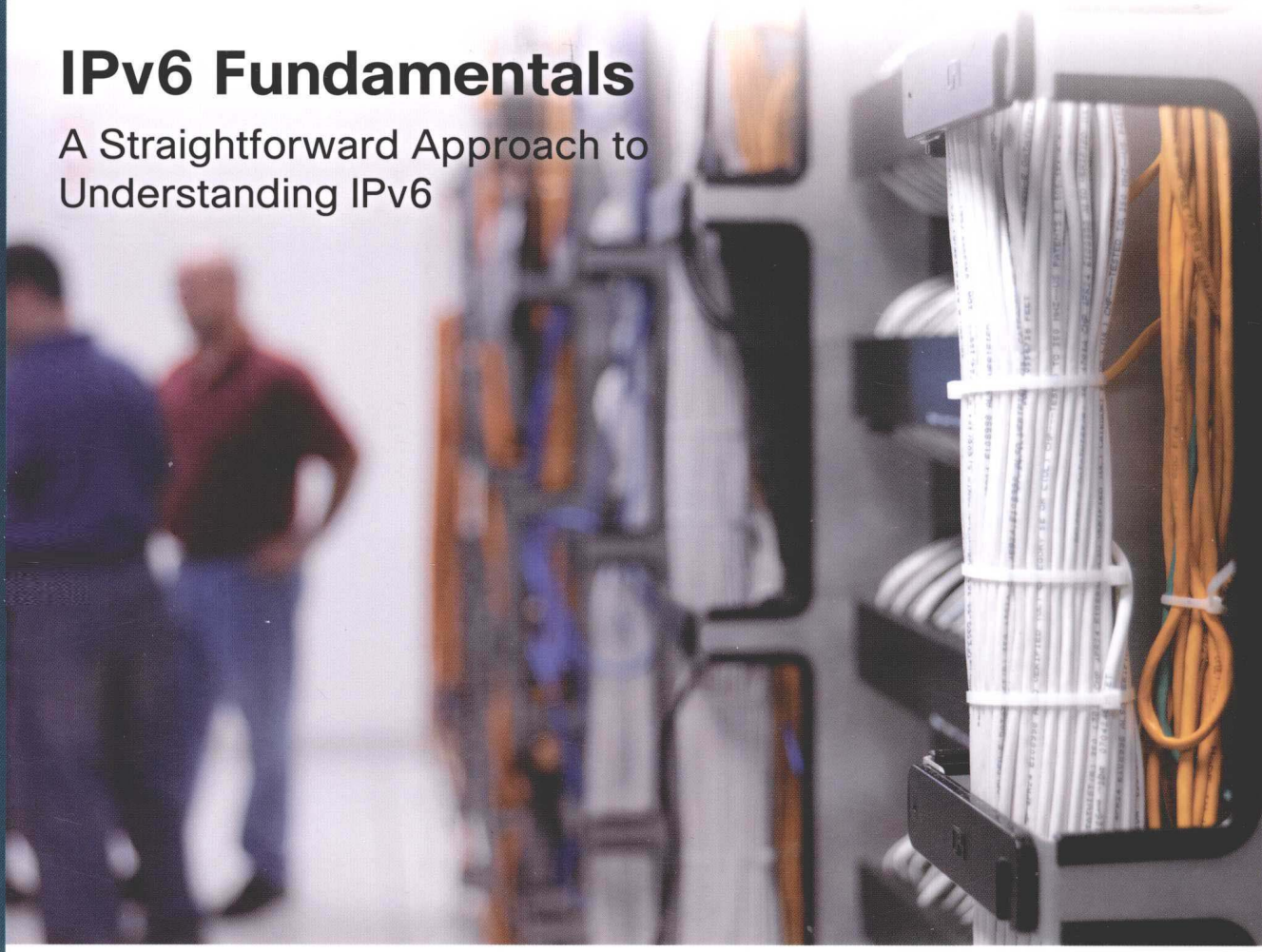
CISCO

ciscopress.com

# IPv6技术精要

## IPv6 Fundamentals

A Straightforward Approach to Understanding IPv6



〔美〕Rick Graziani 著
夏俊杰 译

人民邮电出版社
POSTS & TELECOM PRESS

# IPv6技术精要

- 理解IPv6是如何克服IPv4的一些重要局限性的；
- 通过对比IPv6与IPv4来理解IPv6做了哪些变化，哪些则保持不变；
- 讨论IPv6地址以及子网地址；
- 通过静态、动态、EUI-64、无编号、SLAAC以及DHCPv6等各种方式在路由器接口上启用IPv6功能；
- 利用ICMPv6和邻居发现协议改善网络的运行状况；
- 利用通用拓扑结构配置IPv6地址和访问控制列表；
- 分析IPv6路由器并配置IPv6静态路由；
- 对比、配置并验证每种IPv6 IGP路由协议；
- 部署状态化和无状态DHCPv6服务；
- IPv6与DNS、TCP和UDP等上层协议的集成；
- 利用双栈技术在同一台设备上同时运行IPv4和IPv6；
- 利用手工隧道、6to4隧道或ISATAP隧道实现IPv4与IPv6的共存；
- 利用NAT64实现IPv4向IPv6的平滑迁移。

为了保证商业应用的持续性、增长性和创新性，各类组织机构必须尽快向IPv6迁移，IPv6是定义计算机如何通过网络进行通信的下一代互联网协议。本书以简单易懂的方式向所有需要部署和管理IPv6网络的网络新手或资深网络专家全面介绍了IPv6的相关知识。

优秀的网络教师Rick Graziani以简洁明了的方式一步步地阐释了IPv6的方方面面，为大家成功掌握IPv6提供了所有的细节信息。在厘清各种基本概念的基础上，作者还提供了包括多种协议和多种进程在内的大量有用的技术信息，为大家未来几年将要开展的IPv6实践工作提供了大量宝贵的案头资源。

本书首先讨论了IPv6的必需性以及产生历史和工作方式，然后全面讨论了IPv6编址、配置选项和路由协议（包括RIPng、IPv6 EIGRP、OSPFv3）等内容，大家将了解如何将IPv6与IPv4集成，如何在全面迁移到IPv6之前实现两种协议的平滑共存。

Graziani在书中提供了所需的全部IOS命令语法，包括各种配置案例、示例拓扑以及与Cisco相关的IPv6配置技巧。此外，为了满足广大读者进一步学习的需要，书中还列出了大量Cisco白皮书以及官方IPv6 RFC的链接。

Rick Graziani在美国加利福尼亚的卡布利洛学院（位于阿普托斯）教授计算机科学和计算机网络课程，拥有30多年的计算机网络和IT领域工作与教学经验，目前为Cisco和其他重要客户提供咨询服务。在最近的Cisco网络学院会议上，Graziani有关IPv6基础和IPv6路由的专题演讲吸引了大批现场观众和大量网上观众。Graziani此前还曾在Santa Cruze Operaton公司、Tandem Computers公司和Lockheed公司工作过。

本书是Cisco Press出版的网络技术基础系列丛书中的一本，该系列丛书旨在为网络从业人员讲解各种网络新技术，包括网络拓扑、部署概念示例、协议以及管理技术等内容。

ciscopress.com

美术编辑：王建国

人民邮电出版社-信息技术分社
http://weibo.com/ptpitbooks

分类建议：计算机/网络技术/思科技术
人民邮电出版社网址：www.ptpress.com.cn

ISBN 978-7-115-31310-2

ISBN 978-7-115-31310-2

定价：69.00 元

ciscopress.com

# IPv6技术精要

## IPv6 Fundamentals

A Straightforward Approach to Understanding IPv6

〔美〕Rick Graziani 著
夏俊杰 译

人 民 邮 电 出 版 社
北 京

**图书在版编目（CIP）数据**

IPv6技术精要 / (美) 格拉齐亚尼 (Graziani,R.) 著 ; 夏俊杰译. -- 北京 : 人民邮电出版社, 2013.5
ISBN 978-7-115-31310-2

Ⅰ. ①I… Ⅱ. ①格… ②夏… Ⅲ. ①计算机网络一通信协议 Ⅳ. ①TN915.04

中国版本图书馆CIP数据核字(2013)第049358号

## 版 权 声 明

IPv6 Fundamentals: A Straightforward Approach to Understanding IPv6 (ISBN: 978-1587143137)
Copyright © 2013 Pearson Education, Inc.
Authorized translation from the English language edition published by Cisco Press.
All rights reserved.

本书中文简体字版由美国 Cisco Press 授权人民邮电出版社出版。未经出版者书面许可，对本书任何部分不得以任何方式复制或抄袭。

版权所有，侵权必究。

**IPv6 技术精要**

◆ 著 [美] Rick Graziani
译 夏俊杰
责任编辑 傅道坤

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子邮件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京鑫正大印刷有限公司印刷

◆ 开本：800×1000 1/16
印张：23.5
字数：473 千字 2013 年 5 月第 1 版
印数：1－3 000 册 2013 年 5 月北京第 1 次印刷

著作权合同登记号 图字：01-2012-7069 号

ISBN 978-7-115-31310-2

定价：69.00 元

读者服务热线：(010)67132692 印装质量热线：(010)67129223
反盗版热线：(010)67171154
广告经营许可证：京崇工商广字第 0021 号

# 内容提要

本书从IP的发展以及IPv4的不足入手，全面阐述了IPv6的各种基本知识，包括IPv6编址、ICMPv6与邻居发现协议、IPv6路由、DHCPv6等内容，并详细讨论了包括双栈、隧道、转换在内的各种IPv4向IPv6迁移的过渡技术。为便于读者深入掌握各章所学知识，本书通过大量实用案例详细阐述了IPv6地址以及各种IPv6协议与进程的Cisco IOS配置信息和配置命令，对于理解IPv6以及从IPv4向IPv6迁移提供了非常有用的参考内容。受篇幅限制，本书没有过多地去描述IPv4的相关内容，只是对IPv6与IPv4相关知识点进行了分析和对比，因而读者应具备基本的IPv4基础知识。

本书适用于任何希望学习IPv6基础知识的读者，包括所有从事网络架构和网络设计工作的专业人士以及广大网络专业的在校学生。

# 译者序

随着 IANA 将最后 5 个地址块分配给 5 大 RIR，其主地址池于 2011 年 2 月 3 日已正式宣告耗尽。世界各国范围内的 IPv4 地址枯竭问题已真真切切地摆到了大家面前，特别是中国互联网在经过十几年的快速发展之后，步入了一个崭新的发展天地，网民数早已跃居世界第一。互联网在中国也得到无所不在的广泛应用，互联网在带给大家便利的同时，也让几乎所有的人都无法离开互联网。因此，大家不得不正视 IPv4 向 IPv6 过渡的问题，虽然近些年关于 IPv6 试验或试商用部署的消息不时见诸报端，但业界对 IPv6 的认识却始终未能统一，包括运营商在内的 ISP 很少能真正意识到问题的迫切性和严重性。可喜的是，我国政府很早就认识到这一点，特别是 2012 年开始全面启动我国下一代互联网商用建设工程，并提出了明确的发展目标，计划在 2014～2015 年进入 IPv6 的全面商用阶段。

不过，向 IPv6 的发展演进是一个长期而复杂的进程，涉及互联网产业链的方方面面。由于 IPv6 仍然处于发展变化之中，很多技术还不成熟，再加上因陌生而产生的排斥感，使得大家对 IPv6 还存在着各种各样的疑惑或顾虑。因此相关网络技术人员应提前做好向 IPv6 过渡的技术储备，尽早熟悉 IPv6，尽早制定并测试本公司的 IPv6 迁移方案。

本书作者是互联网领域的资深专家，长期从事 IPv6 等相关领域的咨询和教育工作，具有丰富的实践经验，写作内容精炼实用，涵盖了包括编址、路由、过渡技术等在内的所有 IPv6 基础知识，并提供了大量有益的配置实例。本书不但便于读者学习理解，而且也极具实际参考价值。译者在翻译过程中收获良多，相信本书也一定能够为广大网络工程师提供大量有益信息和最佳实践。

在本书翻译过程中，得到了家人、朋友和编辑的无私支持与帮助，在此表示衷心感谢。本书内容涉及面广，虽然在翻译过程中为了尽量准确表达作者原意，特别是某些专有名词术语的译法，译者在多年网络通信工程经验的基础上，查阅了大量的相关书籍及标准规范，但由于时间仓促，加之译者水平有限，译文中仍难免有不当之处，敬请广大读者批评指正。

夏俊杰

# 关于作者

**Rick Graziani** 在美国加利福利亚州阿普托斯市的卡布利洛学院教授计算机科学和计算机网络课程。在此之前，他曾经在 SCO 公司、天腾电脑公司、洛克希德导弹和太空公司从事信息技术领域方面的工作。Rick 获得了加利福尼亚州立大学蒙特雷湾分校的计算机科学与系统理论的硕士学位，此外，Rick 还为 Cisco 公司提供咨询服务。在工作之余，Rick 通常都在其最喜欢的圣克鲁斯冲浪区享受冲浪带来的无穷乐趣。

# 关于技术审稿人

**Jim Bailey，CCIE #5275**（路由&交换、服务提供商），CCDE #20090008，是 Cisco 公司的服务技术负责人，拥有 20 多年的网络工作经验。作为中央工程网络架构与设计高级服务团队的一员，Jim 专注于企业、服务提供商以及政府客户的网络体系架构、网络设计与网络实施等领域。在过去的 7 年里，Jim 一直致力于如何更好地将 IPv6 集成到这些网络中。

**Yenu Gobena** 不但是 Cisco 公司无边界网络架构与设计领域负责战略方向的杰出服务工程师，而且还是将多项 Cisco 体系架构（如 BYOD、VDI/VXI、IPv6）融合到核心基础设施与数据中心的主要传播者和咨询者。Yenu 还致力于为全球范围内领先的 IPv6 先行者提供咨询与设计服务，拥有非常丰富的 IPv6 客户交付与研讨经验，还赢得了全球范围内几十家客户值得信赖的顾问角色，是 Cisco 在全球 IPv6 核心领域至关重要的行业发言人与代表。Yenu 在北卡罗莱纳州罗利市的三角研究园（Research Triangle Park）工作，研究兴趣广泛，包括路由和交换、MPLS、SP 移动性、IPv6 体系架构与设计、企业与 SP NGN 体系结构等。Yenu 拥有纽约州立大学的电信学士学位。他的两个孩子名叫 Amara 和 Elena，妻子是 Molly。此外，Yenu 还拥有 CCIE 认证，发表了多本白皮书，而且还是 IPv6 论坛认证的金牌培训师。

# 献辞

本书献给我的父母，感谢他们一直以来给予我的无私的爱与支持。

# 致谢

首先感谢我的朋友和同事们的大力支持。特别要感谢周四晚上 DPS（Dropped Packet Society）社团的所有朋友，感谢你们在“会议”期间及以后所给予的无私帮助，这里需要感谢的人很多，特别要感谢 Mark Boolootian、Dave Barnett 和 Jim Warner，感谢你们这些年来的技术探讨与解惑答疑，我们曾经在餐巾纸上涂画了无数个拓扑结构。

Jim Bailey 在 Cisco 公司所获得的荣誉远远超过了作为本书技术审稿人的简短评价，他使我变得更加聪明。为了保证本书的准确性和时效性，Jim 付出了巨大的精力。他的经验和知识背景对我编写本书是不可估量的财富，是本书当之无愧的幕后英雄，感谢 Jim 所做出的惊人工作。

感谢 Cisco 公司的 Yenu Gobena 对本书做的技术审稿工作，使得本书更加准确，从而大大提升了本书读者的阅读体验。

对于 Gerlinde Brady、Sue Nerton 和 Jim Griffin 的友谊与支持，我心存感激。当我致力于本书写作之时，是你们的努力才让卡布利洛学院的 CS/CIS 部门始终保持平稳运行。我为能够与你们以及 CS/CIS 部门的其他同仁共事而倍感欣慰。

写作本书让我享受了很多特权，因为是 Dennis Frezzo、Jeremy Creech、Karen Alderson、Sonya Stott、Wayne Lewis、Bob Vachon 以及其他许多在思科网络学院工作的人们，给了我参与此项工作的机会与荣誉，参与到改变世界范围内的学员生活的工作当中。你们不仅是我的同事，更是值得我尊敬的朋友。

感谢 Pat Farley 让我仍然能够继续我的冲浪课程，对于热爱冲浪的人来说，其重要性不言而喻。

特别感谢 Cisco Press 的执行编辑 Mary Beth Ray。感谢你在本书漫长成稿期间的耐心与理解，你总是给予我沉着的保证与指导，而且还是一名非常了不起的舞者。

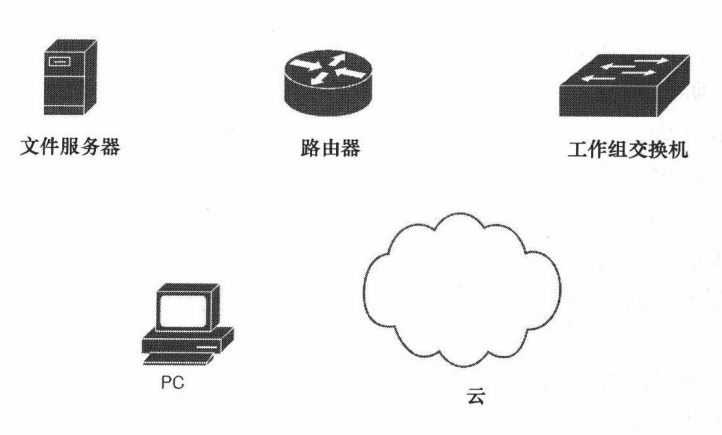
感谢 Cisco Press 的 Marianne Bartow，感谢你在本书出版过程中每天（包括周末）与我一起编辑、排版和策划。你每天都是那么愉快地工作着，非常感谢你为本书付出的努力工作，也让我觉得自己似乎的确是一个好作者。

感谢 Cisco Press 的 Chris Cleveland、Mandie Frank、John Edwards、Tim Wright、Mark Shirar、Sandra Schroeder 等诸位同仁。感谢你们为我做出的一切，我常常惊诧于写作一本技术书籍过程中所呈现出的合作与团队精神，感谢你们给予我的无私帮助。

感谢 Luigi，每天清晨都将我从睡梦中叫醒去沿着海边散步，这让我们俩都受益匪浅。真是我的好狗狗！

最后，我希望感谢我所有的学生，让我有机会去教授如此优秀的你们，而且让我的工作充满乐趣，让我有理由热爱我每一天的工作。

# 本书使用的图标




# 命令语法约定

本书命令语法遵循的惯例与 IOS 命令手册使用的惯例相同。命令手册对这些惯例的描述如下。

- **粗体字**表示照原样输入的命令和关键字，在实际的设置和输出（非常规命令语法）中，粗体字表示命令由用户手动输入（如 **show** 命令）。
- *斜体字*表示用户应提供的具体值参数。
- 竖线（|）用于分隔可选的、互斥的选项。
- 方括号（[]）表示任选项。
- 花括号（{}）表示必选项。
- 方括号中的花括号（[{}]）表示必须在任选项中选择一个。

# 前言

本书主要讨论 IPv6 的基础知识。IPv6 需要学习的东西很多，而不仅仅是拥有更大的地址空间。

写作本书时，我尽量以一种简单的、步骤式的方法来解释每一个概念，同时将关键细节包含在其中。既要向读者展示尽可能多的信息，又不至于让读者迎难而下，这是写作本书时的一大挑战。虽然 IPv6 并不难学，但毕竟包含了很多新协议和新进程。

大家不要被本书的细节信息所迷惑。例如，虽然我在本书中概要列出了各类协议的所有字段信息，但并不需要大家完全理解并掌握所有细节信息，这一点我已经在书中时时提到。我这样做的目的是认为不应该向读者遗漏或隐藏某些细节信息。

本书自始至终都在引用 RFC，将 RFC 作为重要参考的原因有二：首先，希望为广大读者提供权威的参考文件，以便能够获得更详细的有用信息；其次，IPv6 在相当长的时期内，始终处于发展变化之中。虽然 IPv6 已经出现了很多年，但是各种额外的开发和微调工作始终都在进行，如果大家对阅读 RFC 还不熟悉，那么也完全不必担心，因为大多数 RFC 阅读起来并不复杂，只是在尽力解释各种主题而已。

在介绍某种技术或概念时，我有时会声称“已经超出了本书写作范围”。我觉得大家在全面掌握了 IPv6 基本内容之后，最好再去阅读更高级的主题，因而在书中为感兴趣的读者提供了一些有用的参考资源。

本书的目的是尽可能清楚地解释 IPv6，因此有时我也很纠结，总在尽力决定应该在书中涵盖哪些主题。

欢迎广大读者使用我主页（www.cabrillo.edu/~rgraziani）上的 IPv6、CCNA 和 CCNP 资源，可以给我发电子邮件（graziani@cabrillo.edu）以获取上述资源的用户名和密码。

# 目标和方法

本书最重要的目标就是以一种易于理解的方式让广大读者全面掌握 IPv6 的主要内容。本书也希望为广大读者构筑坚实的 IPv6 基础，因而在书中列出了一些不易掌握的主题。

本书的另一个目标是希望成为 IPv6 的资源索引，因而在书中列出了大量命令语法、RFC 以及 Cisco 白皮书的链接，以方便广大读者进一步学习相关主题的知识。

# 本书阅读对象

本书适用于任何希望学习 IPv6 基础知识的读者，包括网络工程师、网络设计人员、网络技术人员、技术部门的员工以及网络专业的学生（思科网络学院的学生在内），读者应该熟悉难度相当于 CCNA 认证或思科网络学院课程的基本的 IPv4 与 IPv4 路由协议。

本书对于那些希望使用 Cisco 技术部署 IPv6 网络、提供 IPv6 连接以及在网络中使用 IPv6 的专业规划人员也能提供非常有用的参考内容。大家可以在书中找到配置 Cisco IOS IPv6 技术的大量案例、拓扑图以及 IOS 命令。

# 本书组织方式

如果从未学习过 IPv6，那么就应该从头至尾阅读本书。如果已经具备了一定的 IPv6 知识，那么就可以在本书的各个章节之间灵活地选择相关感兴趣的主题。

第 1 章到第 5 章介绍了 IPv6 的基本协议、编址类型和 ICMPv6 邻居发现协议，同时列出了一些 Cisco IOS 命令和配置案例。第 6 章到第 9 章通过一个通用示例拓扑解释了 IPv6 编址与 IPv6 路由协议的相关知识，此外，还讨论了 DHCPv6 及其他上层协议。最后两章（第 10 章和第 11 章）则讨论了从 IPv4 向 IPv6 迁移的过渡技术。如果希望通读全书，那么上述章节顺序就非常合适。

下面列出了本书各章的主要内容以及相应的组织方式。

**•第 1 章“IPv6 概述”**：本章讨论当今的互联网为何需要一种新的网络层协议（IPv6）来满足其用户的发展需求。本章首先分析了 IPv4 的一些局限性，描述了 IPv6 在提供各种优势的同时是如何解决这些局限性的。此外，还讨论了 IPv6 的基本原理以及与 IPv4 地址耗尽相关的问题，列出了 IPv4 和 IPv6 的发展简史，并讨论了包括 CIDR 和 NAT 在内的 IPv4 迁移技术。

**•第 2 章“IPv6 协议”**：本章讨论 IPv6 协议及其字段信息。本章首先回顾了 IPv4 协议并与 IPv6 做了详细对比，此外还讨论了分段的处理方式以及 IPv6 扩展报头等内容。

**•第 3 章“IPv6 编址”**：本章讨论 IPv6 编址与地址类型。本章首先解释了十六进制编号系统，讨论了 IPv6 地址表达形式、各种不同的 IPv6 地址格式以及 IPv6 压缩格式的规则等内容。本章概要性地描述了各种类型的 IPv6 地址，并讨论了 IPv6 地址的子网划分问题，包括在半字节边界划分子网和半字节内部划分子网。

**•第 4 章“IPv6 地址类型”**：本章详细讨论各种类型的 IPv6 地址。本章描述了全局单播地址的配置方法（包括手工配置和动态配置），解释了如何利用静态、EUI-64、IPv6 无编号、SLAAC（Stateless Address Autoconfiguration，无状态地址自动配置）和 DHCPv6

（有关 SLAAC 和 DHCPv6 的详细内容将在后续章节进行讨论）等配置方式在路由器接口上启用 IPv6，以具体的静态和动态 IOS 配置示例描述了链路本地地址的配置方式，并讨论了环回地址和未指定单播地址。最后，还讨论了已分配的多播地址、请求节点多播地址和任播地址等内容。

**•第 5 章“ICMPv6 与邻居发现协议”**：本章讨论 ICMPv6。虽然 ICMPv6 与 ICMPv4 很相似，但是 ICMPv6 是一种更健壮的协议。本章首先讨论了 ICMPv6 差错消息，包括目的地不可达、数据包超大、超时以及参数问题，然后讨论了 ICMPv6 通知消息（包括回显请求消息和回显应答消息）和多播侦听发现消息，接下来详细讨论了邻居发现协议、路由器请求消息、路由器宣告消息、邻居请求消息、邻居发现消息和重定向消息。IPv6 不仅解决了更大地址空间的问题，而且 ICMPv6 与邻居发现协议也在网络运行中做出了非常大的变动，包括链路层地址解析（IPv4 中的 ARP）、DAD（Duplicate Address Detection，重复地址检查）、SLAAC 以及 NUD（Neighbor Unreachability Detection，邻居不可达性检测），最后，本章还讨论了 IPv6 邻居缓存表和邻居缓存表状态（类似于 IPv4 ARP 缓存表）。

**•第 6 章“IPv6 配置”**：本章通过一个通用示例拓扑结构解释了 IPv6 编址方面的配置。全局单播地址和链路本地地址使用了不同选项进行配置。此外，本章还给出了邻居缓存表以及修改路由器宣告消息以调节邻居发现参数的配置案例，并以相同的拓扑结构为例解释了 IPv6 访问控制列表的配置。

**•第 7 章“IPv6 路由概述”**：本章讨论 IPv6 路由表以及与 IPv6 相关的配置变化。此外，还讨论了与 IPv4 静态路由相似的 IPv6 静态路由的配置以及 IPv6 CEF 等内容。

**•第 8 章“IPv6 IGP 路由协议”**：本章讨论包括 RIPng、IPv6 EIGRP 和 OSPFv3 在内的三种路由协议。本章分析了这些路由协议与其 IPv4 对等协议之间的区别，并通过通用示例拓扑结构解释了每种路由协议的配置与验证情况。

**•第 9 章“DHCPv6”**：本章讨论 DHCPv6，包括状态化 DHCPv6 和无状态 DHCPv6 服务。本章解释了 DHCPv6 术语和消息类型，并讨论了客户端与服务器之间的 DHCPv6 进程。此外，本章还讨论快速分配选项和中继代理等内容，并在最后简要讨论了 DNS、TCP 和 UDP 等上层协议。

**•第 10 章“双栈与隧道”**：本章讨论 IPv4 向 IPv6 迁移与共存阶段的三大机制中的两种机制：双栈机制与隧道机制。双栈用于同时部署 IPv4 协议和 IPv6 协议的设备，可以让这两种协议共存于同一个网络中，而隧道机制则是将 IP 包封装到另一个 IP 包中。本章讨论了基本的隧道概念，并解释了包括手工隧道、6to4 隧道和 ISATAP 隧道在内的常见隧道技术的配置情况。

**•第 11 章“NAT64”**：本章讨论了 IPv4 向 IPv6 迁移与共存阶段的第三种机制：NAT64。与 IPv4 NAT 相似，IPv6 NAT 也是在 IPv4 协议与 IPv6 协议之间进行转换。NAT64 是 NAT-PT 的替代技术，为了保持较新的 NAT64 技术与目前仍然在用的技术之间的连续性，本章也讨论了 NAT-PT 技术。

# 目录

# 第1章 IPv6概述

IPv6（Internet Protocol version 6，因特网协议版本6）是IPv4的后续版本，本章将分析当今互联网为何需要一套新的网络层协议来满足其用户的需求，在分析IPv4局限性的基础上，将讨论IPv6如何解决这些局限性并提供了许多其他优势。

## 1.1 IPv4

IPv4（Internet Protocol version 4，互联网协议版本4）是目前互联网以及绝大多数网络正在使用的三层协议，至今已经应用了30多年，早已成为互联网发展演进过程中密不可分的组成部分。IPv4最初由RFC 760定义（1980年1月），后来被RFC 791（1981年9月）所取代。

大家能否想象互联网在发展初期的状况呢？即便是在20世纪90年代初期，WWW（World Wide Web，万维网）出现之后，当时全球的互联网用户数大约只有1600万。而到了2011年，互联网用户数已经突破了20亿。考虑到人们通常会同时拥有多台具有上网功能的终端设备（如智能手机、平板电脑和笔记本电脑），因而实际的互联网设备数量呈现激增的状态。

有关从IPv4向IPv6迁移的详细理由将在本章的后面章节中进行讨论，现在大家需要知道的一点就是，当前的互联网与30年前、10年前甚至5年前的互联网是完全不一样的。如今的互联网已不再仅仅是网页、电子邮件和文件传送，而是随着移动终端与对等网络的爆炸式增长，再加上大量具备联网能力的消费产品所带来的潜在影响，原先的计算机互联网（Internet of computers）已逐渐演化成物联网（Internet of things）。术语物联网是一个不断发展变化的概念，最初指代的是具有RFID（Radio-Frequency Identification，射频识别）标签并用于识别和分类的日常用品，目前的概念非常宽泛，

包括智能电网、智慧互连城市、家庭自动化以及自动计算等。

一旦设备之间具备了通信能力，那么这些设备对网络的影响将是非常巨大的。例如，如果日程表显示老板要在第二天开早会，那么该数据会被传送给闹钟，以便能在 10 分钟前触发闹铃并开始为您煮咖啡。在您离开办公室的 5 分钟之前，汽车会自动预热并融化挡风玻璃上的积冰，识别您的驾车计划并及时通知您应该在回家的路上加油。

虽然这些听起来有些遥远，但大家想象一下，如果物品之间能够进行相互通信，那么这些就完全有可能成为现实。具备联网能力的设备可以是电视机、网络摄像头，也可以是关键的汽车引擎部件中的传感器或者是进行医疗诊断时的简易监视器。可以利用具备联网能力的传感器来提醒驾驶员当前的汽车胎压过低，或者能够通过监控中央位置的传感器来测量并实时调整空调系统，以节省能耗，从而节约成本。

本章将在后面的内容中讨论当前 IPv4 地址空间的耗用情况。虽然人们采取了诸如 NAT（Network Address Translation，网络地址转换）等技术来延长 IPv4 的地址寿命，但是很明显，我们已经走到了迫切需要一套新的网络层协议的重要关头。有关 NAT 及其对 IPv4 地址空间的节约信息将随后讨论。

## 1.2 早期的互联网

虽然早期被称为 ARPANET 的互联网起源于 1969 年，但当前互联网中使用的大量协议和技术都是最近的产物。RFC 2235（Hobbes Internet Timeline）对互联网在 1957 年至 1997 年之间的早期发展情况作了总结。

- 1957 年：USSR 发射了 Sputnik（第一颗人造地球卫星）。作为回应，美国 DoD（国防部）发起了 ARPA（Advanced Research Projects Agency，高级研究计划署）计划，以便在科技领域建立美国的领先地位。
- 1962 年：Paul Baran 发表了“关于分布式通信网络”的研究论文，这是分组交换网络概念的先驱。
- 1969 年：DoD 发起 ARPANET 以开展网络互连研究。第一个节点（大型机）位于 UCLA（University of California Los Angeles，加州大学洛杉矶分校）网络测量中心。其余的三个节点分别是 SRI（Stanford Research Institute，斯坦福研究所）、UCSB（University of California Santa Barbara，加州大学圣塔芭芭拉分校）以及犹他大学（University of Utah）。第一台路由器是 IMP（Information Message Processor，信息报文处理器），这是由 BBN（Bolt Beranek and Newman Inc.，贝洛奈克与纽曼公司）开发的拥有 12KB 内存的 Honeywell 516 迷你计算机。

