

路由器 R_1 时，路由器 R_1 先收下它，接着把 TTL 的值减 1。由于 TTL 等于零了，因此 R_1 就把 P_1 丢弃，并向源主机发送一个 ICMP 时间超过差错报告报文。

源主机接着发送第二个数据报 P_2 ，并把 TTL 设置为 2。 P_2 先到达路由器 R_1 ， R_1 收下后把 TTL 减 1 再转发给路由器 R_2 。 R_2 收到 P_2 时 TTL 为 1，但减 1 后 TTL 变为零了。 R_2 就丢弃 P_2 ，并向源主机发送一个 ICMP 时间超过差错报告报文。这样一直继续下去。当最后一个数据报刚刚到达目的主机时，数据报的 TTL 是 1。主机不转发数据报，也不把 TTL 值减 1。但因 IP 数据报中封装的是无法交付的运输层的 UDP 用户数据报，因此目的主机要向源主机发送 ICMP 终点不可达差错报告报文（见 5.2.2 节）。

这样，源主机达到了自己的目的，因为这些路由器和最后目的主机发来的 ICMP 报文正好给出了源主机想知道的路由信息——到达目的主机所经过的路由器的 IP 地址，以及到达其中的每一个路由器的往返时间。图 4-31 是从南京的一个 PC 向新浪网的邮件服务器 mail.sina.com.cn 发出 tracert 命令后所获得的结果。图中每一行有三个时间出现，是因为对应于每一个 TTL 值，源主机要发送三次同样的 IP 数据报。

我们还应注意到，从原则上讲，IP 数据报经过的路由器越多，所花费的时间也会越长。但从图 4-31 可看出，有时正好相反。这是因为互联网的拥塞程度随时都在变化，也很难预料到。因此，完全有这样的可能：经过更多的路由器反而花费更短的时间。

```
C:\Documents and Settings\XXXR>tracert mail.sina.com.cn
Tracing route to mail.sina.com.cn [202.108.43.230]
over a maximum of 30 hops:
1  24 ms    24 ms    23 ms  222.95.172.1
2  23 ms    24 ms    22 ms  221.231.204.129
3  23 ms    22 ms    23 ms  221.231.206.9
4  24 ms    23 ms    24 ms  202.97.27.37
5  22 ms    23 ms    24 ms  202.97.41.226
6  28 ms    28 ms    28 ms  202.97.35.25
7  50 ms    50 ms    51 ms  202.97.36.86
8  308 ms   311 ms   310 ms  219.158.32.1
9  307 ms   305 ms   305 ms  219.158.13.17
10 164 ms   164 ms   165 ms  202.96.12.154
11 322 ms   320 ms   2988 ms 61.135.148.50
12 321 ms   322 ms   320 ms  freemail43-230.sina.com [202.108.43.230]

Trace complete.
```

图 4-31 用 tracert 命令获得目的主机的路由信息

4.5 IPv6

协议 IP 是互联网的核心协议。现在使用的协议 IP（即 IPv4）是在 20 世纪 70 年代末期设计的。互联网经过几十年的飞速发展，在 2011 年 2 月 3 日，IANA 开始停止向地区互联网注册机构 RIR 分配 IPv4 地址，因为 IPv4 地址已经全部耗尽了。不久，各地区互联网地址分配机构也相继宣布地址耗尽。我国在 2014 年至 2015 年也逐步停止了向新用户和应用分配 IPv4 地址，同时全面开始商用部署 IPv6。

解决 IP 地址耗尽的根本措施就是采用具有更大地址空间的新版本的 IP，即 IPv6。经过多年的研究和试验，2017 年 7 月终于发布了 IPv6 的正式标准[RFC 8200, STD86]。

4.5.1 IPv6 的基本首部

IPv6 仍支持无连接的传送，但将协议数据单元 PDU 称为分组(packet)，而不是 IPv4 的数据报(datagram)。为方便起见，本书仍采用数据报这一名词（[KURO17]和[TANE11]也是这样做的）。实际上，在本书中一直把分组和数据报看成是同义词。

