## 7.2.1 互联网协议第 4 版（IPv4）

要想明白 IPv4 的作用，你就需要知道流量是如何在不同网络之间传输的。IPv4 是这一通信过程的「搬运工」，并且不管通信端点在哪，它最终都负责在设备之间携带数据。

如果网络中的所有设备仅使用集线器或者交换机进行连接，那么这个网络称为局域网（local area network, LAN）。如果想将两个局域网连接起来，那么你可以使用路由器做到这一点。在复杂的网络中，可能包含了成千上万的局域网，而这些局域网则是由世界各地成千上万的路由器连接起来的。互联网本身就是由无数局域网和路由器所组成的一个集合。

### 1．IPv4 地址

IPv4 地址是一个 32 位的地址，用来唯一标识连接到网络的设备。由于让人记住一串 32 位长的 01 字符确实比较困难，因此 IP 地址采用点分四组的表示法。

在点分四组表示法中，构成 IP 地址的四组 1 和 0 中的每一组都转换为以十进制并以 A.B.C.D 的格式表示 0～255 之间的数字（见图 7-7）。我们拿这样一个 IP 地址 11000000 10101000 00000000 00000001 举例，这个值显然不容易记忆或者表示，但如果采用点分四组的表示法，我们就可以将其表示为 192.168.0.1。

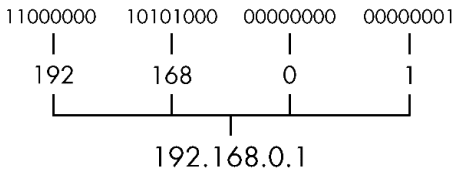


图 7-7　IPv4 地址的点分四组表示法

IP 地址之所以会被分成 4 个单独的部分，是因为每个 IP 地址都包含着两个部分：网络地址和主机地址。网络地址用来标识设备所连接到的局域网，而主机地址则标识这个网络中的设备本身。用来决定究竟 IP 地址哪部分属于网络或者主机的划分通常并不唯一。这实际上是由另一组名为网络掩码（network mask）的地址信息所决定的，有时它也会被称为子网掩码（subnet mask）。

注意

> 在本书中，如果我们提到 IP 地址，那么我们默认是指 IPv4 地址。我们将在后续小节里讲到 IPv6，它的地址有另一套规则。无论何时提到 IPv6 地址，我们都会明确标注出来。

网络掩码用来标识 IP 地址中究竟哪一部分属于网络地址而哪一部分属于主机地址。网络掩码也是 32 位的，并且网络掩码使用 1 的部分都是网络地址，而剩下为 0 的部分则标识着主机地址。

我们以 IP 地址 10.10.1.22 为例，其二进制形式为 00001010 00001010 00000001 00010110。为了能够区分出 IP 地址的每一个部分，我们将网络掩码应用其上。在这个例子中，我们的网络掩码是 11111111 11111111 00000000 00000000。这意味着 IP 地址的前一半（10.10 或者 00001010 00001010）是网络地址，而后一半（1.22 或者 00000001 00010110）标识着这个网络上的主机，如图 7-8 所示。

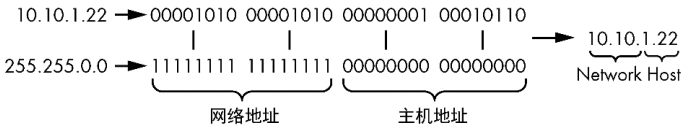


图 7-8　网络掩码决定了 IP 地址中比特位的分配

如图 7-8 所示，网络掩码也可以写成点分四组的形式。比如网络掩码 11111111 11111111 00000000 00000000 可以被写成 255.255.0.0。

为简便起见，IP 地址和网络掩码通常会被写成无类型域间选路（Classless Inter-Domain Routing，CIDR）的形式。在这个形式下，一个完整的 IP 地址后面会跟有一个左斜杠（/），斜杠右边的数字表示网络部分的位数。举例来说，IP 地址 10.10.1.22 和网络掩码 255.255.0.0，在 CIDR 表示法下就会被写成 10.10.1.22/16 的形式。

## 2．IPv4 头

源 IP 地址和目的 IP 地址都是 IPv4 数据报头中的重要组成部分，但除了这两个地址之外，数据包里面还有其他重要的信息。IP 报头比我们刚刚介绍过的 ARP 数据包要复杂得多。这其中包含了很多额外的信息，以便 IP 协议完成其工作。