- 1969 年：Steve Crocker 撰写了第一篇 RFC（Request for Comments，请求注解）“Host Software”。
- 1971 年：ARPANET 拥有了 15 个节点（23 台主机）：UCLA、SRI、UCSB、犹他大学、BBN、MIT（Massachusetts Institute of Technology，麻省理工学院）、RAND 公司、SDC（System Development Corporation，系统开发公司）、哈佛大学、MIT 林肯实验室、斯坦福大学、伊利诺伊大学香槟城分校、凯斯西储大学、卡内基梅隆大学以及 NASA 艾姆斯研究中心。
- 1971 年：BBN 的 Ray Tomlinson 发明了电子邮件程序，可以通过分布式网络发送信息。
- 1973 年：Bob Metcalfe 的哈佛博士论文提出了以太网的概念。
- 1973 年：制定了 FTP（File Transfer Protocol，文件传送协议）规范（RFC 454）。
- 1974 年：Vint Cerf 和 Bob Kahn 发表了论文“分组网络的相互通信协议”，详细描述了 TCP（Transmission Control Protocol，传输控制协议）的设计方案。
- 1982 年：ARPA 将 TCP/IP 作为 ARPANET 的协议簇，第一次将“Internet（互联网）”定义为通过 TCP/IP 互连的多个网络。
- 1982 年：发布了外部网关协议规范（RFC 827）。EGP（External Gateway Protocol，外部网关协议规范）是网络之间的路由协议，于 1994 年被 BGP（Border Gateway Protocol，边界网关协议）所代替（RFC 1656）。
- 1983 年：互联网在 1 月 1 日从 NCP（Network Control Protocol，网络控制协议）迁移到 TCP/IP。
- 1984 年：RFC 920 定义了 DNS（Domain Name System，域名系统）。
- 1984 年：互联网上的主机数突破了 1 000 台。
- 1986 年：NSFNET（National Science Foundation Network，美国国家自然科学基金网络）开始运营速率为 56kbit/s 的骨干网。
- 1987 年：互联网上的主机数突破了 10 000 台。
- 1988 年：NSFNET 骨干网速率升级到 T1（1.544Mbit/s）。
- 1988 年：Jarkko Oikarinen 开发了 IRC（Internet Relay Chat，互联网中继聊天）。
- 1989 年：互联网上的主机数突破了 100 000 台。
- 1989 年：Clifford Stoll 出版了 *Cuckoo's Egg* 一书，揭示了曾经入侵了大量美国设备的某个德国黑客组织的现实生活。
- 1990 年：第一台可远程操作的机器被挂到了互联网上，这台被称为 Internet Toaster 的机器首次出现在 Interop（IT Expo and Conference，IT 博览会暨研讨会）。

- 1991 年，CERN 发布了由 Tim Berners-Lee 开发的 WWW。
- 1991 年：NSFNET 骨干网速率升级到 T3（44.736Mbit/s）。
- 1992 年：互联网上的主机数突破了 1000 000 台。
- 1992 年：Jean Armour Polly 创造了“上网冲浪”一词。
- 1993 年：美国白宫 www.whitehouse.gov 上线。
- 1994 年：首次出现网络购物。
- 1994 年：可以使用 WWW 在网上订购必胜客的比萨。
- 1995 年：WWW 开始超越 FTP，成为互联网上流量最大的应用。
- 1995 年：在线拨号服务提供商 Compuserve、America Online 和 Prodigy 开始提供互联网接入服务。
- 1995 年：梵蒂冈上线。
- 1996 年：互联网电话开始引起美国电信企业的关注，要求美国国会取缔该项技术。
- 1996 年：富有争议的美国 CDA（Communications Decency Act，传播净化法案）正式成为美国法律，禁止通过互联网传播不道德或有伤风化的内容。三个月之后，由三名联邦法官组成的审判小组否决了这项法案。随后美国最高法院在 1997 年一致认为该法案的大部分内容违宪。
- 1996 年：MCI 升级其互联网骨干，将有效速率从 155Mbit/s 提升到 622Mbit/s。
- 1996 年：WWW 浏览器之争（主要在网景和微软之间）引发了新时代软件开发的仓促性。为了满足互联网用户渴望体验新（测试）版本的需求，每个季度都会推出带有新功能的软件版本。
- 1996 年：世界范围内各种限制措施开始施加到互联网用户身上：
  - 德国禁止用户访问 Compuserve 运营的某些新闻组；
  - 沙特阿拉伯限制用户通过互联网访问大学和医院；
  - 新加坡要求提供政治和宗教内容的服务提供商向国家注册。
  - 新西兰将计算机光盘视为“出版物”，可以进行审查和查没。
- 1997 年：whois 数据库中拥有了 101 803 台命名服务器。
- 1997 年：互联网上的主机数突破了 19 000 000 台。

互联网是一个发展变化的网络。1983 年引入 IPv4 的时候，虽然互联网上只有 600 台主机，但 IPv4 已经设计了 43 亿个可能的 IP 地址。虽然如今仍然应用了很多相同的分组交换概念，但如今的互联网用户数以及应用方式都产生了极大的变化。

> 注：如果希望更深入地了解互联网的诞生历史以及相应的奠基人，要求大家阅读由 Katie Lyons 撰写的最佳畅销书 *Where Wizards Stay Up Late: The Origins of the Internet*。

## 1.3　IPv5

IPv5 的情况如何呢？20 世纪 70 年代后期制定了一组实验协议簇，被称为互联网 ST（Stream Protocol，流协议）以及后来的 ST2，最初定义在互联网工程备忘录（Internet Engineering Note）IEN-119（1979）中，后来又在 RFC 1190 和 RFC 1819 中进行了修订。

ST 是一种实验性的资源预留协议，目的是为实时多媒体应用（如视频和语音）提供 QoS（Quality of Service，服务质量）保障。ST 由 ST 和 SCMP（Stream Control Message Protocol，流控制报文协议）两种协议组成。互联网流协议版本 2（ST-II 或 ST2）的设计目的并不是要取代 IPv4，而是希望多媒体应用同时使用这两类协议——IP4 用于传送传统数据包，而 ST-II 则用于传送承载了实时数据的数据包。

虽然业界并没有公开承认 IPv5，但是当 ST 封装在 IP 报文中时，使用的协议号却是 5（RFC 1700）。也就是说，虽然从未真正实现过，但“IP 版本 5”的名称早已确定。目前用于资源预留的标准是传输层协议 RSVP（Resource Reservation Protocol，资源预留协议），可以实现在 IPv4 上由接收端发起的资源预留请求。有关 RSVP 的详细描述定义在 RFC 2205 中。

## 1.4　IPv6 的历史

IETF（Internet Engineering Task Force，互联网工程任务组）自 1990 年代早期就开始研究制定 IPv4 的后续版本，并于 1994 年成立了 IPng（IP Next Generation，下一代 IP）工作组，以制定 IPv6 的相关标准：

- 地址架构与分配方案；
- 支持大尺寸数据包；
- 通过 IPv4 网络隧道化 IPv6 报文；
- 安全与自动配置。

随着互联网路由表的快速增长以及互联网用户数的急剧增大，大家一致认为有必要开始设计和测试一种新的网络层协议以接替 IPv4。图 1-1 显示了 1989 年至 2012 年互联网路由表中的网络数量。从图中可以看出，在 1990 年代早期，互联网路由表中的网络数量出现了爆炸性增长。各类研究预测（包括 IETF 在 20 世纪 90 年代早期完成的研究项目）表明，互联网将在 2005 年至 2011 年左右耗尽 IPv4 的全部地址空间。

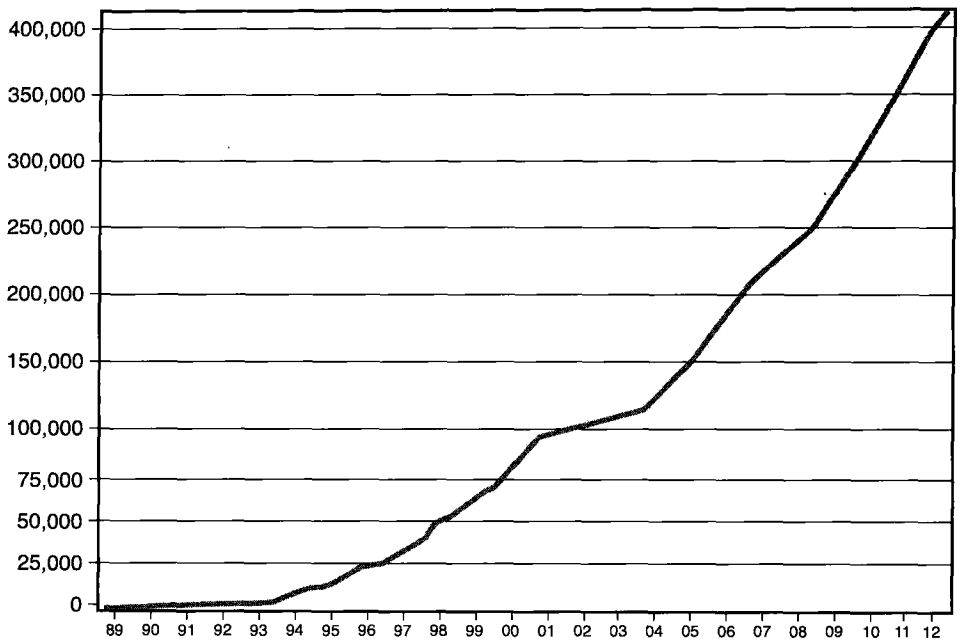



图 1-1 BGP 路由表大小（1989-2011）

除了增加地址空间之外，这也是解决 IPv4 固有缺陷，制定一种协议以满足未来可靠增长并提升性能的绝佳机会。因此，IETF 于 1993 年在 RFC 1550“IP: Next Generation (IPng) White Paper Solicitation”中发出了征求新的 IP 协议的呼吁。有关这些建议的详细情况请参见 RFC 1752（下一代 IP 协议推荐标准）。

当时提出的三种建议方案如下。

- **CATNIP（Common Architecture for the Internet，互联网通用架构）：** CATNIP 建议将 IP、IPX（Internetwork Packet Exchange，网间分组交换）以及 CLNP（Connectionless Network Layer Protocol，无连接网络层协议）集成在一起。其中，IPX 是 IPX/SPX（Internetwork Packet Exchange/Sequenced Packet Exchange，网间分组交换/顺序分组交换）协议簇的一部分，主要应用于部署了 Novell Netware 操作系统的网络中。CLNP 是 ISO 8473 定义的 OSI 标准，是 OSI 协议簇中与 IPv4 对等的协议。有关 CATNIP 的内容定义在 RFC 1707 中。
- **SIPP（Simple Internet Protocol Plus，增强型简单互联网协议）：** SIPP 建议将 IPv4 地址尺寸从 32 比特增加到 64 比特，同时对 IPv4 报头进行一定的改进以提高转发效率。有关 SIPP 的内容定义在 RFC 1710 中。
- **TUBA（TCP and UDP with Bigger Addresses，包含更多地址的 TCP 和 UDP）：**

TUBA 希望尽量减少从 IPv4 迁移到新 IP 地址时的风险，因而建议由 CLNP 来代替 IP，相应的地址尺寸将达到 20 字节（160 比特）。TCP、UDP 以及传统的 TCP/IP 应用都运行在 CLNP 之上，有关 TUBA 的内容定义在 RFC 1347、1526 以及 1561 中。

IETF 最终选择了由 Steve Deering、Paul Francis 和 Bob Hinden 开发的 SIPP，但是将地址尺寸更改为 128 比特。IETF 在 1993 年成立了 IPng 工作组（http://datatracker.ietf.org/wg/ipngwg/charter），并于 1995 年发布了 RFC 1883 "Internet Protocol, Version 6 (IPv6) Specification"，随后被 1998 年发布的 RFC 2460 所替代。2001 年的时候，正式将 IPng 工作组更名为 IPv6 工作组。

1996 年，IETF 建立了名为 6bone 的 IPv6 测试床网络。最初的 6bone 网络利用 IPv6-over-IPv4 隧道/封装技术，在 IPv4-only 互联网上支持 IPv6 的传输，后来逐步迁移到纯 IPv6 链路。6bone 最终于 2006 年结束。

RIR（Regional Internet Registries，地区性互联网注册机构）开始于 1999 年为用户分配 IPv6 地址。最初的 IPv6 地址申请非常缓慢，直至 2007 年，RIR 开始收到大量的有关 IPv6 地址空间的申请，这是因为 RIR 支持在更大范围内部署 IPv6。有关 RIR 分配的 IPv6 地址信息，可以参考 https://labs.ripe.net/Members/mirjam/interesting-graph-ipv6-allocations-since-1999 和 http://en.wikipedia.org/wiki/IPv6_deployment。

自 2000 年开始，大量设备商开始在其主流产品中增加对 IPv6 的支持，Cisco 公司在 Cisco IOS Software Release 12.2(2)T 开始支持 IPv6，Linux 厂商也开始在 2000 年支持 IPv6，Microsoft 宣布于 2001 年在 Windows XP 中支持 IPv6。2010 年 9 月，美国联邦政府承诺部署可运行的 IPv6 网络并使用 IPv6。在美国联邦政府首席信息官 Vivek Kundra 的一份备忘录中，要求 2012 财年底之前，所有面向公众/外部的服务器及业务（如 Web、电子邮件、DNS 和 ISP 服务等）都必须升级支持 IPv6。

目前，世界上有多个论坛和组织机构都在积极推进和支持向 IPv6 的迁移过渡。

- **国际 IPv6 论坛（www.ipv6forum.com）**：是世界范围内的互联网设备商组织，目的是推进 IPv6。
- **北美 IPv6 任务组（www.nav6tf.org）**：是 IPv6 论坛的分支机构，专门在北美洲升级和推广 IPv6。
- **IPv6 门户（www.ipv6tf.org）**：由 Consulintel 公司负责维护的网站，通过论坛、培训、研讨会以及实验室等方式帮助大家部署和支持 IPv6。

## 1.5 IPv6 的优点

本节将介绍 IPv6 的优点，下面简要列举了 IPv6 的主要优点及功能特性。

- **极大扩展的地址空间**：与 IPv4 的 32 比特地址长度相比，IPv6 的源地址和目的地址长度均为 128 比特，可以提供巨大的地址空间——$2^{128}$ 即 340 兆兆兆个地址，足够为地球上的每粒沙子都分配一个 IP 地址。
- **无状态自动配置**：IPv6 提供了一种配置机制，允许主机自己生成一个可路由地址。IPv4 的自动配置地址（RFC 3330 和 5735）只能在本地子网（链路本地）内部使用，路由器不会转发这些地址。IPv6 还支持利用 DHCPv6 进行状态化自动配置。
- **消除了 NAT/PAT（Network Address Translation / Port Address Translation，网络地址转化/端口地址转换）**：由于 IPv6 拥有足够多的公有 IPv6 地址，因而不再需要 NAT/PAT。每个客户站点（从最大型的企业站点到单一的家庭站点）都能得到一个公有 IPv6 地址，从而避免了 NAT 给应用（如 VoIP、视频会议以及其他 P2P 应用）带来的很多负面问题。虽然“不再使用 NAT”被视为 IPv6 的重要优势之一，但 NAT64（Network Address Translation IPv6 to IPv4，IPv6 到 IPv4 的网络地址转换）在目前仍然被用作后向兼容 IPv4 网络的重要工具。另一个与 NAT 相关的问题是，许多组织机构将自己的 IPv4 网络隐藏在 NAT 设备之后，一旦使用了可公开访问的 IPv6 地址之后，这些组织机构就不得不更改其网络安全策略。有关 NAT 的详细讨论，请参考本书第 11 章。
- **消除了广播**：IPv6 不再使用三层广播地址，不过，IPv6 采用了被请求节点的多播地址（solicited node multicast address），这是一种更有效也更具选择性的用于处理地址解析等应用的技术。与 IPv4 使用广播地址进行 ARP（Address Resolution Protocol，地址解析协议）不同，IPv6 使用被请求节点的多播地址来实现相同的目的，而且全部节点多播地址（all-node multicast address）在本质上与 IPv4 广播地址拥有相同的效果。有关多播地址和地址解析的相关内容将在后续章节进行详细讨论。
- **迁移工具**：IPv6 提供了多种工具以支持 IPv4 向 IPv6 的迁移，包括隧道和 NAT。隧道机制将 IPv6 报文封装到 IPv4 报文中，从而可以经由 IPv4-only 网络进行传输。NAT 提供了一种将 IPv4 地址转换为 IPv6 地址或者将 IPv6 地址转换为 IPv4 地址的机制。IPv4 也可以通过隧道机制在 IPv6 网络中进行传输。有关隧道的相关内容将在第 10 章进行详细讨论。

## 1.6 IPv6：何时部署

互联网世界将会在什么时候过渡到 IPv6？是否有明确的日期从 IPv4 切换到 IPv6

呢？首先，没有明确的过渡到 IPv6 的截止日期，从 IPv4 向 IPv6 的过渡工作一直都在进行也仍将继续进行。ISOC（Internet Society，国际互联网协会）创立了世界 IPv6 日（World IPv6 Day），并于 2011 年 6 月举行了首次活动。第一个世界 IPv6 日的主题是为服务提供商和内容提供商测试 IPv6 的实现情况，第二个世界 IPv6 日（2012 年 6 月）的主题是让这些提供商为它们的服务提供永久性的 IPv6 部署。因此可以看出，向 IPv6 的过渡工作才刚刚展开。不过，IPv4 与 IPv6 在相当长的一段时间内将处于共存状态。有关 IPv6 过渡策略的相关内容将在第 10 章进行详细讨论。

当人们谈到对 IPv6 的理解时，首先想到的可能就是巨大的地址空间。本书已经详细讨论了 IPv4 地址的耗尽与 IPv6 地址空间的巨大，但是与 IPv4 相比，IPv6 不仅仅拥有 128 比特的源地址和目的地址，IPv6 还拥有自动配置和邻居发现机制（将在后续章节进行讨论），是一种更有效、更简洁的协议。

## 1.7 IPv4 地址耗尽

如果 IPv4 网络运行状况良好，为什么还需要开始考虑迁移到 IPv6 呢？这就如同在旧汽车状况良好的情况下为何要考虑购置新汽车一样。虽然没有任一种杀手级应用需要用户迁移到 IPv6，但依然有一些不可抗拒的理由（如前所述）驱使网络管理员至少应该为向 IPv6 的迁移做好准备。最明显的理由就是 IPv4 地址已经耗尽，因此可以说，IPv6 的杀手级应用就是保护了互联网，允许互联网继续前进。

虽然 IPv4 在理论上最多可拥有 43 亿个地址，但在实际的地址分配过程中存在很多低效的情况。即便能够以更高效的方式重新分配整个 IPv4 地址空间，也只是短期应对之策，更何况这种做法根本就不切实际。

IPv6 拥护者一直在寻找各种方法，在不迁移到 IPv6 的情况下，尽可能延长 IPv4 的使用时间。这就像重新安排泰坦尼克号上的甲板座椅一样，可能会让大家感到一时的舒适，但大家迟早会意识到，自己已身处麻烦之中。

当 Bob Kahn 和 Vint Cerf 首次开发 TCP/IP 协议集以及 IPv4 及其 32 比特地址时，根本没有预料到互联网会发展到如今的状态。如今，即便 70 多岁的人每天也都会在互联网上使用几个小时的 Facebook，表明如今的互联网已不再是年轻人的专利了。

人们从 20 世纪 90 年代就已经意识到 IPv4 地址空间的耗尽问题。随着 WWW 的出现，互联网用户的数量和多样化都呈现出爆炸式增长，使用互联网的群体不再局限于相对小规模的能够使用较为复杂和较不直观的网络工具（如 FTP、新闻组以及 UUCP[Unix-to-Unix Copy，UNIX 至 UNIX 的复制]等）享用全球网络的电脑高手。随着 Tim Berners-Lee 在日内瓦的 CERN 开发出 HTTP（Hypertext Transfer

Protocol，超文本传输协议），以及由 Marc Andreessen 及其研究团队在伊利诺伊大学香槟分校的国家超级计算应用中心开发出第一款 Web 浏览器 Mosaic，那些从没拥有电脑的人们都纷纷开始接入互联网，这使得 43 亿个地址变得越来越无法满足人们的长期需求。

因此，在 20 世纪 90 年代早期，IETF 开始研究被称为 IPng（下一代 IP）的新 IP 版本，后来成为 IPv6。不过这是解决 IP 地址不足的长期解决方案，人们仍然需要马上能用的短期解决方案，因而 NAT、私有地址空间以及 CIDR（Classless InterDomain Routing，无类别域间路由）等技术应运而生。

### 1.7.1 CIDR

早期的 IPv4 网络地址按照 A 类、B 类、C 类地址进行分配（如表 1-1 所示）。

表 1-1 有类别单播 IP 地址

| 地址类别 | 第一个八位组范围 | 可能的网络数量 | 每个网络的主机数量 |
|---|---|---|---|
| A 类 | 0～127 | 128（其中 2 个被保留） | 16 777 214 |
| B 类 | 128～191 | 16 348 | 65 535 |
| C 类 | 192～223 | 2 097 152 | 254 |

IPv4 地址一共分为 5 类。

- **A 类**：子网掩码是 255.0.0.0 或/8。
- **B 类**：子网掩码是 255.255.0.0 或/16。
- **C 类**：子网掩码是 255.255.255.0 或/24。
- **D 类**：多播地址。
- **E 类**：保留作为试验地址。

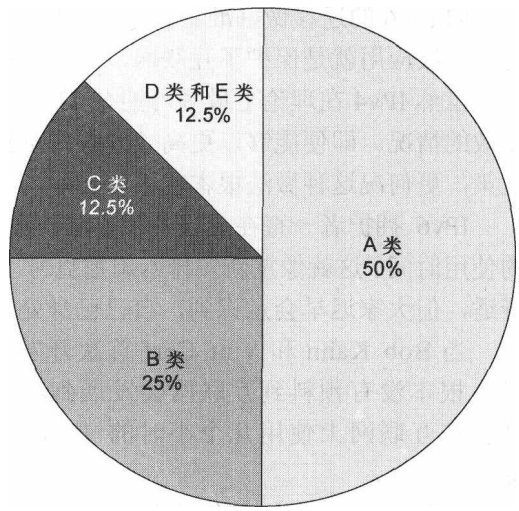


图 1-2 有类别 IP 地址分配

每个子网掩码为/8 的 A 类网络可以拥有 16 777 214 台主机（减去 2 个地址，分别是网络地址和广播地址）。126 个 A 类网络以及每个 A 类网络中的大量主机数占去了 IPv4 地址空间的一半（如图 1-2 所示）。

B 类网络的子网掩码是/16，表示每个 B 类网络可拥有 65 534 台主机。而 16 384 个 B 类网络占去了 IPv4 地址空间的 25%。

C 类网络的子网掩码是/24，表示每个 C 类网络最多可拥有 254 台主机。虽然可用

的 C 类网络有 2 097 152 个，但仅占 IPv4 地址空间的 12.5%。

由于网络管理员只能得到这三类地址中的一个网络地址，因而造成了地址分配方法的低效化。绝大多数 A 类和 B 类网络都存在大量未用地址，而绝大多数 C 类网络都很少有未用地址。

1992 年，IETF 改变了这种地址分配方法，转而采用 CIDR（读成 cider）。有关 CIDR 的信息定义在 RFC 1338 中，后来被 RFC 1519 所废除。此时的网络地址不再基于这三类地址进行分配，而是可以按照任意大小的子网掩码进行分配，如图 1-3 所示（图 1-3 给出了全部可能的主机地址数，而没有减去网络地址和广播地址）。这使得 RIR 和 ISP 能够更灵活地为用户分配 IP 地址，从而极大地提高了有限的 IPv4 地址空间的分配效率。

| | | | |
|---|---|---|---|
| A 类 | 11111111.00000000.00000000.00000000 | /8(255.0.0.0) | 16,777,216 主机地址 |
| | 11111111.10000000.00000000.00000000 | /9(255.128.0.0) | 8,388,608 主机地址 |
| | 11111111.11000000.00000000.00000000 | /10(255.192.0.0) | 4,194,304 主机地址 |
| | 11111111.11100000.00000000.00000000 | /11(255.224.0.0) | 2,097,152 主机地址 |
| | 11111111.11110000.00000000.00000000 | /12(255.240.0.0) | 1,048,576 主机地址 |
| | 11111111.11111000.00000000.00000000 | /13(255.248.0.0) | 524,288 主机地址 |
| | 11111111.11111100.00000000.00000000 | /14(255.252.0.0) | 264,144 主机地址 |
| | 11111111.11111110.00000000.00000000 | /15(255.254.0.0) | 131,072 主机地址 |
| B 类 | 11111111.11111111.00000000.00000000 | /16(255.255.0.0) | 65,536 主机地址 |
| | 11111111.11111111.10000000.00000000 | /17(255.255.128.0) | 32,768 主机地址 |
| | 11111111.11111111.11000000.00000000 | /18(255.255.192.0) | 16,384 主机地址 |
| | 11111111.11111111.11100000.00000000 | /19(255.255.224.0) | 8,192 主机地址 |
| | 11111111.11111111.11110000.00000000 | /20(255.255.240.0) | 4,096 主机地址 |
| | 11111111.11111111.11111000.00000000 | /21(255.255.248.0) | 2,048 主机地址 |
| | 11111111.11111111.11111100.00000000 | /22(255.255.252.0) | 1,024 主机地址 |
| | 11111111.11111111.11111110.00000000 | /23(255.255.254.0) | 512 主机地址 |
| C 类 | 11111111.11111111.11111111.00000000 | /24(255.255.255.0) | 256 主机地址 |
| | 11111111.11111111.11111111.10000000 | /25(255.255.255.128) | 128 主机地址 |
| | 11111111.11111111.11111111.11000000 | /26(255.255.255.192) | 64 主机地址 |
| | 11111111.11111111.11111111.11100000 | /27(255.255.255.224) | 32 主机地址 |
| | 11111111.11111111.11111111.11110000 | /28(255.255.255.240) | 16 主机地址 |
| | 11111111.11111111.11111111.11111000 | /29(255.255.255.248) | 8 主机地址 |
| | 11111111.11111111.11111111.11111100 | /30(255.255.255.252) | 4 主机地址 |
| | 11111111.11111111.11111111.11111110 | /31(255.255.255.254) | 2 主机地址 |
| | 11111111.11111111.11111111.11111111 | /32(255.255.255.255) | “主机路由” |

图 1-3　无类别 IP 地址分配

## 1.7.2　NAT 和私有地址

NAT 允许多台主机共享一个或多个保留地址或公有 IPv4 地址。结合私有 IP 地址，NAT 成为一种非常有用的节约 IPv4 地址空间的方法。IANA（Internet Assigned Numbers Authority，互联网号码分配机构）在 RFC 1918 中分配了三段地址作为私有 IPv4 地址，

这三段私有网络地址可用于私有网络内部，但不允许被路由到全球互联网上。因此，在面向互联网的路由器转发这些分组时，必须首先将这些私有 IPv4 地址转换成公有 IPv4 地址。

注：虽然这是 NAT 最常用的使用方式，但 NAT 不仅仅可以在私有 IP 地址与公有 IP 地址之间进行转换，而是可以应用于任何一对地址的转换场景中。

RFC 1918 定义的定义私有 IPv4 地址如下：
10.0.0.0～10.255.255.255 (10.0.0.0/8)
172.16.0.0～172.31.255.255 (172.16.0.0/12)
192.168.0.0～192.168.255.255 (192.168.0.0/16)
大家可能会发现，绝大多数家庭路由器都使用 192.168.0.0 地址进行 NAT 操作。

注意：新的 RFC 6598 “IANA-Reserved IPv4 Prefix for Shared Address Space” 中分配了另一个共享地址空间 100.64.0.0/10。共享地址空间与 RFC 1918 中定义的私有地址空间不同，共享地址空间主要是留给服务提供商的网络使用的，不过，也可以按照 RFC 1918 私有地址空间的使用方式来使用这些共享地址空间。

PAT 是动态 NAT 的一种形式，利用不同的 TCP/UDP 端口号将多个 IP 地址映射成单个 IP 地址。PAT 也被称为超量 NAT（NAT overloading）、单地址 NAT（single address NAT）或端口级多路复用 NAT（port-level multiplexed NAT）。对 PAT 来说，可以将私有网络中的每台计算机都转换成同一个公有 IP 地址，但需要分配不同的源端口号。有时也将 NAT/PAT 简称为 NAT，对于本书来说，术语 NAT 始终包含 PAT。

图 1-4 给出了一个使用 NAT 的网络示例。XYZ 公司拥有公有 IP 地址 209.165.200.248/29，因而只有 6 个可用的主机地址：209.165.200.249～209.165.200.254。但是 XYZ 公司拥有几百台主机，而且所有主机都希望能够随时访问互联网。由于该公司仅有 6 个公有 IP 地址，因而需要使用私有 IP 地址空间和 NAT 机制来满足其互联网接入需求。

XYZ 公司的网络管理员选用了私有地址空间 10.0.0.0/8。该地址空间提供了足够多的主机 IP 地址，完全能够满足公司的需要。路由器 A 是 XYZ 公司的路由器，负责将数据包转发给 ISP 路由器，从而将公司的私有网络连接到互联网上。图 1-4 中的步骤显示了 NAT 在私有 IP 地址与公有 IP 地址之间的转换操作，为了简化起见，本案例没有演示 PAT 的端口转换操作。

**第 1 步：** XYZ 公司网络中的主机 A 向 Web 服务器 www.example.com 发送数据包，源 IP 地址为 10.0.0.100，目的 IP 地址为 209.165.202.158。

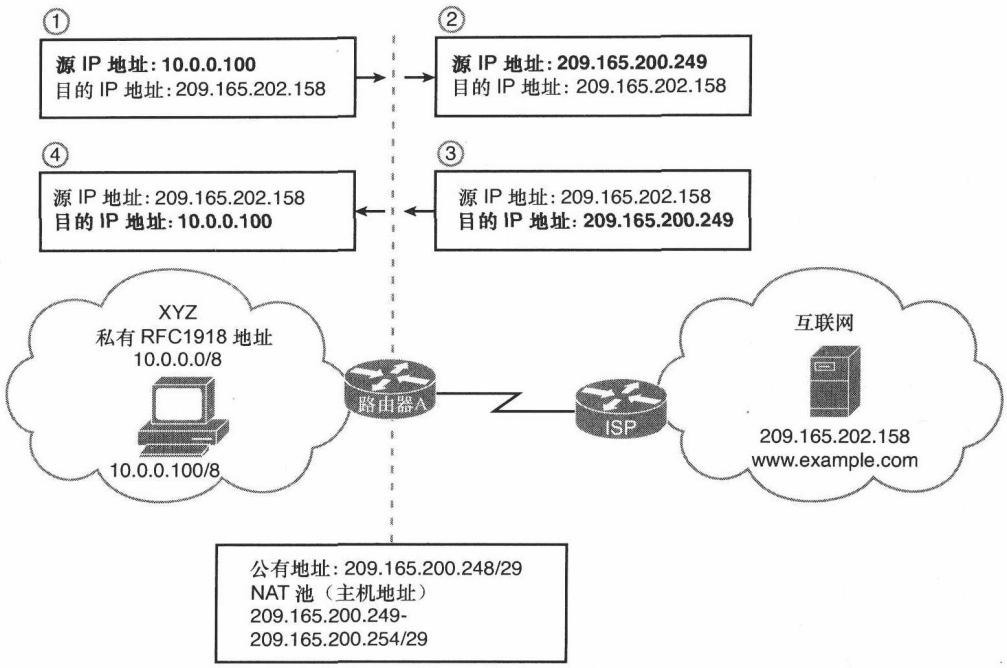



图 1-4　NAT 应用示例

**第 2 步**：路由器 A 收到该数据包后，在转发给 ISP 路由器之前执行 NAT/PAT 操作。路由器 A 将该数据包的私有源 IP 地址 10.0.0.100 转换成为公有 IP 地址 209.165.200.249。对于 PAT 来说，路由器 A 还会更改该数据包的源 TCP/UDP 端口号。为了简化起见，本案例没有演示 PAT 的端口转换操作。路由器 A 会维护一张 NAT 表以记录所有的地址与端口号分配情况，从而能够在第 4 步将返程流量正确定位到原始的地址和端口号。

