第 3 章 TCP/IP 协议分析实验

第 1 章主要讲述了计算机网络体系结构的概念、ISO/OSI 参考模型、TCP/IP 体系结构以及我们接触最多的局域网体系结构——IEEE802 系列标准。第 2 章讲述了 Windows 2000 环境下 TCP/IP 协议的具体应用。通过前两章的学习，读者对计算机网络的整个工作过程有了一定的认识。

以使用 TCP/IP 协议栈的 Internet 中两个用户(用户 A、B)通信为例，暂不考虑数据在 Internet 中的传输过程，可用图 3-1 来说明数据在两个用户间的处理过程。

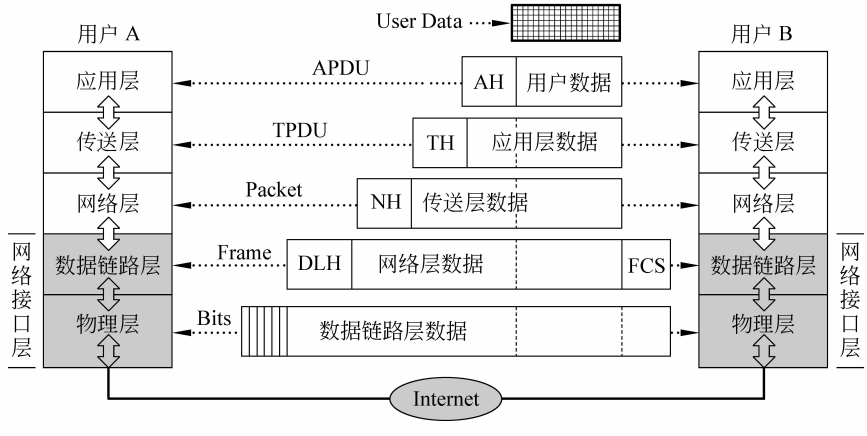

图 3-1　TCP/IP 用户数据处理过程

用户 A 的通信进程递交用户数据(User Data)给应用层，应用层协议对用户数据处理后形成应用层数据单元(APDU)，然后将 APDU 交于传输层协议处理，形成传输层数据单元(TPDU)，同样传输层将 TPDU 向下交于网络层协议处理，形成网络层分组(Packet)。由于 TCP/IP 是一个网际互连协议，网络层以下统称为网络接口层，TCP/IP 协议栈没有对此定义，它可以工作在任何物理层和数据链路层协议之上。图 3-1 中显示了工作在局域网之上的 TCP/IP 协议栈。用户 A 的高层数据在本地端系统经过层层“封装”，最后经物理层以位流形式发出，经过 Internet 最终到达端系统 B。

端系统B经过从物理层从下到上的层层“解封”,最后将A的用户数据递交给B。

图3-1以TCP/IP协议栈为例说明了数据在端用户系统中的处理过程。对于其他协议栈(如SPX/IPX、AppleTalk、DEC和XNS等),整个处理过程基本相似。可能的区别在于协议层次结构不同,每层协议的功能不同而已。

不同层次的协议处理不同结构的数据单元。网络中用户的通信实际上就是数据包的传输过程。检测网络中传输的数据包就可以分析网络中各种协议的功能及运行情况,进一步可以判断出网络的工作情况,如网络中是否有用户在发送大量无用的广播包,网络中是否有用户配置了无用的协议,是否存在网络攻击或感染了一些特征明显的病毒等。

网络协议分析是网络工程中的一个重要环节,主要用来进行网络系统维护及性能检测。如用户经常感觉自己的内部网络突然变得很慢,可能就是网络中的协议配置出了问题,也可能有病毒操纵用户计算机向网络发送大量数据包。目前有许多软、硬件网络协议分析产品出现,如Fluke公司价格昂贵的Optiview协议分析仪、网上各种免费的Sniffer软件等。

本章主要介绍基于局域网的TCP/IP协议栈,对其中的协议数据单元进行详细的分析,最后通过Sniffer软件(Ethereal)来查看、分析协议数据单元的结构和协议之间关系,以加强读者对TCP/IP协议栈的层次关系、各层协议功能的认识和理解。

3.1 局域网中常用的协议栈

IEEE802是局域网参考模型,该参考模型主要包含物理层和数据链路层。针对局域网中共享介质的特殊情况,局域网参考模型又将数据链路层分别称为介质访问控制子层(MAC)和逻辑链路子层(LLC),MAC子层主要解决网络内各节点对共享介质的争用问题,LLC子层主要提供差错控制、流量控制、向高层提供面向连接和无连接服务等功能。简单的层次结构如图3-2所示。

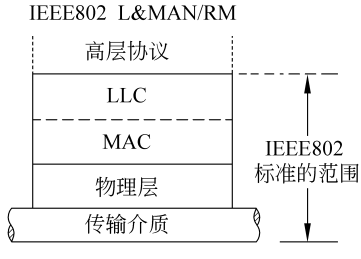

图3-2 IEEE802局域网层次结构

在局域网之上可运行多种高层协议栈。计算机网络中存在多种协议栈,遵循不同的体系结构,大部分协议栈都可直接运行在局域网之上。图3-1为常用的在局域网上运行TCP/IP协议,以实现接入Internet的目的。局域网中常用的还有运行Netware的SPX/IPX协议栈、基于局域网内部协议NetBIOS的协议栈等,这三种协议栈有不同的层次结构和数据封装格式。本章节着重介绍基于IEEE802.3局域网的TCP/IP协议栈结构。

3.2 局域网数据链路层的帧结构

在OSI/RM中,数据链路层协议主要基于点到点的高级链路控制规程(HDLC)。而在局域网中,数据链路层还必须完成网络层的功能,如数据报、虚电路和多路复用等。因

为无须进行路由选择，只要如同 HDLC 中那样，在帧中安排源和目的地址，即可实现数据报服务。而要实现虚电路和多路复用，则必须如同在 X.25 中那样，在分组中安排逻辑信道号。在 IEEE802 标准中，采用的办法是在数据帧中安排源和目的服务访问点(SAP)，以分别对各个局域网站上不同的用户或进程寻址。

IEEE802.3 的帧结构如图 3-3 所示。在 LLC 子层使用 IEEE802.2 的格式——LLC 帧或 LLC 协议数据单元(LLC PDU)，它的源和目的服务访问点(SSAP 和 DSAP)用于实现多路复用和虚电路。LLC 帧再封装到相应的 MAC 帧中，在 MAC 帧中安排有目的和源站地址。地址信息在 LLC 和 MAC 之间进行了分解，目的是为了把数据送往各个局域网站上不同的用户或进程，或从它们那里接收数据。因此 LLC 子层需要知道源和目的服务访问点，而 MAC 子层还要知道服务访问点所在的源和目的站地址。有两种数据链路层帧格式：IEEE802.3 帧和 Ethernet V2 帧。