IPv6 所引进的主要变化如下：

- (1) **更大的地址空间**。IPv6 把地址从 IPv4 的 32 位增大到 4 倍，即增大到 128 位，使地址空间增大了 2^{96} 倍。这样大的地址空间在可预见的将来是不会用完的。
- (2) **扩展的地址层次结构**。IPv6 由于地址空间很大，因此可以划分为更多的层次。
- (3) **灵活的首部格式**。IPv6 数据报的首部和 IPv4 的并不兼容。IPv6 定义了许多可选的扩展首部，不仅可提供比 IPv4 更多的功能，而且还可提高路由器的处理效率，这是因为路由器对扩展首部不进行处理（除逐跳扩展首部外）。
- (4) **改进的选项**。IPv6 允许数据报包含有选项的控制信息，因而可以包含一些新的选项。但 IPv6 的首部长度是固定的，其选项放在有效载荷中。我们知道，IPv4 所规定的选项是固定不变的，其选项放在首部的可变部分。
- (5) **允许协议继续扩充**。这一点很重要，因为技术总是在不断地发展的（如网络硬件的更新），而新的应用也还会出现。但我们知道，IPv4 的功能是固定不变的。
- (6) **支持即插即用**（即自动配置）。因此 IPv6 不需要使用 DHCP。
- (7) **支持资源的预分配**。IPv6 支持实时视像等要求保证一定的带宽和时延的应用。
- (8) IPv6 首部改为 8 字节对齐（即首部长度必须是 8 字节的整数倍）。原来的 IPv4 首部是 4 字节对齐。

IPv6 数据报由两大部分组成，即**基本首部(base header)**和后面的**有效载荷(payload)**。有效载荷也称为**净负荷**。有效载荷允许有零个或多个**扩展首部(extension header)**，再后面是数据部分（如图 4-32 所示）。但请注意，所有的扩展首部并不属于 IPv6 数据报的基本首部。

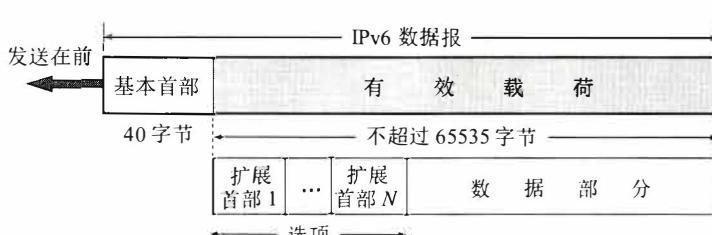


图 4-32 具有多个可选扩展首部的 IPv6 数据报的一般形式

与 IPv4 相比，IPv6 对首部中的某些字段进行了如下的更改：

- 取消了首部长度字段，因为它的首部长度是固定的（40 字节）。
- 取消了服务类型字段，因为优先级和流标号字段实现了服务类型字段的功能。
- 取消了总长度字段，改用有效载荷长度字段。
- 取消了标识、标志和片偏移字段，因为这些功能已包含在分片扩展首部中。
- 把 TTL 字段改称为跳数限制字段，但作用是一样的（名称与作用更加一致）。
- 取消了协议字段，改用下一个首部字段。
- 取消了检验和字段，这样就加快了路由器处理数据报的速度。我们知道，在数据链

路层对检测出有差错的帧就丢弃。在运输层，当使用 UDP 时，若检测出有差错的用户数据报就丢弃。当使用 TCP 时，对检测出有差错的报文段就重传，直到正确传送到目的进程为止。因此在网络层的差错检测可以精简掉。