**第 3 步**：Web 服务器 www.example.com 向主机 A 返回数据包，源 IP 地址为 209.165.202.158，目的 IP 地址为 209.165.200.249。

**第 4 步**：路由器 A 收到发送给主机 A 的流量后，使用 NAT 表，即可识别正确的目的 IP 地址和目的端口号，从而以原始地址 10.0.0.100 和原始端口号进行修改。

虽然利用 NAT 与 RFC 1918 私有地址可以减缓 IPv4 地址的耗尽速度，但是 NAT 也存在许多限制条件。由于实际目的 IPv4 地址的不确定性以及单个公有地址可以被多个用户共享，因而极大地限制了 P2P 应用。以两台均位于 NAT 之后的设备为例，如果这两台设备需要进行相互通信，那么它们应该将数据包发送给哪个公有地址呢？纯粹的 P2P 通信仅包含两台相互通信的设备，但 NAT 的使用使得这两台设备之间的通信

变得难以实现。因此，像 Skype 等许多服务都使用公共可访问的服务器在用户之间充当中继点。

如果大家对 IPSec（Internet Protocol Security，互联网协议安全）有所了解，那么就会知道 NAT 会给传输模式下的 IPSec 应用带来混乱。因为在使用了 IPSec AH（Authentication Header，认证头）的情况下，由于 NAT 转换会在数据包传输过程中修改数据包，因而会破坏完整性检查。NAT 会修改 TCP/UDP 校验和，使得对端的完整性检查失败。因此，不能在传输模式下的 IPSec 中使用 NAT。虽然 NAT 可以工作于隧道模式下的 IPSec，但也同样存在一些问题。

对大多数用户来说，NAT 不是问题，因为互联网主要是由大量客户端/服务器网络组成。一般来说，大部分用户使用互联网的目的是下载各种内容，如网页、电子邮件以及音乐等。

在 2000 年代的第一个 10 年里，互联网更多地被用作双向介质。随着大量社交网站（如 Myspace 和 Facebook）的出现，人们越来越多地开始利用互联网进行协作和交互。用户不再仅仅是信息的消费者，而是越来越多地成为内容参与网站（如 YouTube 和博客）的内容制造者，并利用 P2P 应用（如 Napster 和 BitTorrent）分享各自的内容。

所有的一切都表明，与几年前相比，用户使用互联网的方式发生了翻天覆地的变化。未来的互联网仍然会出现各种现在不可知的新应用和新需求。互联网将更多地承载双向通信，任何设备都可能是一台客户终端或者是一台服务器，甚至是两者兼而有之。在很多情况下，这都需要通信双方拥有公有地址，而 NAT 则无法支撑这些应用。

### 1.7.3 IPv4 地址空间耗尽

由于互联网用户数一直都在快速增长，因此即使部署了 NAT 等短期解决方案，人们仍然不得不面对可用公有 IPv4 地址的最后阶段。表 2-1 给出了世界互联网使用情况及人口统计情况。从中可以看出一些有趣的 IPv4 地址分配情况。北美的互联网渗透率大约为 78%。互联网渗透率（Internet Penetration Rate）指的是给定国家或地区的互联网用户数占总人口数的百分比。这意味着北美的大部分人都在使用互联网。而在亚洲等其他地区，它们的互联网渗透率大约只有 29%。如果进一步对比这两个地区的互联网用户数，大家一定会更加吃惊。北美的互联网用户数是 2720 万，而亚洲的互联网用户数是 9220 万。可以看出，随着亚洲、非洲及其他地区越来越多的用户开始访问互联网，即便拥有 NAT，IPv4 地址也绝对无法满足互联网用户数快速增长的需求。

表 1-2　　世界互联网使用及人口统计情况（2011 年 3 月）

| 世界地区 | 人口（2011 年预测数） | 互联网用户数（2000 年 12 月 31 日） | 互联网用户数（最新数据） | 渗透率（占人口的百分比） | 增长率，2000-2011（%） | 互联网用户比例(%) |
|---|---|---|---|---|---|---|
| 非洲 | 1,037,524,058 | 4,514,400 | 118,609,620 | 11.4 | 2,527.4 | 5.7 |
| 亚洲 | 3,879,740,877 | 114,304,000 | 922,329,554 | 23.8 | 706.9 | 44.0 |
| 欧洲 | 816,426,346 | 105,096,093 | 476,213,935 | 58.3 | 353.1 | 22.7 |
| 中东 | 216,258,843 | 3,284,800 | 68,553,666 | 31.7 | 1,987.0 | 3.3 |
| 北美 | 347,394,870 | 108,096,800 | 272,066,000 | 78.3 | 151.7 | 13.0 |
| 拉美/加勒比海 | 597,283,165 | 18,068,919 | 215,939,400 | 36.2 | 1,037.4 | 10.3 |
| 大洋洲/澳大利亚 | 35,426,995 | 7,620,480 | 21,293,830 | 60.1 | 179.4 | 1.0 |
| 世界合计 | 6,930,055,154 | 360,985,492 | 2,095,006,005 | 30.2 | 480.4 | 110.0 |

表 1-2 中的信息来自 www.internetworldstats.com，Copyright©2001–2011，Miniwatts Marketing Group

随着无线产业的飞速发展，WLAN、3G、4G、WiMax 和 MBWA（Mobile Wireless，移动无线）等蜂窝网络和无线网络增加了大量需要访问互联网的设备。一位美国大学的网络工程师曾经分析过，每年学生放假返校后的无线网络利用率都会增加 25%，因为学生们返校时都带着大量的笔记本电脑、蜂窝手机、iPad 以及其他各类无线设备。

那么 IPv4 地址会在什么时候被耗尽呢？IANA 负责为 5 个 RIR 分配/8 地址块。利用这些地址空间，RIR 再将这些网络地址分配给 ISP 及其他终端用户，5 个 RIR 的管理范围如下。

- **AfriNIC（African Network Information Centre，非洲网络信息中心）**：非洲。
- **ARIN（American Registry for Internet Numbers，美国互联网号码注册中心）**：美国、加拿大以及部分加勒比海地区和南极洲。
- **APNIC（Asia-Pacific Network Information Centre，亚太地区网络信息中心）**：亚洲、澳大利亚、新西兰以及相邻国际。
- **LACNIC（Latin America and Caribbean Network Information Centre，拉美及加勒比海地区网络信息中心）**：中美洲、南美洲以及加勒比海大部分地区。
- **RIPE（Réseaux IP Européens Network Coordination Centre，欧洲 IP 资源网络协调中心）**：欧洲、中东和中亚。

2011 年 1 月 31 日星期一，IANA 将最后两块 IPv4 地址空间 39.0.0.0/8 和 106.0.0.0/8 分配给了 APNIC（亚太地区的 RIR），并触发了一项 IANA 政策：今后将剩余的/8 地址段平均分配给 5 个 RIR。至此，IANA 已经分配完了所有 IPv4 地址。

这是否意味着终端用户无法再得到 IPv4 地址了呢？答案为否，因为 ISP 及其他用

户仍然可以从RIR获取IPv4地址，反过来绝大多数终端用户也就可以从ISP处获取IPv4地址。目前业界有多个预测项目分析这些RIR何时将耗尽其IPv4地址空间，其中一个预测项目就是RIR IPv6地址耗尽模型（RIR IPv6 Address Run-Down Model）。大家还可以从Hurricane Electric（http://ipv6.he.net/statistics）下载适用不同系统平台的IPv4地址耗尽计数器，该计数器可以显示RIR可用的/24地址空间。随着IPv4地址的耗尽，从事IPv4地址交易的“黑市”也应运而生。很多网站都为希望出售或租借IPv4地址空间的组织机构充当代理商角色。ISOC告诫IPv4地址的买卖方，要求地址转让一定要经过恰当的RIR程序，从而避免对网络的正常运行或安全性造成影响。

在IPv4地址空间具有43亿的固有限制、互联网用户数不断增长、用户使用互联网的方式发生变更，以及NAT存在各种各样问题的情况下，向IPv6迁移已经迫在眉睫，至少应该开始关注并了解IPv6。

## 1.8 IPv6的迁移

与生活中的大多数事物一样，网络也有其黄金法则：“只要网络没有崩溃，就不要处理它。”既然如此，为何在IPv4网络依然能够正常运行的情况下，依然要考虑向IPv6进行迁移呢？部署IPv6会给我们带来什么样的杀手级应用呢？

如前所述，IPv6的杀手级应用就是互联网本身。因为IPv4无法支撑互联网的可持续发展，互联网因缺乏地址而无法继续增长和发展演进，解决之道就是IPv6。由于IPv4互联网已经达到了其全部潜能，开始经历性能与可用性难题。具有联网能力的设备在人们日常生活、工作等各个角落中的增长速度越来越快，这些设备已不再局限于家用电脑，而是包含了电视机、智能电话、平板设备、网络摄像头、安全监控等各种设备。

世界将最终过渡到IPv6，这是不可避免的事实，除非希望将自己排除在全球互联网社区之外。现在已别无选择，唯有开始准备并最终在网络中部署IPv6。虽然不要求所有网络都在确定的日期迁移到IPv6，也没有最终的截止日期，但大家必须开始做好准备。

现在是IT部门准备过渡到IPv6的最佳时机，以免届时因压力或需求急切而仓促应事。对IT部门来说，现在还有时间去学习IPv6，培训/培养自己的员工，测试/验证自己的IPv6部署方案，应该将准备工作聚焦于IT部门，如网络运行、网络安全、应用开发者、内容开发者以及数据中心等。下面列举了向IPv6迁移前所必须完成的一些准备工作。

- **将IPv6纳入IT战略**：必须将IPv6纳入公司的长期和短期IT战略计划之中，IPv6必须成为公司IT规划进程、例会、培训及采购计划的一部分。
- **培训**：进行IPv6培训或至少熟悉IPv6的最佳时机就是在全面部署IPv6之前。

学习/培训是消除 IT 人员对 IPv6 恐惧或排斥心理的最好方式。当然，培训也可以不局限于 IT 人员，还可以包含其他非 IT 人员，如销售人员。一个拥有 25 年网络工程师工作经历的朋友曾经告诉我，他的最大希望就是能赶在必须学习 IPv6 之前退休。不过，当大家学习完本书之后，就会发现 IPv6 并没有想象中的那么复杂，甚至在某些方面比 IPv4 更简单。例如，子网划分时不再需要比特映射，因此 IPv6 子网的配置和识别更加简单。

- **开列设备清单**：编制网络中的设备列表（或者使用现有的盘存清单），以确定当前设备是否支持 IPv6 或者需要做哪些升级/替换，包括：
  - 客户端和服务器；
  - 路由器和多层交换机；
  - 二层交换机；
  - 防火墙和其他安全设备；
  - 打印机、网络摄像头、电话机以及网络中具有 IP 地址的其他设备；
  - 应用程序、操作系统和软件服务。
- **购置**：要确定设备购置计划中的所有新设备都支持 IPv6，询问设备商其硬件/软件是否支持 IPv6。如果不支持，一定要了解这些产品的 IPv6 支持计划。
- **测试实验室**：准备工作的最佳起点就是搭建与生产性网络相隔离的测试实验室，开始学习和摆弄 IPv6。一旦熟悉了 IPv6，就可以在继续运行 IPv4 的情况下引入 IPv6。

## 1.9 本章小结

通过本章的学习，大家应该对 IPv4 的局限性有了更清晰的认识，并且了解了为何需要向 IPv6 进行过渡。虽然有关 IPv6 优点的讨论将贯穿本书始终，但大家对 IPv6 的优点应该有了一个基本认识。

本章主要内容如下。

- 目前人们使用互联网的方式与 IPv4 创建之初的巨大差异：用户更多、设备更多、新需求层出不穷，已经从计算机互联网发展到了物联网。
- 虽然没有人能够确切地知道究竟何时会耗尽 IPv4 的 43 亿个地址，但已经到了可用公有 IPv4 地址的最后阶段却是不争的事实。
- 拥有 128 比特地址的 IPv6 提供了足够多的全球唯一的地址，以支撑互联网的持续发展。
- IPv4 和 IPv6 将在相当长的一段时期内长期共存。IPv6 提供了很多过渡工具和迁移策略，以允许这两种协议的共存。

- 虽然通过 CIDR、NAT 以及私有地址可以延缓 IPv4 地址空间的耗尽速度，但 NAT 会给很多互联网应用带来不利影响，如 P2P 网络互连。而且随着全球互联网社区的日益普及，这些针对 IPv4 地址的增强手段已无能为力。
- 除了更大的地址空间之外，IPv6 还提供了包括无状态自动配置和无需 NAT 的扩展地址空间等诸多增强功能。
- 在必须升级到 IPv6 之前，现在是 IT 部门开始熟悉并掌握 IPv6 的最佳时机。IPv6 的杀手级应用就是互联网本身。

IPv6 是确保互联网持续发展的 IPv4 后续版本。

## 1.10 参考文献

RFC：

RFC 1, Host Software , Steve Crocker, UCLA, www.ietf.org/rfc/rfc1.txt , April 1969

RFC 454, File Transfer Protocol , A. McKenzie, BBN, IETF, www.ietf.org/rfc/rfc454.txt , February 1973

RFC 827, Exterior Gateway Protocol , Eric C. Rosen, BBN, IETF, www.ietf.org/rfc/rfc827.txt , October 1982

RFC 760, DoD Standard, Internet Protocol , USC, IETF, www.ietf.org/rfc/rfc760.txt ,January 1980

RFC 791, Internet Protocol, DARPA Internet Program Protocol Specification ,USC, www.ietf.org/rfc/rfc791.txt , September 1981

RFC 920, Domain Requirements , J. Postel, J. Reynolds, ISI, IETF, www.ietf.org/rfc/rfc920.txt , October 1984

RFC 1190, Experimental Internet Stream Protocol, Version 2 (ST-II) , CIP Working Group, IETF, www.ietf.org/rfc/rfc1190.txt , October 1990

RFC 1338, Supernetting: An Address Assignment and Aggregation Strategy , V.Fuller, BARRNet, IETF, www.ietf.org/rfc/rfc1338 , June 1992

RFC 1347, TCP and UDP with Bigger Addresses (TUBA), A Simple Proposal for Internet Addressing and Routing , Ross Callon, DEC, IETF, www.ietf.org/rfc/rfc1347.txt , June 1992

RFC 1519, Classless Inter-Domain Routing (CIDR): An Address Assignment and Aggregation Strategy , V. Fuller, BARRNet, IETF, www.ietf.org/rfc/rfc1519.txt ,September 1993

RFC 1526, Assignment of System Identifiers for TUBA/CLNP Hosts , D. Piscitello,Bellcore, IETF, www.ietf.org/rfc/rfc1526.txt , September 1993

RFC 1550, IP: Next Generation (IPng) White Paper Solicitation , S. Bradner, Harvard

University, IETF, www.ietf.org/rfc/rfc1550.txt , December 1993
RFC 1561, Use of ISO CLNP in TUBA Environments , D. Piscitello, Core Competence, IETF, www.ietf.org/rfc/rfc1561.txt , December 1993
RFC 1656, BGP-4 Protocol Document Roadmap and Implementation Experience , P. Traina, Cisco Systems, IETF, www.ietf.org/rfc/rfc1656.txt , July 1994
RFC 1700, Assigned Numbers , J. Reynolds, ISI, IETF, www.ietf.org/rfc/rfc1700.txt , October 1994
RFC 1707, CATNIP: Common Architecture for the Internet , M. McGovern, Sunspot Graphics, IETF, www.ietf.org/rfc/rfc1707.txt , October 1994
RFC 1710, Simple Internet Protocol Plus White Paper , R. Hinden, Sun Microsystems, IETF, www.ietf.org/rfc/rfc1710.txt , October 1994
RFC 1752, The Recommendation for the IP Next Generation Protocol , S. Bradner, Harvard University, IETF, www.ietf.org/rfc/rfc1752.txt , January 1995
RFC 1819, Internet Stream Protocol Version 2 (ST2) Protocol Specification -Version ST2+ , ST2 Working Group, IETF, www.ietf.org/rfc/rfc1819.txt , August 1995
RFC 1883, Internet Protocol, Version 6 (IPv6) Specification , S. Deering, Xerox PARC, IETF, www.ietf.org/rfc/rfc1883.txt , December 1995
RFC 1918, Address Allocation for Private Internets , Y. Rekhter, Cisco Systems, IETF, www.ietf.org/rfc/rfc1918.txt , February 1996
RFC 2205, Resource ReSerVation Protocol (RSVP) -Version 1 Functional Specification , R. Braden, ISI, IETF, www.ietf.org/rfc/rfc2205.txt , September 1997
RFC 2235, Hobbes'Internet Timeline , R. Zakon, MITRE, IETF, www.ietf.org/rfc/rfc2235.txt , November 1997
RFC 2460, Internet Protocol, Version 6 (IPv6) Specification , S. Deering, Cisco Systems, IETF, www.ietf.org/rfc/rfc2460.txt , December 1998
RFC 3330, Special-Use IPv4 Addresses , IANA, IETF, www.ietf.org/rfc/rfc3330.txt , September 2002
RFC 3927, Dynamic Configuration of IPv4 Link-Local Addresses , S. Cheshire, Apple Computer, IETF, www.ietf.org/rfc/rfc3927.txt , May 2005

网站：

International IPv6 Forum, www.ipv6forum.com
Internet World Stats, www.internetworldstats.com
IPv4 Address Report, www.potaroo.net/tools/ipv4/index.html
North American IPv6 Task Force, www.nav6tf.org

The IPv6 Portal, www.ipv6tf.org
The top 500 sites on the web, www.alexa.com/topsites
Global IPv6 Deployment Progress Report, http://bgp.he.net/ipv6-progress-report.cgi
Internet Protocol Version 6 Address Space, www.iana.org/assignments/ipv6-addressspace/ipv6-address-space.xml
Comparison of IPv6 application support, http://en.wikipedia.org/wiki/Comparison_of_IPv6_application_support
Comparison of IPv6 support in operating systems, http://en.wikipedia.org/wiki/Comparison_of_IPv6_support_in_operating_systems

# 第2章 IPv6协议

本章将详细描述IPv6协议的相关内容。首先分析IPv4和IPv6报头的各个字段，并分析两者的异同点，然后解释为什么IPv6所提供的不仅仅是更大的地址空间，而是一种更灵活、更有效的新协议。

有关IPv6报头结构的信息定义在RFC 2460“Internet Protocol, Version 6 (IPv6) Specification”中。本章除了介绍IPv6的基本报头之外，还将介绍IPv6的扩展报头，并在本章小结中归纳了IPv4与IPv6报头的之间差异。

## 2.1 IPv4报头

为了更好地理解IPv6报头，首先回顾一下IPv4报头，这样既能复习一下IPv4报头的信息，又能加深对IPv4报头的理解。无论哪种情况，都将有助于理清与IPv6报头的差异。图2-1给出了RFC 791“Internet Protocol, DARPA Internet Program, Protocol Specification”中定义的IPv4报头结构。IPv6报头的结构如图2-2所示。快速对比这两种报头结构可以看出，IPv6报头是一种更为简单的协议——字段较少，这无疑会有助于简化协议并实现更高效的处理。

> 注：有关IPv4报头中各个字段的详细介绍将放到下一节，本节的主要目的是为需要IPv4知识的读者提供参考和回顾。这方面的信息对于后面将要分析的IPv4与IPv6报头差异非常有用。

IPv4报头中的各个字段如下。

- **版本（Version，4比特）**：该字段包含IP报头的版本号。对IPv4来说，该字段的值始终为4。

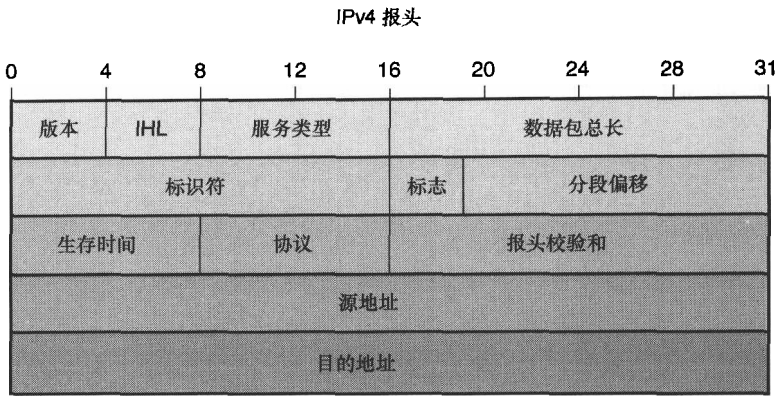



图 2-1 IPv4 报头

- **IHL（4 比特）**：IHL（Internet Header Length，互联网报头长度）字段指示 IP 报头长度（包括所有选项字段），以 32 比特字为单位。事实上，该字段指出了 IP 报头的结束位置以及数据或净荷的开始位置，最小值为 5（5×32 比特字=160 比特或 20 个八位组[字节]），等于 IPv4 报头的最小长度（不包含任何选项和填充比特）。
- **ToS（8 比特）**：ToS（Type of Service，服务类型）字段指示了该数据包将受到路由器的何种处理方式。利用不同的优先级，ToS 信息可以提供 QoS（Quality of Service，服务质量）功能。当多个数据包都在排队从同一个接口向外发送时，可以利用 ToS 值来确定发送顺序。由于 ToS 字段并没有得到设计之初的广泛应用，因此 IETF 于 1998 年在 RFC 2747 中使用 DS（Differentiated Service，差分服务）技术进行了重新定义。虽然本章将在后面进一步介绍 DSCP（Differentiated Services Code Point，差分服务代码点）的详细信息，但有关 ToS 和 DS 的内容已经超出了本书范围。
- **数据包总长（Total Length，16 比特）**：该字段表示 IP 包（包括 IP 报头和数据）的长度，以八位组（字节）为单位。由于该字段为 16 比特，因而 IPv4 数据包最大为 65 535 字节。大多数 IPv4 包都远小于该数值。

接下来的三个字段用于数据包的分段与重组。IP 的设计初衷是可以适应各种传输链路，但大多数传输链路都有被称为 MTU（Maximum Transmission Unit）的最大包长限制。当传输路径上的 MTU 小于发送端的 MTU 时，由于允许路由器对 IP 包进行分段，因而能够适应不同的 MTU 差异。如果路由器收到一个大于其出接口 MTU 的 IPv4 包，就可以通过 IPv4 报头中的选项对数据包进行分段。有时数据包在源端就会被分段成多个数据包，最后的 IP 包目的地址负责将分段后的数据包重组成原始的全尺寸 IP 包。

- **标识符（Identifier，16 比特）**：大多数经网络传送的消息都包含多个数据包，消息中的每个数据包都通过 16 比特标识符字段分配一个唯一值。当某个数据

包需要被分段成两个或多个数据包时，所有被分段后的数据包的标识符字段都相同，这样就能帮助接收端重组这些分段。

- **标志（Flag，3 比特）**：该字段的第一个比特为 0，表示被保留或未使用。第二个比特称为 DF（Don't Fragment，不分段）比特，DF 比特为 1 表示该数据包不应该被分段，但很多协议并不关心分段进程，因而将该比特置 0，意思是可以根据需要对数据包进行分段。第三个比特是更多分段标志（More Fragments Flag），表示该分段是最后一个分段（比特为 0）或者后面还有更多的分段（比特为 1）。如果数据包未被分段，那么就只有一个分段（也就是整个数据包），该标志比特就被置为 0。

注：DF 比特在测试源端与目的端路径上的 MTU 时非常有用，如果 DF 为 1，那么数据包就不应该被分段，对路径上的任意路由器来说，如果其 MTU 小于数据包，那么就丢弃该数据包并向源端返回一条 ICMP（Internet Control Message Protocol，因特网控制消息协议）消息“目的地不可达”，该 ICMP 消息将包含该路由器出接口的 MTU。有关 IPv4 的路径 MTU 发现（Path MTU Discovery）的相关内容已经超出了本书范围，如果感兴趣，可以进一步参考 RFC 1191 “Path MTU Discovery”。本章稍后将讨论 IPv6 的路径 MTU 发现。

- **分段偏移（Fragment Offset，13 比特）**：数据包被分段后，该字段将指示该数据包的偏移量或者在原始数据中的位置，以 8 个八位组（64 字节）为单位，从本质上来看，分段偏移字段为接收端指示了该被分段数据包与其他被分段数据包之间的关系，第一个分段的偏移值为 0，如果数据包未被分段，那么该字段值为 0。
- **TTL（8 比特）**：TTL（Time to Live，生存时间）字段可以确保发生路由环路时数据包不会无止境的在网络中生存下去，每经过一台路由器，数据包的 TTL 值就递减 1，当该字段值达到 0 时，该数据包就会被丢弃并向该数据包的源端发送一条 ICMPv4 超时（类型 11）消息。

注：TTL 最初的设计意图是希望表示允许数据包在网络中传送的最大实际时间，而不必是路由器的跳数，RFC 791 中提到“即使本地没有关于实际花费时间的相关信息，该字段仍然必须被递减 1，时间度量单位为秒（也就是说，值 1 表示 1 秒），因而，生存时间的最大值为 255 秒或 4.25 分钟。”由于路由器并不计算时间量，而仅仅对 TTL 执行递减 1 操作，因而使得该字段在事实上成为跳数。

- **协议（Protocol，8 比特）**：该字段表示 IP 包中数据部分所承载的协议类型，RFC 1700（Assigned Numbers）指定了不同协议的取值，大家可以通过 IANA 负责维护的在线数据库 www.iana.org/assignments/protocol-numbers/protocol-numbers.xml 查看

已分配的协议号，常见值有 ICMP 为 1、TCP 为 6、UDP 为 17。

注：有时该字段也被称为承载的上层协议，由于 IP 包所承载的数据或净荷可能是其他三层协议（如 ICMP，甚至是 IP），因而这可能会让人产生误解。

- **报头校验和（Header Checksum，16 比特）**：IP 报头中的校验和可以为数据包传输过程中的损害提供保护机制，该字段并不是以太网中使用的复杂的 CRC（Cyclic Redundancy Check，循环冗余校验），而是较为简单的针对 IP 报头的 16 比特校验和，路径上的每台路由器都会验证并重新计算该字段，如果校验和检查失败，路由器就会丢弃该数据包。
- **源地址（Source Address，32 比特）**：源地址字段是数据包发起方的 32 比特 IP 地址。
- **目的地址（Destination Address，32 比特）**：目的地址字段是最终目的地或数据包接收方的 32 比特 IP 地址，路由器利用该字段将数据包沿路径转发到最终目的地。

注：NAT 可以将源地址或目的地址更改为转换网关的某个地址，通常是 RFC 1918 的私有 IP 地址。有关 NAT 的内容已超出了本书范围，大家可参考 RFC 4787。

- **选项（Options，可变长度）**：选项字段为可选字段，因而可以出现在 IP 包中，也可以不出现在 IP 包中，该字段长度可变，大多数数据包都没有该字段，常见选项有记录路由选项、时间戳选项以及用来增强 traceroute 程序的跟踪路由选项，这些都定义在 RFC 1393（Traceroute Using an IP Option）中。
- **填充（Padding，可变长度）**：如果使用了一个或多个选项，并且 IP 报头的大小不再是 32 比特的整数倍，那么就会在报头填充比特 0 直至 32 比特边界。
- **数据（可变长度）**：是在 IP 包中进行传输并通过协议字段进行标识的数据，数据可以是其他三层协议（如 ICMP），也可以是 TCP 或 UDP 等高层协议。

## 2.2 IPv6 报头

IPv6 报头定义在 RFC 2460（Internet Protocol, Version 6 (IPv6) Specification）中，图 2-2 给出了 IPv6 报头的基本结构，有时也称为 IPv6 基本报头，IPv6 基本报头可以包含一个或多个扩展报头，有关扩展报头的内容将在本章后面进行讨论。

IPv6 报头及其各个字段如图 2-2 所示。

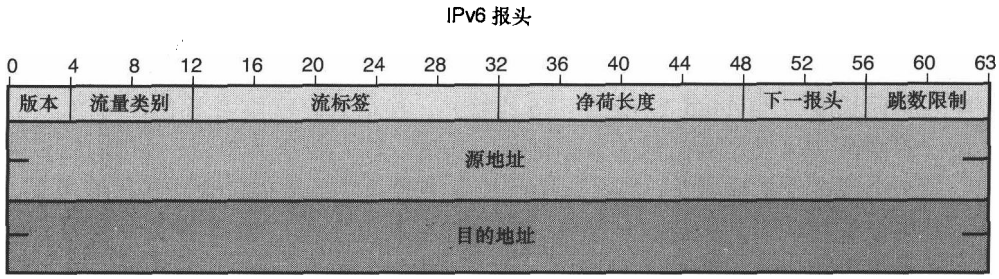



图 2-2　IPv6 报头

- **版本（Version，4 比特）**：版本（Version）字段包含了 IP 报头的版本号，其值始终为 6。
- **流量类别（Traffic Class，8 比特）**：流量类别字段的功能与 IPv4 报头中的 ToS 字段相似，并且长度与 IPv4 的 ToS 字段相同，只是名称发生的变化，流量类别字段用于识别并区分不同类别或优先级的 IPv6 数据包，IPv6 利用 RFC 2647（Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers）中定义的差分服务技术，使用 6 个比特作为 DSCP，从而能够提供 64 种可能的标记值，与最初的 IPv4 优先级（IPv4 Precedence）只有 3 比特（只有 8 种可选标记值）相比，这样就能实现更加精细化的优先级选择能力。

注：有关 DSCP 和 IP 优先级的内容已经超出了本书范围。一个有趣的事实是 IP 优先级的值实际上就是 DSCP 值的前 3 个比特（如图 2-3 所示），因此无法同时使用这两种值，如果使用了带有其余 3 个比特的 DSCP，那就意味着取代了 IP 优先级。

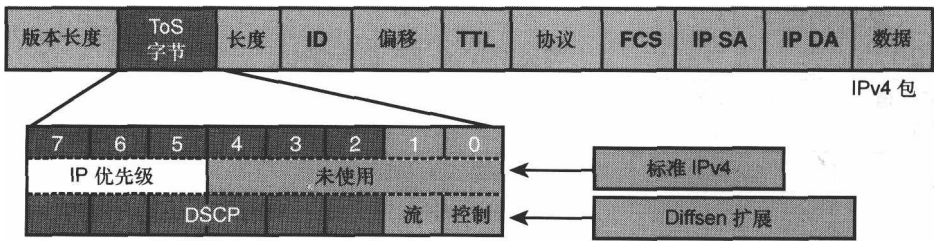



图 2-3　IPv4 ToS 字节

- **流标签（Flow Label，20 比特）**：流标签字段用来标记从源节点发送给一个或多个目的节点的一串 IPv6 数据包序列或 IPv6 数据包流，源节点可以利用该字段来标记那些请求 IPv6 路由器进行特殊处理（如实时业务）的数据包序列，流标签字段可以标识同一个流中的所有数据包，从而保证所有数据包都能得到 IPv6 路由器的相同处理。有关流标签的详细使用信息定义在 RFC 6437 “IPv6 Flow Label Specification” 中，路由器会记录这些数据包流，由于路由器无需

独立地处理每个数据包的报头，因而对于拥有多个数据包的流来说，处理效率更高。截至本书写作之时，该字段的使用方式仍处于试验阶段。

- **净荷长度（Payload Length，16 比特）**：净荷长度字段表示 IPv6 基本报头后的净荷（也就是数据包的数据部分）长度，以八位组为单位。如果 IPv6 包有一个或多个扩展报头，那么该净荷长度字段的字节数也包含这些扩展报头，扩展报头被认为是净荷的一部分。IPv6 净荷长度字段与 IPv4 报头中的数据包总长字段相似，但两者之间存在一个非常重要的差异，IPv4 的数据包总长字段包含 IPv4 报头和数据，而 IPv6 净荷长度字段仅指示数据部分的字节数，而不包含 IPv6 基本报头。由于 IPv4 报头有填充和选项字段，因而 IPv4 报头长度是可变的，而 IPv6 报头固定为 40 字节。