如图 7-9 所示，IPv4 头有着下列的几个字段。

**版本号（Version）**：IP 所使用的版本。

**首部长度（Header Length）**：IP 头的长度。

**服务类型（Type of Service）**：优先级标志位和服务类型标志位，被路由器用来进行流量的优先排序。

**总长度（Total Length）：**IP 头与数据包中数据的长度。

| 互联网协议4（IPv4） | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 偏移位 | 八位组 | 0 | | 1 | 2 | | 3 | |
| 八位组 | 位 | 0–3 | 4–7 | 8–15 | 16–18 | 19–23 | 24–31 | |
| 0 | 0 | 版本号 | 首部长度 | 服务类型 | 总长度 | | | |
| 4 | 32 | 标识符 | | | 标识 | 分片偏移 | | |
| 8 | 64 | 存活时间 | | 协议 | 首部校验和 | | | |
| 12 | 96 | 源IP地址 | | | | | | |
| 16 | 128 | 目的IP地址 | | | | | | |
| 20 | 160 | 选项 | | | | | | |
| 24+ | 192+ | 数据 | | | | | | |

图 7-9　IPv4 数据包结构

**标识符（Identification）：**一个唯一的标识数字，用来识别一个数据包或者被分片数据包的次序。

**标识（Flags）：**用来标识一个数据包是否是一组分片数据包的一部分。

**分片偏移（Fragment Offset）：**一个数据包是一个分片，这个域中的值就会被用来将数据包以正确的顺序重新组装。

**存活时间（Time to Live）：**用来定义数据包的生存周期，以经过路由器的跳数/秒数进行描述。

**协议（Protocol）：**用来识别在数据包序列中上层协议数据包的类型。

**首部校验和（Header Checksum）：**一个错误检测机制，用来确认 IP 头的内容没有被损坏或者篡改。

**源 IP 地址（Source IP Address）：**发出数据包的主机的 IP 地址。

**目的 IP 地址（Destination IP Address）：**数据包目的地的 IP 地址。

**选项（Options）：**保留作额外的 IP 选项。它包含着源站选路和时间戳的一些选项。

**数据（Data）：**使用 IP 传递的实际数据。

### 3．存活时间

存活时间（TTL）值定义了在该数据包被丢弃之前所能经历的时间，或者能够经过的最大路由数目。TTL 在数据包被创建时就会被定义，而且通常在每次被发往一个路由器的时候减 1。举例来说，如果一个数据包的存活时间是 2，那么当它到达第一个路由器的时候，其 TTL 会被减为 1，并被发向第二个路由。接着这个路由会将 TTL 减为 0，这时如果这个数据包的最终目的地不在这个网络中，那么这个数据包就会被丢弃，如图 7-10 所示。由于 TTL 的值在技术上还是基于时间的，因此一个非常繁忙的路由器可能会将

TTL 的值减掉不止 1，但通常情况下，我们还是可以认为一个路由设备在多数情况下只会将 TTL 值减 1。

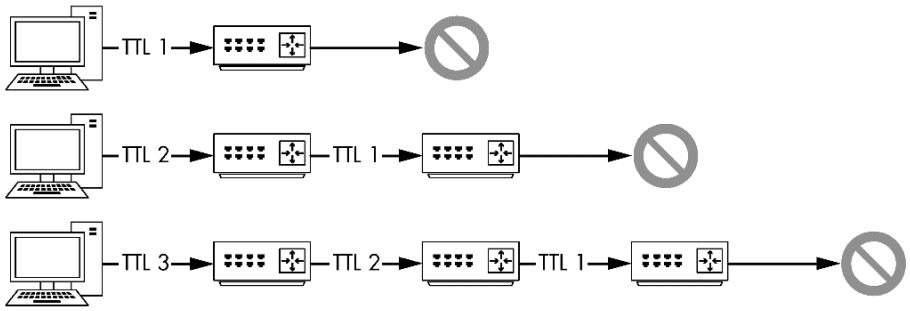


图 7-10　数据包的 TTL 在每次经过一个路由器的时候减少

为什么 TTL 的值会这样重要？我们通常所关心的一个数据包的生存周期，只是其从源前往目的地所花去的时间。但是考虑到一个数据包想要通过互联网发往一台主机需要经过数十个路由器，在这个数据包的路径上，它可能会碰到被错误配置的路由器，而失去其到达最终目的地的路径。在这种情况下，这个路由器可能会做很多事情，其中一件就是将数据包发向一个网络，而产生一个死循环。

如果你有编程背景，那么你就会知道死循环会引发各种问题，一般来说它会导致一个程序或者整个操作系统的崩溃。理论上，同样的事情也会以数据包的形式发生在网络上。数据包可能会在路由器之间持续循环。随着循环数据包的增多，网络中可用的带宽就会减少，直至拒绝服务（DoS）的情况出现。IP 头中的 TTL 域就是为了防止出现这个潜在的问题。