- 取消了选项字段，而用扩展首部来实现选项功能。

由于把首部中不必要的功能取消了，使得 IPv6 首部的字段数减少到只有 8 个（虽然首部长度增大了一倍）。

下面解释 IPv6 基本首部中各字段的作用（参见图 4-33）。

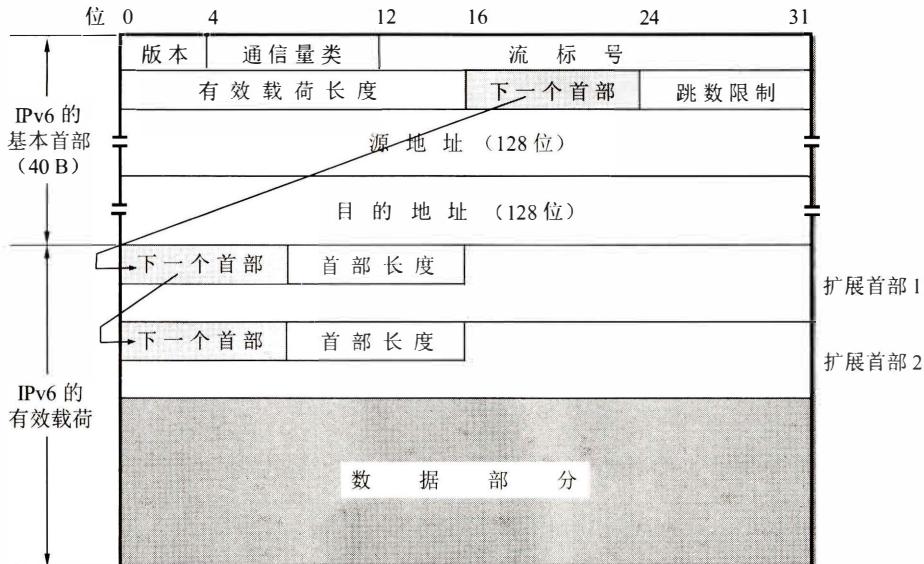


图 4-33 IPv6 基本首部和有效载荷（这里画出了两个扩展首部作为例子）

- 版本(version)** 占 4 位。它指明了协议的版本，对 IPv6 该字段是 6。
- 通信量类(traffic class)** 占 8 位。这是为了区分不同的 IPv6 数据报的类别或优先级，和 IPv4 的区分服务字段的作用相似。目前正在对不同的通信量类性能进行实验。
- 流标号(flow label)** 占 20 位。IPv6 的一个新的机制是支持资源预分配，并且允许路由器把每一个数据报与一个给定的资源分配相联系。IPv6 提出流(flow)的抽象概念。所谓“流”就是互联网络上从特定源点到特定终点（单播或多播）的一系列数据报（如实时音频或视频传输），而在一个“流”所经过的路径上的路由器都保证指明的服务质量。所有属于同一个流的数据报都具有同样的流标号。因此，流标号对实时音频/视频数据的传送特别有用。对于传统的电子邮件或非实时数据，流标号则没有用处，把它置为 0 即可。关于流标号的规约可参考建议标准[RFC 6437]。
- 有效载荷长度(payload length)** 占 16 位。它指明 IPv6 数据报除基本首部以外的字节数（所有扩展首部都算在有效载荷之内）。这个字段的最大值是 64 KB（65535 字节）。
- 下一个首部(next header)** 占 8 位。它相当于 IPv4 的协议字段或可选字段。
 - 当 IPv6 数据报没有扩展首部时，下一个首部字段的作用和 IPv4 的协议字段一样，它的值指出了基本首部后面的数据应交付 IP 层上面的哪一个高层协议（例如：6 或 17 分别表示应交付运输层 TCP 或 UDP）。
 - 当出现扩展首部时，下一个首部字段的值就标识后面第一个扩展首部的类型。

(6) 跳数限制(hop limit) 占 8 位。用来防止数据报在网络中无限期地存在。和 IPv4 的生存时间字段相似。源点在每个数据报发出时即设定某个跳数限制（最大为 255 跳）。每个路由器在转发数据报时，要先把跳数限制字段中的值减 1。当跳数限制的值为零时，就要把这个数据报丢弃。

(7) 源地址 占 128 位。是数据报的发送端的 IP 地址。

(8) 目的地址 占 128 位。是数据报的接收端的 IP 地址。

下面我们简单介绍一下 IPv6 的扩展首部。

在 RFC 8200 中定义了以下六种扩展首部：(1) 逐跳选项；(2) 路由选择；(3) 分片；(4) 鉴别；(5) 封装安全有效载荷；(6) 目的站选项。

每一个扩展首部都由若干个字段组成，它们的长度也各不同。但所有扩展首部的第一个字段都是 8 位的“下一个首部”字段。此字段的值指出了在该扩展首部后面的扩展首部是什么。当使用多个扩展首部时，应按以上的先后顺序出现。高层首部总是放在最后面。

大家知道，IPv4 的数据报若在其首部中使用了选项，则在数据报转发路径中的每一个路由器，都必须检查首部中的所有选项，看是否与本路由器相关。这必然要花费相当的时间。IPv6 把原来 IPv4 首部中选项的功能都放在扩展首部中。IPv6 数据报若使用了扩展首部，则其基本首部的“下一个首部”字段会指出，在“有效载荷”字段中使用了何种扩展首部。而所有扩展首部的第一个字段都是“下一个首部”，用来指出在后面还有何种扩展首部。这就使得路由器能够迅速判断待转发的 IPv6 数据报有无需要本路由器处理的选项。