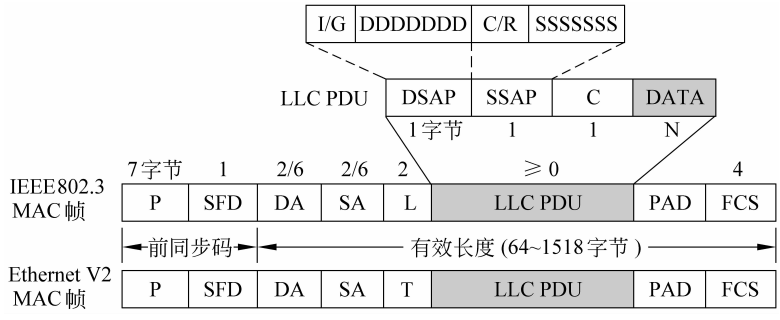

图 3-3　IEEE802.3 和 Ethernet V2 帧格式

3.2.1　IEEE802.2 LLC PDU 结构

在图 3-3 中，LLC PDU 的首部比较简单，只有三个字段。DSAP、SSAP 字段各占一个字节，分别是目的和源服务访问点。DSAP 的第一位为 I/G 位。当 I/G 位为 0 时，其后面的 7 位代表单个目的访问点 DSAP；当 I/G 位为 1 时，DSAP 为组地址。组地址能实现数据发往特定站的一组服务访问点。SSAP 的第一位为 C/R 位。当 C/R 为 0 时，指示 LLC 帧为命令帧；当 C/R 为 1 时，指示 LLC 帧为响应帧。C/R 位后 7 位用来表示源服务访问点。

该服务访问点主要用来表示高层协议(进程)类型。这些协议类型要向 IEEE 注册，任何人都可以注册自己的协议。当用户访问某协议时，用 DSAP 字段中的一个值来表示它。如 IEEE 给 IPX 协议分配的十六进制值是 0xE0，分配给 NetBIOS 的 DSAP 为 0xF0，分配给 STP 的为 0x42。通常情况下，DSAP 和 SSAP 会取一样的值。

C 为控制字段，其结构类似于 HDLC 的控制字段，功能比较复杂，在此不做详细介绍。局域网中该字段为 0x03，表示无编号信息帧 UI，意味着 LLC 层提供无连接服务。

DATA 是为上层协议传送的数据。

LLC PDU 封装在 MAC 帧中，所以 LLC PDU 不设帧标志字段和帧校验序列字段。LLC PDU 的数据字段长度不受限制，应是整数个字节，但它受 MAC 帧长度的限制。

此外,为了支持其他各种非IEEE的数据链路层协议,除前面讲述的IEEE802 LLC帧结构之外,还有一种通常称作IEEE802.2 SNAP的帧格式。该帧结构是在IEEE802.2 LLC的3字节的帧头后加一个子网访问协议(Sub-network Access Protocol,SNAP)首部,此时LLC首部为0xAAAA03(DSAP、SSAP均为AA,保留用于传送非IEEE的局域网所产生的上层数据包)。

IEEE802.2 SNAP的帧格式如图3-4所示。

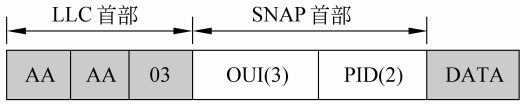

图3-4 IEEE802.2 SNAP的帧格式

三个字节OUI(Organizationally Unique Identifier,组织唯一标识符)标识制定高层协议的组织或公司,后面的两个字节PID(Protocol Identifier,协议标识符)表示该组织或公司制定的各种协议。二者合在一起标识了一个独特的路由或桥接协议,增强了IEEE802.2帧的通用性。常用的OUI和PID的值如表3-1所示。

表3-1 常见的SNAP首部

OUI	PID
0x000000(以太网)	0x0800(IPv4)
	0x86dd(IPv6)
0x00000c(Cisco公司)	0x2000(CDP)
	0x2003(VTP)
	0x2002(HDLC)
	0x2004(CDP/VTP)
0x000081(Bay公司)	0x01a2;0x01a1
0x00e0fc(华为)	0x2000(CDP/VTP)

3.2.2 IEEE802.3帧结构

IEEE802.3的MAC帧结构如图3-3所示。

- P为7字节的前导码,包含速率及定时信息,用于使PLS(物理收发信号)电路和收到的帧定时达到稳态同步。
- SFD为帧首定界符,1个字节,取10101011二进制序列,紧跟在前导码的后面,表示一个帧的开始。

P和SFD这两部分统称为帧的前同步码,通常由物理层硬件在帧发送和接收时加上和除去,不算在MAC帧的有效长度范围内。

- DA和SA分别为目的站和源站的物理地址(MAC地址),目的站和源站的地址

DA、SA 可采用 6 字节(48 位)或 2 字节(16 位)两种编址方式中的一种。6 字节的地址字段对于一个小范围内使用的局域网显得太长,但可使全世界所有的局域网上的站都有不相同的地址,因此这种标准深受广大用户欢迎。IEEE 负责分配地址字段 6 个字节的前 3 个字节(高 24 位),生产局域网卡的厂家都必须向 IEEE 购买这三个字节构成的一个号,称为“地址块”或“厂商代码”。地址字段的后三个字节(低 24 位)由厂家自行分配。在生产网卡时,这种 6 字节的 MAC 地址已经被固化在网卡中。MAC 地址又称为物理地址,也就是通常所说的计算机的硬件地址。在计算机网络中,网卡从网上每收到一个 MAC 帧就首先检查其目的物理地址。如果是发往本站的帧,就收下放入缓冲区,然后再进行其他处理;否则就将此帧丢弃,不再进行其他处理。

- L 为长度字段,表示 LLC PDU 的字节数。它的范围为 46～1500 字节。如果 LLC PDU 的字节数小于 46 字节,则发送站的 MAC 子层自动填 0 代码于填充字段 PAD 中。
- PAD 为填充字段,长度不定。由于 CSMA/CD 协议的正常运行要求帧满足最小长度,必要时可填充附加位以达到要求。
- FCS 为帧校验序列,通常收发双方都使用循环冗余校验(CRC-32)来产生 32 位的 FCS 字段。FCS 处理一般由网卡硬件自动完成。

3.2.3　Ethernet V2 帧结构

通常所称的“以太网”是 Ethernet 的中文译名。而 Ethernet 的标准 DIX Ethernet V2 与 IEEE802.3 标准略有不同。在 DIX Ethernet V2 标准中,IEEE802.3 MAC 帧的数据长度(L)字段被类型(T)字段代替(如图 3-3 所示),其他字段完全相同。在不涉及到网络协议细节时,可以将 802.3 局域网简称为以太网。

图 3-3 中表示了以太网帧和 IEEE802.3 帧的结构,对于 IEEE802.3 帧来说,由于它的高层协议基于逻辑链路控制子层(LLC),即 IEEE802.2 标准,因此在 IEEE802.3 帧的 DATA 段是 LLC 协议数据单元(LLC PDU)。而以太网的 MAC 帧可以直接承载网络层的数据,且用类型(T)字段指明上层封装的数据是何种网络层协议的分组,故以太网的 MAC 帧的 DATA 段为网络层的分组(如 IP 分组),不存在 LLC 子层。

为使以太网和 IEEE802.3 两种帧能兼容,即使两种帧都能在网上正常发送和接收,怎样处理 TYPE 和 L 字段是问题的关键。解决的办法是:取值 1536(0x0600)作为界限,大于或等于 0x0600 认为是以太网帧,此字段按类型(Type)处理。例如 IP 为 0x0800,ARP 为 0x 0806,RARP 为 0x8035,IPX 为 0x8137 和 0x8138,AppleTalk 为 0x809b 等。反之,小于 0x0600 认为是 IEEE802.3 帧,该字段为长度值 L。

两种帧结构的区别还有:以太网帧的 DA 字段上只有最高位才有定义,区分单址(0)还是多址(1),而 IEEE802.3 帧却是最高两位有定义。次高位为 0 表示全局管理地址,1 表示局部管理地址。一般来说地址总是全局管理,因此该位总设置为 0。对于广播地址来说,两种帧结构中 DA 的最高位均设置为 1。

