：

组播MAC地址，第六个字节的第一位为1

单播MAC地址，第六个字节的第一位为0

图6-12 MAC 地址

为了更进一步了解组播的概念，我们先从 MAC 层的数据帧接收过程说起。

6.1.2 MAC层数据帧的接收

在网卡的内部保留一张接收地址列表（可以理解为一个可读写的随机存储器），其中至少有两个 MAC 地址，即网卡的物理 MAC 地址和全 1 的广播 MAC 地址。每当计算机想接收一个多播数据，也就是说要加入一个多播组，那么上层软件会给网络层一个通知，网络层做完自己的处理后，也会发一个通知给数据链路层，于是，数据链路层根据网络层想加入的组播组的组地址（一般是一个组播的 IP 地址），根据一定的规则影射为一个组播的 MAC 地址，然后把该 MAC 地址加入接收地址列表。

每当数据链路层接收到一个数据帧的时候，就提取该数据帧的帧头，找出目的 MAC 地址，跟接收地址列表中的地址项目比较，在比较完整个接收地址列表之前，如果遇到一个地址，跟该数据帧的目的 MAC 地址是相同的，于是停止比较，接收该数据帧，并把该数据帧放到上层协议对应的接收队列中；如果比较完整个接收列表也没有找到一个匹配的 MAC 地址，则丢弃该数据帧。

现假设接收到的数据帧是发给自己的单播数据帧，于是该数据帧的目的 MAC 地址就是自己的硬件地址。数据链路层接收到该数据帧，跟接收列表中的地址比较，第一次比较就会通过，因为接收地址列表中的第一个 MAC 地址就是自己的硬件地址。所以在任何情况下，发给自己的数据帧一定能接收下来；

假设接收到的数据帧是一个广播数据帧，则在比较的时候，最后一项是匹配的，因为接收地址列表中肯定包含一个广播的 MAC 地址，这样就保证了任何广播数据报都会被正确接收；

假设上层软件想接收组播组 G 的数据，经过一番影射到数据链路层之后，数据链路层会在自己的接收数据列表中添加一项组播组 G 对应的 MAC 地址，假设为 MAC_G，当计算机接收到一个数据帧，该数据帧的目的地址为 MAC_G 的时候，该数据帧会被接收并传递到上层，因为接收列表中有一项 MAC_G 记录。

6.1.3 组播转发表

在前面交换机的转发方式课程中讲述到，交换机在转发组播数据包的时候，也是根据一个转发表来进行的，但跟单播转发表不同的是，该组播转发表对应的出口不是一个元素，而是一组元素，即一个出口列表。在转发数据的时候，交换机根据组播数据帧的目的组播地址查找组播转发表，如果在组播转发表中查不到相应的转发项，则把该组播数据帧当做广播数据帧处理，向所有端口上转发该数据帧（如果实现了 VLAN，则仅仅向接收端口所属同一个 VLAN 的端口上转发）；如果能查找到一个结果，则结果肯定是一个接口列表，于是，交换机把这个组播数据报复制成许多份，每份提交到一个出接口，从而完成数据的交换。

需要注意的是，这个组播转发项不象单播那样，是通过学习的方式获得的（在处理单播数据的时候，交换机每转发一个数据帧，会进行相应的学习过程，学习到一个单播转发项），而是通过一些二层的组播协议获得的，这些流行的组播协议有 IGMP 窥探，GMRP 等。下面部分详细解释每一种二层组播协议。

6.2 二层组播协议

上面我们提到，用于转发组播数据帧的组播转发项不是通过学习获得的，而是通过一些其他的组播协议，比如 IGMP 窥探，GMRP 等协议获得的。至于为什么不能通过学习获得，是因为学习过程是分析一个数据帧的源地址，但组播地址在任何情况下不可能出现在源地址的位置（因为组播 MAC 地址是一个逻辑的 MAC 地址，网络上没有一个终端的 MAC 地址是一个组播 MAC 地址）。