4.5.2 IPv6 的地址

一般来讲，一个 IPv6 数据报的目的地址可以是以下三种基本类型地址之一：

扫一扫



视频讲解

(1) 单播(unicast) 单播就是传统的点对点通信。

(2) 多播(multicast) 多播是一点对多点的通信，数据报发送到一组计算机中的每一个。IPv6 没有采用广播的术语，而是将广播看作多播的一个特例。

(3) 任播(anycast) 这是 IPv6 增加的一种类型。任播的终点是一组计算机，但数据报只交付其中的一个，通常是按照路由算法得出的距离最近的一个。

IPv6 把实现 IPv6 的主机和路由器均称为节点。由于一个节点可能会使用多条链路与其他的一些节点相连，因此一个节点可能有多个与链路相连的接口。这样，IPv6 给节点的每一个接口（请注意，不是给某个节点）指派一个 IPv6 地址。一个具有多个接口的节点可以有多个单播地址，而其中任何一个地址都可当作到达该节点的目的地址。不过有时为了方便，若不会引起误解，也常说某个节点的 IPv6 地址，而把某个接口省略掉。

在 IPv6 中，每个地址占 128 位，地址空间大于 3.4×10^{38} 。如果整个地球表面（包括陆地和水面）都覆盖着计算机，那么 IPv6 允许每平方米拥有 7×10^{23} 个 IP 地址。如果地址分配速率是每微秒分配 100 万个地址，则需要 10^{19} 年的时间才能将所有可能的地址分配完毕。可见在想象到的将来，IPv6 的地址空间是不可能用完的。

为了体会一下 IPv6 的地址有多大，可以看一下目前已经分配出去的最大的地址块。法国电信 France Telecom 和德国电信 Deutsche Telekom 各分配到一个/19 地址块，相当于各有 35×10^{12} 个地址，远远大于全部的 IPv4 地址（IPv4 地址还不到 4.3×10^9 个）。

巨大的地址范围还必须使维护互联网的人易于阅读和操纵这些地址。IPv4 所用的点分十进制记法现在也不够方便了。例如，一个用点分十进制记法的 128 位的地址为：

104.230.140.100.255.255.255.255.0.0.17.128.150.10.255.255

为了使地址再稍简洁些，IPv6 使用冒号十六进制记法 (colon hexadecimal notation, 简写为 colon hex)，它把每个 16 位的值用十六进制值表示，各值之间用冒号分隔。例如，如果前面所给的点分十进制数记法的值改为冒号十六进制记法，就变成了：

68E6:8C64:FFFF:FFFF:0:1180:960A:FFFF

在十六进制记法中，允许把数字前面的 0 省略。上面就把 0000 中的前三个 0 省略了。

冒号十六进制记法还包含两个技术使它尤其有用。首先，冒号十六进制记法可以允许零压缩(zero compression)，即一连串连续的零可以为一对冒号所取代，例如：

FF05:0:0:0:0:0:0:B3

可压缩为：

FF05::B3

为了保证零压缩有一个不含混的解释，规定在任一地址中只能使用一次零压缩。该技术对已建议的分配策略特别有用，因为会有许多地址包含较长连续的零串。

其次，冒号十六进制记法可结合使用点分十进制记法的后缀。我们下面会看到这种结合在 IPv4 向 IPv6 的转换阶段特别有用。例如，下面的串是一个合法的冒号十六进制记法：

0:0:0:0:0:0:128.10.2.1

请注意，在这种记法中，冒号所分隔的每个值是两个字节（16 位）的值，但点分十进制每个部分的值是一个字节（8 位）的值。再使用零压缩即可得出：

::128.10.2.1

下面再给出几个使用零压缩的例子。

1080:0:0:0:8:800:200C:417A 记为 1080::8:800:200C:417A

FF01:0:0:0:0:0:0:101 (多播地址) 记为 FF01::101

0:0:0:0:0:0:0:1 (环回地址) 记为 ::1

0:0:0:0:0:0:0:0 (未指明地址) 记为 ::

CIDR 的斜线表示法仍然可用。例如，60 位的前缀 12AB00000000CD3 (十六进制表示的 15 个字符，每个字符代表 4 位二进制数字) 可记为：

12AB:0000:0000:CD30:0000:0000:0000:0000/60

或 12AB::CD30:0:0:0:0:0/60

或 12AB:0:0:CD30::/60

但不允许记为：

12AB:0:0:CD3/60 (不能把 16 位地址 CD30 块中的最后的 0 省略)