让我们看一下 Wireshark 中的实例。文件 ip_ttl_source.pcap 包含着两个 ICMP 数据包。ICMP（我们会在这章之后介绍到）利用 IP 传递数据包，我们可以通过在 Packet Details 面板中展开 IP 头区段看到（见图 7-11）。

你可以看到 IP 的版本号为 4，IP 头的长度是 20 字节，首部和载荷的总长度是 60 字节，并且 TTL 域的值是 128。

ICMP ping 的主要目的就是测试设备之间的通信。数据从一台主机发往另一个主机作为请求，而后接收主机会将那个数据作为响应发回。这个文件中，一台 IP 地址为 10.10.0.3 的设备将一个 ICMP 请求发向了地址为 192.168.0.128 的设备。这个原始的捕获文件是在源主机 10.10.0.3 上被创建的。

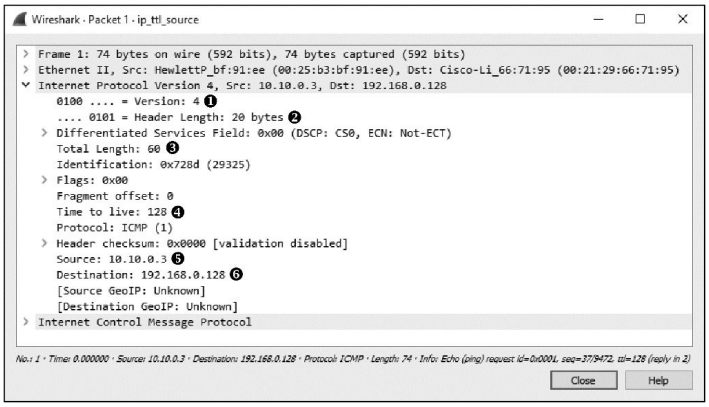


图 7-11　源数据包的 IP 头

现在打开文件 ip_ttl_dest.pcap。在这个文件中，数据在目的主机 192.168.0.128 处被捕获。展开这个捕获中第一个数据包的 IP 头，来检查它的 TTL 值（见图 7-12）。

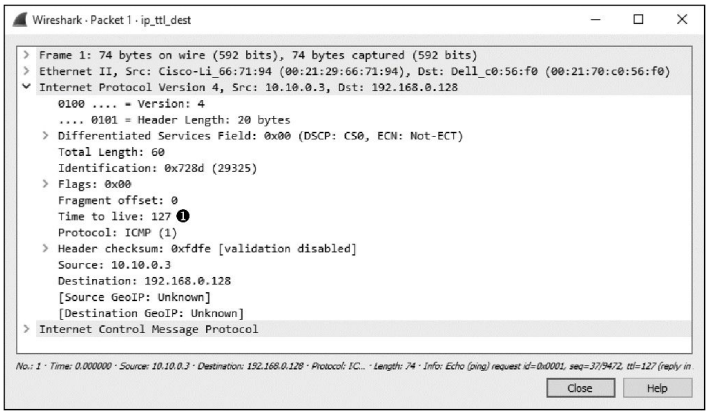


图 7-12　IP 头告诉我们 TTL 已经被减 1 了

你可以立刻注意到 TTL 的值变为了 127，比原先的 TTL 减少了 1。即使不知道网络的结构，我们也可以知道这两台设备是由一台路由器隔开的，并且经过这台路由器的路径会将 TTL 值减 1。

### 4．IP 分片

数据包分片将一个数据流分为更小的片段，它是 IP 用于解决跨越不同类型网络时可靠传输的一个特性。

一个数据包的分片主要基于第 2 层数据链路协议所使用的最大传输单元（maximum transmission unit, MTU）的大小，以及使用这些第 2 层协议的设备配置情况。在多数情况下，第 2 层所使用的数据链路协议是以太网。以太网的默认 MTU 是 1500，也就是说以太网的网络上所能传输的最大报文大小是 1500 字节（并不包括 14 字节的以太网头本身）。

> 注意

> 虽然存在着标准的 MTU 设定，但是一个设备的 MTU 通常可以手工设定。MTU 是基于接口进行设定的，其可以在 Windows 或者 Linux 系统上修改，也可以在托管路由器的界面上修改。

当一个设备准备传输一个 IP 数据包时，它将会比较这个数据包的大小，以及将要把这个数据包传送出去的网络接口 MTU，用以决定是否需要将这个数据包分片。如果数据包大小大于 MTU，那么这个数据包就会被分片。将一个数据包分片包括下列的步骤。

（1）设备将数据分为若干个将要接下来进行传输的数据包。

（2）每个 IP 头的总长度字段会被设置为每个分片的片段长度。

（3）除了最后一个分片数据包外，之前所有分片数据包的标志位都被标识为 1。

（4）IP 头中分片部分的分片偏移将会被设置。

（5）数据包被发送出去。

文件 ip_frag_source.pcap 是从地址为 10.10.0.3 的计算机上捕获而来的。它向一个地址为 192.168.0.128 的设备发送 ping 请求。注意，在 ICMP（ping）请求之后，Packet List 面板的 Info 列中列出了两个被分段的 IP 数据包。