由于净荷长度字段为 16 比特，因而最大净荷尺寸是 65 535 字节，如果需要支持更大的数据包，那么就可以使用 IPv6 提供的巨包（Jumbogram）扩展报头，RFC 2675“IPv6 Jumbograms”指定了一个额外的 32 比特字段，用于传输净荷在 65 535～4 294 967 295 之间的 IPv6 包。有关扩展报头以及巨包净荷选项的详细内容将在本章后面讨论。

- **下一报头（Next Header，8 比特）**：下一报头字段有两个作用，如果只有 IPv6 基本报头而无扩展报头，那么下一报头字段指示的是 IPv6 包的数据部分所承载的协议，这一点类似于 IPv4 报头中的协议字段，而且与 IPv4 报头的协议字段使用相同的协议值，并有所增加，表 2-1 列出了常见的 IPv6 下一报头值，完整列表可参见 www.iana.org/assignments/protocol-numbers/protocol-numbers.xml。这些值看起来可能比较眼熟，因为大部分值都与 IPv4 协议字段相同，如 UDP 为 6，TCP 为 17。图 2-4 给出了一些包含下一报头字段的示例，表明该 IPv6 包是 TCP 报文段的数据部分。

**表 2-1　　常见的 IPv6 下一报头值**

| 下一报头值（十进制） | 下一报头值（十六进制） | 描述 |
|---|---|---|
| 0 | 0 | 用于 IPv6 的逐跳选项扩展报头 |
| 1 | 1 | ICMPv4（Internet Control Message Protocol version 4，因特网控制消息协议版本 4） |
| 2 | 2 | IGMPv4（Internet Group Management Protocol version 4，因特网组管理协议版本 4） |
| 4 | 4 | IPv4 封装 |
| 5 | 5 | IST（Internet Stream Protocol，因特网流协议） |
| 6 | 6 | TCP（Transmission Control Protocol，传输控制协议） |
| 8 | 8 | EGP（Exterior Gateway Protocol，外部网关协议） |
| 17 | 11 | UDP（User Datagram Protocol，用户数据报协议） |
| 41 | 29 | IPv6 封装 |

续表

| 下一报头值（十进制） | 下一报头值（十六进制） | 描述 |
|---|---|---|
| 43 | 2B | 用于 IPv6 的路由扩展报头 |
| 44 | 2C | 用于 IPv6 的分段报头 |
| 46 | 2E | RSVP（Resource Reservation Protocol，资源预留协议） |
| 47 | 2F | GRE（Generic Routing Encapsulation，通用路由封装） |
| 50 | 32 | ESP（Encapsulation Security Payload，封装安全净荷） |
| 51 | 33 | AH（Authentication Header，认证头） |
| 58 | 3A | ICMPv6（Internet Control Message Protocol version 6，因特网控制消息协议版本 6） |
| 59 | 3B | 用于 IPv6 的无下一报头 |
| 60 | 3C | 用于 IPv6 的目的地选项扩展报头 |
| 88 | 58 | EIGRP（Enhanced Interior Gateway Routing Protocol，增强型内部网关路由协议） |
| 89 | 59 | OSPF（Open Shortest Path First，开放最短路径优先） |

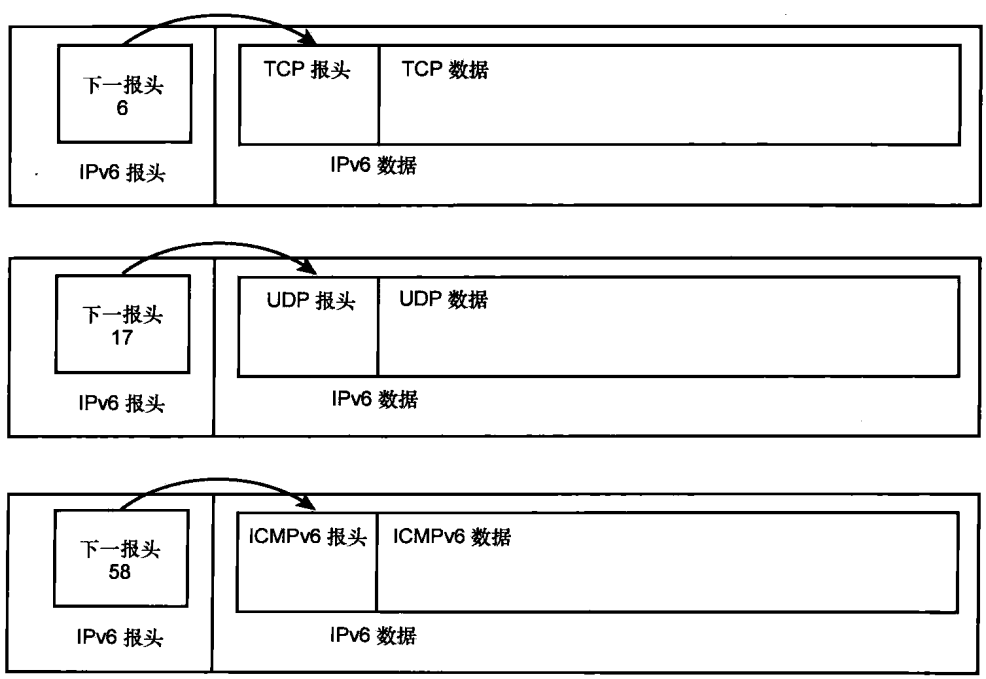



图 2-4　下一报头字段

注：虽然有关隧道的内容将在第 10 章进行讨论，不过现在需要说明的是，IP 包可以封装其他 IP 包。如果下一报头是 IPv4 报头，那么下一报头字段的值为 4，如果下一报头是另一个 IPv6 报头，那么下一报头字段的值为 41。

- **跳数限制（Hop Limit，8 比特）**：跳数限制字段与 IPv4 报头中的 TTL 字段相同，不过名字更能真实地反映路由器处理该字段的方式，即对跳数限制字段进行递减 1 操作。与 IPv4 的 TTL 字段一样，如果路由器将跳数限制字段的值从 1 递减到 0，那么就会丢弃该数据包，对 IPv6 来说，此时会发送一条 ICMPv6 超时消息，以通知数据包的源端“已丢弃该数据包”。有关 ICMPv6 的详细内容将在第 5 章进行讨论。
- **源地址（Source Address，128 比特）**：源地址字段是 IPv6 包发起方的 128 比特 IP 地址，与 IPv4 一样，该地址是最初发送该数据包的节点地址，源地址必须是单播地址。
- **目的地址（Destination Address，128 比特）**：目的地址字段是 IPv6 包最终目的节点或接收方的 128 比特 IP 地址，该字段表示最终目的地，可以是单播地址或多播地址。与 IPv4 不同，IPv6 无广播地址，但是有一个全部节点多播地址。有关 IPv6 地址的详细内容将在第 3 章与第 4 章进行讨论。

## 2.3 Wireshark 报文分析

下面将利用报文分析工具（如 Wireshark）来查看一下 IPv6 包的情况。以图 2-5 所示网络为例，从 PC1 向 PC2 发起简单的 ping 操作。

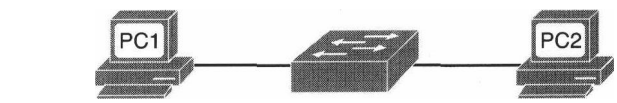



图 2-5 PC1 向 PC2 上的 IPv6 地址发起 ping 操作

```
PC1> ping 2001:0db8:aaaa:0001::0200

Pinging 2001:db8:aaaa:1::200 from 2001:db8:aaaa:1::100 with 32 bytes of data:
Reply from 2001:db8:aaaa:1::200: time<1ms
Reply from 2001:db8:aaaa:1::200: time<1ms
Reply from 2001:db8:aaaa:1::200: time<1ms
Reply from 2001:db8:aaaa:1::200: time<1ms
```

```
Ping statistics for 2001:db8:aaaa:1::200:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC1>
```

从输出结果可以看出，地址中的某些 0 被省略了，有关 IPv6 地址及其精简格式将在第 3 章进行讨论，此处大家只要知道这些都是同一个地址的不同表示方式即可。

IPv6 地址是 128 比特地址，以十六进制数字表示，虽然现在看见来有点陌生，不过不要紧，从第 3 章开始，大家就会慢慢熟悉这些地址了。图 2-6 显示了 Wireshark 抓取的 ICMPv6 回显请求（ping），并将在表 2-2 中进行详细分析，而有关 ICMPv6 的详细内容将在第 5 章进行讨论。

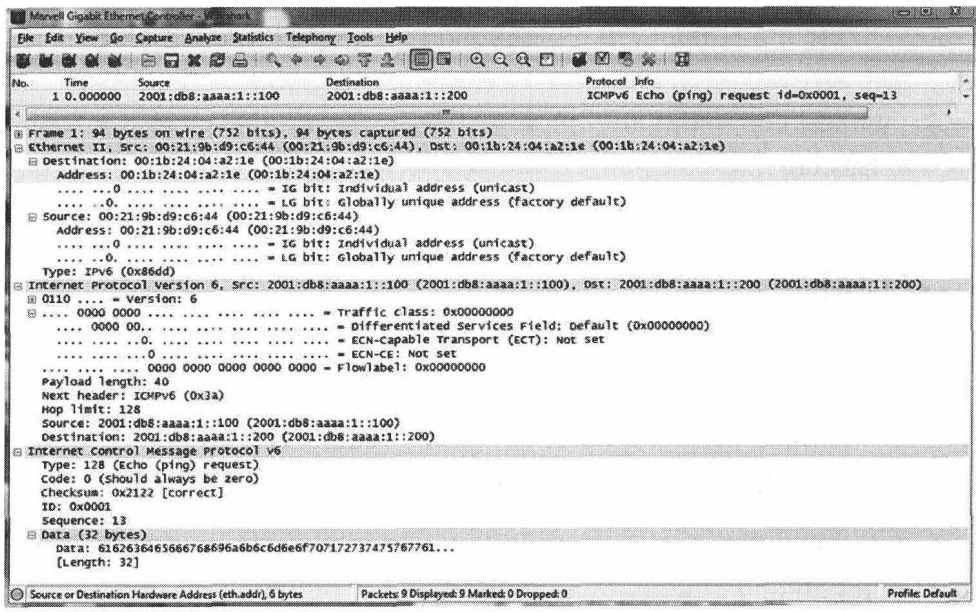



图 2-6　Wireshark 抓取的 IPv6 包

**表 2-2　　IPv6 包的分析**

| | 字段名称 | 大小（比特） | 值-描述 |
|---|---|---|---|
| IPv6 报头（40 字节） | 版本 | 4 | 6-表示 IP 版本 6 |
| | 流量类别 | 8 | 0-默认为 0 |
| | 流标签 | 20 | 0-默认为 0 |
| | 净荷长度 | 16 | 40 字节-表述数据部分的长度，本案例是 ICMPv6 消息，请注意，ICMPv6 消息长度为 40 字节 |

续表

| | 字段名称 | 大小（比特） | 值-描述 |
|---|---|---|---|
| | 下一报头 | 8 | 58-标识下一报头是 ICMPv6 报头，参见表 2-1 可以了解常见的 IPv6 下一报头值列表 |
| | 跳数限制 | 8 | 128-表示数据包被丢弃前最多所能经过的路由器数量 |
| | 源地址 | 128 | 2001:0db8:aaaa:1::100-以十六进制表示的源 IPv6 地址 |
| | 目的地址 | 128 | 2001:0db8:aaaa:1::200-以十六进制表示的目的 IPv6 地址 |
| ICMPv6 报头（40 字节） | 类型 | 8 | 128-表示这是一条 ICMPv6 回显请求消息 |
| | 代码 | 8 | 0-未使用；默认为 0 |
| | 校验和 | 16 | 0x2122-16 比特校验和的作用是验证 ICMPv6 报头的完整性 |
| | ID | 16 | 0x0001-用于帮助匹配 ICMPv6 回显请求和回显应答消息 |
| | 序列 | 16 | 13-用来帮助匹配 ICMPv6 回显请求和回显应答消息 |
| | 数据 | 256 | 可变长度的可选数据，取决于 ICMPv6 消息的类型 |

注：Wireshark 是一种适用于 IPv4 和 IPv6 的网络协议分析软件，大家可以从 www.wireshark.org 免费下载各种操作系统版本的 Wireshark 软件。

## 2.4 扩展报头

理解扩展报头相对较为困难，所以本节将采取步步深入的方式进行介绍。有些扩展报头很简单，而有些扩展报头则比较复杂，因此，即便有些内容理解起来有些含糊，也不必太在意，本节的目的是让大家熟悉扩展报头的概念以及相应的使用方式。

扩展报头是可选项，位于 IPv6 基本报头之后，如前所述，IPv6 报头包含下一报头字段，该字段的作用有两个：

- 一是标识 IPv6 包数据部分所承载的协议；
- 二是指示扩展报头的存在。

在前面已经说过，下一报头字段能够标识 IPv6 包数据部分所承载的协议，这一点与 IPv4 报头的协议字段相似（如图 2-5 所示）。

第二个作用是对 IPv6 报头的重要补充，用于指示被称为扩展报头的额外报头的存在。在必需的 IPv6 基本报头之后，可以有 0 个、1 个或多个扩展报头。所有扩展报头中都有的一个字段是另外的下一报头字段，表示接下来还有其他扩展报头，或者是数据（净荷）协议（如 TCP 报文段）。因此，最后的扩展报头总是指示哪种协议被封装在数据部分（净荷），这一点与 IPv4 的协议字段相似。

目前在 RFC 2460 中定义了 6 种扩展报头（如表 2-3 所示）。大家可能还记得，在 IPv4 报头中有一个很少使用的可变长度的选项字段，可以为 IPv4 提供一定的灵活性，而 IPv6 也有两个扩展报头提供了类似的功能：逐跳选项（Hop-by-Hop Option）报头和目的选项（Destination Option）报头。图 2-7 给出了使用这两种报头的 IPv6 包示例。

- IPv6 基本报头拥有前面讨论过的全部字段，包括源地址和目的地址字段，IPv6 基本报头中的下一报头字段为 0 则表示后面紧跟的是逐跳选项扩展报头。
- 逐跳选项扩展报头位于 IPv6 基本报头之后，有关扩展报头的详细内容将在下一节讨论，这里需要注意的是，该字段也包含自己的下一报头字段，值 51 表示后面还有其他的扩展报头，即 AH（Authentication Header，认证头）扩展报头。
- 最后的扩展报头就是 AH，其下一报头字段值为 6，表示后面跟的是 TCP 上层协议报头，也就意味着本数据包没有其他的扩展报头了。

表 2-3　　IPv6 扩展报头

| 下一报头值（十进制） | 扩展报头名 | 扩展报头长度（字节） | 使用可变长选项（TLV）？ | 扩展报头描述 |
|---|---|---|---|---|
| 0 | 逐跳选项 | 可变长度 | 是 | 用于承载选项信息，数据包传送路径上的每台路由器都必须检查该报头 |
| 43 | 路由 | 可变长度 | 否 | 允许数据包源端指定去往目的端的路径 |
| 44 | 分段 | 8 | 否 | 用于对 IPv6 包进行分段 |
| 50 | ESP | 可变长度 | 否 | 用于提供认证、完整性和加密功能 |
| 51 | AH | 可变长度 | 否 | 用于提供认证和完整性功能 |
| 60 | 目的选项 | 可变长度 | 是 | 用于承载选项信息，该信息只需数据包的目的节点处理 |

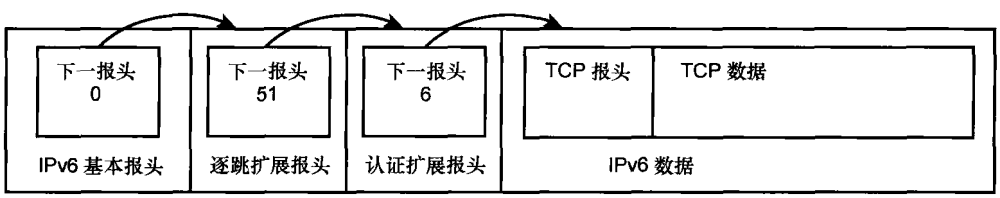



图 2-7　扩展报头中下一报头字段的使用

注：下一报头字段的作用是将多个 IPv6 报头链接在一起，链条的末尾是 IPv6 包的数据部分。

RFC 2460 建议同一个数据包使用多个扩展报头时，这些扩展报头的出现顺序如下：

1．IPv6 基本报头；

2. 逐跳选项报头；
3. 目的选项报头；
4. 路由报头；
5. 分段报头；
6. AH 报头；
7. ESP 报头；
8. 目的选项报头；
9. 上层协议报头。

### 2.4.1 逐跳选项扩展报头

逐跳选项报头用于承载选项信息，并且数据包传送路径上每台路由器都必须处理这些信息。逐跳选项报头是与 IPv4 选项字段类似的包含可变长度选项字段的两个扩展报头之一，顾名思义，逐跳选项报头是一种要求传送路径上每台路由器都必须处理的扩展报头。

> 注：目的选项报头是另一种使用选项的扩展报头，顾名思义，该扩展报头包含的信息只传递给目的端，本节将在最后详细讨论目的选项报头。

下面就来分析一下选项的使用。选项为 IPv6 包提供了很好的灵活性，可以通过扩展报头标准组中未定义的一组值对 IPv6 包进行补充定义，这组值也被称为 TLV（Type-Length-Value，类型-长度-值）三元组。目前已经定义了两种使用选项的扩展报头：逐跳选项报头和路由选项报头。如图 2-8 所示，这两类扩展报头都有下一报头字段和报头扩展长度字段，然后是一个或多个选项组，每个选项都包含一组选项类型（Option Type）字段、选项长度（Option Length）字段和选项数据（Option Data）字段（即 TLV）。

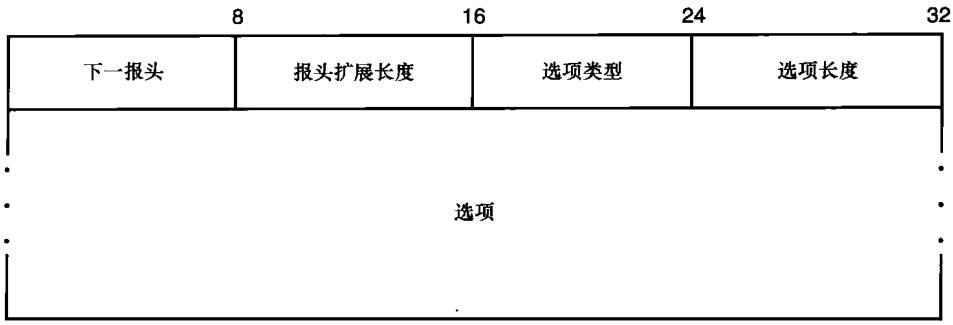



图 2-8 扩展报头选项

图 2-9 显示了包含巨包净荷选项（Jumbo Payload Option）的逐跳选项报头格式。巨包净荷选项用于表示该 IPv6 包大于 65 535 字节。由于这是逐跳选项，因而路径上的所有路由器都必须检查该信息。

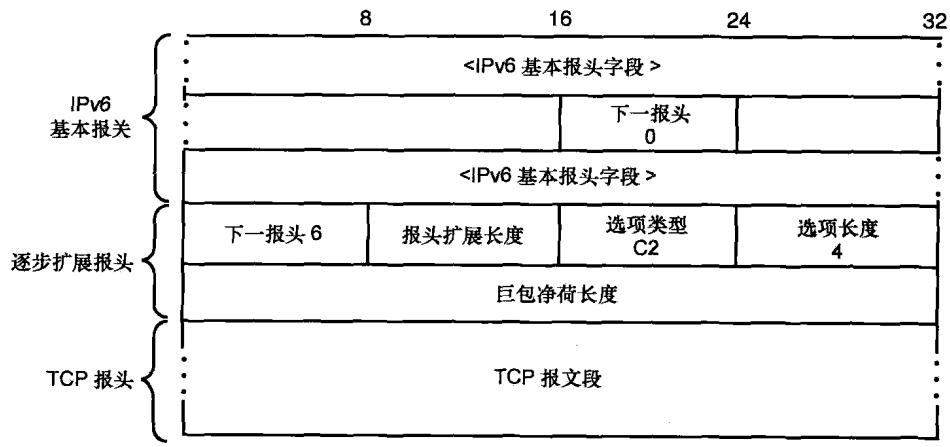



图 2-9　包含巨包净荷选项的逐跳选项报头

下面列出了与逐跳选项扩展报头相关的字段信息。

- **IPv6 基本报头：下一报头（Next Header，8 比特）**：除了 IPv6 基本报头的其他信息之外，还有一个值为 0 的下一报头字段，表明基本报头后面有一个逐跳选项扩展报头。
- **逐跳选项扩展报头。**
  - **下一报头（Next Header，8 比特）**：下一报头字段值为 6，表明该报头后面是 TCP 报头，没有其他扩展报头。
  - **报头扩展长度（Header Extension Length，8 比特）**：表示以八位组为单位的逐跳选项报头的长度，不包含第一个八位组。可以有多个选项，每个选项都包含一个 TLV（选项类型、选项长度和选项数据字段）。
  - **选项类型（Option Type，8 比特）**：表示该报头所承载的选项类型，十六进制值 C2 表明这是一个巨包净荷选项。
  - **选项长度（Option Length，8 比特）**：表示选项数据字段中的字节数，值 4 表示选项数据长度为 4 字节（32 比特）。
  - **选项数据（可变长度）**：本例中的数据是巨包净荷长度。巨包净荷长度是一个 32 比特字段，表示以字节为单位的 IPv6 包大小，不包括 IPv6 报头，但包括逐跳选项扩展报头以及其他扩展报头。巨包净荷长度必须大于 65 535 字节，最大可以达到 4 294 967 295 字节。

- **TCP 报文段**：由于只有一个选项且没有其他扩展报头，因而后面跟着的是 TCP 报文段，从前面的逐跳选项扩展报头中下一报头值为 6 即可看出。

如果使用了逐跳选项扩展报头，那么该报头必须紧跟在 IPv6 基本报头之后。

### 2.4.2 路由扩展报头

路由扩展报头允许数据包源端指定去往目的端的路径，该报头包含去往数据包目的端路径上的一台或多台中间路由器，该功能与 IPv4 使用的松散源路由选项非常类似，路由扩展报头由前一个报头中的下一报头值 43 来标识。

图 2-10 显示了路由类型为 2（在 IPv6 中支持移动性）的路由扩展报头结构。该扩展报头允许将数据包从通信端直接路由到移动节点的转交地址（Care-of Address），扩展报头提供了移动节点的当前位置信息。

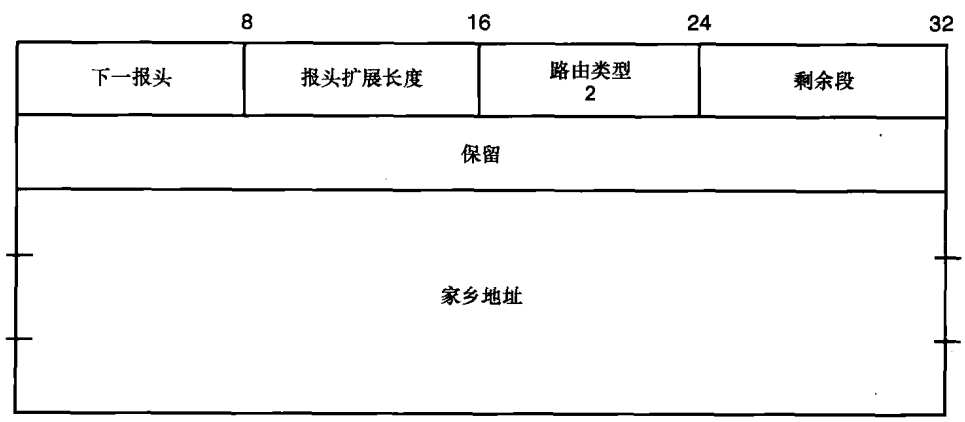



图 2-10 类型 2 的路由扩展报头

下面列出了与路由扩展报头相关的字段信息。

- **下一报头（Next Header，8 比特）**：表示路由扩展报头后面下一个报头的类型，要么是其他的扩展报头，要么是净荷协议。
- **报头扩展长度（Header Extension Length，8 比特）**：表示以八位组为单位的路由报头的长度，不包含第一个八位组。
- **路由类型（Routing Type，8 比特）**：值为 2。
- **剩余段（Segments Left，8 比特）**：值为 1。
- **保留（Reserved，32 比特）**：该字段被保留，传输时被初始化为 0，并且被接收端忽略。
- **家乡地址（Home Address，128 比特）**：表示目的端移动节点的家乡地址。

注：有关路由器如何处理路由扩展报头的内容超出了本书范围，如果感兴趣，可以参考 RFC 2460 或 Cisco Press 出版的 *Cisco Self-Study: Implementing Cisco IPv6 Networks*”。许多 ISP 对源节点在数据包的下一跳选择问题上持怀疑态度，通常都会根据管理需要阻塞包含了路由扩展报头的数据包，目前唯一有效的路由扩展报头就是类型 2，该类型报头用于 IPv6 中的移动性支持。

### 2.4.3　分段扩展报头

如图 2-11 所示，分段扩展报头与 IPv4 报头中用于分段目的的字段相似。当 IPv6 数据包的源端需要将数据包分段并将每个分段都作为一个单独的数据包进行发送时，就需要用到分段扩展报头。数据包的接收端再重组这些分段后的数据包，每个数据包都有自己的 IPv6 基本报头和分段扩展报头。

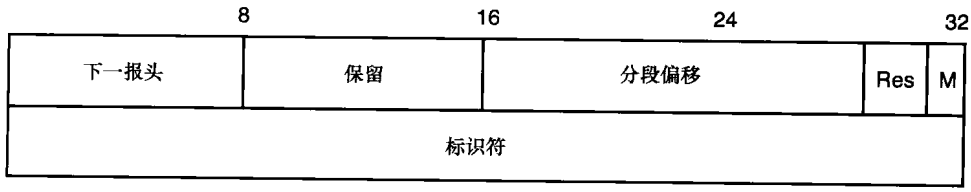



图 2-11　分段扩展报头

与 IPv4 不同，对每个被分段的数据包来说，源端会生成一个唯一的标识值，并且每个分段后的数据包中都会包含该标识值。该标志值可以确保接收端能够正确地重组来自原始数据包的各个片段。如果源端还需要对同一条消息中的其他数据包进行分段，那么就要使用不同的标志值。

下面列出了与分段扩展报头相关的字段信息。

- **下一报头（Next Header，8 比特）**：表示原始数据包被分段后的数据的协议号。
- **保留（Reserved，8 比特）**：该字段被保留，传输时被初始化为 0，并且被接收端忽略。
- **分段偏移（Fragment Offset，13 比特）**：表示被分段后的数据在该报头之后的相对偏移量或位置，以八位组为单位。与 IPv4 报头中的分段偏移相似，该字段的目的是指示接收端如何将该分段后的数据包与其他分段后的数据包进行排列。
- **Res（2 比特）**：该字段被保留，传输时被初始化为 0，并且被接收端忽略。
- **M 标志（M flag，3 比特）**：M（More Fragments，更多分段）标记用来表示

是否是最后一个分段（比特 0）或者后面还有更多分段（比特 1），该字段与 IPv4 报头的更多分段标志（More Fragments Flag）相似。

- **标识符（Identification，32 比特）**：该字段与 IPv4 报头的标识符字段相似，用来唯一的标识同一个原始数据包中的所有分段数据包，该字段从 IPv4 中的 16 比特扩充到 IPv6 中的 32 比特。

### 2.4.4 IPSec：AH 和 ESP 扩展报头

IPv6 利用以下两种扩展报头来实现 IPSec 中的两种关键的安全协议：

- AH（Authentication Header，认证头）；
- ESP（Encapsulating Security Payload，封装安全净荷）。

#### 1. IPSec

在讨论 AH 和 ESP 扩展报头之前，首先简要回顾一下 IPSec 以及这两种安全协议的主要功能，目的不是要求大家完全掌握 IPSec、AH 和 ESP，而是希望为大家理解其重要性以及在 IPv6 中的使用分方法提供足够的背景知识。

IPSec 是保障数据包在 IP 网络中传输安全性的一组协议集。

AH 和 ESP 是为整个 IPv6 包或部分 IPv6 包提供认证与完整性功能的主要安全协议，此外，ESP 还能够提供加密功能。

> 注：IPSec 是 IPv4 和 IPv6 的一部分，但实现 IPv4 协议栈的设备并不强制要求提供 IPSec 功能，早期的 RFC 要求所有的 IPv6 实现都必须强制支持 IPSec，描述方式是“必须支持”IPSec，不过最新的 RFC 6343“IPv6 Node Requirements”放松了这一要求，相应的描述方式调整为“应该实现”IPSec。

AH 可以提供数据包的真实性和完整性保证，认证（Authentication）的作用是确保数据包的发送方和接收方身份都是真实的，完整性（Integrity）的作用是保证数据包在发送途中未被更改。AH 可以提供认证和完整性功能，但不提供加密功能，加密（Encryption）是利用一定的算法（被称为密码）转换信息（通常是明文）的过程，以确保只能由拥有特殊信息（通常称为密钥）的接收方读取。

ESP 可以提供认证、完整性和加密功能，ESP 不仅可以保护数据包不被中间设备更改，而且还能保护数据包的内容不被查看，ESP 有自己的认证方案，也可以与 AH 配合使用。总得来说，AH 仅提供认证和完整性功能，而 ESP 除了提供认证和完整性功能之外，还能对数据包进行加密。

到现在为止，还没有说到究竟需要对数据包的多少内容进行认证或加密，这个问题的答案取决于 IPSec 的工作模式，即传输模式或隧道模式。

### 2. 传输模式与隧道模式

顾名思义，传输模式保护传输层及更高层，仍然使用原始的 IP 报头。由于原始的源 IP 地址和目的 IP 地址都在 IP 基本报头内，因而中间设备路由器不是 IPSec 的参与者。传输模式通常用于主机之间的通信。

隧道模式用于保护 IP 包的全部内容，包括 IP 报头。实现方式是将原始 IP 包（包括 IP 报头）封装到一个新的 IP 报头中，隧道端点作为新的源 IP 地址和目的 IP 地址，隧道端点可以是路由器或主机本身。隧道模式可以保护整个 IP 包，而传输模式不行。图 2-12 解释了传输模式与隧道模式之间的区别。

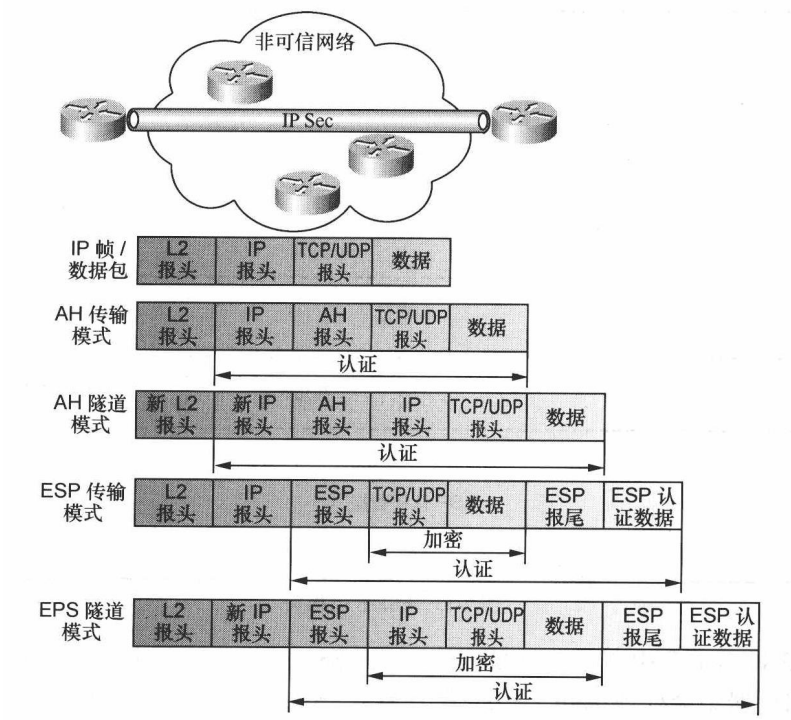



图 2-12　传输模式与隧道模式

下面将开始讨论 AH 扩展报头和 ESP 扩展报头，如果是初次接触 IPSec，那么对某些内容感到困惑是完全可以理解的。

## 2.4.5　ESP 扩展报头

ESP 扩展报头是一种可变长度的扩展报头，如前所述，该报头用于提高认证、完整性和加密等功能，ESP 扩展报头是由前一个报头中的下一报头字段值 50 来标识的。

图 2-13 显示了 ESP 扩展报头的结构，ESP 扩展报头可以分为 4 部分。

- **ESP 报头**：SPI（Security Parameter Index，安全参数索引）和序列号字段。
- **净荷**：ESP 净荷数据字段。
- **ESP 报尾**：填充、填充长度以及下一报头字段。
- **ESP 认证数据**。