或 12AB::CD30/60 (这表示 12AB:0:0:0:0:0:0:CD30/60)

或 12AB::CD3/60 (这表示 12AB:0:0:0:0:0:0:0CD3/60)

但是，IPv6 取消了子网掩码。

斜线的意思和 IPv4 的情况相似。例如，

CIDR 记法的 $2001:0DB8:0:CD30:123:4567:89AB:CDEF/60$, 表示 IPv6 的地址是: $2001:0DB8:0:CD30:123:4567:89AB:CDEF$
而其子网号是: $2001:0DB8:0:CD30::/60$
IPv6 的地址分类如表 4-7 所示[RFC 4291]。

表 4-7 IPv6 的常用地址分类

地址类型	地址块前缀	前缀的 CIDR 记法
未指明地址	00…0 (128 位)	::/128
环回地址	00…1 (128 位)	::1/128
多播地址	11111111	FF00::/8
本地站点单播地址	1111111011	FEC0::/10
本地链路单播地址	1111111010	FE80::/10
全球单播地址	见图 4-34	

对表 4-7 所列举的几种常用地址简单解释如下。

未指明地址 这是 16 字节的全 0 地址, 可缩写为两个冒号 “::”。这个地址不能用作目的地址, 而只能将某台主机当作源地址使用, 条件是这台主机还没有配置到一个标准的 IP 地址。这类地址仅此一个。

环回地址 IPv6 的环回地址是 $0:0:0:0:0:0:0:1$, 可缩写为 ::1。它的作用和 IPv4 的环回地址一样。这类地址也是仅此一个。

多播地址 功能和 IPv4 的一样。这类地址占 IPv6 地址总数的 $1/256$ 。

本地站点单播地址(site-local unicast address) 有些单位的内部网络使用 TCP/IP 协议, 但并没有连接到互联网上。连接在这样的内部网络上的主机都可以使用这种本地站点地址进行通信, 但不能和互联网上的其他主机通信。这类地址占 IPv6 地址总数的 $1/1024$, 其用途和 IPv4 的专用地址是一样的。

本地链路单播地址(link-local unicast address) 这种地址是在单一链路上使用的。当一个节点启用 IPv6 时就自动生成本地链路地址(请注意, 这个节点现在并没有连接在某个网络上)。当需要把分组发往单一链路的设备而不希望该分组被转发到此链路范围以外的地方时, 就可以使用这种特殊地址。这类地址占 IPv6 地址总数的 $1/1024$ 。

全球单播地址 IPv6 的这一类单播地址是使用得最多的一类。曾提出过多种方案来进一步划分这 128 位的单播地址。根据 2006 年发布的草案标准 RFC 4291 的建议, IPv6 单播地址的划分方法非常灵活, 可以是如图 4-34 所示的任何一种。这就是说, 可把整个的 128 位都作为一个节点的地址。也可用 n 位作为子网前缀, 用剩下的 $(128 - n)$ 位作为接口标识符(相当于 IPv4 的主机号)。当然也可以划分为三级, 用 n 位作为全球路由选择前缀, 用 m 位作为子网前缀, 而用剩下的 $(128 - n - m)$ 位作为接口标识符。

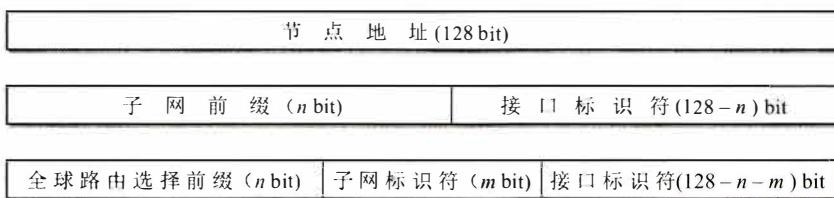


图 4-34 IPv6 的单播地址的几种划分方法

4.5.3 从 IPv4 向 IPv6 过渡

扫一扫



视频讲解

由于现在整个互联网的规模太大，因此，“规定一个日期，从这一天起所有的路由器一律都改用 IPv6”，显然是不可行的。这样，向 IPv6 过渡只能采用逐步演进的办法，同时，还必须使新安装的 IPv6 系统能够向后兼容。这就是说，IPv6 系统必须能够接收和转发 IPv4 分组，并且能够为 IPv4 分组选择路由。