下面我们分析二层组播协议，这些协议在构建组播网络的时候，有不可替代的重要地位。尤其是将来的流媒体应用会很广泛，在这种情况下，二层的组播将是必然的要求。

6.2.1 IGMP协议

在讲述 IGMP 窥探之前，先看一下 IGMP 协议，理解了 IGMP 协议，IGMP 窥探就很容易了。

一般情况下，组播数据是从路由器的上游流下来的，而组播的接收端一般位于路由器的下游，典型的情况是，路由器通过一个以太网口连接企业网，另外通过一个串口连接 INTERNET。组播数据源，比如流媒体服务器一般位于 INTERNET 上，组播数据流通过路由器到达企业网上的数据接收端。这种情况下，流媒体服务器是长时间工作的，也就是说，一天 24 小时，一周 7 天都在不停的发送媒体流信息，相当于电视频道。但企业网络中的数据接收端却不是这样，只有有限的时间接收端才接收数据，其他时间都是不接收数据的。这样，在企业网上没有数据接收端的时候，如果路由器也把大量的媒体流信息发到企业网内部，必然会浪费大量的资源，理想的情况是，如果企业网上没有数据接收端，则路由器就不转发媒体流，但如果只要有一个接收端，路由器就必须把媒体流引入企业网。这样必须有一种机制来保证路由器对企网上的数据接收端有一个清楚的了解，什么时候网络上有数据接收端，什么时候没有数据接收端，这种机制就是 IGMP 协议。

从上面的分析中，我们可以看出 IGMP 协议是数据接收端和路由器之间的交互协议，数据接收端使用该协议来通知路由器，自己是否想接收组播数据流。如果想接收的话，接收哪个组播组的数据流。

一般情况下，存在两种类型的 IGMP 消息（IGMP V1）：组成员报告和组成员查询，其中组成员报告消息由计算机发出，用来告诉路由器，自己想加入某个组播组，而组成员查询消息则由路由器发出，用来查询网络上是否存在相应组的成员。一个指导性的原则是，只要网络上有组播组的数据接收端，不管该接收端的数量是多少，路由器必须把该组播组的数据转发到网络上。

IGMP 消息也是通过组播地址发出的，在 IGMP 加入消息中，该组播地址（IGMP 消息的目的地址）就是该计算机想加入的组播组的组地址。比如，计算机想加入 224.0.0.2 这个组播组，则该计算机发出的 IGMP 加入消息的目的 IP 地址就是 224.0.0.2。这样当路由器接收到该组播消息后，就知道网络上有一个主机，该主机想接收到组播组 224.0.0.2 的数据，于是，每当从上游接收到目的地址是该组地址的数据的时候，就把该数据包向企业网上转发。

还有一类消息就是组播组成员查询消息。在 IGMP 协议第一版中，没有规定主机的离开消息，即如果一个主机不想接收某一组播组的数据了，它也不会通过某种消息通知路由器，而是静悄悄的离开该组播组。这样如果路由器不采取某种措施来掌握网络上组播组接收端的数目情况，就会产生问题，假设网络上所有的主机都不想接收组播数据了，根据 IGMP 协议第一版，这些主机不会通知路由器，而是不做任何处理。这样路由器不知道这些这些原来接收组播数据的主机现在已经不接收数据了，而且还以为网络上有一大批的接收端，于是仍源源不断的发送组播数据，这样必然浪费带宽。而引入了组成员查询消息后，路由器可以每隔一段时间发出该消息，用来查询网络上还有没有主机在接收组播数据。注意的是，该查询消息跟成员加入消息一样，采用的目的 IP 地址也是查询的组的组播 IP 地址。假设路由器想查询 245.2.2.1 这样一个组播组是否还有成员，发出的查询消息的目的 IP 地址就是 245.2.2.1。当网络上接收该组播组数据的成员接收到这个查询消息后，就发出一个响应，该响应还是一个组播组成员加入消息。这样只要路由器接收到一个这样的响应，它就知道网络上必然还有终端在接收数据，于是不能停止转发。但路由器发出查询消息后，一段时间内没有接收到任何响应，它就判断网络上没有了数据接收端，于是停止转发组播数据。