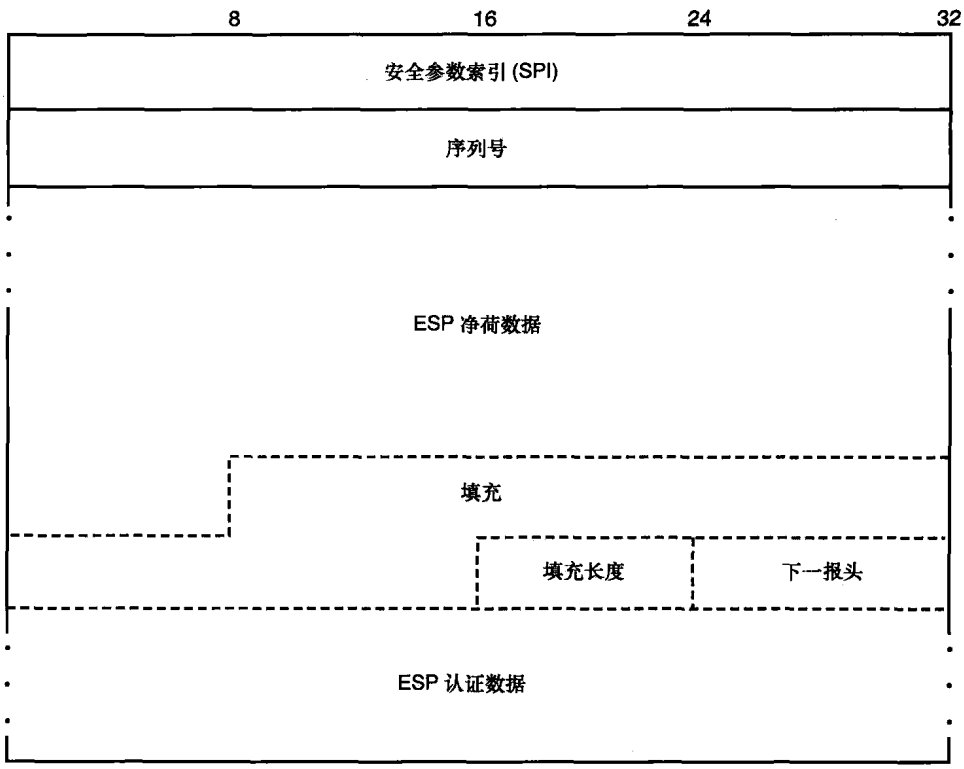



图 2-13 ESP 扩展报头

图 2-13 显示了 ESP 扩展报头的各个字段，ESP 被视为端到端通信机制，也就是说，路径上的路由器不会处理 ESP 报头。请注意，ESP 为原始数据包提供了认证完整性与机密性机制，因此，ESP 扩展报头被封装在 IPv6 基本报头、逐跳选项扩展报头、路由扩展报头以及分段扩展报头之后（如图 2-14 所示）。对 IPv6 来说，加密操作会涵盖整个传输层报文段、ESP 报尾以及目的选项扩展报头（如果目的选项扩展报头位于 ESP 报头之后），目的选项扩展报头可以位于 ESP 报头之前或之后，这取决于设计意图。

与前面讨论过的扩展报头不同，有关 ESP 扩展报头各个字段的内容已超出了本书范围，需要大家对 IPSec 有比较深入的认识和理解，如果大家初次接触 IPSec，那么对某些内容感到困惑是完全可以理解的，有关 IPSec 的详细信息，可参考 Cisco Press 出版的由 James Henry Carmouche 编写的图书 *IPSec Virtual Private Network Fundamentals*，这是一本非常好的学习 IPSec 知识的图书。

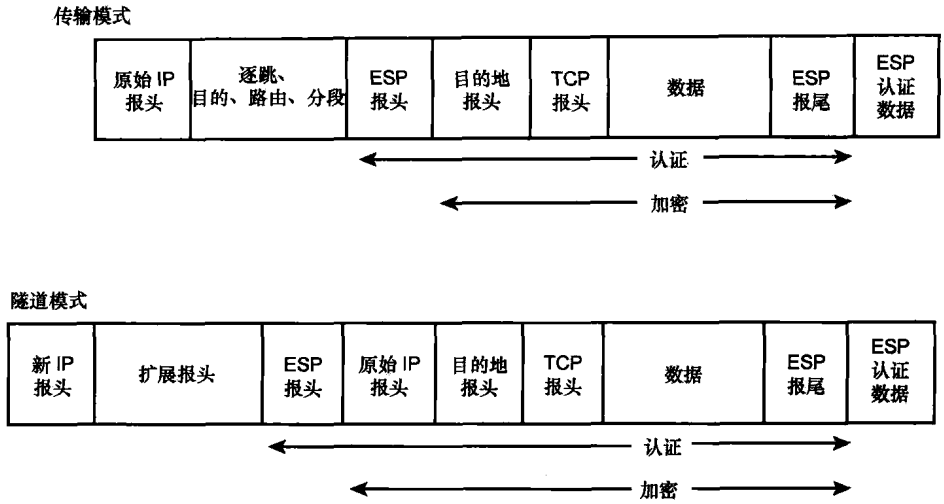



图 2-14　ESP—传输模式和隧道模式

### AH 扩展报头

AH 扩展报头也是一种可变长度的扩展报头，与 ESP 不同的是，AH 仅提供认证和完整性功能，并不使用加密机制来提供机密性功能，AH 扩展报头是由前一个报头中的下一报头字段值 51 来标识的。

图 2-15 显示了 AH 扩展报头的结构，与 ESP 一样，AH 也被视为端到端通信机制，请注意，AH 仅提供数据的完整性机制，可以确保通信参与方的身份，并且接收方可以据此检测出数据包内容在传送过程中是否被更改。与 ESP 相似，AH 扩展报头也被封装在 IPv6 基本报头、逐跳选项扩展报头、路由扩展报头以及分段扩展报头之后（如图 2-16 所示）。目的选项扩展报头可以位于 ESP 报头之前或之后，这取决于设计意图。

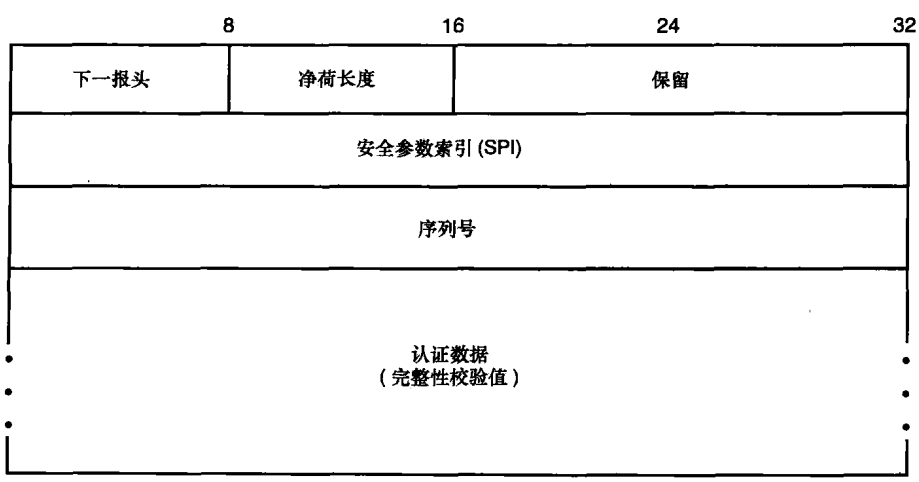



图 2-15　AH 扩展报头

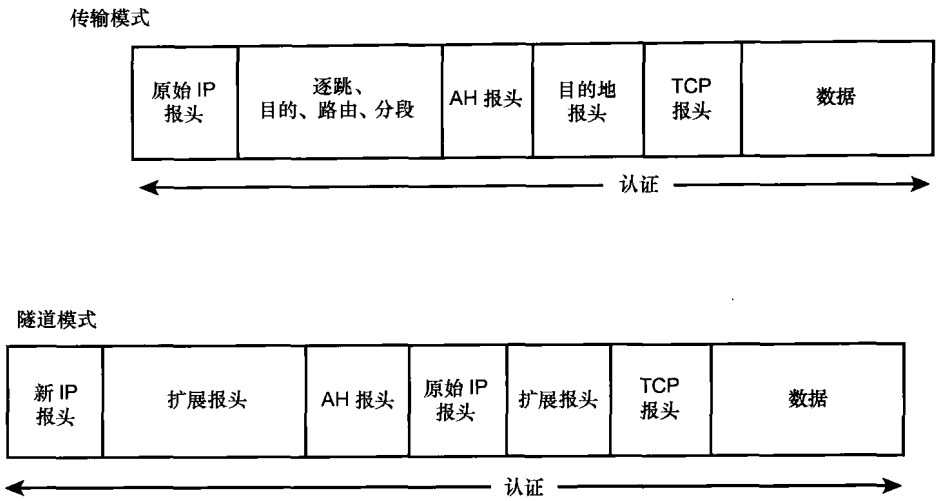



图 2-16 AH—传输模式和隧道模式

与其他 IPSec 扩展报头一样，有关 AH 扩展报头各个字段的内容已超出了本书范围，需要大家对 IPSec 有比较深入的认识和理解。

### 2.4.6 目的选项扩展报头

目的选项扩展报头用于承载仅需要数据包目的节点处理的选项信息，除了逐跳选项扩展报头之外，目的选项扩展报头是另一个使用选项的扩展报头，目的选项扩展报头是由前一个报头中的下一报头字段值 60 来标识的。如图 2-17 所示，目的选项扩展报头的格式如下。

- **下一报头（Next Header，8 比特）**：用于标识目的选项扩展报头后面的报头类型，既可以是其他扩展报头，也可以是净荷协议。
- **报头扩展长度（Header Extension Length，8 比特）**：表示以八位组为单位的目的选项扩展报头的长度，不包含第一个八位组。
- **选项（可变长度）**：该字段包含一个和多个采取 TLV 编码的选项。
  - **选项类型（Option Type，8 比特）**：表示该报头所承载的选项类型。
  - **选项长度（Option Length，8 比特）**：表示选项数据字段中的字节数。
  - **选项数据（可变长度）**：即数据包的数据内容。

注：有关目的选项扩展报头的一种建议使用方式是 IPv6 的移动性支持，相关内容定义在 RFC 6275 “Mobility Support in IPv6”。

“对每个移动节点来说，无论其连接互联网的当前附着点在哪里，都是由其家乡地址来标识的。当移动节点离开家乡时，就会与一个转交地址产生关联，转交地址可以提

供该移动节点的当前位置信息，传送给移动节点的 IPv6 数据包都会被透明路由到这个转交地址，协议要求 IPv6 节点缓存移动节点的家乡地址与转交地址之间的绑定关系，然后就可以将发送给该移动节点的数据包直接发送给转交地址。为了支持该操作，移动 IPv6 定义了一个新的 IPv6 协议和一个新的目的选项，这样一来，所有 IPv6 节点（移动节点或固定节点）都能与移动节点进行通信。”

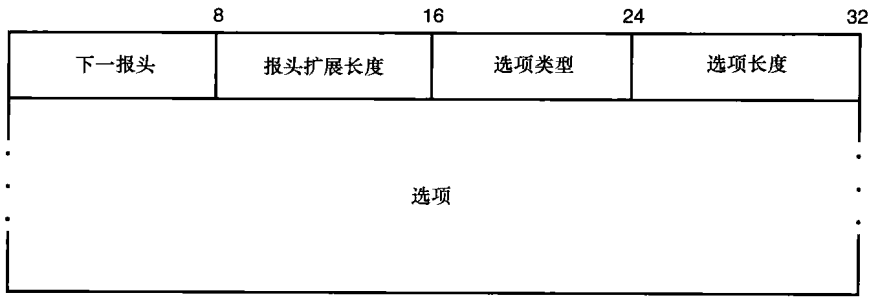



图 2-17　目的选项扩展报头

### 2.4.7　无下一报头

下一报头字段值为 59 时表示该报头后面无其他数据。它仅仅是一个占位符，表示该报头后面无任何数据。假如净荷长度指示该报头后面还有其他字节，那么这些字节都会被忽略。

## 2.5　IPv4 与 IPv6 对比

了解了 IPv4 与 IPv6 报头细节之后，很自然就会想到两者之间到底有何重要区别，由于有很多知识需要我们消化，因而以图 2-1 和图 2-2 为参考进行简要归纳。

### 2.5.1　IPv4 与 IPv6 报头对比

以下 IPv4 报头字段的名称与 IPv6 报头保持一致。

- **版本（IPv4 和 IPv6）**：这是最简单的一个字段，在 IPv4 中值为 4，在 IPv6 中值为 6。
- **源地址和目的地址（Pv4 和 IPv6）**：最大的差别就是 IPv4 的源地址和目的地址是 32 比特，而 IPv6 的源地址和目的地址为 128 比特。

以下 IPv4 报头字段在 IPv6 中的名称出现了变化，而且在某些场合下功能也有所变化。

- **ToS（IPv4）→流量类别（IPv6）**：IPv4 既可以使用 3 比特 IP 优先级字段，

同时将其余 3 比特用作时延、吞吐量和可靠性，也可以使用 6 比特 DS 技术，而 IPv6 在设计时就规定了使用 6 比特 DS 技术。

- **数据包总长度（IPv4）→净荷长度（IPv6）**：IPv4 的总长度字段包含 IPv4 报头和数据部分，而 IPv6 的净荷长度字段仅指示数据部分（即净荷）的字节数，包括所有扩展报头，但不包括 IPv6 基本报头。
- **TTL（IPv4）→跳数限制（IPv6）**：这两个字段在 IPv4 和 IPv6 中的功能相同，只是 IPv6 中的名称更能反映该字段的实际使用方式。
- **协议（IPv4）→下一报头（IPv6）**：IPv4 中的协议字段用于标识 IPv4 数据部分（即净荷）所承载的协议类型，IPv6 中的下一报头字段也提供了相同的功能，同时还能标识 IPv6 基本报头之后还有扩展报头。

以下 IPv4 报头字段被 IPv6 取消了。

- **IHL（IPv4）**：由于 IPv6 基本报头的长度固定为 40 字节，因而 IPv6 不需要 IPv4 中的该字段。IPv6 中除基本报头之外的其他报头都通过下一报头字段进行指示。
- **标识符（IPv4）、标志（IPv4）以及分段偏移（IPv4）**：IPv4 报头利用这些字段进行数据包的分段操作，而 IPv6 对分段操作采取了不同的处理方式，使用的是分段扩展报头。
- **报头校验和（IPv4）**：由于二层数据链路层技术（如以太网）会执行自己的校验和与差错控制机制，上层协议（如 TCP 和 UDP）也有自己的校验和机制，因而在三层执行校验和操作显得多余且不必要。对于 UDP 校验和来说，在 IPv4 中是可选操作，而在 IPv6 中则是强制操作。
- **选项（IPv4）**：IPv4 中的选项字段被 IPv6 中的扩展报头所取代，IPv6 中的逐跳选项扩展报头和目的选项扩展报头都有自己的 TLV 选项集。
- **填充（IPv4）**：由于 IPv6 基本报头固定为 40 字节，因而无需通过填充比特来确保其长度为 32 比特的整数倍。

以下字段是 IPv6 报头中的新增字段。

- **流标签（IPv6）**：该字段是 IPv6 报头的新增字段，目前有关该字段的使用方式仍在 IETF 的讨论之中。RFC 2460 仅讨论了使用流标签字段来标记数据包的顺序，以便由 IPv6 路由器对“实时”业务实施特殊处理。RFC 6437“IPv6 Flow Label Specification”则定义了流标签字段的一些额外细节信息。

### 2.5.2 其他差异

IPv4 与 IPv6 还存在一些其他重要差异。IPv6 使用了逐跳扩展报头和巨包净荷选项扩展报头，极大地扩展了 IP 包的可能尺寸，由 IPv4 的最大 65 535 字节到 IPv6 的最大

4 294 967 295 字节。

### 1. 更大的 MTU

IPv4 要求每个节点在不进行分段的情况下都能转发 68 字节的 IP 包，这是因为 IPv4 报头最长可达 68 字节或者最小分段尺寸为 8 字节。对于 IPv4 包的最终目的地来说，每个 IPv4 节点都必须能够接收最小为 576 字节的 IPv4 包（可以是整个原始数据包，也可以是多个分段后的数据包）。

IPv6 要求每条链路的最小 MTU 为 1280 字节，建议 MTU 为 1500 字节，而 IPv4 的最小 MTU 为 68 字节。

> 注：RFC 1981 “Path MTU Discovery for IP version 6” 建议 IPv6 应该执行 PTMU（Path Maximum Transmission Unit，路径最大传输单元）发现操作，以避免分段。

### 2. UDP

IPv4 报头中的 UDP 校验和字段是可选项，虽然 IPv6 报头中也有同样的字段，但 IPv6 的校验和字段却是强制性的，这是因为 IPv4 报头有自己的校验和字段，而 IPv6 报头则取消了该字段。校验和字段的作用是验证 UDP 报头及数据的完整性。

> 注：对 IPv4 和 IPv6 来说，TCP 中的校验和字段都是强制性的，运行在 IPv6 上的 TCP 和 UDP 都没有做结构性的修改，本书将在第 9 章讨论 TCP、UDP 以及其他上层协议。

### 3. 分段

在前面讨论分段扩展报头时说过，与 IPv4 不同，IPv6 路由器不对数据包进行分段，除非该路由器是数据包的源端。只有 IPv6 数据包的源节点才执行分段操作，如果中间节点（如路由器）收到一个需要被分段的 IPv6 包，就会丢弃该数据包并向源节点发送一条 ICMPv6 “分组过大” 差错消息。有关分段和路径 MTU 发现的详细内容将在第 5 章进行讨论。

## 2.6　本章小结

本章详细分析了 IPv4 报头与 IPv6 报头，对比了这两种协议之间的异同点。IPv6 报头的字段较少，在很多方面都是一个较为简单的协议。某些字段从 IPv4 直接迁移到 IPv6 中保持不变，某些字段则更改了名称，并存在一定的功能差异，还有一些字段则完全被 IPv6 所取消，同时还增加了一个新的流标签字段。

IPv6 引入了扩展报头，提升了 IPv6 的灵活性和效率。本章还解释了 IPv6 对 UDP 和 MTU 所带来的影响。

第 3 章将详细讨论 IPv6 地址的表示方式以及 IPv6 单播地址的通用结构。

## 2.7 参考文献

RFC：

RFC 791, Internet Protocol, DARPA Internet Program Protocol Specification , USC, www.ietf.org/rfc/rfc791.txt , September 1981

RFC 1191, Path MTU Discovery , J. Mogul, Stanford University, www.ietf.org/rfc/rfc1191.txt , November 1990

RFC 1393, Traceroute Using an IP Option , G. Malkin, Xylogics, Inc., www.ietf.org/rfc/rfc1393.txt , January 1993

RFC 1700, Assigned Numbers , J. Reynolds, ISI, IETF, www.ietf.org/rfc/ rfc1700.txt ,October 1994

RFC 1981, Path MTU Discovery for IP version 6 , J. McCann, Digital Equipment Corporation, www.ietf.org/rfc/rfc1981 , August 1996

RFC 2460, Internet Protocol, Version 6 (IPv6) Specification , S. Deering, Cisco Systems, IETF, www.ietf.org/rfc/rfc2460.txt , December 1998

RFC 2474, Using a Technique Called Differentiated Services (DS) , K. Nichols, Cisco Systems, www.ietf.org/rfc/rfc2474.txt , December 1998

RFC 2675, IPv6 Jumbograms , D. Borman, Berkeley Software Design, www.ietf.org/rfc/rfc2675.txt , August 1999

RFC 3775, Mobility Support in IPv6 , D. Johnson, Rice University, www.ietf.org/rfc/rfc3775.txt , June 2004

RFC 6434, IPv6 Node Requirements , E. Jankiewicz, SRI International, www.ietf.org/rfc/rfc6434 , December 2011

RFC 6437, IPv6 Flow Label Specification , S. Amante, Level 3, www.ietf.org/rfc/rfc6437 , November 2011

网站：

www.iana.org/assignments/protocol-numbers/protocol-numbers.xml

# 第3章 IPv6编址

IPv4与IPv6之间最显著的区别是地址空间。IPv4地址长度为32比特，采取点分十进制表示方式，而IPv6地址长度为128比特，采用十六进制表示方式。有关IPv6编址形式的详细内容请参见RFC 4291“IP Version 6 Addressing Architecture”。

通过本章的学习，大家将会熟悉IPv6地址的表示方式并能识别地址的不同部分。首先简要介绍各种不同类型的IPv6地址以及全局单播地址的基本结构，然后利用IPv6地址配置路由器的接口并使用ping命令来验证接口的可达性。此外，本章还会讨论如何对IPv6地址划分子网。在大多数情况下，IPv6的子网划分比IPv4的子网划分要简单得多。

虽然初看起来，较长且以十六进制表示的IPv6地址会令人望而生畏，但实际上，大家完全不必有此担忧。与IPv4相比，IPv6地址的阅读更为简单，而且在子网划分上更为简单。我相信，等大家学习完本书之后，一定会更加喜欢IPv6地址，而不再是IPv4地址！现在的当务之急就是让大家先理解十六进制编号系统以及IPv6的地址标记法。

## 3.1 十六进制编号系统

本节主要针对于对十六进制编号系统不熟悉的读者，如果已经熟悉了十六进制编号系统，那么就完全可以跳过本节。IPv6地址的长度是128比特，大家将会发现十六进制是表示长比特字符串的理想编号系统。

只要理解了十进制（即基数10）编号系统，那么就会很容易理解各种编号系统，包括十六进制编号系统（即基数16）。以下内容假定大家已掌握二进制（即基数为2），不过即便大家不了解二进制，通过本节的学习也将会理解基数16，因为所有编号系统的原理是共通的。

考察整数编号系统时，存在以下三条通用准则。

准则 1：基数 n 编号系统有 n 个数字。

- 基数 10（十进制）编号系统有 10 个数字；
- 基数 2（二进制）编号系统有 2 个数字；
- 基数 16（十六进制）编号系统有 16 个数字。

准则 2：所有编号系统的起始数字均为 0。

结合准则 1 与准则 2，可以发现：

- 基数 10 编号系统有 10 个以 0 开始的数字：0、1、2、3、4、5、6、7、8、9；
- 基数 2 编号系统有 2 个以 0 开始的数字：0、1；
- 基数 16 编号系统有 16 个以 0 开始的数字：0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F（稍后将会讨论字符型数字）。

准则 3：第 1 列（即最右列或最小有效数）始终是 1 的倍数列，而前一列总是上一列的 n 倍（*n* 表示基数 n 编号系统）。以基数 10 为例，第 1 列是 1 的倍数列（个位），下一列是 1 的倍数列的 10 倍，也就是 10 的倍数列（十位），再下一列是 10 的倍数列的 10 倍，即 100 的倍数列（百位），依此类推。因此将其他编号系统转换成基数 10 的编号系统非常简单（如表 3-1 所示）。

表 3-1　　编号

| 基数 n 编号系统 | $n^3$ | $n^2$ | $n^1$ | $n^0$ |
|---|---|---|---|---|
| 基数 10 | 1000 | 100 | 10 | 1 |
| 基数 2 | 8 | 4 | 2 | 1 |
| 基数 16 | 4096 | 256 | 16 | 1 |

- **基数 10**：10 000、1000 倍、100 倍、10 倍、1 倍；
- **基数 2**：128 倍、64 倍、32 倍、16 倍、8 倍、4 倍、2 倍、1 倍；
- **基数 16**：256 倍、16 倍、1 倍。

只要理解了这三条准则，就可以进一步了解十六进制编号系统了。十六进制编号系统（基数 16）有 16 个从 0 开始的数字，表 3-2 列出了这 16 个数字以及与之对等的十进制和二进制数。

表 3-2　　十进制、十六进制与二进制

| 十进制（基数 10） | 十六进制（基数 16） | 二进制（基数 2） |
|---|---|---|
| 0 | 0 | 0000 |
| 1 | 1 | 0001 |
| 2 | 2 | 0010 |
| 3 | 3 | 0011 |
| 4 | 4 | 0100 |
| 5 | 5 | 0101 |

续表

| 十进制（基数 10） | 十六进制（基数 16） | 二进制（基数 2） |
|---|---|---|
| 6 | 6 | 0110 |
| 7 | 7 | 0111 |
| 8 | 8 | 1000 |
| 9 | 9 | 1001 |
| 10 | A | 1010 |
| 11 | B | 1011 |
| 12 | C | 1100 |
| 13 | D | 1101 |
| 14 | E | 1110 |
| 15 | F | 1111 |

将上述三条准则应用于十六进制编号系统即可得到：

- **准则 1**：十六进制编号系统有 16 个数字；
- **准则 2**：表 3-2 列出了以 0 开始的 16 个十六进制数字以及与之相对等的十进制数和二进制数。需要注意的是，十六进制以 A～F 的字符型数字来表示十进制值 10～15；
- **准则 3**：为了以十六进制方式表示 IPv6 地址，只需使用第 1 列（即 1 的倍数列）。

那么，为何要使用十六进制编号系统来表示 IPv6 地址呢？十六进制是计算机科学、计算机网络以及其他计算机技术领域中应用非常普遍的编号系统。这是因为任意 4 个比特（半个字节或半个八位组）都可以被表示为一个十六进制数。也就是说，4 个比特有 16 种不同的组合，而十六进制编号系统也刚有 16 个数字，两者非常匹配。由于一个十六进制数可以表示 4 个比特，因此两个十六进制数可以表示一个字节或一个八位组。

> 注：由于 4 个比特是半个字节或半个八位组，所以有时也被称为半字节（nibble），有时也会出现半字节的其他拼写方式 nybble 或 nyble。

## 3.2　IPv6 地址表示形式

IPv6 地址长度为 128 比特，以一串十六进制数字来表示。每 4 个比特表示一个十六进制数，一共有 32 个十六进制数值（4×32=128）。十六进制中的字符型数字不区分

大小写，即大写字母与小写字母完全相同。

> 注：RFC 5952“A Recommendation for IPv6 Address Text Representation”建议 IPv6 地址使用小写字母，不过本书在很多场合下仍使用大写字母，目的是让大家看起来更容易一些，也更能发现不同地址类型的差异。

如 RFC 4291 所述，优选的 IPv6 地址格式是 x:x:x:x:x:x:x:x。其中，每个 *x* 都是一个十六比特数，可以最多使用 4 个十六进制数字来表示（用冒号来分隔），因此 IPv6 地址包括 8 个 16 比特段，一共 128 比特（如图 3-1 所示）。

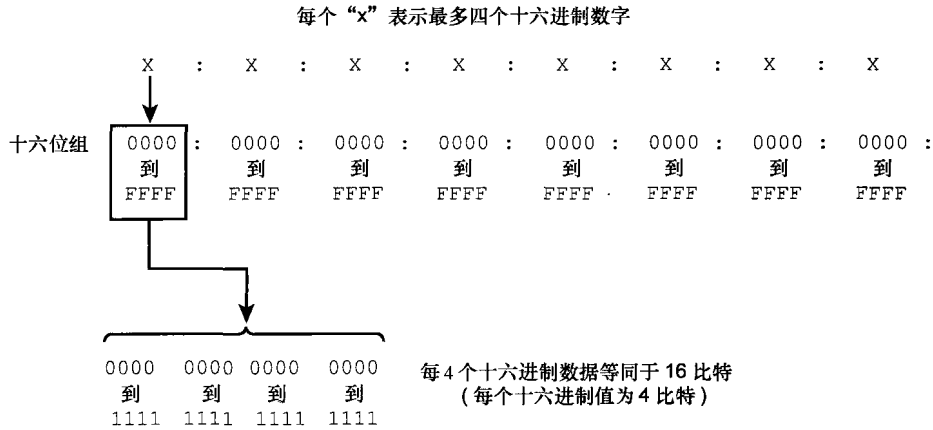



图 3-1 IPv6 地址优选表述形式

推荐的 IPv6 地址表述形式最长，使用了 32 个十六进制数值，并用冒号分隔每 4 个十六进制数字组成的段，而且每个十六进制数包括 4 个比特。

> 注：通常将 4 个十六进制数组成的段称为十六位组（hextet），类似于 IPv4 编址中的八位组（octet），因此，一个 IPv6 地址包括 8 个由冒号分隔的十六位组。如图 3-1 所示，每个由 4 位十六进制数组成的十六位组等于 16 比特。为了清楚起见，本书在指代 16 比特段时将始终使用术语十六位组。表 3-3 给出了使用优选格式表示的多个 IPv6 地址示例。请注意，最后两个地址是第 2 章中拓扑结构的 PC1 和 PC2 的 IPv6 地址。

表 3-3 使用优选格式的 IPv6 地址示例

| 使用优选格式的 IPv6 地址 |
| --- |
| 0000:0000:0000:0000:0000:0000:0000:0000 |
| 0000:0000:0000:0000:0000:0000:0000:0001 |
| FF02:0000:0000:0000:0000:0000:0000:0001 |
| FC00:0001:A000:0B00:0000:0527:0127:00AB |

续表

| 使用优选格式的 IPv6 地址 |
| --- |
| 2001:DCBA:1111:000A:00B0:0000:9000:0200 |
| 2001:0000:0000:0000:ABCD:0000:0000:1234 |
| 2001:0DB8:AAAA:0001:0000:0000:0000:0100 |
| 2001:0DB8:AAAA:0001:0000:0000:0000:0200 |

乍看起来，这些地址很复杂，但是大家完全不必担心，因为通过本章的学习之后，大家阅读和使用 IPv6 地址的信息会快速增加。除了优选的 IPv6 地址格式之外，RFC 2373 和 RFC 5952 还提供了两种有助于简化这些地址表达形式的规则。

### 3.2.1　规则 1：省略前导 0

所有十六位组（16 比特段）中的前导 0 都可以被省略。该规则仅适用于前导 0，而不适用于尾部 0，否则地址会出现歧义。以前面使用优选格式的 IPv6 地址为例，表 3-4 显示了如何省略这些地址中的前导 0。

**表 3-4　　省略十六位组中前导 0 的 IPv6 地址示例**

| 格式 | IPv6 地址 |
| --- | --- |
| 优选格式 | 0000:0000:0000:0000:0000:0000:0000:0000 |
| 省略前导 0 的格式 | 0:　0:　0:　0:　0:　0:　0:　0<br>或 0:0:0:0:0:0:0:0 |
| 优选格式 | 0000:0000:0000:0000:0000:0000:0000:0001 |
| 省略前导 0 的格式 | 0:　0:　0:　0:　0:　0:　0:　1<br>或<br>0:0:0:0:0:0:0:1 |
| 优选格式 | FF02:0000:0000:0000:0000:0000:0000:0001 |
| 省略前导 0 的格式 | FF02:0:　0:　0:　0:　0:　0:　1<br>或<br>FF02:0:0:0:0:0:0:1 |
| 优选格式 | FC00:0001:A000:0B00:0000:0527:0127:00AB |
| 省略前导 0 的格式 | FC00: 1:　A000: B00: 0:　527: 127:　AB<br>或<br>FC00:1:A000:B00:0:527:127:AB |
| 优选格式 | 2001:DCBA:1111:000A:00B0:0000:9000:0200 |
| 省略前导 0 的格式 | 2001: DB8:　1111: A:　B0:　0:　9000: 200<br>或<br>2001:DB8:1111:A:B0:0:9000:200 |