先检查数据包 1 的 IP 头（见图 7-13）。

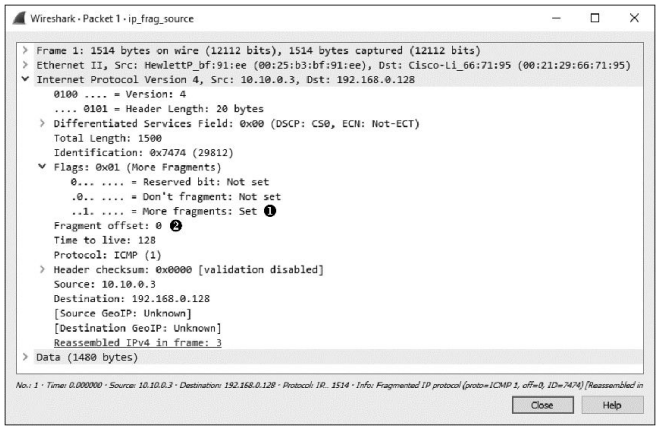


图 7-13　更多分片和分片偏移值可以用来识别分片数据包

根据更多分片和分片偏移域，你可以断定这个数据包是分片数据包的一部分。被分片的数据包可能有一个大于 0 的分片偏移，或者设定了更多分片的标志位。在第一个数据包中，更多分片标志位被设定，意味着接收设备应该等待接收序列中的另一个数据包。分片偏移被设为 0，意味着这个数据包是这一系列分片中的第一个。

第二个数据包的 IP 头（见图 7-14），同样被设定了更多分片的标志位，但在这里分片偏移的值是 1480。这里明显意味着 1500 字节的 MTU，减去了 IP 头的 20 字节。

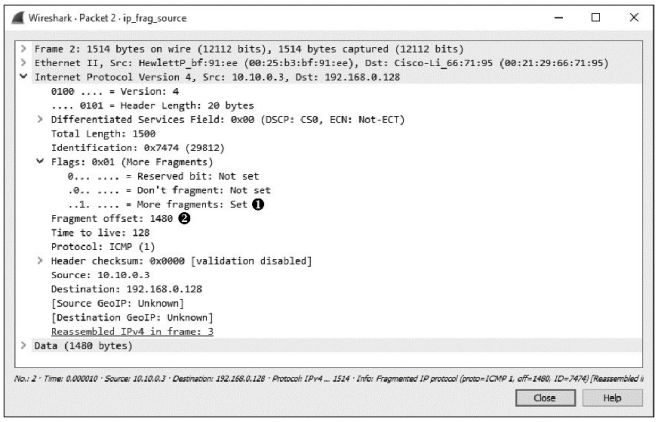


图 7-14　分片偏移值会根据数据包的大小而增大

第三个数据包（见图 7-15），并没有设定更多分片标志位，也就意味着它被标记为整个数据流中的最后一个分片。并且其分片偏移被设定为 2960，也就是 1480+（1500－20）的结果。这些分片可以被认为是同一个数据序列的一部分，因为它们在 IP 头中的标识位字段中拥有相同的值。

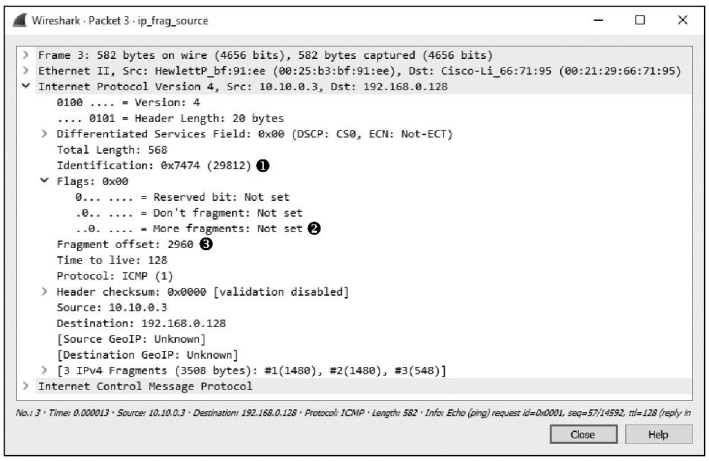


图 7-15　没有设置更多分片标志位意味着这是最后一个分片

虽然网络上被分片的包不怎么常见，但明白数据包为什么会被分片是有用的，这样当遇到它们时，你就可以诊断问题所在或认出丢失的分片。