3.2.4 IEEE802.3 帧和 Ethernet V2 帧实例

根据以上所述,读者已经对局域网中常用的两种帧结构有了一个比较清楚的认识。下面的两个数据包是从局域网环境中用软件 Ethernet 捕获的数据帧,分别为 IEEE802.3 帧和 Ethernet V2 帧。图中阴影部分表示的是帧的首部。对应图 3-3,下面分别予以说明。

要注意的是,用软件捕获的数据帧已经经过 CRC(循环冗余校验)校验,帧中的 FCS 字段已经被去掉。

1. IEEE802.3 MAC 帧

图 3-5 为一个 IEEE802.3 MAC 帧结构实例,图中阴影部分包含 14 个字节的帧首部,之后有 3 个字节的 LLC PDU 首部。各字段解释如表 3-2 所示。

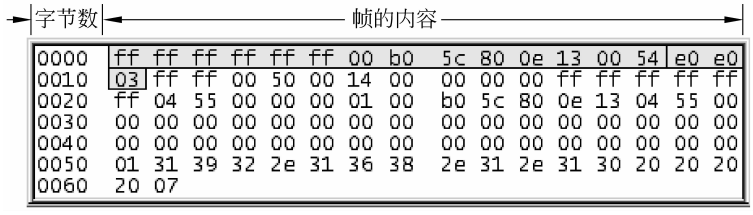

图 3-5 IEEE802.3 MAC 帧实例

表 3-2 IEEE802.3 帧头说明

IEEE802.3 MAC 帧首部(14 字节)			LLC PDU 首部(3 字节)		
DA(6 字节)	SA(6 字节)	L(2 字节)	DSAP(1 字节)	SSAP(1 字节)	C(1 字节)
ffffffffffff	00b05c800e13	0054	e0(IPX)	e0(IPX)	03

注:目的地址 DA 为一个广播地址,说明图 3-5 为一个广播包。且 LLC 上层为一个 Netware IPX 数据包。

2. Ethernet V2 MAC 帧

图 3-6 为一个 Ethernet V2 MAC 帧,图中阴影部分包含 14 个字节的 MAC 帧首部。

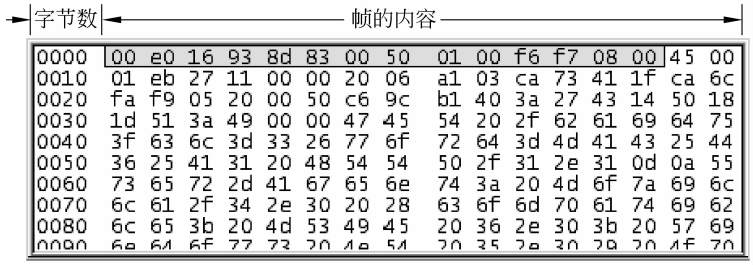

图 3-6 Ethernet V2 MAC 帧实例

帧首部各字段解释如表 3-3 所示。

表 3-3　Ethernet V2 帧头说明

Ethernet V2 MAC 帧首部(14 字节)		
DA(6 字节)	SA(6 字节)	T(类型)(2 字节)
00e016938d83	00500100f6f7	0800

注：Ethernet V2 帧的 T(类型)字段为 0800，表示其后(上层)数据段为 IP 数据包。

3.3　IP 协 议 栈

TCP/IP 协议栈是目前应用最广的协议栈。该协议栈每层包括多个协议，在此仅介绍常用的协议数据格式。

3.3.1　网际层协议

网际协议(Internet Protocol，IP)是 TCP/IP 协议栈中重要的网际层协议，它向高层提供无连接的服务。网际层传输的数据单元是分组(Packet)，一般称为 IP 数据报(Datagram)。其主要功能是提供一种从源端经互连网络到目的端的、尽最大努力传输数据报的方法。属于同一层协议的还有 Internet 控制报文协议(ICMP)、地址解析协议(ARP)和逆地址解析协议(RARP)等。ICMP 用于监控和报告差错信息。ARP/RARP 是实现物理地址和 IP 地址之间的转换，起着屏蔽物理地址细节的重要作用。网际层还包含内/外部寻径协议 RIP/EGP，负责路由的建立和刷新。

IP 协议有 IPv4 和 IPv6 之分，IPv6 是对 IPv4 功能的增强版本。目前 Internet 使用的是 IPv4 版本，故本节以 IPv4 为例进行介绍。

1. IP 数据报格式

IP 数据报由头部(Header，也称为首部)和正文部分构成。头部有一个 20 字节的固定长度部分和一个长度为 4N(N 取值范围为 1～10)选项的部分。头部格式如图 3-7 所示。

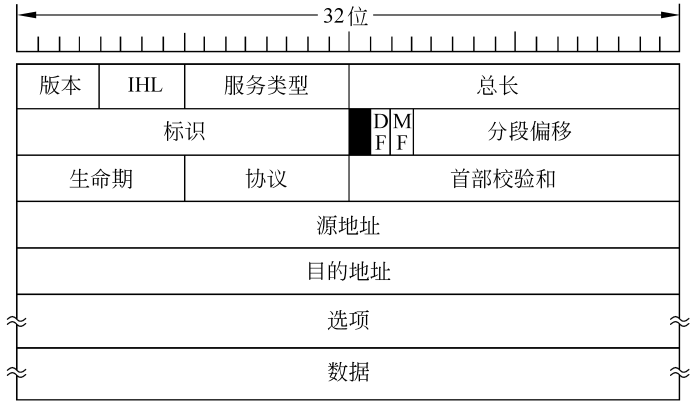

图 3-7　IP 数据报头部的组成

各字段解释如下：

- 版本(Version)。该字段指出该数据报是按哪一个版本的IP协议定义的。通过在每个数据报中引入版本字段，可以在运行不同版本IP的机器间传输数据。其中4代表IPv4，6代表IPv6。
- 头部长度(IHL)字段。用来指明头部包含的长度为32位(4字节)的字的个数。该4位字段最小值是5(当没有可选项出现时)，最大值是15。于是头部的最大长度是60字节，可选字段最大为40字节。对于像记录分组所经路由的这类选项，40字节可能太短。之所以需要IHL，是因为IP分组的头部长度可变。
- 服务类型(Type of Service)字段。使主机可以告诉通信子网它需要的服务。例如可以是各种可靠性和速度的组合。对于数字化声音传输，速度要求高于准确性要求。对于文件传输，准确性又比速度重要。

该字段从左到右分别是3位优先顺序(Precedence)子字段；3位标志位子字段，分别为D、T和R(Delay延迟，Throughput吞吐量，Reliability可靠性)，还有2位未用。优先顺序子字段标志优先级，从0(一般)到7(网络控制分组)。标志位使主机能说明它最关注D、T和R中的哪一项。路由器可以对这三项进行选择，例如是采用具有高吞吐量和高延迟的卫星线路，还是用低吞吐量和低延迟的租用线路。实际上，很多路由器可能会忽略该字段。

- 总长(Total Length)。该字段包括数据报中的头部和数据的长度，最长为65 535字节。当长数据报要分段传送时，“总长”是指分段后每段的首部长度与数据长度的总和。
- 标识(Identification)。该字段用来让目的主机判断新来的分段属于哪个数据报，所有属于同一数据报的分段包含同样的标识值。此“标识”没有顺序号的意思，因为IP是无连接服务，不存在按序接收的问题。