续表

| 格式 | IPv6 地址 |
| --- | --- |
| 优选格式 | 2001:**0000**:**0000**:**0000**:ABCD:**0000**:**0000**:1234 |
| 省略前导 0 的格式 | 2001: DB8: 0: 0: ABCD: 0: 0: 1234<br>或<br>2001:DB8:0:0:ABCD:0:0:1234 |
| 优选格式 | 2001:0DB8:AAAA:**0001**:**0000**:**0000**:**0000**:**0**100 |
| 省略前导 0 的格式 | 2001: DB8:AAAA: 1: 0: 0: 0: 100<br>或<br>2001:DB8:AAAA:1:0:0:0:100 |
| 优选格式 | 2001:0DB8:AAAA:**0001**:**0000**:**0000**:**0000**:**0**200 |
| 省略前导 0 的格式 | 2001: DB8:AAAA: 1: 0: 0: 0: 200<br>或<br>2001:DB8:AAAA:1:0:0:0:200 |

注：表 3-4 中所有被省略的 0 都被标注成黑体，地址中保留的空格是为了更好地形象化表示哪些 0 被省略了。

请注意，只能省略前导 0，否则会导致 IPv6 地址出现歧义。例如，如果允许省略尾部 0，那么就无法知道正确的地址。由于只有一种正确的理解方式，因此只能省略前导 0：

| 省略 0 后的 IPv6 地址 | 2001 :1944 :100 :A :0 :BC :ABCD :D0B |
| --- | --- |
| 错误（尾部 0） | 2001 :1944 :100 **0** :A **000** :0 **000** :BC**00** :ABCD :D0B **0** |
| 正确（前导 0） | 2001 :1944 : **0** 100 : **000** A : **000** 0 : **00** BC :ABCD : **0** D0B |

### 3.2.2 规则 2：省略全 0

双冒号（::）可以表示任何一个连续的由一个或多个全 0 组成的十六位组（16 比特段），这样就能进一步简化 IPv6 地址的大小，表 3-5 解释了双冒号的使用方式。

表 3-5 省略全 0 十六位组的连续字符串后的 IPv6 地址示例

| 格式 | IPv6 地址 |
| --- | --- |
| 优选格式 | **0000** :**0000** :**0000** :**0000** :**0000** :**0000** :**0000** :**0000** |
| (::) 全 0 段压缩的格式 | :: |
| 优选格式 | **0000**:**0000**:**0000**:**0000**:**0000**:**0000**:**0000**:**0001** |
| (::) 全 0 段压缩的格式 | ::0001 |
| 优选格式 | FF02:**0000**:**0000**:**0000**:**0000**:**0000**:**0000**:**0001** |
| (::) 全 0 段压缩的格式 | FF02::0001 |

续表

| 格式 | IPv6 地址 |
|---|---|
| 优选格式 | FC00:0001:A000:0B00:**0000**:0527:0127:00AB |
| (::) 全 0 段压缩的格式 | FC00:0001:A000:0B00::0527:0127:00AB |
| 优选格式 | 2001:DCBA:1111:000A:00B0:**0000**:9000:0200 |
| (::) 全 0 段压缩的格式 | 2001:DCBA:1111:000A:00B0::9000:0200 |
| 优选格式 | 2001:**0000:0000:0000**:ABCD:0000:0000:1234 |
| (::) 全 0 段压缩的格式 | 2001::ABCD:0000:0000:1234<br>注：该地址也可以表示为 2001:0000:0000:0000:ABCD::1234 |
| 优选格式 | 2001:0DB8:AAAA:0001:**0000:0000:0000**:0100 |
| (::) 全 0 段压缩的格式 | 2001:0DB8:AAAA:0001::0100 |
| 优选格式 | 2001:0DB8:AAAA:0001:**0000:0000:0000**:0200 |
| (::) 全 0 段压缩的格式 | 2001:0DB8:AAAA:0001::0200 |

注意：表 3-5 的优选格式地址中，所有被标记为粗体的 0 都被替换成::。

只有一个连续的全 0 段才能用双冒号进行替换，否则地址会出现歧义。

- 同时使用了两个双冒号的错误地址：

```
2001::ABCD::1234
```

- 可能的地址选项：

```
2001:0000:0000:0000:0000:ABCD:0000:1234
2001:0000:0000:0000:ABCD:0000:0000:1234
2001:0000:0000:ABCD:0000:0000:0000:1234
2001:0000:ABCD:0000:0000:0000:0000:1234
```

从上面的例子可以看出，如果使用了两个双冒号，那么就会存在多种可能的理解，也就无法知道确切的地址。

注意：RFC 5952 建议用双冒号替代最长的 0 字符串。

### 3.2.3　同时运用规则 1 和规则 2

同时运用规则 1 和规则 2 可以进一步简化 IPv6 地址的表达形式。表 3-16 解释了这三种地址格式。同样，为了更好地形象化表示被省略的 0，地址中保留了空格。

表 3-6　　同时运用规则 1 和规则 2

| 格式 | IPv6 地址 |
|---|---|
| 优选格式 | FF02:**0000:0000:0000:0000:0000:0000:000**1 |
| 省略前导 0 的格式 | FF02:0:　0:　0:　0:　0:　0:　1 |

续表

| 格式 | IPv6 地址 |
|---|---|
| (::) 全 0 段压缩的格式 | :: |
| 优选格式 | 0000:0000:0000:0000:0000:0000:0000:0001 |
| 省略前导 0 的格式 | 0: 0: 0: 0: 0: 0: 0: 1 |
| (::) 全 0 段压缩的格式 | ::0001 |
| 优选格式 | FF02:0000:0000:0000:0000:0000:0000:0001 |
| 省略前导 0 的格式 | FF02:0: 0: 0: 0: 0: 0: 1 |
| (::) 全 0 段压缩的格式 | FF02::0001 |
| 优选格式 | FC00:0001:A000:0B00:0000:0527:0127:00AB |
| 省略前导 0 的格式 | FC00: 1: A000: B00: 0: 527: 127: AB |
| (::) 全 0 段压缩的格式 | FC00:1:A000:B00::527:127:AB |
| 优选格式 | 2001:DCBA:1111:000A:00B0:0000:9000:0200 |
| 省略前导 0 的格式 | 2001: DB8: 1111: A: B0: 0: 9000: 200 |
| (::) 全 0 段压缩的格式 | 2001:DCBA:1111:A:B0::9000:200 |
| 优选格式 | 2001:0000:0000:0000:ABCD:0000:0000:1234 |
| 省略前导 0 的格式 | 2001: DB8: 0: 0: ABCD: 0: 0: 1234 |
| (::) 全 0 段压缩的格式 | 2001::ABCD:0:0:1234<br>注：该地址也可以写成 2001:0:0:0:ABCD::1234 |
| 优选格式 | 2001:0DB8:AAAA:0001:0000:0000:0000:0100 |
| 省略前导 0 的格式 | 2001: DB8:AAAA: 1: 0: 0: 0: 100 |
| (::) 全 0 段压缩的格式 | 2001:DB8:AAAA:1::100 |
| 优选格式 | 2001:0DB8:AAAA:0001:0000:0000:0000:0200 |
| 省略前导 0 的格式 | 2001: DB8:AAAA: 1: 0: 0: 0: 200 |
| (::) 全 0 段压缩的格式 | 2001:DB8:AAAA:1::200 |

表 3-7 显示了 IPv6 地址的优选格式以及同时运用两种规则后的压缩格式。

表 3-7 IPv6 地址的优选格式与压缩格式

| IPv6 地址的优选格式 | IPv6 地址的压缩格式 |
|---|---|
| 0000:0000:0000:0000:0000:0000:0000:0000 | :: |
| 0000:0000:0000:0000:0000:0000:0000:0001 | ::1 |
| FF02:0000:0000:0000:0000:0000:0000:0001 | FF02::1 |
| FC00:0001:A000:0B00:0000:0527:0127:00AB | FC00:1:A000:B00::527:127:AB |
| 2001:DCBA:1111:000A:00B0:0000:9000:0200 | 2001:DCBA:1111:A:B0::9000:200 |
| 2002:0000:0000:0000:ABCD:0000:0000:1234 | 2002::ABCD:0:0:1234 |
| 2001:0DB8:AAAA:0001:0000:0000:0000:0100 | 2001:DB8:AAAA:1::100 |
| 2001:0DB8:AAAA:0001:0000:0000:0000:0200 | 2001:DB8:AAAA:1::200 |

虽然运用这两个规则可以压缩 IPv6 地址的表述形式，但 IPv6 地址看起来似乎仍然略显笨拙。接下来将讨论“3-1-4”技术，这就有助于大家更好地认识 IPv6 地址。

## 3.3　前缀标记

对 IPv4 来说，IPv4 地址的前缀或网络部分是通过点分十进制网络掩码（通常称为子网掩码）来标识的。例如，255.255.255.0 表示该 IPv4 地址的网络部分或前缀长度是最左侧的 24 比特。

根据 RFC 4291“IP Version 6 Addressing Architecture”的定义，IPv6 地址前缀的表示方式类似于采用 CIDR（Classless InterDomain Routing）标记法的 IPv4 地址前缀。IPv6 地址前缀（地址的网络部分）的表示格式如下：

*ipv6-address/prefix-length*

其中，prefix-length（前缀长度）是一个十进制数值，表示该地址最左侧连续比特的数量。前缀长度用于确定该地址的前缀或网络部分。

下面以地址 2001:0DB8:AA AA:1111:0000:0000:0000:0000/64 为例予以说明，图 3-2 解释了前缀长度/64 是如何标识该地址的前缀或网络部分的。除去前缀长度/64 之后还剩余其他的 64 个比特，这些比特称为该 IPv6 地址的接口 ID（Interface ID）部分，也就是 IPv4 地址中的主机部分。有关接口 ID 的相关内容将在下一节进行讨论。

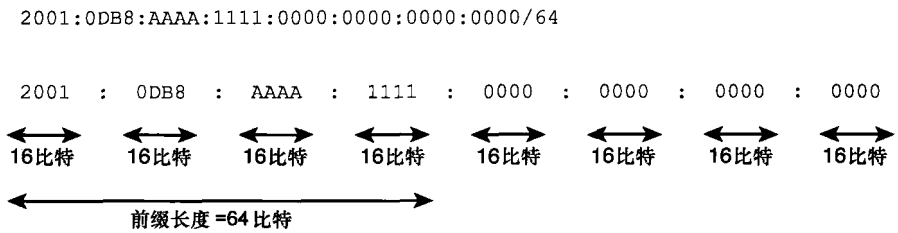



图 3-2　IPv6 前缀

利用前面讨论过的地址压缩规则，该 IPv6 地址的其他有效表示形式有：

- 2001:0DB8:AAAA:1111:0:0:0:0/64
- 2001:0DB8:AAAA:1111::/64
- 2001:DB8:AAAA:1111::/64

如图 3-3 所示，主机电脑等设备将拥有属于该前缀或网络地址的 IPv6 地址。以第 2 章的拓扑结构为例，两个有效的主机地址分别为：

- 2001:0DB8:AAAA:1111:0000:0000:0000:0100/64

或

2001:DB8:AAAA:1111::100/64

- 2001:0DB8:AAAA:1111:0000:0000:0000:0200/64

或

2001:DB8:AAAA:1111::0200/64

每一个十六进制值是 4 比特，一个十六位组是 16 比特。

```
2001:0DB8:AAAA:1111:0000:0000:0000:0100/64
```

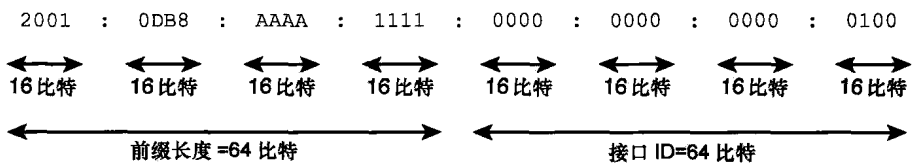



图 3-3 IPv6 前缀长度和接口 ID

与 IPv4 相似，对 IPv6 来说，网络中可以容纳的设备数量也取决于前缀长度。如第 1 章所述，随着互联网的不断增长，有限的 IPv4 地址空间被迅速耗尽，客户申请一个 IPv4 地址及前缀长度（子网掩码）以满足其网络需求时，必须提供相应的证明，因此大多数站点都极度依赖于 NAT 技术来满足其网络中内部 IPv4 主机的数量需求。

而 IPv6 则不存在这样的问题，这归功于 IPv6 充足的地址空间。绝大多数人都已经习惯于节约分配网络中的 IPv4 地址，通常都为 IPv4 网络中的点到点串行链路分配/30 地址，虽然 IPv6 时代无需如此，但这必定是一个难以克服的习惯。

IAB（Internet Architecture Board，互联网架构委员会）和 IESG（Internet Engineering Steering Group，互联网工程指导组）在 RFC 3177 “IAB/IESG Recommendations on IPv6 Address Allocations to Sites” 中发布了一系列 IPv6 地址分配建议，这是由 IAB/IESG 提供给 5 个 RIR 的 IPv6 地址分配建议，其中提到了以下重要信息：

“有关地址分配的技术原理需要在大量的节约实践与易于操作的明智性之间做出平衡。一方面，在管理潜在有限的资源时，大家必须从各个方面节约资源以便在预期的生命期内防止资源被耗尽；另一方面，IPv6 地址空间与 IPv4 地址空间完全不一样，完全不是有限的资源，而且妨碍市场的不必要的保守性做法都会成为其他抑制因素。因此从市场发展的角度来看，我们希望用户或 ISP 能够很容易地得到他们确实需要的足够多的 IPv6 地址，而无需立即重新编号或实施低效的地址扩展方案。”

在 RFC 3177 中，IESG 和 IAB 建议为不同规模的网络使用特定的前缀长度，其中的一条建议就是所有站点都应该获得一个/48 地址，包括家庭网络或者小型企业网，甚至是大型企业网。RIR 于 2002 年采纳了这一建议，但 2005 年开始重新斟酌该建议，直至 2011 年，RFC 6177 “IPv6 Address Assignments to End Sites” 废除了 RFC 3177 并提出以下意见：

“对一个可运作的社区来说，为端站点准确分配多大的地址空间确实是一个问题。在这种情况下，IETF 的角色应该专注于为 IPv6 的架构与运作提供指导……此外，这份文件需要澄清的一个问题是，为所有站点分配同样的/48 地址空间无法完全反映大量端

站点的实际情况，因此不再将/48 地址空间作为唯一的默认分配建议。”

也就是说，RIR 向其客户（如 ISP）分配地址空间时将取决于 RIR 各自的策略。ARIN（负责北美的 RIR）目前的策略是向 ISP 分配最小/32、最大/24 的 IPv6 地址空间，否则必须提供足够的证明。如果端站点能够提供足够的证明，那么就能获得最少/48 或更大的 IPv6 地址空间。由于/48 地址空间看起来仍然是端站点的常规分配标准，因此后面的这些例子也将使用该地址空间。RFC 6177 建议，家庭站点可能不需要/48 地址空间，/56 地址空间可能更合适，这就为 ISP 提供了更多可分配的地址。对家庭站点来说，唯一的区别就是每个家庭站点网络的子网数。

> 注：可以分配给端用户组织机构的两类地址是：PI（Provider-Independent，提供商独立）地址和 PA（Provider-Aggregatable，提供商可聚合）地址。PI 地址空间由 RIR 直接分配给端用户组织机构，PI 地址空间允许组织机构在无需获得新地址空间的情况下灵活更换服务提供商。PA 地址空间由 RIR 分配给 ISP，允许 ISP 聚合器地址空间以提高路由效率，这些地址属于 ISP。与 PI 地址不同，如果端用户更换服务提供商，那么 PA 地址则无法随端用户进行迁移。

如下节所述，一个典型的 IPv6 单播地址的主机部分是 64 比特（即接口 ID）。如果某个站点收到一个/48 前缀，那么就能允许 65 535 个子网，而且每个子网可以提供 18 446 744 073 709 551 616 个接口地址（主机）！为家庭站点分配一个/56 前缀，也就意味着每个子网也可以拥有如此多的主机数，只是子网数量减少至 256 个，而这对绝大多数家庭站点来说都是绰绰有余的。

> 注：一个主机可以有多个接口，每个接口可以有一个或多个 IPv6 地址。

## 3.4　IPv6 地址类型概述

本节将简要介绍基本的 IPv6 地址类型，详细内容将在第 4 章中讨论。IPv4 有单播地址、多播地址和广播地址，而 IPv6 则无广播地址，IPv6 的三种地址类型分别是：

- 单播（Unicast）地址；
- 任播（Anycast）地址；
- 多播（Multicast）地址。

### 3.4.1　单播地址

单播地址唯一地标识 IPv6 设备上的某个接口。发送给单播地址的数据包会被传送

给该地址所标识的接口。IPv6 地址可以更精确地标识主机上的接口，而不是主机本身。一个接口可以拥有多个 IPv6 地址和一个 IPv4 地址。

IPv6 中的单播地址类型很多，常见的主要有：

- 全局单播地址（Global unicast address）；
- 唯一本地单播地址（Unique local unicast address）（2004 年 9 月废除了站点本地地址）；
- 链路本地单播地址（Link-local unicast address）；
- 未指定地址（Unspecified address）；
- 环回地址（Loopback address）。

全局单播地址中还有一些特殊用途的地址，如内嵌 IPv4 地址的 IPv6 地址。有关全局单播地址的结构将在后面的章节进行介绍。

### 3.4.2 任播地址

任播地址是分配给多台设备的单播地址。发送给任播地址的数据包只传送给配置了该地址的其中一台设备。任播数据包会被路由到最近的设备。

与 IPv6 类似，IPv4 中也有任播地址，而且是分配给多台设备的单播地址。对 IPv4 和 IPv6 来说，从语法上很难区分任播地址与单播地址，IPv6 会对分配了任播地址的设备进行显式配置，以便能够识别任播地址，IPv4 则无需如此。

### 3.4.3 多播地址

多播地址用于标识一组接口，而且这些接口通常属于不同设备。发送给多播地址的数据包会被传送给该多播地址所标识的所有设备，多播组的所有成员都会处理该数据包。因而多播地址与任播地址之间的区别就是，任播数据包仅发送给一台设备，而多播数据包会发送给多台设备。

IPv6 中没有广播地址，取而代之的是全部节点多播地址（all-nodes multicast address）。

## 3.5 全局单播地址的结构

下面将介绍全局单播地址的基本结构，全局单播地址也被称为可聚合全局单播地址（aggregatable global unicast address），是 IPv6 互联网上全局可路由且可达的地址，相当于 IPv4 的公有地址。

图 3-4 显示了某典型站点的全局单播地址的结构。

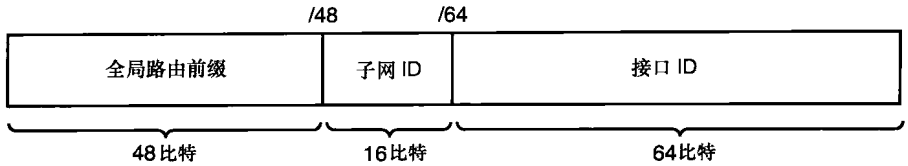



图 3-4　某典型站点的全局单播地址的结构

### 3.5.1　全局路由前缀

全局路由前缀是由提供商（如 ISP）向客户或站点分配的地址的前缀，尽管 IESF 和 IAB 不再推荐为不同规模的网络分配特定的前缀长度，但 RIR（如 ARIN）的常见策略仍然是为端站点分配 48 比特前缀（/48），图 3-4 显示的就是一个典型的/48 全局路由前缀。

注：RFC 4291 没有指定子网 ID（Subnet ID）的大小，图 3-4 中的 16 比特子网 ID 来源于获得/48 全局路由前缀的站点，由于接口 ID 为 64 比特，因而剩下的 16 比特就是子网 ID。有关详细信息请参考 RFC 3587“IPv6 Global Unicast Address Format”。

### 3.5.2　子网 ID

IPv4 地址与 IPv6 地址之间的最大区别就是地址子网部分的位置。IPv4 从地址的主机部分借位来创建子网，而 IPv6 地址中的子网 ID 是一个独立部分，并不属于地址的主机部分（即 IPv6 地址的接口 ID）。

如图 3-4 所示，IPv6 地址有 16 比特子网 ID，因此可以拥有 65 536 个子网。正如大家所疑惑的那样，IPv6 可以使用全 0 和全 1 的子网。有关子网的详细内容将在下一节进行讨论。

### 3.5.3　接口 ID

接口 ID 唯一标识了子网上的接口，如前所述，64 比特的接口 ID 允许每个子网都拥有 18 446 744 073 709 551 616 个地址。这里用到术语接口 ID（Interface ID）而不是主机 ID（Host ID）的原因是，一个主机可以有多个接口，每个接口可以有一个或多个 IPv6 地址。

IPv4 地址与 IPv6 地址之间的另一个重要区别是：全 0 与全 1 地址都是合法的 IPv6 接口地址。IPv6 接口 ID 可以包含全 0 和全 1 地址，而 IPv4 地址的主机部分全 0 则被

保留给网络地址或子网地址，IPv4 地址的主机部分全 1 则是广播地址。请注意，IPv6 中没有广播地址。

### 3.5.4 3-1-4 法则

IPv6 全局单播地址看起来比较复杂且难以识别地址的各个部分。下面就来介绍我创造的 3-1-4 法则，将有助于大家快速识别 IPv6 地址的各个部分（如图 3-5 所示）。这里的数字代表 IPv6 地址各部分所拥有的十六位组数量或 16 比特段。一个简单的记忆方式就是可以将该法则称为 pi 法则（pi=3.14）。

- 3：表示全局路由前缀是 3 个十六位组，即 48 比特。
- 1：表示子网 ID 是 1 个十六位组，即 16 比特。
- 4：表示接口 ID 是 4 个十六位组，即 64 比特。

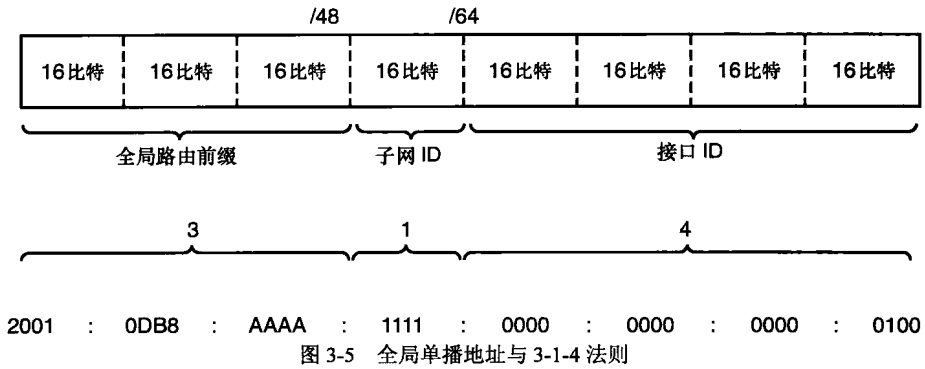



图 3-5 全局单播地址与 3-1-4 法则

注：该法则对于/48 全局路由前缀和 64 比特接口 ID 来说总是非常有用的，这也是目前通用的前缀分配方式。需要注意的是，本书后面还会谈到，全局路由前缀和接口 ID 并不必须分别是 48 比特和 64 比特。

表 3-8 给出了使用 3-1-4 法则的多个/48 全局单播地址示例。虽然双冒号可以压缩 IPv6 地址的表述形式，但有时也会给识别 IPv6 地址三个部分带来困难。有时先从接口 ID 开始识别 IPv6 地址各个部分会更容易些，有时也可以从两头向中间的子网 ID 来定位各个部分。

表 3-8 使用 3-1-4 法则的/48 全局单播地址

| /48 全局单播地址 | 全局路由前缀 3 | 子网 ID 1 | 接口 ID 4 |
|---|---|---|---|
| 2001:0DB8:AAAA:1234:1111:2222:3333:4444 | 2001:0DB8:AAAA | 1234 | 1111:2222:3333:4444 |
| 2001:0DB8:BBBB:4321:AAAA:BBBB:CCCC:DDDD | 2001:0DB8:BBBB | 4321 | AAAA:BBBB:CCCC:DDDD |

续表

| /48 全局单播地址 | 全局路由前缀　3 | 子网 ID　1 | 接口 ID　4 |
|---|---|---|---|
| 2001:0DB8:AAAA:0001:0000:0000:0000:0100 | 2001:0DB8:AAAA | 0001 | 0000:0000:0000:0100 |
| 2001:0DB8:AAAA:9:0:0:0:A | 2001:0DB8:AAAA | 0009 | 0000:0000:0000:000A |
| 2001:0DB8:AAAA:0001::0200 | 2001:0DB8:AAAA | 0001 | 0000:0000:0000:0200 |
| 2001:DB8:AAAA::200 | 2001:0DB8:AAAA | 0000 | 0000:0000:0000:0200 |
| 2001:DB8::ABC:0 | 2001:0DB8:0000 | 0000 | 0000:0000:0ABC:0000 |
| 2001:DB8:ABC:: | 2001:0DB8:0ABC | 0000 | 0000:0000:0000:0000 |
| 2001:DB8:ABC::FFFF:FFFF:FFFF:FFFF | 2001:0DB8:0ABC | 0000 | FFFF:FFFF:FFFF:FFFF |
| 2001:DB8::FFFF:FFFF:FFFF:FFFF:FFFF | 2001:0DB8:0000 | FFFF | FFFF:FFFF:FFFF:FFFF |

请注意，对 IPv6 来说，以下两个地址都是合法的接口（主机）地址。

- **全 0 地址**：2001:DB8:ABC:: or 2001:0DB8:0ABC:0000:0000:0000:0000:0000；
- **全 1 地址**：2001:DB8::FFFF:FFFF:FFFF:FFFF:FFFF or 2001:DB8:0000:FFFF:FFFF:FFFF:FFFF:FFFF

## 3.6　合在一起

现在大家对 IPv6 全局单播地址应该有了一个基本概念了，图 3-6 给出了一个 IPv6 网络示例。

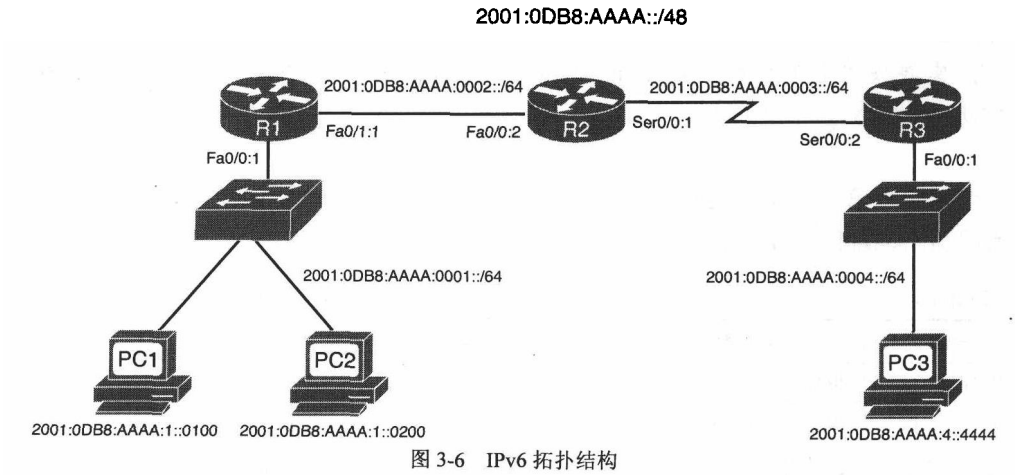



图 3-6　IPv6 拓扑结构

首先来看地址 2001:0DB8:AAAA::/48。前 3 个十六位组标识全局路由前缀，也就是从提供商获得的 IPv6 网络地址。/48 网络被分成 4 个/64 子网——0001、0002、0003 和 0004。这 4 个子网分别是：

- 2001:0DB8:AAAA:0001::/64
- 2001:0DB8:AAAA:0002::/64
- 2001:0DB8:AAAA:0003::/64
- 2001:0DB8:AAAA:0004::/64

利用 3-1-4 法则，可以快速发现前 3 个十六位组是全局路由前缀（2001:0DB8:AAAA），第四个十六位组是子网 ID。

注：2001:0DB8:AAAA::/48 是地址块 2001:0DB8::/32 的一部分，该地址块是专门为示例和文档保留的地址块。

为路由器接口配置 IPv6 地址与 IPv4 极其相似。如本书各种案例所示，大多数命令都是相同的，只是将 **ip** 替换成 **ipv6** 而已。

表 3-9 列出了为路由器接口手工配置 IPv6 所需的命令。第 4 章解释了命令 **ipv6 address** 的一些可选参数。

**表 3-9　　命令 ipv6 address**

| 命令 | 描述 |
|---|---|
| Router(config)# **interface** *interface-type interface-number* | 指定接口类型和接口号 |
| Router(config-if)# **ipv6 address** *ipv6-address/prefix-length* | 指定分配给接口的 IPv6 地址和前缀长度。为了删除接口的 IPv6 地址，需要使用该命令的 **no** 形式 |

注：有关 IPv6 地址配置的内容将在第 4 章进行讨论。这里仅列出了接口命令 **ipv6 address** 以说明 IPv6 命令与 IPv4 命令之间的相似性。为了让路由器能够路由 IPv6 数据包，需要使用全局配置命令 **ipv6 unicast-routing**。有关该命令的相关内容也将在第 4 章进行讨论。

现在配置第一个子网的设备。使用 3-1-4 法则与图 3-6 所示的拓扑结构，可以很容易地识别出地址的 3 个部分（如表 3-10 所示）。

**表 3-10　　子网 2001:0DB8:AAAA:0001::/64 的 IPv6 地址**

| 设备 | 全局路由前缀　3 | 子网 ID　1 | 接口 ID　4 |
|---|---|---|---|
| 路由器 R1 | 2001:0DB8:AAAA | 0001 | 0000:0000:0000:0001 |
| PC1 | 2001:0DB8:AAAA | 0001 | 0000:0000:0000:0100 |
| PC2 | 2001:0DB8:AAAA | 0001 | 0000:0000:0000:0200 |

例 3-1 给出了以 IPv6 地址 2001:0DB8:AAAA:0001::0100 和前缀长度/64 配置路由器 R1 的快速以太网接口 0/0 的配置示例。

**例 3-1　Cisco 路由器 IPv6 接口配置**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 address 2001:0db8:aaaa:0001::1/64
R1(config-if)# no shutdown
R1(config-if)# end
R1#
```

注：在 Cisco IOS 中配置 IPv6 地址时，IPv6 地址与前缀长度之间没有空格。

图 3-7 显示了 PC1 的 IPv6 配置示例。路由器 R1 的 IPv6 地址被用作 PC 的默认网关，大家可以利用命令 **ipconfig** 在两台 Windows PC 上验证该配置。在这个案例中，IPv6 地址是手工配置的，不过对绝大多数终端设备来说，最可能的方式是利用无状态地址自动配置（Stateless Address Autoconfiguration）或 DHCPv6 服务器获取其 IPv6 地址。有关这些地址分配技术的内容，将在第 4 章进行详细讨论。

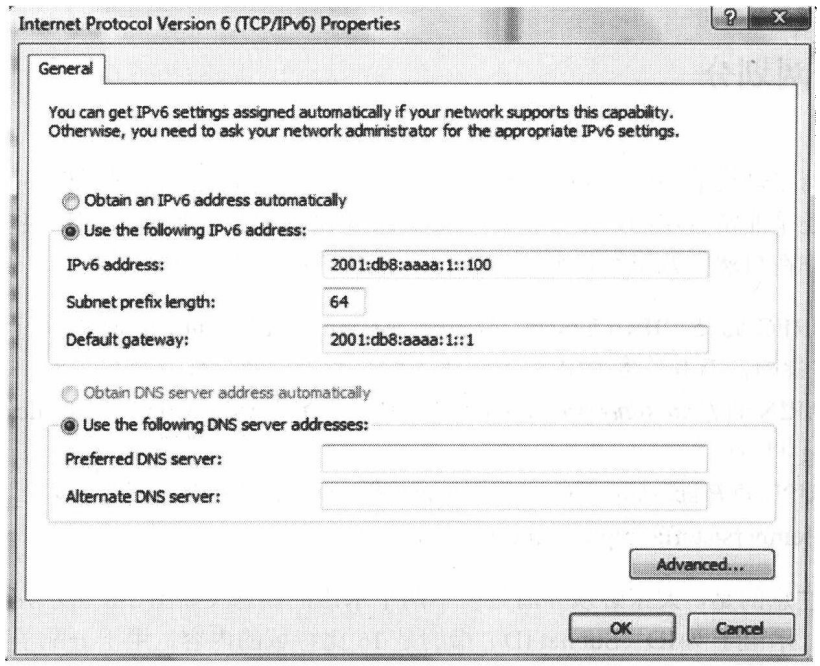



图 3-7　PC1 的 IPv6 接口配置

例 3-2 通过向 PC1 和 PC2 执行 ping 操作即可验证 IPv6 通信，请注意。这里的 **ping** 命令与 IPv4 相似，唯一的区别就是目的地址为 IPv6 地址。**ping** 命令的作用是向 IPv6 目的地址发送 ICMPv6 回显请求（ICMPv6 Echo Request）消息，“!”表明已经收到了从目的接口发出的 ICMPv6 回显应答（ICMPv6 Echo Reply）消息，从而验证了端到端

通信正常。有关 ICMPv6 的详细信息将在第 5 章进行讨论。

**例 3-2 路由器 R1 向 PC1 和 PC2 执行 ping 操作**