到此为止，IGMP 协议的一些细节已经介绍完毕，接下来介绍 IGMP 窥探。需要注意的是，IGMP 协议是三层协议，而 IGMP 窥则是二层组播协议，但它利利用了 IGMP 的三层特性。注意，IGMP 协议是运行在交换机上的一种协议，它使用该协议来形成转发组播数据的组播转发表。

下面是启动了 IGMP 窥探的交换机工作过程：

1. 交换机从每个端口上监听接收到的数据帧，如果是单播数据帧就按照通常的转发方式进行转发，是组播数据帧则进行下一步处理；

2. 如果接收到的数据帧是组播数据帧，则分析该组播数据帧的协议类型字段，看是否是 IP 协议，如果不是，则按照通常的方式转发（这时候可能查询组播转发表转发，如果没有查到结果，则向所有端口上转发）；

3. 如果是 IP 协议数据，则进一步判断该数据是不是一个 IGMP 加入消息 IGMP 查询消息。如果不是，则进行通常的组播数据发送，否则转下一步；

4. 如果是一个 IGMP 加入或查询消息，则该交换机就可以判断，在接收到该数据帧的端口上一定连接一个组播数据接收端，该接收端想接收 IGMP 加入消息中目的地址所在的组播组数据。于是，交换机就检查组播转发项，看对应的组播 MAC 地址有没有在组播转发项中出现。如果出现了，则把接收到该数据帧的接口加入组播转发项对应的接口集合，如果没有出现，说明该组播转发项还没有创建，于是创建一个组播转发项，该转发项的组播地址是 IGMP 消息的目的 IP 地址的影射 MAC 地址，接口列表初始化为接收到该数据帧的接口，接下来继续按照通常的过程转发该数据帧。

到此为止，IGMP 窥探的具体工作过程已经明了，可以看出，这种协议最大的一个缺点就是效率较低，交换机需要分析每个组播数据帧，看该数据帧是否是组播数据帧，如果是，继续看是否是 IP 数据帧，如果是，继续看是否是 IGMP 加入消息等等。

6.2.2 GMRP协议

当主机不支持 IGMP 协议的时候就无法实现窥探。我们考虑能不能开发一种专门的针对组播的协议，该协议可以高效的完成其他组播协议完成的任务。

这就是 GMRP（General Multicast Register Protocol，通用组播注册协议）协议。该协议需要计算机网卡和交换机一起工作，因此需要计算机网卡的支持。由于该协议是一个较新的协议，故大多数网卡都没有支持，但作为一种标准的协议，将来的网络设备都会支持。

该协议的运行过程很简单，想接收组播数据的计算机只要告诉连接的交换机，它想接收哪个组的组播数据即可。交换机接收到通知后，就创建相应的组播转发项，把接收到组播请求的端口加入组播转发项中。当计算机不再想接收组播数据了，只简单的告诉交换机就可以了，交换机接收到通知后，就从组播转发项中把相应的端口删除掉。

6.3 总结

到此为止，二层组播的一些基础概念就讲解完了，在这些内容中，组播 MAC 地址，组播和单播数据的接收过程以及三种组播协议是重点。这些内容也是

学习后续内容（比如三层组播路由协议）的基础，在实际中，这些组播概念也是应用得最多的。

第7章 生成树协议

现在的大型网络设计一般都采用分层结构，即分为核心层，汇聚层和访问层三个层次，其中核心层使用核心的设备，比如高端以太网交换机承担，汇聚层采用一些中低端的交换机担当，而接入层则一般采用低端交换机来负责，这样，汇聚层完成接入层业务的汇聚，然后送到骨干层上传输。

为了保证冗余特性，汇聚层交换机或接入层交换机一般通过两条以上的链路跟上层连接，而骨干层各个设备之间也不是单一的链路，而是组成一个全网状结构或一个半网状结构。不论何种结构，核心层交换机之间的链路肯定多于一条。在这些冗余链路的环境中，就会产生很多问题，最典型的是广播风暴。