紧跟着的是1个未用的位，然后是分别占1位的标志字段DF和MF。

- DF(Don't Fragment)。该位为1时，代表不要分段，意味着数据报必须绕过可能在最优路径上的小分组网络，而使用次优路由。DF位为0时，表示该分组可以分段。
- MF(More Fragment)。该位为1时，代表后面还有分段。除最后一个分段外，所有其他分段都应设置该位，它用来标志原数据报的所有分段是否都已被接收方收到。
- 分段偏移(Fragment Offset)。说明分段处在原数据报的什么位置，即相对于数据报的起点该分段的开始位置。基本分段长8个字节。因为该字段为13位，所以每个数据报最长是8191个基本分段，这样，最大的数据报长度是65 528字节。注意，65 528比总长字段提供的最大值小7。
- 生命期(Time To Live)。该字段是用于限制分组生命周期的计数器。推荐以时间(秒)计数，最长为255s，在每个节点都要递减，减到0时即丢弃该分组。但实际上，生命期是以节点数计数的，当它减到0时，节点就丢弃该分组，并向源主机发送一个警告分组，这就能防止数据报在网中无限制地漫游，当路由选择表崩溃时

就会发生这种情况。生命周期字段还可被原站用于限制数据报的行程，若设为1，数据报到达第一个路由器就会被丢弃，从而将数据报限制在本局域网内。

- 协议(Protocol)。该字段说明将数据报的数据部分送给哪个上层实体，可能是TCP、UDP、ICMP 或其他基于 IP 的协议实体。协议的编号在整个因特网上是全球通用的，它定义于 RFC 1700 中。常用的基于 IP 的协议即协议字段值为 ICMP(1)、IGMP(2)、GGP(3)、TCP(6)、EGP(8)、UDP(17)和 OSPF(89)等。
- 首部检验和(Header Checksum)。仅用于校验头部，不包括数据部分。不检验数据部分是因为数据报每经过一个节点，节点处理机都要重新计算首部校验和字段。因为至少有一个字段总是在变(生命周期字段)。如将数据部分一起校验，工作量太大。

为了简化计算，检验和不采用 CRC 检验码。IP 检验和的计算方法是将 IP 数据报首部看成 16 位的序列，先将检验和字段置 0，将首部 16 位字相加后，将和的二进制反码填入检验和字段，当节点收到 IP 数据报后，将首部的 16 位字再次相加，若首部未发生变化，其和必为全 1，否则出错丢弃。

- 源地址(Source Address)和目的地址(Destination Address)。指源 IP 地址字段和目的 IP 地址字段。
- 可选项(Options)。提供了一种灵活性，可使后续版本的协议能引入最初版本中没有的信息，以便适应新的应用，还可避免为很少使用的信息分配头部字段。

2. IP 数据报实例

在图 3-6 所示 Ethernet V2 MAC 帧中，数据字段为 IP 数据报。图 3-8 标识了此 IP 数据报的首部，并在图 3-9 中给出了相应字段的解释。

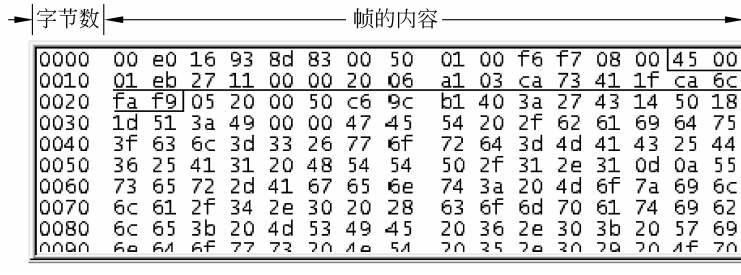

图 3-8　IP 数据报首部示例

图 3-9 中的数据可参考前面的说明理解其中的含义，要说明以下几点：

(1) IHL 字段为 5，说明此 IP 数据报的首部长度为 20 个字节，即首部不含选项字段。

(2) DF、MF、分段偏移所占的 16 位为 0x0000，分别表示数据报可以分段、此分段为最后一个分段、此分段距数据报首部为 0(即此数据报是第一个分段，也是最后一个分段)。

(3) 协议字段为 0x06，说明 IP 的数据字段封装的是一个 TCP 数据报。

(4) 源 IP 地址为 0xca73411f，即 202. 115. 65. 31。目的地址为 oxca6cfaf9，即 202. 108. 250. 249。

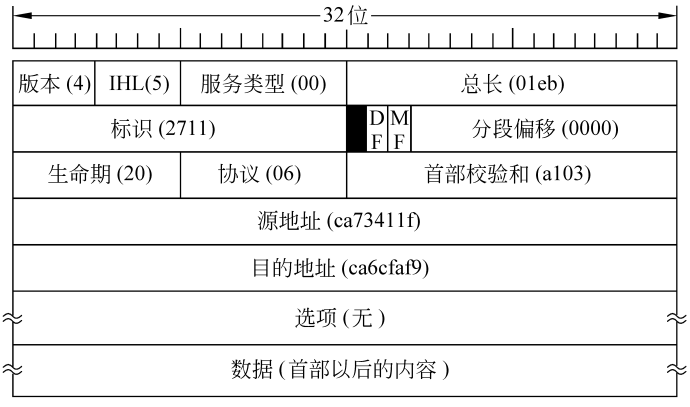

图 3-9　图 3-8 所示 IP 首部解释

3.3.2　地址解析协议

1. 协议原理

IP 地址是主机在网络层的地址。Internet 中的路由器可根据路由协议和 IP 数据报中的目的 IP 地址,将 IP 数据报送到下一路由器或目的主机所在的网络。但要实现这一目标,当前路由器要做的是将 IP 数据报传送到数据链路层,转变成 MAC 帧,然后经物理层发出。此时 MAC 帧中的源 MAC 地址为此路由器外出端口的 MAC 地址,而下一路由器或目的主机的 MAC 地址如何得到呢?

从 IP 地址到 MAC 地址的转换是由 ARP 协议实现的,而从 MAC 地址到 IP 地址的转换是由 RARP 协议实现的,这两个协议的工作原理、协议数据单元格式都基本相同,都是局域网内部协议。此处以介绍 ARP 协议为主。

通常网络中每个主机都有一个 ARP 缓存,存放目前主机知道的 IP 地址和 MAC 地址的对应关系表。当主机 A 要向本局域网上的主机 B 发送一个 IP 数据报时,就先在 ARP 缓存中查看有无主机 B 的 IP 地址。若有,取出对应的 MAC 地址,写入主机 A 要发送的 MAC 帧首部目的地址字段,然后通过局域网发往主机 B。此过程如图 3-10(a)所示。

也有可能从缓存中查不到主机 B 的 IP 地址,这可能由于主机 B 刚入网,也可能是主机 A 刚开机,其 ARP 缓存还是空的。在这种情况下,主机 A 就自动运行 ARP,按下列步骤找出主机 B 的 MAC 地址:

(1) 主机 A 的 ARP 进程在本局域网上广播发送一个 ARP 请求分组,分组中含有主机 B 的 IP 地址。

(2) 在本局域网中所有主机上运行的 ARP 进程都可收到此 ARP 请求分组。

(3) 主机 B 在 ARP 请求分组中见到自己的 IP 地址,就向主机 A 发送一个 ARP 响应分组,此分组中包含了主机 B 的 MAC 地址。其他主机对此分组不作任何处理。