下面介绍两种向 IPv6 过渡的策略，即使用双协议栈和使用隧道技术[RFC 2473, 2529, 3056, 4038, 4213]。

1. 双协议栈

双协议栈(dual stack)是指在完全过渡到 IPv6 之前，使一部分主机（或路由器）同时装有 IPv4 和 IPv6 这两种协议栈。因此双协议栈主机（或路由器）既能够和 IPv6 的系统通信，又能够和 IPv4 的系统通信。双协议栈的主机（或路由器）记为 IPv6/IPv4，表明它同时具有 IPv6 地址和 IPv4 地址（如图 4-35 所示）。

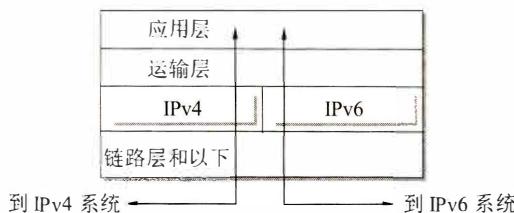


图 4-35 使用双协议栈进行从 IPv4 到 IPv6 的过渡

双协议栈的主机在和 IPv6 主机通信时采用 IPv6 地址，而和 IPv4 主机通信时则采用 IPv4 地址。但双协议栈主机怎样知道目的主机是采用哪一种地址呢？它是使用域名系统 DNS 来查询的。若 DNS 返回的是 IPv4 地址，则双协议栈的源主机就使用 IPv4 地址。但当 DNS 返回的是 IPv6 地址，源主机就使用 IPv6 地址。

双协议栈需要付出的代价太大，因为要安装上两套协议。因此在过渡期，最好采用下面的隧道技术。

2. 隧道技术

向 IPv6 过渡的另一种方法是隧道技术(tunneling)。图 4-36 给出了隧道技术的工作原理。这种方法的要点就是在 IPv6 数据报要进入 IPv4 网络时，把 IPv6 数据报封装成为 IPv4 数据报。现在整个的 IPv6 数据报变成了 IPv4 数据报的数据部分。这样的 IPv4 数据报从路由器 B 经过路由器 C 和 D，传送到 E，而原来的 IPv6 数据报就好像在 IPv4 网络的隧道中传输，什么都没有变化。当 IPv4 数据报离开 IPv4 网络中的隧道时，再把数据部分（即原来的 IPv6 数据报）交给主机的 IPv6 协议栈。图中的一条粗线表示在 IPv4 网络中好像有一个从 B 到 E 的“IPv6 隧道”，路由器 B 是隧道的入口而 E 是出口。请注意，在隧道中传送的数据报的源地址是 B 而目的地址是 E。

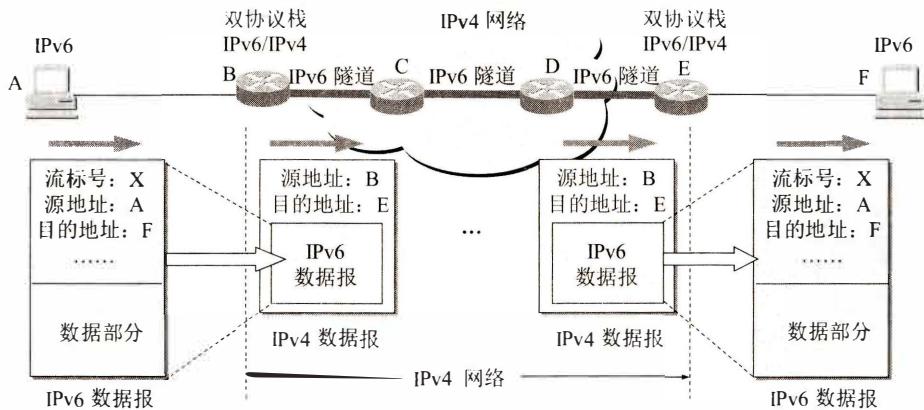


图 4-36 使用隧道技术进行从 IPv4 到 IPv6 的过渡

要使双协议栈的主机知道 IPv4 数据报里面封装的数据是一个 IPv6 数据报，就必须把 IPv4 首部的协议字段的值设置为 41（41 表示数据报的数据部分是 IPv6 数据报）。