7.1 广播风暴

考虑下面的网络结构：

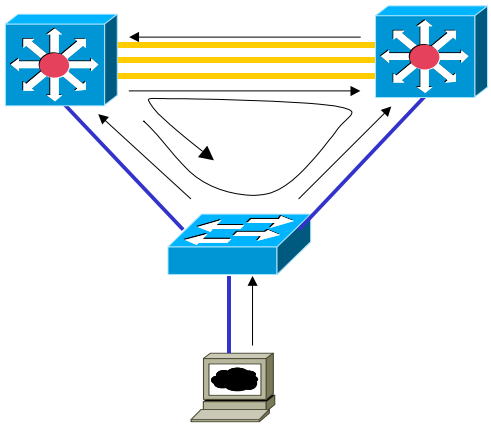


图7-13 环路

在该网络中，汇聚层交换机通过两条链路分别跟核心层的两个核心交换机连接。这样可以做到充分的冗余，比如，当其中一条链路出现问题，还有另外一条链路做备用。但如果转发广播数据报，会产生严重问题：假设 PC 机发出了一个广播数据帧，汇聚层交换机接收到以后，会通过两个上行的端口转发。核心层接收到这个广播数据帧之后，会互相转发，这样下来，必然造成广播数据帧在整个网络上来回震荡，造成网络的瘫痪，这就是所谓的广播风暴。

在这种情况下，我们必须引入一种机制来避免这种危险，这种机制就是生成树协议。

7.2 生成树协议基本概念

生成树协议的关键是其几个基本概念，理解了这些基本概念，其运行过程也就明了了。下面结合图形来具体说明几个概念。

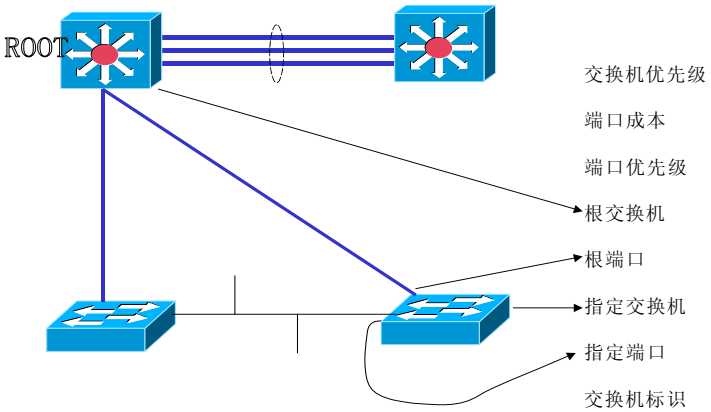


图7-14 生成树协议概念示意图

7.2.2 根交换机

生成树协议的运做结果是生成一棵无环树，既然是一棵无环树，必然存在一个树根。这个树根是由选举产生的，选举出来当做树根的交换机就是根交换机。

选举出根交换机以后，其他交换机就以根交换机为依据来建立自己的转发拓扑。每个交换机的转发拓扑组合起来，就形成了一棵树。

7.2.3 交换机标识和交换机优先级

交换机标识用来唯一表示网络上的一台交换机。该标识是由两部分组成的，第一部分是 2 字节的优先级，该优先级说明了在选举根交换机的时候交换机的优先程度，接下来是六字节的 MAC 地址，该 MAC 地址就是交换机的 MAC 地址。这样的组合就保证了交换机的标识唯一性，即使优先级相同，但每个交换机有唯一的 MAC 地址，这样就可以保证了交换机标识的唯一性。

交换机标识是选举根交换机的唯一依据。在选举根交换机的时候，各个交换机比较其他交换机的标识，具有最低标识的交换机被选中。

可以看出，在比较交换机的标识的时候，起决定作用的是 2 个字节的优先级。因为这两个字节位于标识的高位部分。当优先级相同的时候，交换机的 MAC 地址才起作用。所以，在实际中，只要更改交换机的优先级就可以轻松控制根交换机的选举。