(4) 主机 A 收到主机 B 的 ARP 响应分组后,就在其 ARP 缓存中写入主机 B 的 IP 地址到 MAC 地址的映射。

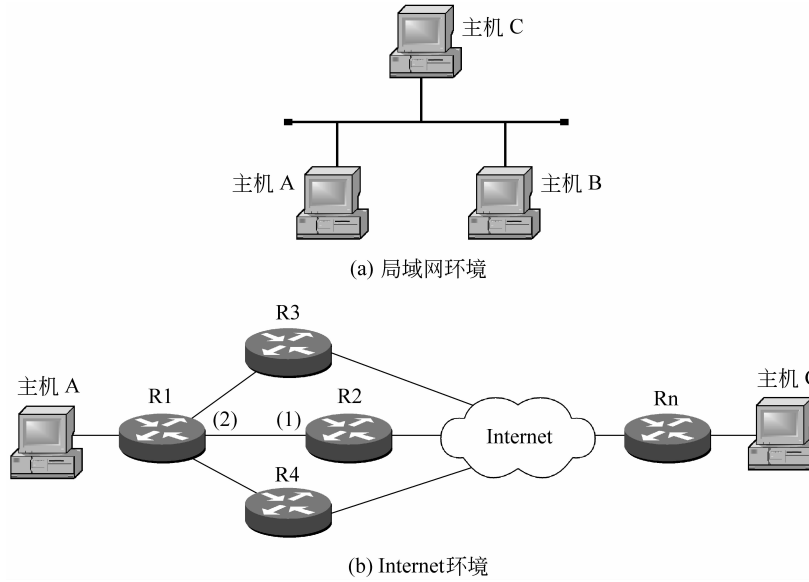

图 3-10　ARP 地址解析过程

通常情况下，当主机 A 向主机 B 发送数据报时，很可能不久以后主机 B 还要向主机 A 发送数据，因而主机 B 也可能要向主机 A 发送 ARP 请求分组。为了减少网络上的通信量，主机 A 在发送其 ARP 请求分组时，就将自己的 IP 地址到物理地址的映射写入到 ARP 请求分组（从本节后面的实例可以看出这一点）。当主机 B 收到主机 A 的 ARP 请求分组时，主机 B 就将主机 A 的这一地址映射写入主机 B 自己的 ARP 缓存中。这样，主机 B 以后向主机 A 发送数据就方便了。

当主机 A 要向 Internet 上某一主机 C 发送数据时（如图 3-10(b)所示），在 IP 数据报头，源 IP 地址是主机 A 的 IP 地址，目的 IP 地址是主机 C 的 IP 地址。当把 IP 数据报交给下层——数据链路层——封装成 MAC 帧时，源 MAC 地址是主机 A 的 MAC 地址，目的 MAC 地址是主机 A 的默认网关（路由器 R1）的靠近主机 A 的接口的 MAC 地址。当默认网关收到此 MAC 帧，取出 IP 数据报，进行路由选择。若没到达目的网络，则通过路由算法找出本路由器的输出端口(2)和下一个路由器 R2，得到 R2 接收端口(1)的 IP 地址，通过 ARP 找出 R2 的(1)端口的 MAC 地址。然后将默认网关 R1 的输出端口(2)的 MAC 地址作为源地址，下一个路由器(R2)的接收端口(1)作为目的 MAC 地址，将 IP 数据报重新封装成 MAC 帧，发给下一个路由器。

可以想象，IP 数据报在 Internet 中传输时，IP 数据报头中的源 IP 地址和目的地址始终不变（暂不考虑 NAT 功能），而 MAC 帧中的源 MAC 地址和目的 MAC 地址每经过一个路由器就要发生一次变化。

由此看出，在 TCP/IP 协议栈中，经过网际层、网络接口层（此处为局域网中的数据链路层和物理层）的处理，Internet 中任何两个计算机之间都可以互通，源主机发出的数据包最终可以到达目的主机，建立了端到端的连接。在此之上，就是由高层协议来处理两个主机之间的交互信息了。

2. ARP协议数据格式

ARP协议属于网络层协议,其协议数据单元被封装在Ethernet V2 MAC帧中,此时Ethernet V2 MAC帧中的类型(T)字段为0x0806。ARP分组有ARP请求分组和ARP响应分组,数据格式基本相同。ARP分组通常长度固定,应用程序可以直接作用于ARP协议,这一点类似于ICMP协议,但也正好说明了TCP/IP协议栈层次结构不够严谨,也说明了TCP/IP协议栈的灵活性。ARP分组格式如图3-11所示。

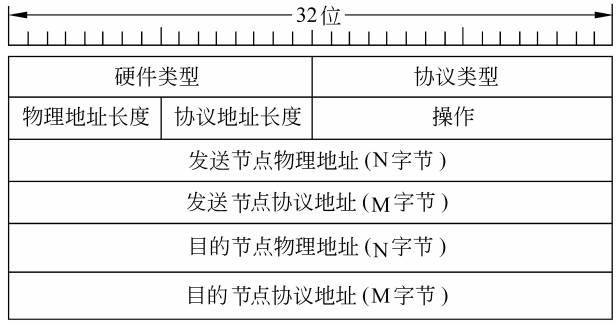

图3-11 ARP分组格式

与其他大多数协议不同,ARP分组没有固定的首部,以适应各种不同的网络技术,支持不同的物理网络和协议。此处以以太网(Ethernet)之上运行TCP/IP协议栈为例,各字段说明如下:

- 硬件类型(2字节)。指示硬件接口类型,0x0001表示Ethernet。
- 协议类型(2字节)。所使用的网络层协议类型,0x0800表示IP协议(与Ethernet V2帧中类型(T)字段相同)。
- 物理地址长度(1字节)。Ethernet物理地址长度为0x06字节(48位)。
- 协议地址长度(1字节)。网络层IP长度为0x4字节(32位)。
- 操作(2字节)。指示ARP分组的类型,其中0x0001表示ARP请求分组,0x0002表示ARP响应分组,0x0003表示RARP请求分组,0x0004表示RARP响应分组。
- 发送节点物理地址。发送ARP分组的主机的物理地址。
- 发送节点协议地址。发送ARP分组的主机的IP地址。
- 目的节点物理地址。真正接收ARP分组的主机的物理地址。对ARP请求分组,由于发送节点此时不知道目的节点的物理地址,故全部填入0。
- 目的节点协议地址。接收ARP分组的主机的协议地址。对RARP请求分组,由于发送节点此时不知道目的节点的协议地址,故全部填入0。

可以看出,RARP协议分组格式与ARP相同。RARP协议一般用于无盘工作站,可根据硬件地址查找IP地址。

3. ARP请求分组实例

在图3-12所示的Ethernet V2 MAC帧中,前面14字节为MAC帧首部,可以看出该帧为一个广播帧。帧的数据字段为ARP请求分组(Ethernet V2帧中类型字段为

0x0806)。图 3-13 中给出了各字段的解释。

图 3-12　ARP 请求分组实例

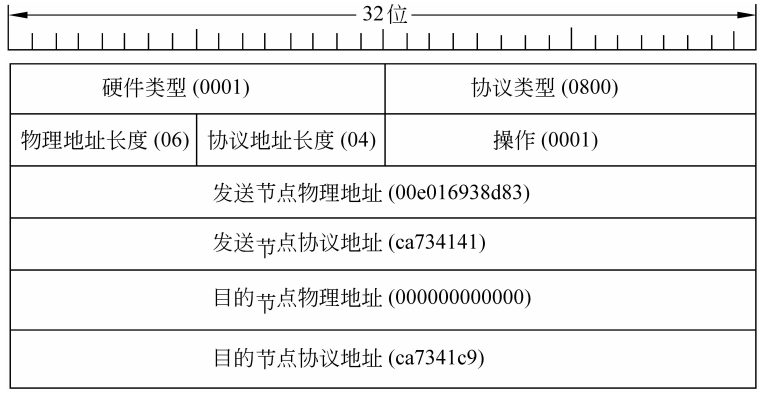

图 3-13　图 3-12 中 ARP 分组各字段的解释

可以看出,ARP 请求分组为 202.115.65.65(即 0xca734141)询问 202.115.65.201(即 0xca7341c9)的 MAC 地址。图 3-12 所示 Ethernet V2 帧中最后的 18 个 00 字节为帧的填充字节,以满足帧的最小长度。