4.5.4 ICMPv6

和 IPv4 一样，IPv6 也不保证数据报的可靠交付，因为互联网中的路由器可能会丢弃数据报。因此 IPv6 也需要使用 ICMP 来反馈一些差错信息。新的版本称为 ICMPv6，它比 ICMPv4 要复杂得多。地址解析协议 ARP 和网际组管理协议 IGMP 的功能都已被合并到 ICMPv6 中（如图 4-37 所示）。



图 4-37 新旧版本中的网络层的比较

ICMPv6 是面向报文的协议，它利用报文来报告差错，获取信息，探测邻站或管理多播通信。ICMPv6 还增加了几个定义报文功能及含义的其他协议。在对 ICMPv6 报文进行归类时，不同的文献和 RFC 文档使用了不同的策略，有的把其中的一些报文定义为 ICMPv6 报文，而把另一些报文定义为邻站发现 ND(Neighbor-Discovery)报文或多播听众交付 MLD(Multicast Listener Delivery)报文。其实所有这些报文都应当是 ICMPv6 报文，只是功能和作用不同而已。因此我们把这些报文都列入 ICMPv6 的不同类别。使用这种分类方法的原因是所有这些报文都具有相同的格式，并且所有报文类型都由 ICMPv6 协议处理。其实，像 ND 和 MLD 这样的协议都是运行在 ICMPv6 协议之下的。基于这样的考虑，可把 ICMPv6 报文分类，如图 4-38 所示。请注意，邻站发现报文和组成员关系报文分别是在 ND 协议和 MLD 协议的控制下进行发送和接收的。

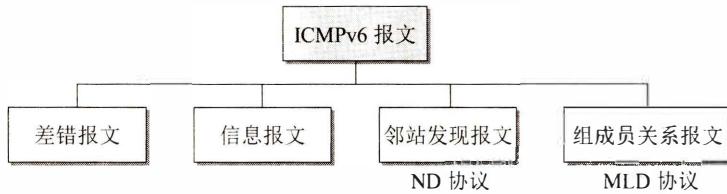


图 4-38 ICMPv6 报文的分类

关于 ICMPv6 的进一步讨论可参阅[FORO10]，这里从略。

4.6 互联网的路由选择协议

本节将讨论几种常用的路由选择协议，也就是要讨论转发表中的路由是怎样得出的。按照 4.1.2 节所述的观点，路由选择协议属于网络层控制层面的内容。不过本节仍然按传统的思路进行讨论，也就是说，路由选择协议规定了互联网中有关的路由器应如何相互交换信息并生成出路由表。

4.6.1 有关路由选择协议的几个基本概念

1. 理想的路由算法

路由选择协议的核心就是路由算法，即需要何种算法来获得路由表中的各项目。一个理想的路由算法应具有如下的一些特点[BELL86]：

- (1) **算法必须是正确的和完整的。** 这里，“正确”的含义是：沿着各路由表所指引的路由，分组一定能够最终到达目的网络和目的主机。
- (2) **算法在计算上应简单。** 路由选择的计算不应使网络通信量增加太多的额外开销。
- (3) **算法应能适应通信量和网络拓扑的变化，** 这就是说，要有自适应性。当网络中的通信量发生变化时，算法能自适应地改变路由以均衡各链路的负载。当某个或某些节点、链路发生故障不能工作，或者修理好了再投入运行时，算法也能及时地改变路由。有时称这种自适应性为“稳健性”(robustness)。^①
- (4) **算法应具有稳定性。** 在网络通信量和网络拓扑相对稳定的情况下，路由算法应收敛于一个可以接受的解，而不应使得出的路由不停地变化。
- (5) **算法应是公平的。** 路由选择算法应对所有用户（除对少数优先级高的用户）都是平等的。例如，若仅仅使某一对用户的端到端时延为最小，但却不考虑其他的广大用户，这就明显地不符合公平性的要求。
- (6) **算法应是最佳的。** 路由选择算法应当能够找出最好的路由，使得分组平均时延最小而网络的吞吐量最大。虽然我们希望得到“最佳”的算法，但这并不总是最重要的。对于某些网络，网络的可靠性有时要比最小的分组平均时延或最大吞吐量更加重要。因此，所谓“最佳”只能是相对于某一种特定要求下得出的较为合理的选择而已。

一个实际的路由选择算法，应尽可能接近于理想的算法。在不同的应用条件下，对以

^① 注：robustness 一词在自动控制界的标准译名是“鲁棒性”，但在[MINGCI94]则译为“稳健性”。