7.2.4 端口成本和端口优先级

生成树协议的最终运行结果表现在交换机上，就是闭塞一些冗余的端口，打开一些端口进行转发。当打开的端口坏掉了，则把闭塞的端口再打开，做到备用。这样就面临一个问题：怎样选择闭塞的端口？

一个显著的想法就是，选择带宽最高的端口进行转发，其他带宽较低的统统闭塞。这样就端口成本便是这样一个选择依据，一般情况下，端口成本就是端口带宽的正比例反映，但这个端口成本是可以配置的，就是说，你可以手工指定端口的成本，而不管实际的带宽如何。这样完全是出于策略的考虑。

但存在一种情况，就是两个端口有同样的带宽。这样的情况下，我们就需要用到另外一个参数——端口优先级了。该参数指定，当端口成本相同的时候，优先级高的端口优先闭塞，优先级最低的端口进行转发。这样就保证了转发端口的唯一性。默认情况下，所有端口的优先级是相同的，这样为了打破僵局，交换机会参考端口号（端口号在交换机上有唯一性），端口号最低的端口优先进行转发。

7.2.5 根端口

顾名思义，根端口就是通向根的端口。交换机运行生成树协议，最终形成一棵转发树，每个交换机都通过一个且仅仅一个端口连接到这棵树的根，这个端口就是根端口。

7.2.6 指定交换机和指定端口

一个交换机选择出根端口以后，并不是把其他端口都堵塞，而且还必须选出一个指定端口。该端口的目的是转发下游数据（所谓下游，是针对根而言的）。考虑上图中最下面的一个冲突域，该冲突域可能通过一个 HUB 连接了很多计算机，这个冲突域跟右边的一个交换机连接，这时候，如果右边的交换机选择出根端口之后，把其他所有端口都闭塞，则该冲突域就不能把自己的数据转发出去。因此，右边的交换机还必须把连接该冲突域的端口打开，来转发该冲突域内的通信。

可以看出，在上图中这个冲突域连接了两个交换机，通过什么样的规则选择其中的一个交换机端口作为指定端口呢？还是端口成本。两个交换机比较从对方到根的成本，选择成本小的作为指定端口，其他端口闭塞。

到此为止，生成树协议的一些基本概念就介绍完了，下面看一下生成树协议的运行过程。

7.3 生成树协议的运行过程

生成树协议运行过程分几个过程：

1. 选择根交换机：交换机启动的时候，向各个端口发送 BPDU（桥接协议数据单元），该数据帧的目的 MAC 地址是一个保留的组播 MAC 地址，这样就保证了以太网上所有的交换机都可以接收到该数据帧。BPDU 包含下列内容：发送交换机的标识，发送端口的成本，根交换机的标识（初始化为自己），到根交换机的成本等等。这样交换机接收到其他交换机发送的 BPDU 后，跟自己进行比较，看交换机标识是否比自己小。如果是，则选择它为根交换机，这样长时间的选择，最终网络中的交换机会达成共识，选择某个交换机为根交换机（该交换机的标识最低）；

2. 选择根端口：交换机从接收到的 BPDU 中可以计算出自己哪个端口到根交换机最近（成本最低），成本最低的端口被选定为根端口，并转换到转发状态；

3. 选择指定端口和指定交换机：选择出根端口之后，交换机会向所有其他未阻塞端口（所谓阻塞，是由于物理链路没有起来或手工关闭）发送从根端口接收到的 BPDU。不过发送的时候，把 BPDU 进行修改，把其中的发送交换机标识填写成自己的标识，到根交换机的成本改为端口成本加上根端口到根交换机的成本，比如，根端口到根交换机的成本为 100，而某个端口的成本为 20，则从该端口把 BPDU 发送出去后，相应的到根交换机的成本就转换为 120。在发送的同时，也可能从其他交换机接收到 BPDU，这时候就把自己刚才发送的 BPDU 同接收到的 BPDU 比较，看哪个距离根比较近（成本低），如果自己的成本低，则该端口跳转到转发状态，也就是说，该端口成为相应冲突域的指定端口，而该交换机就是相应冲突域的指定交换机。而刚才发送 BPDU 的交换机因为发现自己到根的成本高，就会阻塞该端口。