3.4　TCP 协议栈

传输层是网络层之上的、端到端的协议。如前所述,经过网络层以下的处理,最终在 Internet 中为两用户建立了一条端到端通信链路。当然,由于层次之间的透明关系,传输层并不了解也不需要了解下层是如何工作的(正如两人通过电话交谈而不需要知道电话网是如何工作的一样)。

传输层的作用是提高服务质量(QoS)。虽然数据链路层的差错控制、流量控制功能保证了点到点的可靠性,但由于网络层是不可靠的,因此并不能保证传输层端到端的可靠性。传输层仍要完成差错控制、流量控制、检查分组是否丢失、多路复用、故障恢复等功能。

TCP/IP 协议栈的传输层协议有两个:面向连接的、提供可靠的按序传输的 TCP 协议,以及面向无连接的、无可靠性保证的 UDP 协议。

3.4.1　传输控制协议

1. TCP 报文格式

由于 TCP(传输控制协议)向高层提供面向连接的、可靠的服务,其协议数据单元(报

文)首部比较复杂。且TCP协议层次相对较高,接近用户应用程序,因此用户在开发网络应用程序时(如基于Socket编程)经常与TCP协议打交道,尤其对TCP报文的首部各字段含义及作用要熟练掌握。

图3-14表示了TCP报文格式。TCP报文分为首部和数据两部分,首部的前20字节是固定的,后面有4N字节的可选项(N为整数),因此TCP首部最小长度为20字节。通常Internet上的TCP报文的首部长度为20字节。

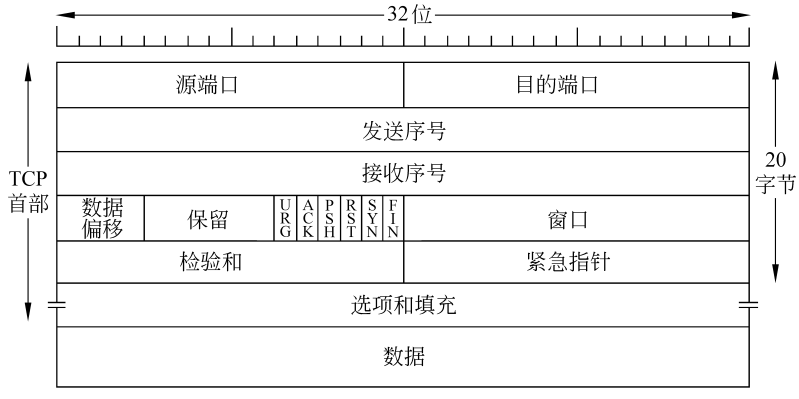

图3-14 TCP报文段的格式

首部固定部分各字段的意义如下:

- 源端口和目的端口。各占2个字节。端口(Port)是传输层与高层的服务接口,它可使若干高层应用向下复用在IP上。传输层通过端口向应用层特定应用进程服务。

端口的范围为0～65 535,分为以下三种类型:

(1) 公认端口(Well Known Ports)。从0到1023,它们紧密绑定于一些服务。通常这些端口的通信明确表明了某种服务的协议。例如,21表示FTP服务,23表示Telnet服务,25表示SMTP服务,53表示DNS服务,80表示WWW服务,110表示POP3服务等。当要访问远方WWW服务器时,对应TCP报文中,目的端口即为0x0050。

(2) 注册端口(Registered Ports)。从1024到49151,它们松散地绑定于一些服务。也就是说有许多服务绑定于这些端口,这些端口同样用于许多其他目的。例如,许多系统处理动态端口从1024左右开始。

(3) 动态和/或私有端口(Dynamic and/or Private Ports)。从49 152到65 535。理论上,不应为服务分配这些端口。实际上,机器通常从1024起分配动态端口。但也有例外,Sun公司的RPC端口从32 768开始。

随着网络应用的不断增加,目前实际的端口分配没有严格遵守上述范围划分。很多大于1023的端口也逐渐变成了公认端口。

16位的端口号加上32位的IP地址,总共48位,构成了插口或套接字(Socket)。在整个Internet中,通过采用插口这样的层次型地址结构,就使得在传输层进行通信的任何一对插口都是可以互相区分的。

- 发送序号(Seq)。4 字节,指明本报文段所发送的数据部分的第一个字节在 TCP 传送的数据流中的序号。TCP 传送的数据报文是按字节编号的,因此 TCP 是面向数据流的协议。
- 确认序号(期待接收的序号,Ack)。4 字节,是期望收到的下一个报文段的首部中的发送序号。

32 位长的序号字段可对 4GB 的数据进行编号。对一般应用的数据传输速率,当序号循环一周重复使用时,可保证旧序号的数据已不存在于网中。

- 数据偏移。占 4 位,指数据部分开始处与 TCP 报文段起始处的距离,即 TCP 报文段首部的长度,单位是 32 位。因可能存在长度不确定的选项字段,故首部长度不定,因此必须设置数据偏移字段。

数据偏移字段后面有 6 位是保留字段,供今后使用,目前应置为 0。

随后的 6 位是说明本报文段性质的控制字段(或称为标志),各位的意义解释如下:

- 紧急位 URG(URGent)。当 URG=1 时,表明本报文段应尽快传送,相当于加速数据,而不要按排队的次序来传送。例如,已经向远地主机发送了运行一个程序的命令,但随后发现有问题,要取消该程序的运行,因此从键盘发出中断信号,这就属于紧急数据。此时要与第 5 个 32 位字中的后一半“紧急指针(Urgent Pointer)”字段配合使用。需要加速传送的紧急数据放在数据部分的最前面,紧急指针指出在本报文段中紧急数据的最后一个字节的序号。紧急指针使接收方可以知道紧急数据共有多长。即便窗口大小为 0 时也可发送紧急数据。
- 确认位 ACK。只有当 ACK=1 时,确认序号字段才有意义。当 ACK=0 时,确认序号没有意义。注意同确认序号 ACK 的区别。
- 急迫位 PSH(PuSH)。当 PSH =1 时,表明请求远地 TCP 进程将本报文段立即传送给其应用层,而不要等到整个缓冲区都填满后再向上交付。
- 重建位 RST(ReSeT)。当 RST = l 时,表明出现严重差错(如由于主机崩溃或其他原因),必须释放连接,然后再重建传输连接。重建位也可用于拒绝一个非法的报文段或拒绝打开一个连接。
- 同步位 SYN。在建立连接时使用。当 SYN=1 而 ACK=0 时,表明这是一个连接请求报文段。对方若同意建立连接,则应在回应的报文段中置 SYN=1 和 ACK=1。同步位 SYN 置为 1,表示这是连接请求或连接接受报文,而 ACK 位则用于区分是哪一种报文。
- 终止位 FIN(FINal)。用于释放一个连接。当 FIN=1 时,表明要发送的字符串已经发送完毕,并要求释放传输连接。
- 窗口。2 字节,窗口字段表示该报文段发送方的接收窗口,单位为字节。通过此窗口告诉对方,在收到下一个确认前,发送方最多能发送此窗口大小的字节数。
- 检验和。2 字节,检验和字段检验的范围包括首部和数据。在计算检验和时,要在 TCP 报文段的前面加上一个 12 字节的伪首部(同 UDP)。伪首部的格式除了将第 4 个字段中的 17 改为 6(Internet 规定 TCP 的协议号是 6),以及第 5 个字段中的 UDP 长度改为 TCP 长度以外,其余部分与 UDP 数据报的伪首部一样。检