```
R1# ping 2001:db8:aaaa:0001::0100

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:AAAA:1::100, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
R1# ping 2001:db8:aaaa:0001::0200

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:AAAA:1::200, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
R1#
```

## 3.7 子网划分

根据网络规模的不同，制定 IPv6 编址方案需要周密规划。不过，IPv6 地址的基本子网划分还是非常直观的，多数情况下都要比 IPv4 地址的子网划分简单很多。对 IPv4 来说，除非在自然的八位组边界划分子网，否则子网总是有些模糊不清。

注：RFC 5375 “IPv6 Unicast Address Assignment Considerations” 为子网前缀提供了操作指南，而且许多 RIR 也都提供了相应指南以协助客户制定 IPv6 编址方案。

- ARIN 的 *IPv6 Addressing Plans (IPv6 编址方案)*: www.getipv6.info/index.php/IPv6_Addressing_Plans
- RIPE 的 *Preparing an IPv6 Addressing Plan (准备制定 IPv6 编址方案)*: https://labs.ripe.net/Members/steffann/preparing-an-ipv6-addressing-plan

需要注意的是，大家必须弄清楚以下两个术语的概念（如图 3-8 所示）：子网 ID 和子网前缀。术语子网 ID（Subnet ID）指的是 16 比特域的内容，用于分配子网。而术语子网前缀（Subnet Prefix）则指的是全局路由前缀与子网 ID 的编址比特。

一个典型的 IPv6 站点通常会拥有提供商（如 ISP）分配的/48 前缀，从而会创建 16 比特子网 ID，能够创建 216 或 65 536 个子网。全 0 和全 1 子网都是有效子网，剩余的 64 比特都是接口 ID，每个子网都能容纳 264 或 18 quintillion 个接口（主机）。相应的站点前缀与子网划分都会在第 4 章进行讨论。

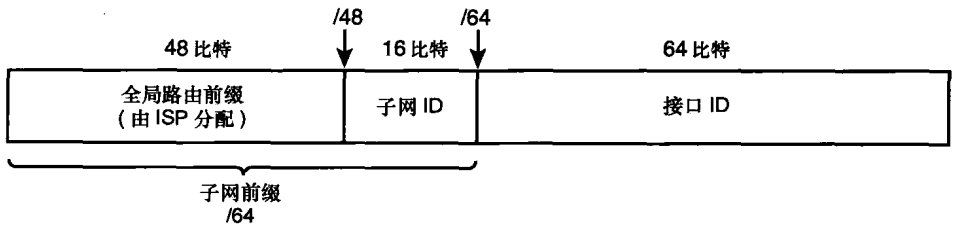



图 3-8　子网前缀

以图 3-6 所示拓扑结构为例，/48 为例被划分成 4 个/64 子网，具体情况请见表 3-11。

表 3-11　　2001:0DB8:AAAA::/48 的子网划分情况

| 子网前缀 | |
|---|---|
| 全局路由前缀 | 子网 ID |
| 2001:0DB8:AAAA: | 0001 |
| 2001:0DB8:AAAA: | 0002 |
| 2001:0DB8:AAAA: | 0003 |
| 2001:0DB8:AAAA: | 0004 |

这 4 个子网的缩写形式如下：

- 2001:DB8:AAAA:1::/64
- 2001:DB8:AAAA:2::/64
- 2001:DB8:AAAA:3::/64
- 2001:DB8:AAAA:4::/64

对 16 比特子网 ID 来说，取值范围是 0000～FFFF，可以提供 65 536 个子网。由于可以从 0000 开始按照递增 1 的方式来划分子网，因而子网划分很简单。请注意，这里所说的都是十六进制，因而 0009 之后的下一个子网 ID 应该是 000A。所以说，利用 16 比特子网 ID 进行子网划分是一件非常简单的事情（如表 3-12 所示）。

表 3-12　　使用 16 比特子网 ID 划分子网

| 范围 | 子网 ID |
|---|---|
| 初始的 16 个子网 | 0001 |
| | 0002 |
| | 0003 |
| | …… |
| | 0009 |
| | 000A |
| | 000B |
| | 000C |
| | 000D |
| | 000E |
| | 000F |

续表

| 范围 | 子网 ID |
|---|---|
| 接下来的 16 个子网 | 0010 |
| | 0011 |
| | 0012 |
| | ..... |
| | 001F |
| 接下来的 16 个子网 | 0020 |
| | 0021 |
| | 0022 |
| | ..... |
| | 002F |
| 依此类推 | 0030 |
| | 0031 |
| | ..... |

### 3.7.1 扩展子网前缀

IPv6 的子网划分并不局限于 16 比特子网 ID，可以选择任意数量的子网比特作为子网 ID。就像 IPv4 一样，如果希望增加子网数量或者减少每个子网中的主机数量，就必须从接口 ID 借位。需要引起重视的是，这种做法应该仅用于网络基础设施中的链路，任何包含端系统的网段都应该保持/64 前缀，因为/64 前缀长度是支持无状态地址自动分配的前提。

如图 3-9 所示，可以使用/112 前缀长度，将原先的/48 前缀扩展 64 比特（4 个十六位组），使得该地址前缀为/112。

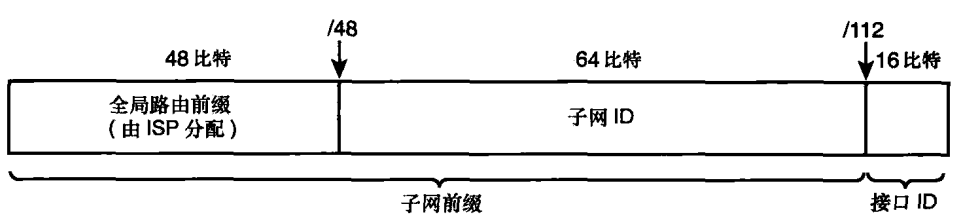



图 3-9 /112 子网前缀

前 4 个子网分别为：

- 2001:0DB8:AAAA:0000:0000:0000:0000::/112
- 2001:0DB8:AAAA:0000:0000:0000:0001::/112
- 2001:0DB8:AAAA:0000:0000:0000:0002::/112

■　2001:0DB8:AAAA:0000:0000:0000:0003::/112

图 3-10 显示了使用/112 前缀长度时的子网前缀范围。

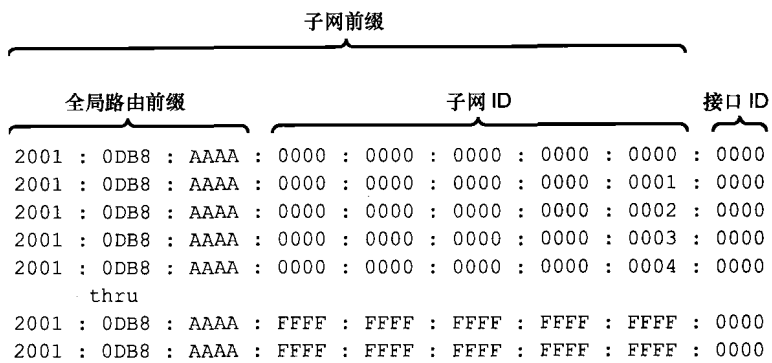



图 3-10　/112 子网前缀范围

注：SURFnet（一家旨在为荷兰信息技术领域的高校与研究机构建立合作关系的非盈利性组织）曾经与其客户召开了一次 IPv6 研讨会，并在该研讨会上制定了 IPv6 编址方案。可以从区域互联网注册机构 RIPE 网站下载相应的文档 *Preparing an IPv6 Addressing Plan*: https://labs.ripe.net/Members/steffann/preparing-an-ipv6- addressing-plan

即使扩展了子网 ID，但是只要子网边界位于半字节（nibble）边界，那么 IPv6 的子网划分依然非常直观。

### 3.7.2　在半字节边界划分子网

如果需要扩展子网 ID（也就是说从接口 ID 借位），那么最佳做法就是在半字节边界划分子网。半字节是 4 比特（如表 3-13 所示）。

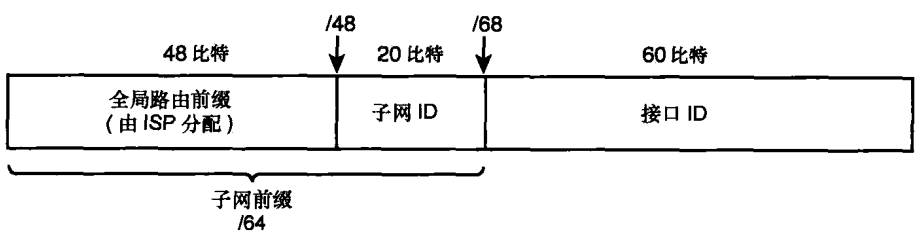
表 3-13　　十进制、十六进制和二进制

| 十进制 | 十六进制 | 二进制（半字节） |
|---|---|---|
| 0 | 0 | 0000 |
| 1 | 1 | 0001 |
| 2 | 2 | 0010 |
| 3 | 3 | 0011 |
| 4 | 4 | 0100 |
| 5 | 5 | 0101 |
| 6 | 6 | 0110 |
| 7 | 7 | 0111 |

续表

| 十进制 | 十六进制 | 二进制（半字节） |
|---|---|---|
| 8 | 8 | 1000 |
| 9 | 9 | 1001 |
| 10 | A | 1010 |
| 11 | B | 1011 |
| 12 | C | 1100 |
| 13 | D | 1101 |
| 14 | E | 1110 |
| 15 | F | 1111 |

如图 3-11 所示，将子网前缀/64 扩展了 4 比特（即半字节），达到/68，从而将子网 ID 从 16 比特增加到 20 比特，从而可以增加更多的子网并减小接口 ID 的大小。对于本案例来说，这么做并不是出于任何实际需要，而仅仅是为了解释扩展子网前缀的概念。通过将子网前缀扩展 4 比特（即半字节），就可以实现在半字节边界划分子网的最佳做法了。由于 20 比特是 4 比特的整数倍，因而可以很容易地列出全部子网（如图 3-11 所示）。




在半字节（4 比特）边界划分子网能够很容易地列出全部子网。

```
2001:0DB8:AAAA:0000:0000::/68
2001:0DB8:AAAA:0000:1000::/68
2001:0DB8:AAAA:1111:2000::/68
      thru
2001:0DB8:AAAA:FFFF:F000::/68
```

图 3-11　在半字节边界划分子网

### 3.7.3　在半字节内划分子网

对绝大多数客户网络来说，都不推荐在半字节内划分子网，因为这么做不但没什么好处，而且还会增加子网划分的实施难度与故障排查的难度。不过在某些场合下，如果在半字节边界划分子网会造成地址浪费，此时就需要在 4 比特的半字节内划分子网。前

面讨论过的编址方案可能在某些方面说明这些问题。

在半字节内划分子网时，事情就会变得非常复杂。如图 3-12 所示，图中使用的是/70 子网前缀，将非常简单的/68 扩展到非常复杂的/70。由于仅扩展了 2 比特而非半字节（4 比特），因此转换工作相当棘手。不过这种做法确实是合法的，只是处理起来很复杂而已。

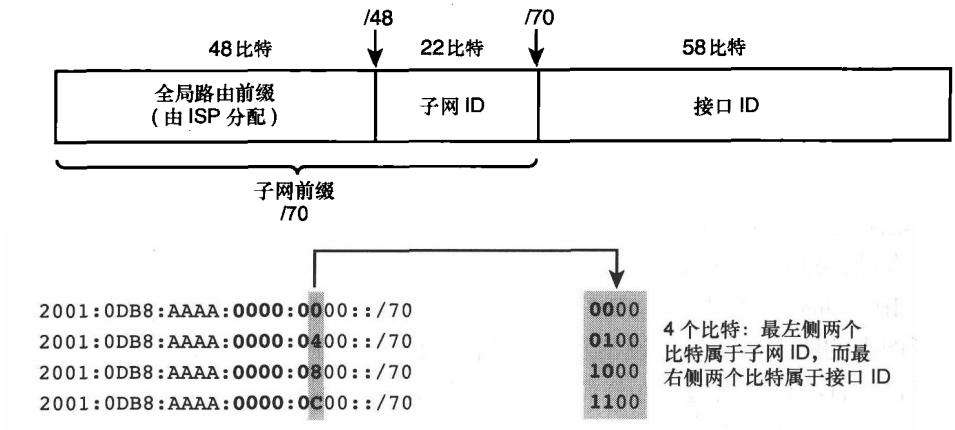



图 3-12　在半字节内划分子网

前 4 个子网分别为：

- 2001:0DB8:AAAA: 0000:00 00::/70
- 2001:0DB8:AAAA: 0000:04 00::/70
- 2001:0DB8:AAAA: 0000:08 00::/70
- 2001:0DB8:AAAA: 0000:0C 00::/70

第一个子网很容易计算，但第二个子网就需要考虑一下了。IPv6 地址使用十六进制数值来表示每 4 个比特。由于选择的是/70 子网前缀，因此最后一个十六进制数值的前半部分属于子网 ID，而另一半则属于接口 ID，因此仅修改最后一个数的前 2 个比特（如表 3-14 所示）。

**表 3-14　　在半字节内划分子网**

| 在半字节内划分子网 | 子网 ID 的最后一位数<br>二进制到十六进制 |
|---|---|
| 2001:0DB8:AAAA: 0000:**00** 00::/70 | 0000 = 0 |
| 2001:0DB8:AAAA: 0000:**04** 00::/70 | 0100 = 4 |
| 2001:0DB8:AAAA: 0000:**08** 00::/70 | 1000 = 8 |
| 2001:0DB8:AAAA: 0000:**0C** 00::/70 | 1100 = C |

### 3.7.4 限制接口 ID 空间

虽然 IPv6 地址空间足够大，但是仍然有必要限制网络中的接口 ID 大小。关于这个问题的争论很多，本节将简要分析对于该问题的理解。

RFC 6164“Using 127-Bit IPv6 Prefixes on Inter-Router Links”出于安全等原因，推荐在路由器之间的点到点链路上使用 127 比特的 IPv6 前缀（/127）。这与 IPv4 使用 31 比特前缀（/31）相似。一个值得关注的争论焦点就是 NDP（Neighbor Discovery Protocol，邻居发现协议）耗尽攻击。

64 比特接口 ID 可以提供非常多的接口地址，每个子网的接口（主机）数可以达到 18 quintillion。IPv4 主机利用 ARP（地址解析协议）缓存来维护 IPv4 地址与相关的二层 MAC 地址之间的关系列表，而 IPv6 则使用类似的邻居缓存（Neighbor Cache）。有关邻居缓存的详细内容将第 5 章进行讨论。

由于 IPv6 拥有巨大的接口 ID 空间，因而攻击者可以通过成千上万个伪造的源 IPv6 地址向子网上的路由器或其他设备发送连续的分组流，导致接收端需要为每个地址创建邻居缓存，从而消耗了大量内存资源。根据所发送的分组类型，接收端可能会响应大量的邻居请求（Neighbor Solicitation）分组，但却接收不到任何回应，从而消耗了大量内存与处理资源（邻居请求消息属于 NDP 的相关内容，也将在第 5 中进行讨论），这就等同于 IPv4 中充塞设备的 ARP 缓存。

RFC 3756“IPv6 Neighbor Discovery (ND) Trust Models and Threats”提出了一些解决 NDP 耗尽攻击的防范技术，如 SeND（Secure Neighbor Discover，安全邻居发现），但该问题已经超出了本书写作范围，有关信息可以参考 Jeff S Wheeler 的 *“IPv6 NDP Table Exhaustion Attack”*，可以从 http://inconcepts.biz/~jsw/IPv6_NDP_Exhaustion.pdf 下载。

在实际应用中也有理由使用比 64 比特更大的接口 ID。如第 4 章所述，对应用于网段内终端系统的 SLAAC（Stateless Address Autoconfiguration，无状态地址自动配置）来说，就需要/64 前缀和 64 比特接口 ID。

## 3.8 本章小结

本章解释了 IPv6 编址的基本概念。128 比特的 IPv6 地址，其优选格式是 8 个采用冒号分隔的 16 比特段（十六位组）。通过省略前导 0 并利用双冒号代替连续的全 0 十六位组，可以极大地简化 IPv6 地址的表述形式。

然后讨论了 IPv6 地址的前缀长度以及不同组织机构所推荐的全局路由前缀的大

小，并简要介绍了 IPv6 地址的类型：单播地址、任播地址和多播地址。

全局单播地址的基本结构包括三部分：全局路由前缀、子网 ID 和接口 ID。利用 3-1-4 法则可以帮助大家快速识别/48 全局单播地址中的各个十六位组。

最后，本章还解释了利用子网 ID 或者在半字节边界扩展的子网 ID 进行 IPv6 地址子网划分的简便性。虽然可以在半字节内划分子网，但实施起来比较困难，所以不建议大家这么做。

## 3.9 参考文献

RFC：

RFC 2374, An IPv6 Aggregatable Global Unicast Address Format , R. Hinden,Nokia, IETF, www.ietf.org/rfc/rfc2374.txt , July 1998

RFC 2460, Internet Protocol, Version 6 (IPv6) Specification , S. Deering, Cisco Systems, IETF, www.ietf.org/rfc/rfc2460.txt , December 1998

RFC 3177, IAB/IESG Recommendations on IPv6 Addresses , IAB, IESG, www.ietf.org/rfc/rfc3177.txt , September 2001

RFC 3587, IPv6 Global Unicast Address Format , R. Hinden, Nokia, IETF, www.ietf.org/rfc/rfc3587.txt , August 2003

RFC 3756, IPv6 Neighbor Discovery (ND) Trust Models and Threats , P. Nikander, Ericsson Research Nomadic Lab, IETF, www.ietf.org/rfc/rfc3756.txt , May 2004

RFC 4291, IP Version 6 Addressing Architecture , R. Hinden, Nokia, IETF, www.ietf.org/rfc/rfc4291. txt , February 2006

RFC 5453, Reserved IPv6 Interface Identifiers , S. Krishnan, www.ietf.org/rfc/rfc5453.txt , February 2009

RFC 6164, Using 127-Bit IPv6 Prefixes on Inter-Router Links , M. Kohno, Juniper Networks, www.ietf.org/rfc/rfc6164.txt , April 2011

RFC 6177, IPv6 Address Assignment to End Sites , IAB, IESG, www.ietf.org/rfc/rfc6167.txt , March 2011

网站：

IPv6 NDP Table Exhaustion Attack, Jeff S Wheeler, http://inconcepts.biz/ ~ jsw/IPv6_NDP_Exhaustion.pdf

AfriNIC IPv6 Resource website: www2.afrinic.net/IPv6/index.htm

ARIN IPv6 Resource website: http://ipv6.com/articles/general/ipv6-the-nextgeneration-internet.htm
APNIC IPv6 Resource website: http://icons.apnic.net/display/IPv6/home
LACNIC IPv6 Resource website: http://portalipv6.lacnic.net
RIPE NCC IPv6 Resource website: www.ripe.net/internet-coordination/ipv6

# 第 4 章　IPv6 地址类型

本章将详细讨论第 3 章中曾经提到过的三种 IPv6 地址：

- 单播（Unicast）地址；
- 任播（Anycast）地址；
- 多播（Multicast）地址。

图 4-1 以图表方式给出了这三种地址类型的详细信息。本章将首先讨论单播地址。大家不要被种类繁多的单播地址所吓倒，因为最重要的地址就是全局单播地址（已经在第 3 章介绍过），它们类似于 IPv4 的公有地址，所以本章的很多内容都是围绕着如何利用不同方式配置全局单播地址而展开的。

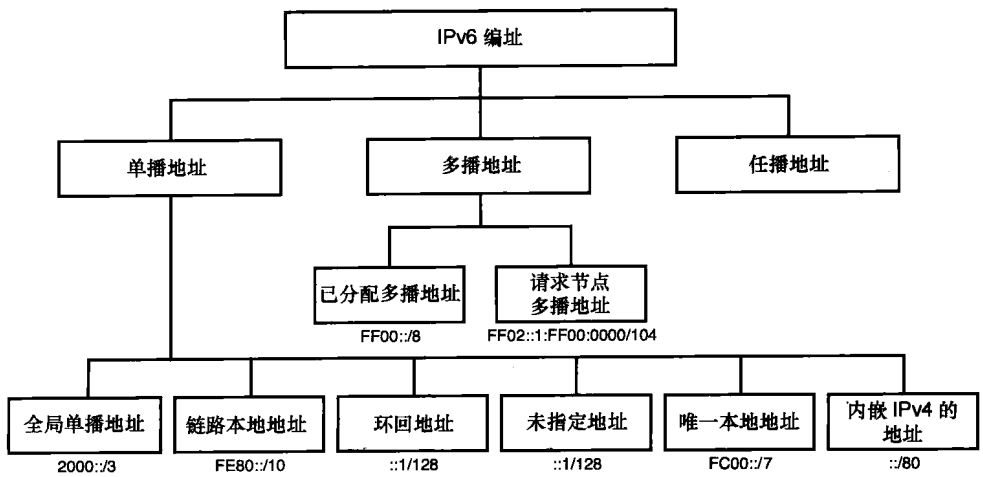



图 4-1　IPv6 地址类型

另一种单播地址是链路本地地址。该类单播地址被限制于单一链路。IPv6 路由协议使用链路本地地址来交换更新消息以及其他路由消息。虽然本章也会讨论其他单播地

址——环回地址、未指定地址、唯一本地地址和内嵌 IPv4 的地址，不过由于这些地址不像全局单播地址有那么多选项，因而无需过多解释。有关内嵌 IPv4 的单播地址的详细内容将在第 10 章进行讨论。

接下来本章将讨论多播地址，包括已分配的多播地址和请求节点多播地址。第 5 章将讨论协议（ND 或 NDP）对多播地址的使用情况。路由协议在很大程度上依赖于已分配的多播地址，有关详细信息将在第 8 章中进行讨论。

最后，本章将简要介绍任播地址。除了分配方式，任播地址与单播地址很难区分。单播地址分配给单个接口，而任播地址则分配给多台设备的多个接口。

## 4.1 IPv6 地址空间

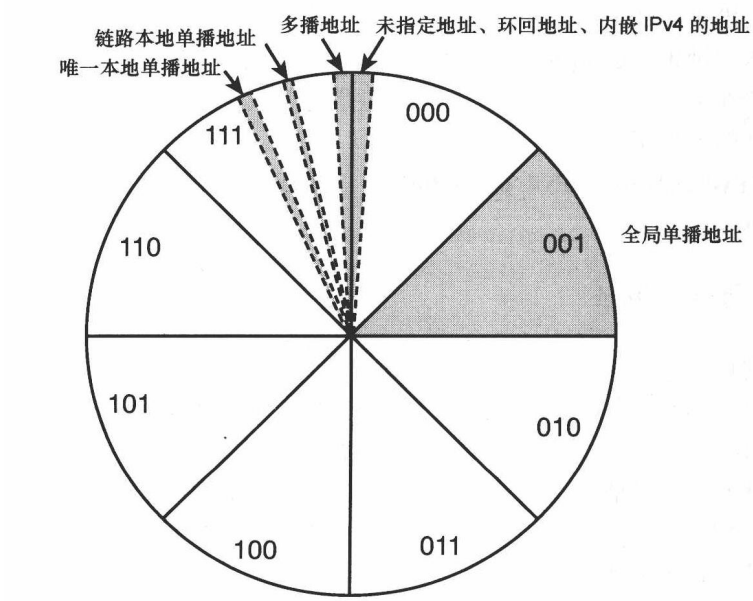
表 4-1 列出了 IANA 已分配的 IPv6 地址空间，并用黑体标识了已分配的全局单播地址、唯一本地单播地址、链路本地单播地址以及多播地址。本章将在后面逐一讨论这些地址。图 4-2 则以图表方式给出了 IPv6 地址空间的构成情况。

表 4-1　　IANA 已分配的 IPv6 地址空间

| 前导比特 | 地址 | 第一个十六位组的范围 | 分配情况 | 地址空间比例 |
|---|---|---|---|---|
| 000x | | | | 1/8 |
| **0000 0000** | **0000::/8** | **0000<br>00FF** | **未指定地址<br>环回地址<br>内嵌 IPv4 的地址** | **1/256** |
| 0000 0001 | 0100::/8 | 0100<br>01FF | IETF 保留 | 1/256 |
| 0000 001x | 0200::/7 | 0200<br>03FF | IETF 保留 | 1/128 |
| 0000 010x | 0400::/6 | 0400<br>05FF | IETF 保留 | 1/64 |
| 0000 1xxx | 0800::/5 | 0800<br>0FFF | IETF 保留 | 1/32 |
| **001x** | **2000::/3** | **2000<br>3FFF** | **全局单播地址** | **1/8** |
| 010x | 4000::/3 | 4000<br>5FFF | IETF 保留 | 1/8 |
| 011x | 6000::/3 | 6000<br>7FFF | IETF 保留 | 1/8 |
| 100x | 8000::/3 | 8000<br>9FFF | IETF 保留 | 1/8 |
| 101x | A000::/3 | A000<br>BFFF | IETF 保留 | 1/8 |

续表

| 前导比特 | 地址 | 第一个十六位组的范围 | 分配情况 | 地址空间比例 |
|---|---|---|---|---|
| 110x | C000::/3 | C000<br>DFFF | IETF 保留 | 1/8 |
| 111x | | | | 1/8 |
| 1110 xxxx | E000::/4 | E000<br>EFFF | IETF 保留 | 1/16 |
| 1111 0xxx | F000::/5 | F000<br>F7FF | IETF 保留 | 1/32 |
| 1111 10xx | F800::/6 | F800<br>FBFF | IETF 保留 | 1/64 |
| **1111 110x** | **FC00::/7** | **FC00<br>FDFF** | **唯一本地单播地址** | **1/128** |
| 1111 1110 0 | FE00::/9 | FE00<br>FE74 | IETF 保留 | 1/512 |
| **1111 1110 10** | **FE80::/10** | **FE80<br>FEBF** | **链路本地单播地址** | **1/1024** |
| 1111 1110 11 | FEC0::/10 | FEC0<br>FEFF | IETF 保留，以前为站点本地地址（已废除） | 1/1024 |
| **1111 1111** | **FF00::/8** | **FF00<br>FFFF** | **多播** | **1/256** |




其余的 IPv6 地址空间被 IETF 预留给将来使用

图 4-2　IANA 以分的 IPv6 地址空间（以 1/8 段为单位）

请注意，表 4-1 的“第一个十六位组的范围”列并没有显示全部地址空间范围。例如，全局单播地址空间的范围应该是 2000::～3FFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF。

在表 4-1 和图 4-2 中，IPv6 地址空间按照 3 个前导比特（000、001、010、011、100、101、110、111）划分为 8 段。目前看起来这些信息可能会让大家感到困惑，不过学习完本章知识之后，大家就能彻底掌握了。

## 4.2 单播地址

单播地址能够唯一地标识 IPv6 设备上的接口。发送到单播地址的数据包会被分配了该地址的接口接收到。与 IPv4 相似，源 IPv6 地址必须是单播地址。

本节将讨论各种不同类型的单播地址（如图 4-1 所示）。在此之前，首先简要介绍每种类型的单播地址。

- **全局单播地址（Global unicast）**：是 IPv6 互联网上可路由的地址，与 IPv4 公有地址相似。
- **链路本地地址（Link-local）**：仅被用来与同一本地链路上的设备进行通信。
- **环回地址（Loopback）**：不能将该地址分配给任何物理接口，主机可以利用此地址向自己发送 IPv6 数据包。
- **未指定地址（Unspecified address）**：仅被用做源地址，表示无 IPv6 地址。
- **唯一本地地址（Unique local）**：与 IPv4 私有地址（RFC 1918）相似，这些地址不能在 IPv6 互联网上进行路由。但是与 RFC 1918 地址不同的是，唯一本地地址的前缀极有可能是全局唯一的。
- **内嵌 IPv4 的地址（IPv4 embedded）**：在低阶 32 比特中内嵌 IPv4 地址的一个 IPv6 地址。

### 4.2.1 全局单播地址

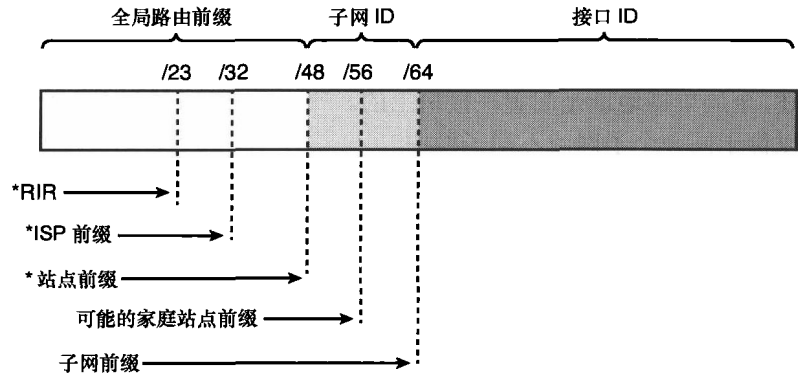
全局单播地址也被称为可聚合全局单播地址，是 IPv6 互联网全局范围内可路由、可达的 IPv6 地址，等同于 IPv4 的公有地址，在 IPv6 编址架构中充当了非常重要的角色。迁移到 IPv6 的一个主要动机就是 IPv4 地址的耗尽。

图 4-3 给出了全局单播地址的一般结构。在第 3 章的图 3-4 中也给出了大多数站点普遍使用的/48 全局单播地址的格式。图 4-3 则在未指定全局路由前缀大小的情况下给出了更为通用的全局单播地址的结构。

注：虽然接口 ID 可以是任意长度，但 64 比特接口 ID 却是 SLAAC 的前提。有关 SLAAC 的内容将在本章后面进行讨论。


图 4-3　全局单播地址结构

图 4-4 解释了地址空间时如何分配给 RIR 和 ISP 的。这些都是最小分配单位，也就是说，RIR 可以获得前缀长度为/23 甚至更小的 IPv6 地址空间，而 ISP 则可以获得前缀长度为/32 甚至更小的 IPv6 地址空间，站点则可以获得前缀长度为/48 甚至更小的 IPv6 地址空间。前缀长度越小也就意味着更大的地址空间。例如，如果站点可以向 ISP 提交足够的证明，那么就可以获得/40 地址而不是/48 地址，从而能够得到更多的地址。图 4-4 给出了提供商可聚合（provider-aggregatable）模型，其中终端客户从其 ISP 获得 IPv6 地址。终端客户也可以直接从 RIR 申请提供商独立（provider-independent）地址空间。如本例所示，在实际中能够向 ISP 证明其应该获得/32 前缀的地址空间并不罕见。




*这些都是最小分配单位，只要能提供足够了的证明，就可以获得更大的地址空间

图 4-4　全局路由前缀大小

注：某些文档中可能会包含术语 TLA（Top-Level Aggregation Identifier，顶级聚合标识符）、NLA（Next-Level Aggregation Identifier，次级聚合标识符）以及 SLA（Site-Level Aggregation Identifier，站点级聚合标识符）。这些术语源于 RFC 2374 “An IPv6 Aggregatable Global Unicast Address Format”，但目前已不再使用。RFC 2374 已被 RFC 3587 废除。RFC 3587 使用了更为简单的地址格式（如图 4-3 所示）。

作为 IANA 的运营方，ICANN（Internet Corporation for Assigned Names and Numbers，互联网名称与编号分配机构）向 5 大 RIR 分配 IPv6 地址块。目前 IANA 分

配的全局单播地址块是从二进制数值 001（即 2000::/3）开始的，因此全局单播地址范围为 2000::/3～3FFF::/3（如表 4-1 所示）。

那么如何获得使用 2000::/3 前缀的地址呢？/3 前缀长度表示在匹配前缀 2000 时只有前三个比特是有意义的。第一个十六位组的前 3 个比特值 2 就是 001x，第 4 个比特 x 是无意义的，可以是 0，也可以是 1，因此第一个十六位组可以是 2（0010）或 3（0011），十六位组（16 比特段）中的其余 14 个比特可以是 0 或 1（如表 4-2 所示）。

**表 4-2　　全局单播地址的范围**

| 全局单播地址（十六进制） | 第一个十六位组的范围 | 第一个十六位组的范围（二进制） |
|---|---|---|
| 2000::/3 | 2000<br>3FFF | 001 0 0000 0000 0000<br>001 1 1111 1111 1111 |

回顾图 4-2 可以看出，IPv6 单播地址空间包含了除 FF00::/8（预留用作多播地址）之外的全部 IPv6 地址空间。目前 IANA 将全局单播地址的分配范围局限在 2000::/3 范围内。有关该地址块的分配情况全部登记在 IANA 注册文件 *"IPv6 Global Unicast Address Assignments"*，大家可以登录 www.iana.org/assignments/ipv6-unicast-address-assignments/ipv6-unicast-address-assignments.xml 查看。请注意，许多 RIR 都拥有比/23 更短的前缀，因此有更多的地址可分配给 ISP。

如前所述，全局单播地址是配置在接口上的。一个接口可以配置多个全局单播地址，这些地址可以位于同一个子网或不同子网中。

接口并不一定要配置全局单播地址，但至少必须要配置一个链路本地地址。也就是说，如果接口有全局单播地址，就必须有链路本地地址。但是，如果接口有链路本地地址，并不一定要有全局单播地址。有关链路本地地址的详细内容将在下一节进行讨论。

与 IPv4 相似，可以采取手工方式或动态方式为设备（准确来说应该是接口）分配全局单播地址。图 4-5 显示了不同的配置选项，下面将从手工配置开始逐一介绍。

### 1. 手工配置全局单播地址

手工配置全局单播地址时存在多个选项，本节将讨论以如下方式手工配置全局单播地址。

- **静态**：静态配置类似于配置静态 IPv4 地址，需要在接口上配置 IPv6 地址与前缀长度。
- **EUI-64**：该配置方式允许指定前缀与前缀长度，而以动态方式创建接口 ID。
- **无编号 IP（IP unnumbered）**：IPv6 的无编号 IP 与 IPv4 相似，允许接口使用同一台设备上其他接口的 IP 地址。

#### 静态配置方式

如第 3 章所述，在 Cisco 路由器上静态配置全局单播地址与配置 IPv4 地址相似。

图 4-6 列出了本节将要讲解的全局单播编址技术。

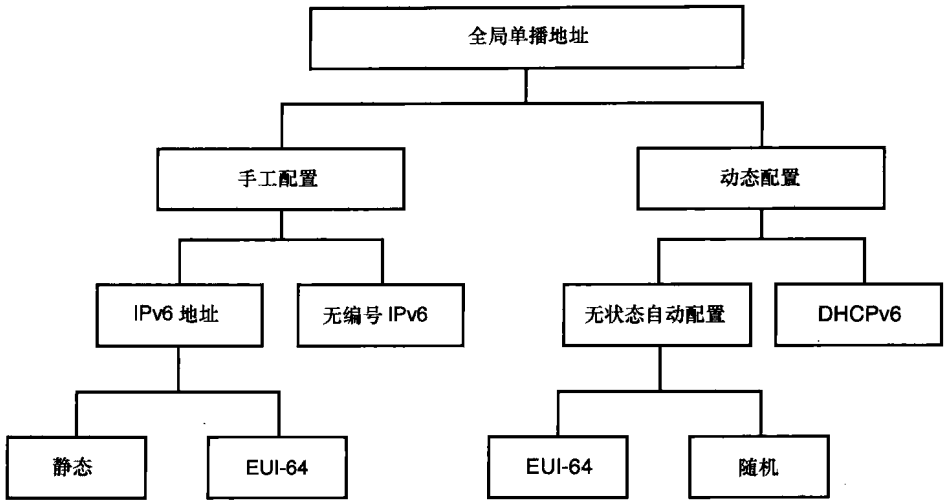



图 4-5　单播地址

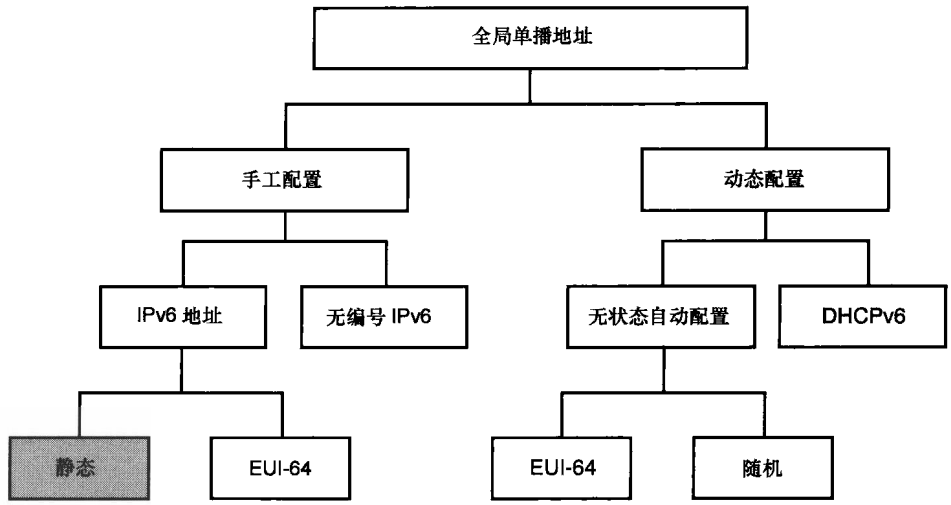



图 4-6　静态 IPv6 地址

表 4-3 列出了静态配置全局单播地址的命令格式。有关该命令在配置其他类型地址时的形式与选项将在本章后面介绍。命令 **ipv6 address** 如下所示。