为了更好的理解指定端口和指定交换机的选择，考虑下面的网络图形：

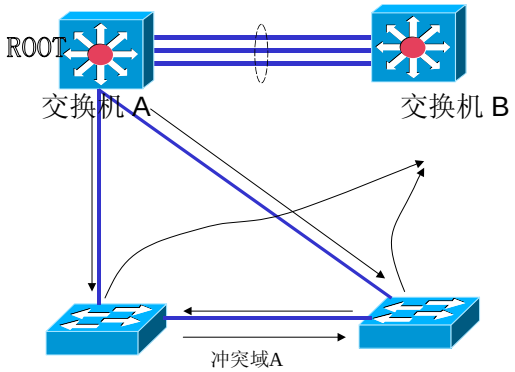


图7-15 根端口选择示意图

交换机 A 和交换机 B 选择出根端口后，就分别向冲突域 A 转发从根端口接收到的 BPDU。该 BPDU 在转发的时候会进行修改，假设交换机 A 连接冲突域 A 的端口的成本是 10，交换机 B 连接冲突域 A 的端口成本是 20，交换机 A 从根端口到根的成本和交换机 B 从根端口到根的成本都是 100，则交换机 A 在发送 BPDU 的时候，BPDU 中到根的成本修改为 110（100 + 10），交换机 B 在发送 BPDU 的时候，BPDU 中到根的成本修改为 120（100 + 20）。

这样交换机 A 和交换机 B 分别接收到对方的 BPDU。在交换机 A 接收到交换机 B 的 BPDU 的时候，把 BPDU 中的端口成本（120）提取出来，跟刚才自己发送的 BPDU 中的端口成（110）本比较，发现比自己刚才发送的成本要高，于是交换机 A 就把连接冲突域 A 的端口转换到转发状态，成为该冲突域的指定交换机，相应的，交换机 A 成为该冲突域的指定交换机。但交换机 B 接收到交换机 A 的 BPDU 后，比较两个 BPDU（自己发出去的和接收到的）到根交换机的成本，发现比自己的低，于是，交换机 B 就把连接到冲突域 A 的端口置为阻塞状态。这样就完成了指定端口和指定交换机的选择。

上述几个步骤完成之后，各个交换机就停留在了一个稳定的转发状态，进行通常的数据转发。

需要注意的是，运行生成树协议的交换机的各个端口会经历几个状态，刚开机的时候是阻塞状态，这种状态下只能接收和发送 BPDU，这样做的目的是因为刚开始的时候没有形成一棵无环的转发树，可能存在环路问题。但 BPDU 是一个组播数据帧，不会发生广播风暴。相互发送 BPDU，选择出根交换机，选择出根端口和指定交换机等，这样就形成了一棵无环的转发树。但这时候为了避免其他交换机还没有完成生成树协议的计算，于是交换机的端口还需要经历一个学习的状态，在该状态下，端口可以接收数据包，并进行学习过程（形成 CAM 表），但不做转发。当学习过程结束后，交换机端口才真正转换到转发状态，这种状态下才能进行正常的数据转发。

附录 缩略词表

CSMA/CD	Carrier Sense Multiple Access with Collision Detection	载波侦听与冲突检测
GMRP	General Multicast Register Protocol	通用组播注册协议
ISO	International Standardization Organization	国际标准化组织
LLC	Logical Link Control	逻辑链路控制子层
MAC	Media Access Control	媒体访问控制子层
OSI	Open System Interconnection	开放式系统互连模型
PDU	Protocol data unit	协议数据单元
QOS	Quality of Service	服务质量保证
VLAN	Virtual Local Area Network	虚拟局域网