验和的计算方法与IP数据报首部检验和的计算方法一样,只是还要加上TCP报文的伪首部。接收端收到此报文段后,仍要加上这个伪首部来计算检验和。

- 选项。长度可变,其中一种选项是最长报文段(Maximum Segment Size,MSS)。该选项通知对方的TCP,本方缓冲区所能接收的报文段的最大长度是MSS。

选择MSS的最佳值比较困难。当MSS长度减小时,网络利用率就降低。若TCP报文段太长,在IP层传输时就可能需要分解成多个短数据报片。在目的站则要将这些片装配成原来的TCP报文段,而且当传输出错时还需重传。这些都会使开销增大。

研究认为,在IP层传输时不需要再分片的前提下,MSS应尽可能大一些。由于Ethernet最大帧长1518字节的限制,MSS通常取1460字节。在连接建立的过程中,双方都将自己能够支持的MSS写入这一字段。在以后的数据传送阶段,MSS取双方提出的较小的那个数值。若主机未填写此项,则MSS的默认值是536字节长的净负荷。因此,Internet上的主机都应能够接收的报文段长度是536+20=556字节。

2. TCP报文格式实例

同样以前面讲述Ethernet V2帧和IP数据报的例子为例,IP数据报中的数据字段即为一个TCP报文。如图3-15显示了此TCP报文的首部,其中各字段解释如图3-16所示。

图3-15 TCP数据报首部示例

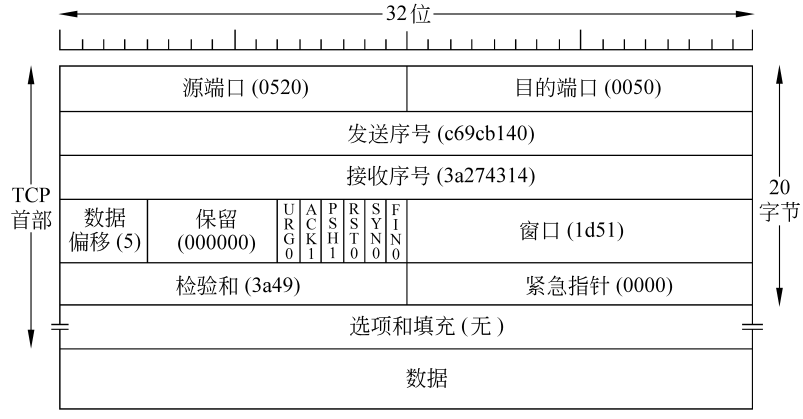

图3-16 图3-15所示TCP首部解释

可以看出，该报文是主机 202.115.65.31 访问远方主机 202.108.250.249 的 80 端口（IP 地址由 IP 数据报头得知），因此 TCP 报文中的数据字段应当是应用层 WWW 服务的 HTTP 协议格式。如若继续分析应用层 HTTP 协议，就可以知道该 Ethernet V2 帧到底要干什么。实际上该帧数据的含义是从 202.108.250.249（百度网站）这个 Web 服务器上获取关键词 MAC 的检索结果。在此不对应用层 HTTP 协议作详细分析。

3. TCP 连接的建立

TCP 协议是面向连接的传输层协议。收发双方通信之前先要建立连接，然后发方按序发送字节流，收方按序接收字节流并确认，通信完成后释放此连接。在前面介绍 TCP 报文首部字段时，介绍了几个与 TCP 连接建立与释放相关的位，分别是 SYN、ACK 和 FIN。

建立一个 TCP 连接需要解决的问题包括：使每一方确知对方的存在；允许双方协商一些参数（如报文段长度、窗口大小和服务质量等）；对传输实体使用的资源（如缓冲区和 TCP 连接表中的表项等）进行分配与登记等。

设一个服务器进程运行在主机 B 中，如图 3-17 所示。该服务器进程先发出被动打开（Passive Open）命令，通知 TCP 实体准备接收客户进程的连接请求。随后服务器进程处于“监听（Listen）”的状态，不断检测是否有客户进程的连接请求到来（实际系统可能有所不同），如有，即作出响应。

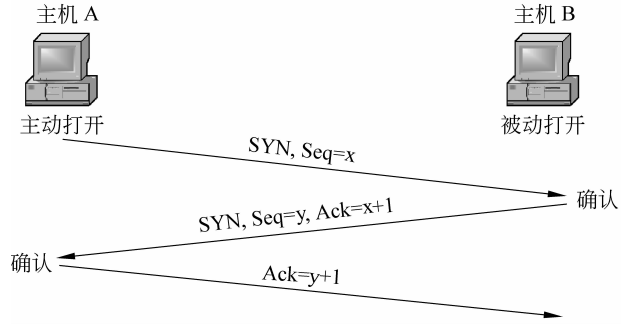

图 3-17　用三次握手建立 TCP 连接

连接建立采用三次握手的方法。

设在主机 A 中运行的客户进程先向其 TCP 实体发出主动打开（Active Open）命令，请求与主机 B 的某个端口建立传输连接。此请求连接建立报文将 TCP 首部中 SYN 位置 1，ACK 位置 0，主机 A 随机选一报文发送，序号为 Seq（此处设为 x）。

主机 B 的 TCP 实体收到连接请求报文后，如同意接收，则发回确认。在确认报文中将 SYN 置为 1，ACK 位置为 1，确认序号（Ack）应为 x+1，同时也为自己选择一个发送序号 y。

主机 A 的 TCP 实体收到此报文后，还要向 B 给出确认。确认报文的 ACK 位置 1，确认序号（Ack）为 y + 1。运行客户进程的主机 A 的 TCP 实体通知上层应用进程，连接已经建立。当运行服务器进程的主机 B 的 TCP 实体收到主机 A 的确认后，也通知其上层应用进程，连接已经建立。

4. TCP 连接的释放

在数据传输结束后,通信的双方都可以发出释放连接请求,以释放系统资源。释放连接过程如图 3-18 所示。

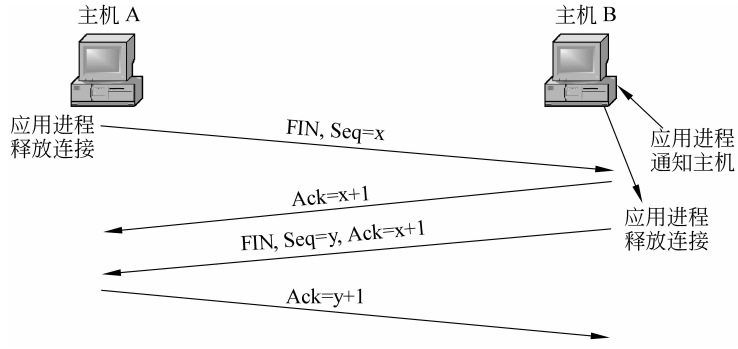

图 3-18 TCP 连接释放的过程

设图 3-18 中主机 A 的应用进程先向其 TCP 实体发出连接释放请求,并且不再将数据传送给传输层。主机 A 的 TCP 实体通知对方要释放从 A 到 B 这个方向的连接,将发往主机 B 的 TCP 实体的报文段首部的终止位 FIN 置 1,发送序号 x 等于前面已传送过的数据的最后一个字节的序号加 1。

主机 B 的 TCP 实体收到释放连接的通知后即发出确认,其确认序号为 x+1,同时通知高层的应用进程。这样,从 A 到 B 的连接就释放了,连接处于半关闭(Half-Close)状态,即主机 A 不再向主机 B 发送数据,但仍可接收 B 发过来的数据。主机 A 只要收到数据,仍应向主机 B 发送确认。

在主机 B 向主机 A 的数据发送结束后,其应用进程就通知 TCP 实体释放连接。主机 B 发出的连接释放报文段同样需将终止位 FIN 置 1,并使其发送序号 y 等于已传送过的数据的最后一个字节的序号加 1,还必须重复上次已发送过的确认序号 Ack=x+1。主机 A 必须对此发出确认,给出 Ack=y+1。于是,从 B 到 A 的反向连接也释放掉了。主机 A 的 TCP 实体再向其应用进程报告,整个连接已经完全释放。

上述连接释放过程和连接建立的三次握手方法类似。连接建立和释放过程可以从分析 TCP 报文之间的关系看出。借助于协议分析软件,整个过程更加清晰且易于理解。

3.4.2 用户数据报协议

1. UDP 数据报首部

UDP(用户数据报协议)是 TCP/IP 协议栈中另一个传输层协议。与 TCP 相比,UDP 提供无连接的服务,在传送数据之前不需要建立连接,远地主机的运输层在收到 UDP 数据报后不需要给出任何应答,因而传输效率较高。TCP/IP 体系中的某些应用,例如 TFTP、BOOTP、DNS 和 NFS 等就使用这种工作方式。

UDP 封装在 IP 数据报中,协议功能简单,只是在 IP 的数据报服务之上增加了端口的功能。UDP 分为数据字段和首部字段。如图 3-19 所示,首部字段只有 8 个字节,由 4

个字段组成，每个字段均占 2 字节。各字段意义如下：

- 源端口和目的端口字段。概念与 TCP 端口相同。
- 长度字段。UDP 数据报的总长度。
- 检验和字段。用于对 UDP 数据报进行差错检验。该字段的计算涉及伪首部的概念。

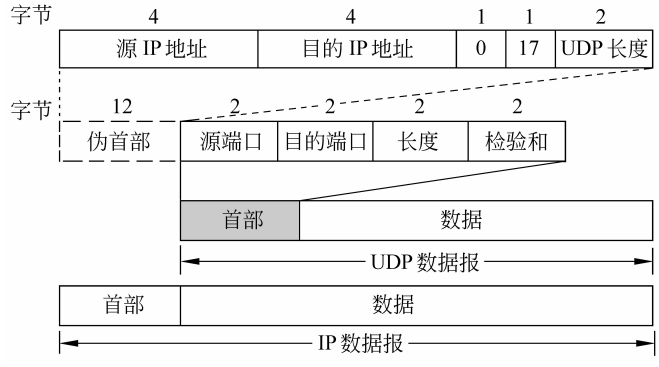

图 3-19　UDP 数据报的首部和伪首部

检验和字段的检验区间包括 UDP 数据报和在它之前添加的一个 12 字节长的伪首部。伪首部只是临时和 UDP 数据报连接在一起，以计算检验和，它既不向下传送，也不向上递交。伪首部各字段的内容如图 3-19 所示，其中第 3 字段是全 0，第 4 字段是 IP 首部中协议字段的值，对于 UDP，此协议字段值为 17。第 5 字段是 UDP 数据报的长度。

2. UDP 数据报实例

图 3-20 为捕获的一个 Ethernet V2 帧。除去前面 14 个字节的帧首部和 20 个字节的 IP 首部，之后即为一个 UDP 数据报。图 3-21 为此 UDP 数据报首部各字段解释。

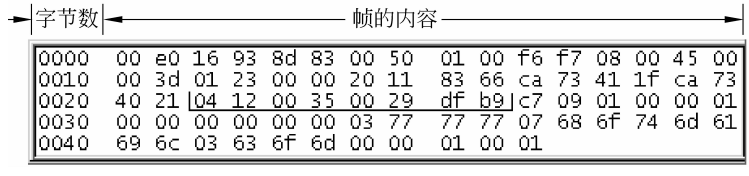

图 3-20　UDP 首部示例

源端口(0412)	目的端口(0035)	长度(0029)	检验和(dfb9)

图 3-21　图 3-20 中 UDP 首部字段解释

同样，UDP 数据报的数据字段为高层应用层协议。由目的端口(53)号可知，此 UDP 中封装的应用层协议为 DNS 协议。若按 DNS 协议继续分析 UDP 数据报的数据字段，即可知道该数据帧的含义。实际上，该 UDP 报文是 202.115.65.31 向 202.115.64.33(由 IP 首部可知)询问 www.hotmail.com 的 IP 地址(域名解析)。

传输层协议之上的应用层协议有很多种，如 HTTP、Telnet、FTP、SNMP 和 DNS 等，每种协议的内容比较复杂，不是这里介绍的重点，有兴趣的同学可以参看相关资料。

3.5 协议分析实验

3.5.1 实验目的

本章讲述局域网中常用的协议栈。由于局域网中用户计算机操作系统不同、运行的网络协议各不相同,甚至许多用户搞不清楚各种协议的作用,因此经常将所有的协议全部运行,以至于造成网络中有许多无用的协议数据包,造成网络资源的严重浪费。

在对局域网常用协议有了一定了解之后,本节完成一个网络协议分析实验,对捕获到的数据包逐字段分析,以加深对协议、协议封装及协议数据单元格式的理解。

3.5.2 实验内容

(1) 安装 WinPcap 和 Ethereal 应用软件;

(2) 运行 Ethereal 应用程序,抓取网络上的数据包;

(3) 分析本章讲述的各种协议的数据包,完成本章的思考题。

3.5.3 实验环境

本实验环境十分简单,在一台正常接入 Internet 的计算机上运行 Ethereal 软件。

Internet 上免费的包捕获与分析软件有很多,如 Windows 的网络监视器、Netxray、Sniffer Pro、Etherpeek、Ethereal 和 Tcpdump 等,这些软件的使用非常简单。本实验以 Windows 环境下的包捕获与分析软件 Ethereal(新的版本称为 wireshark)为例。

3.5.4 实验步骤

1. 安装与启动 Ethereal

当从网上下载下来,安装之后,由于该软件是基于 Winpcap 库,因此还需安装 Winpcap 库。Ethereal 应用程序的主界面如图 3-22 所示。

2. 启动抓包过程

在系统窗体上方的主菜单中选择 Capture→Start 命令,出现一个参数设置窗体。在 Interface 选项框中选择要捕获的网络接口(网卡),单击 OK 按钮。之后出现一个小窗体,动态显示程序的运行状况(已捕获各种协议的包的个数)。此时用户可进行一些网络操作(如浏览网页、收发邮件等),以便产生想要捕获的协议类型包。

3. 协议分析

当认为捕获的包数量足够时,单击小窗体的 Stop 按钮,程序将把所有捕获的包显示到最上面的窗体中,如图 3-22 所示。

窗体分为上、中、下三个子窗体。最上面的窗体显示了每一个数据包的摘要信息,如序号(No)、时间(Time)(No:应用程序按捕获的先后顺序编的号;Time:从程序开始运行起的计时信息。二者均非协议数据单元内容)、源地址(Source)、目的地址(Destination)、协议类型(Protocol:该数据包的最高层次协议类型)和该数据包的含义(Info)等。