```
Router(config-if)# ipv6 address ipv6-address/prefix-length
```

**表 4-3　　命令 ipv6 address**

| 命令 | 描述 |
| --- | --- |
| Router(config)# **interface** *interface-type interface-number* | 指定接口类型和接口号 |

续表

| 命令 | 描述 |
| --- | --- |
| Router(config-if)# **ipv6 address** *ipv6-address/prefix-length* | 指定分配给接口的 IPv6 地址和前缀长度，如果要删除接口的 IPv6 地址，就需要使用该命令的 **no** 形式 |

与 IPv4 相似，这是在路由器接口和服务器上手工配置 IPv6 地址的最佳做法。图 4-7 显示了第 3 章曾经介绍过的拓扑结构。

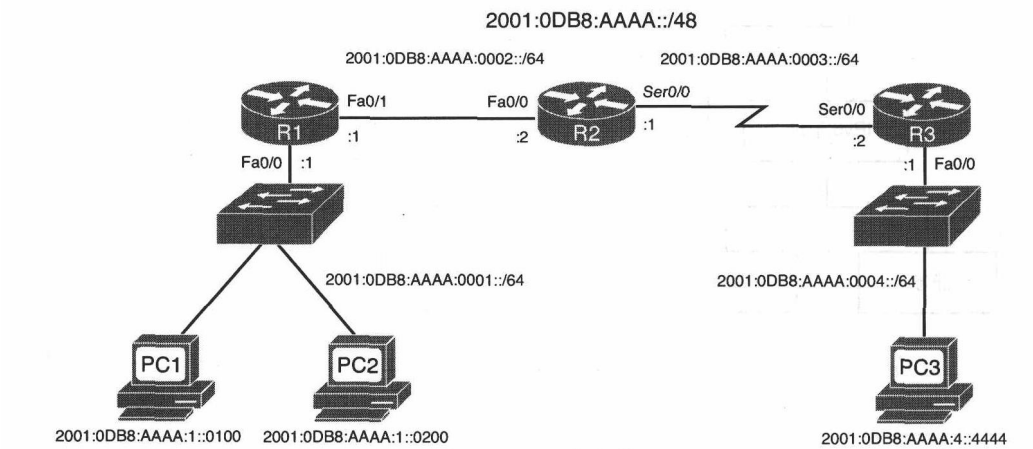



图 4-7 IPv6 拓扑结构

如前所述，网络 2001:0DB8:AAAA::/48 被划分成 4 个/64 子网：

- 2001:0DB8:AAAA:0001::/64
- 2001:0DB8:AAAA:0002::/64
- 2001:0DB8:AAAA:0003::/64
- 2001:0DB8:AAAA:0004::/64

例 4-1 给出了为路由器 R1、R2 和 R3 配置全局单播地址的命令（虽然没有显示 **clock rate** 和 **no shutdown** 等必需的命令）。需要注意的是，在配置 IPv6 地址时，前缀与前缀长度之间没有空格。

**例 4-1 在路由器 R1、R2 和 R3 上配置全局单播地址**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 address 2001:db8:aaaa:1::1/64
R1(config-if)# exit
R1(config)# interface fastethernet 0/1
R1(config-if)# ipv6 address 2001:db8:aaaa:2::1/64
R1(config-if)# exit
```

```
R2(config)# interface fastethernet 0/0
R2(config-if)# ipv6 address 2001:db8:aaaa:2::2/64
R2(config-if)# exit
```

```
R2(config)# interface serial 0/0
R2(config-if)# ipv6 address 2001:db8:aaaa:3::1/64
R1(config-if)# exit
```

```
R3(config)# interface fastethernet 0/0
R3(config-if)# ipv6 address 2001:db8:aaaa:4::1/64
R3(config-if)# exit
R3(config)# interface serial 0/0
R3(config-if)# ipv6 address 2001:db8:aaaa:3::2/64
R3(config-if)# exit
```

注：为了让路由器能够路由 IPv6 数据包，需要启用全局配置命令 **ipv6 unicast-routing**。

例 4-2 显示了路由器 R1 的运行配置（running-config）。显示结果“*no ip address*”表示该接口无 IPv4 地址。在配置接口时，即使不使用省略形式的 IPv6 地址，运行配置也会使用压缩格式。可以在同一个接口上同时配置 IPv6 地址和 IPv4 地址，这种情况被称为“双栈（dual stack）”，相关内容将在第 10 章进行讨论。

**例 4-2　在路由器 R1 执行命令 show running-config 的输出结果**

```
R1# show running-config
<output omitted for brevity>
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
 ipv6 address 2001:DB8:AAAA:1::1/64
!
interface FastEthernet0/1
 no ip address
 duplex auto
 speed auto
 ipv6 address 2001:DB8:AAAA:2::1/64
```

IPv4 中的大多数验证命令都可以用于 IPv6，只要将 **ip** 替换成 **ipv6** 即可。例 4-3 显示了命令 **show ipv6 interface brief** 的输出结果。该输出结果与命令 **show ip interface brief** 相似，但也有一些区别。请注意，线路协议与这两个接口的状态都是 up/up。

**例 4-3 在路由器 R1 执行命令 show ipv6 interface brief 的输出结果**

```
R1# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::203:6BFF:FEE9:D480    ! Link-local unicast address
    2001:DB8:AAAA:1::1          ! Global unicast address
FastEthernet0/1            [up/up]
    FE80::203:6BFF:FEE9:D481
    2001:DB8:AAAA:2::1
R1#
```

虽然每个接口只配置了一个 IPv6 地址，但每个接口都有两个 IPv6 地址。以 FE80 开头的 IPv6 地址是链路本地单播地址。只要为接口分配了全局单播地址，就会在接口上自动配置链路本地单播地址。链路本地单播地址只能与同一条链路上的其他设备进行通信，有关链路本地地址的相关内容将在本章后面进行讨论。

例 4-4 显示了命令 **show ipv6 interface fastethernet 0/0** 的输出结果。请注意已配置的全局单播地址和链路本地地址，在“Joined group address”下面有一组以 FF02 开头的其他地址。只要在路由器的至少一个接口上配置了全局单播地址，路由器就会自动成为这些多播组的成员。有关多播地址的相关内容将在本章后面进行讨论。输出结果中还有一些行与 ND（Neighbor Discovery，邻居发现）有关。其中，ND 是 ICMPv6 的一部分，有关 ICMPv6 和 ND 的相关内容将在第 5 章进行讨论。

**例 4-4 在路由器 R1 执行命令 show ipv6 interface fastethernet 0/0 的输出结果**

```
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::203:6BFF:FEE9:D480
  Global unicast address(es):
    2001:DB8:AAAA:1::1, subnet is 2001:DB8:AAAA:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1
    FF02::1:FFE9:D480
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
```

```
  ND router advertisements live for 1800 seconds
  Hosts use stateless autoconfig for addresses.
R1#
```

可以看出，虽然只是配置了全局单播地址（与 IPv4 地址相当），但是对 IPv6 来说，并不仅仅是更长的 IP 地址而已。

### EUI-64 配置方式

另一种手工配置 IPv6 全局单播地址的方式就是使用 IEEE 定义的 EUI-64（64-bits Extended Unique Identifier，64 比特扩展的唯一标识符）或改进型 EUI-64 进程。采取 EUI-64 配置方式时，在以手工方式配置 IPv6 地址的前缀（网络部分）的同时，利用 EUI-64 进程自动分配接口 ID。而改进型 EUI-64 则使用接口的以太网 MAC 地址来生成 64 比特接口 ID（地址的主机部分）。结合手工配置的前缀，就可以为接口生成一个全局单播地址。图 4-8 列出了手工配置 IPv6 地址的另一种方法——EUI-64 技术。

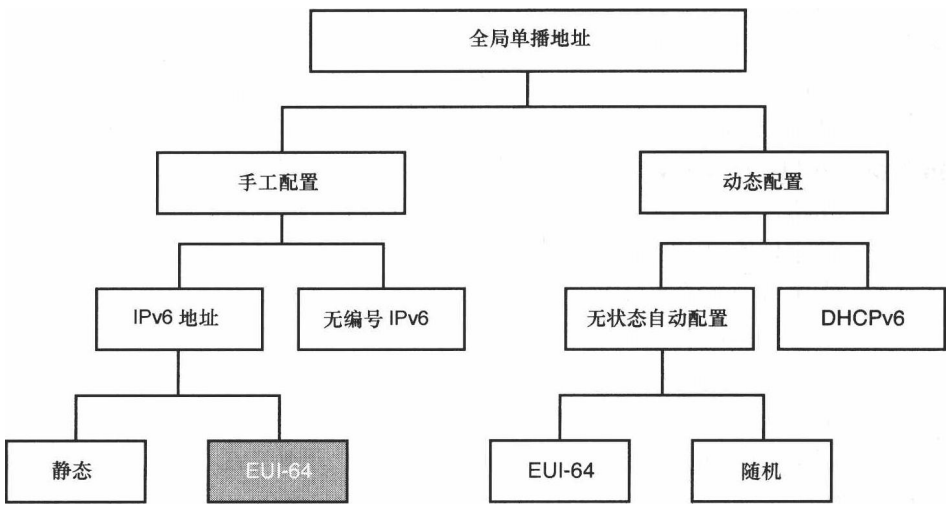



图 4-8　EUI-64

以太网 MAC 地址长度是 48 比特，也有 64 比特标准的 MAC 地址。如图 4-9 所示，以太网 48 比特 MAC 地址包括 24 比特 OUI（Organizational Unique Identifier，组织机构唯一标识符）和 24 比特设备标识符（以十六进制表示）。以太网 NIC（Network Interface Card，网络接口卡）制造商拥有一个或多个 OUI 或厂商代码。每个 24 比特 OUI 都有一个唯一标识以太网 NIC 的 24 比特设备标识符。也就是说，在 24 比特 OUI 后面加上 24 比特设备标识符，就可以唯一地标识一块以太网 NIC。

注：如果希望了解 IEEE OUI 代码列表，大家可以访问 http://standards.ieee.org/develop/regauth/oui/oui.txt。

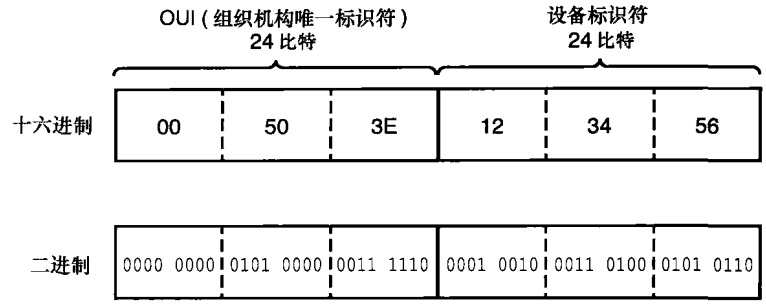



图 4-9 以太网 MAC 地址

改进型 EUI-64 则由 24 比特 OUI、求反后的 U/L 比特、16 比特值 FFFE 以及 24 比特设备标识符组成，其实现过程分为三步（如图 4-10 所示）。

**第 1 步**：将 MAC 地址转换成二进制之后，将 MAC 从中间分为左边的 24 比特 OUI 和后边的 24 比特设备标识符。

**第 2 步**：在 OUI 与设备标识符之间插入 FFFE。FFFE 的二进制形式为 1111 1111 1111 1110。其中，FFFE 是 IEEE 保留值，用来表示 EUI-64 地址是由 48 比特 MAC 地址生成的。

**第 3 步**：U/L（Universally/Locally，全球/本地）比特也被称为本地/全局（Local/Global）比特，是第一个字节的第 7 个比特，用来确定该地址是全球统一管理的还是本地管理的。如果该比特为 0，那么就表示该地址由 IEEE 管理（通过指定唯一的企业 ID）；如果该比特为 0，就表示该地址由本地管理。网络管理员会改写制造商地址并指定一个不同的地址。

在 U/L 比特是否被反转（从 0 到 1 或者从 1 到 0）的问题上似乎存在一些矛盾之处，有些文档提到，仅当 U/L 比特为 0 时才应该被反转为 1（不过对 Cisco 设备来说，无论该比特是什么值，都要被反转）。从图 4-10 可以看出，反转 U/L 比特会修改接口 ID 中的十六进制数值。

> 注：RFC 5342 “IANA Considerations and IETF Protocol Usage for IEEE 802 Parameters” 讨论了 U/L 比特反转背后的逻辑。反转 U/L 比特的目的是为了简化网络管理员输入本地范围标识符的工作。

EUI-64 进程利用 48 比特 MAC 地址来创建 64 比特 IPv6 地址的接口 ID。MAC 地址 00-50-3E-12-34-56 会被转换成 EUI-64 格式 02-50-3E-FF-FE-12-34-56。EUI-64 将 FFFE 插入地址之间并通过修改第 7 比特将第二个十六进制数值从 0 改为 2。

> 注：IPv6 的接口 ID 为 64 比特，能够容纳较长的 64 比特 MAC 地址。当接口 MAC 地址为 48 比特时，先行标准就是使用改进型 EUI-64 格式。

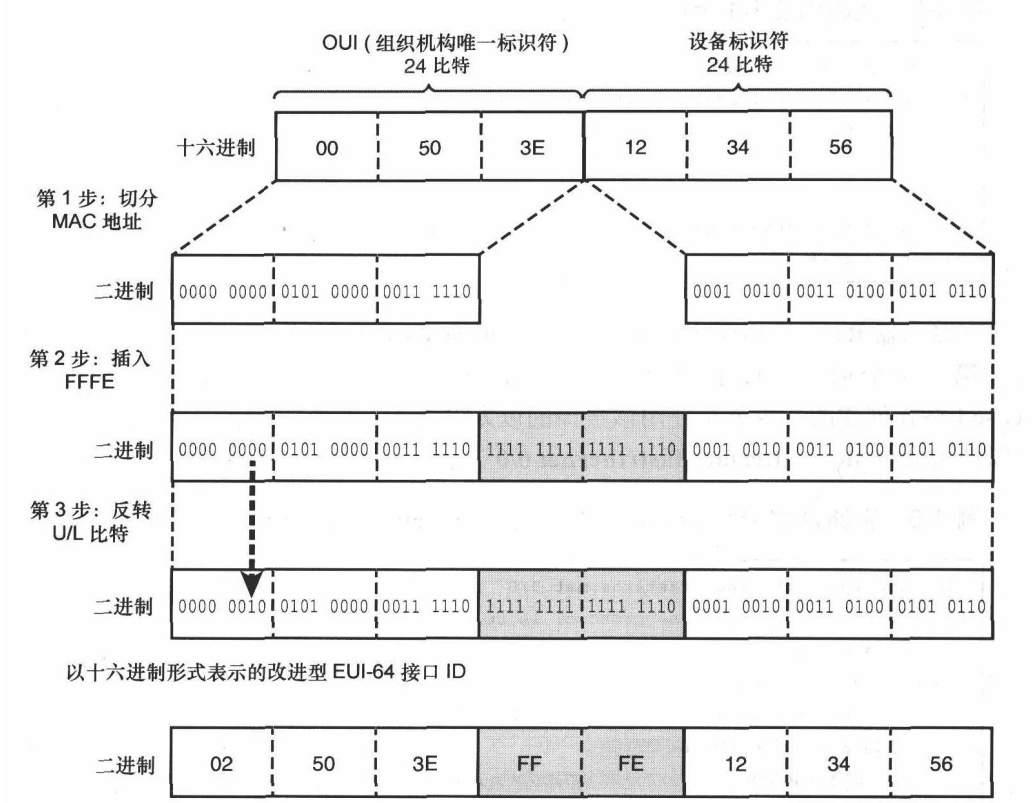



图 4-10　改进型 EUI-64 格式

表 4-4 给出了以改进型 EUI-64 格式配置 IPv6 地址的相关命令。请注意，仅需指定 IPv6 地址的前缀（网络部分）和前缀长度，接口 ID（主机部分）则由 EUI-64 进程来确定。采用 EUI-64 选项的 **ipv6 address** 命令的形式为：

```
Router(config-if)# ipv6 address ipv6-prefix/prefix-length eui-64
```

**表 4-4**　　IPv6 EUI-64 接口

| 命令 | 描述 |
|---|---|
| Router(config)# **interface** *interface-type interface-number* | 指定接口类型和接口号 |
| Router(config-if)# **ipv6 address** *ipv6-address/prefix-length* **eui-64** | 指定分配给接口的 IPv6 地址前缀和前缀长度，IPv6 地址的低阶 64 比特使用 EUI-64 接口 ID |

例 4-5 给出了使用 EUI-64 格式配置路由器 R1 的快速以太网接口 Fast Ethernet 0/0 地址的示例。该命令仅指定了前缀和前缀长度/64，后面附加了选项 **eui-64**。如果不包含选项 **eui-64**，那么接口 ID 将由全 0 构成，这也是一个有效的 IPv6 接口地址。

**例 4-5 在路由器 R1 上配置 EUI-64 地址**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 address 2001:0db8:aaaa:0001::/64 ?
  eui-64  Use eui-64 interface identifier
  <cr>

R1(config-if)# ipv6 address 2001:0db8:aaaa:0001::/64 eui-64
R1(config-if)#
```

在路由器 R1 上应用命令 **show ipv6 fastethernet 0/0**（如例 4-6 所示）。请注意该接口上第二个全局单播地址 2001:DB8:AAAA:1:203:6BFF:FEE9:D480。接口 ID 是由 EUI-64 格式创建的，也就是使用该接口的以太网 MAC 地址 0003.6BE9.D480 创建的。可以使用命令 **show interface fastrthernet 0/0** 来验证以太网 MAC 地址。

**例 4-6 在路由器 R1 执行命令 show ipv6 interface fastethernet 0/0 的输出结果**

```
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::203:6BFF:FEE9:D480
  Global unicast address(es):
    2001:DB8:AAAA:1::1, subnet is 2001:DB8:AAAA:1::/64
    ! Address using EUI-64 format
    2001:DB8:AAAA:1:203:6BFF:FEE9:D480, subnet is  2001:DB8:AAAA:1::/64
Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1
    FF02::1:FFE9:D480
<output omitted for brevity>

R1# show interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is AmdFE, address is 0003.6be9.d480 (bia 0003.6be9.d480)
! Ethernet MAC Address
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
<output omitted for brevity>
```

图 4-11 显示了 EUI-64 将路由器 R1 的以太网 MAC 地址转换为接口 ID 的过程。

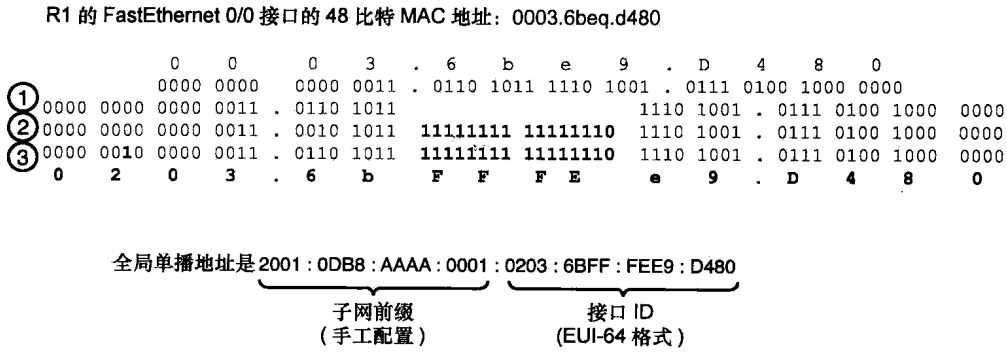



图 4-11　路由器 R1 快速以太网接口 Fast Ethernet 0/0 的 EUI-64 格式

地址转换过程分为以下 3 个步骤。

**第 1 步**：将快速以太网接口 Fast Ethernet 0/0 的 48 比特 MAC 地址分成两个 24 比特段。

**第 2 步**：将 FFFE (1111 1111 1111 1110)插入中间。

**第 3 步**：将第 7 比特（U/L 比特）由 0 反转为 1，从而将第二个十六位组的数值由 0 更改 2。

简单总结一下，从例 4-5 可以看出，EUI-64 配置方式就是手工配置子网前缀并利用选项 **eui-64** 创建接口 ID。通过将手工配置的子网前缀（2001:DB8:AAAA:1::/64）附加到由 EUI-64 生成的接口 ID（6BFF:FEE9:D480）上，即可得到接口的全局单播地址 2001:DB8:AAAA:1:203:6BFF:FEE9:D480。

回顾例 4-6 所示的命令 **show interface fastrthernet 0/0** 的输出结果，可以注意到该接口有两个全局单播地址：静态配置的地址 2001:DB8:AAAA:1::1（来自于例 4-1）和使用 EUI-64 配置的地址 2001:DB8:AAAA:1:203:6BFF:FEE9:D480。该接口上的两个地址都位于同一个子网 2001:DB8:AAAA:1::/64 中。

因此，当接口上配置了多个位于同一子网的地址时，究竟哪个地址会被用作该路由器向外发送数据包的源地址呢？这个问题的答案可以参考 RFC 3484 “Default Address Selection for Internet Protocol version 6”。如果希望给某个接口分配多个位于同一子网的全局单播地址，那么就必须有足够的理由这么做，而且还必须了解相应的选择进程。此外，RFC 3484 还讨论了同一个接口上的全局单播地址和链路本地单播地址。由于这两个地址的类型不同，因而能够很容易地知道会使用哪个地址（取决于应用或协议）。有关链路本地地址的相关内容将在本章后面进行讨论。

为了简化起见，下面就删除路由器 R1 上由 EUI-64 创建的 IPv6 地址（如例 4-7 所示）。使用命令 **no ipv6 address** 以及相应的 IPv6 前缀/前缀长度即可删除指定的 IPv6 地址。IPv6 允许在同一个接口上配置多个 IPv6。利用命令 **show ipv6 interface brief** 即可验证此时路由器 R1 的快速以太网接口 Fast Ethernet0/0 上只有一个全局单播地址了。

**例 4-7 在路由器 R1 执行命令 show ipv6 interface brief 的输出结果**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# no ipv6 address 2001:DB8:AAAA:1:203:6BFF:FEE9:D480/64
R1(config-if)# end
R1# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::203:6BFF:FEE9:D480    ! Link-local unicast address
    2001:DB8:AAAA:1::1          ! Global unicast address
Serial0/0                  [administratively down/down]
    unassigned
FastEthernet0/1            [up/up]
    FE80::203:6BFF:FEE9:D481
    2001:DB8:AAAA:2::1
Serial0/1                  [administratively down/down]
    unassigned
R1#
```

注：后面将讨论 EUI-64 进程与 SLAAC（Stateless Address Autoconfiguration，无状态地址自动配置）共同使用时的应用场景。

### 无编号 IPv6 配置方式

另一种手工配置 IPv6 全局单播地址的形式是命令 **ipv6 unnumbered**。命令 **ipv6 unnumbered** 通过为接口分配一个来自其他接口的全局单播地址来启动该接口的 IPv6 处理能力。图 4-12 列出了手工配置 IPv6 地址的另一种方法——无编号 IPv6 技术。

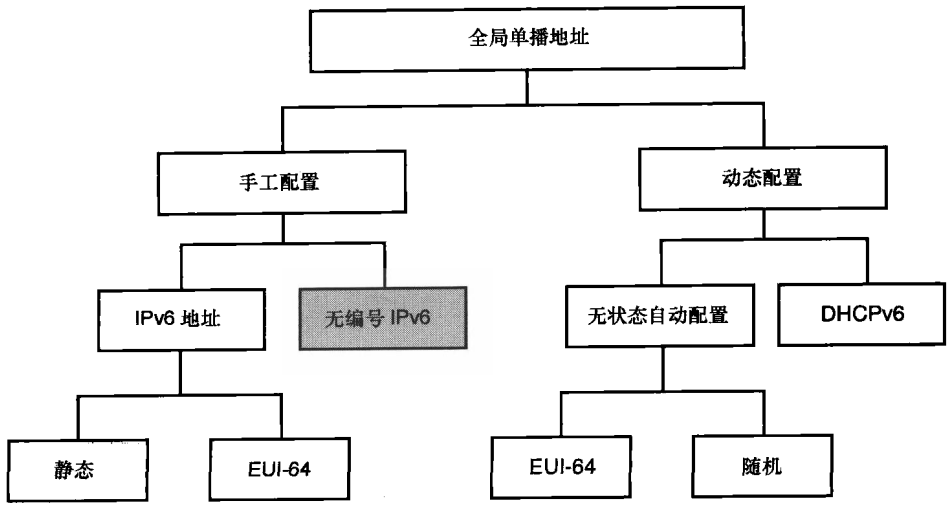



图 4-12 无编号 IPv6

表 4-5 列出了命令 **ipv6 unnumbered**，该命令用于指定接口向外发送 IPv6 数据包时用到的源地址，命令格式如下：

```
Router(config-if)# ipv6 unnumbered interface-type interface-number
```

表 4-5　　命令 ipv6 unnumbered

| 命令 | 描述 |
| --- | --- |
| Router(config)# **interface** *interface-type interface-number* | 指定接口类型和接口号 |
| Router(config-if)# **ipv6 unnumbered** *interface-type interface-number* | 指定无编号接口向外发送 IPv6 数据包时用到的源地址的接口类型和接口号 |

注：IPv4 中的命令 **ip unnumbered** 主要用来帮助节约 IPv4 地址空间，很明显，对于 IPv6 网络来说，地址空间完全不是问题，因而该命令的作用显得更加有限。

例 4-8 解释了命令 **ipv6 unnumbered** 在路由器上的配置方式。该路由器的串口 serial 0/1 被配置为无编号，因此从接口 serial 0/1 发送的 IPv6 数据包都将使用快速以太网接口 Fast Ethernet 0/0 的地址作为自己的源地址。

**例 4-8　命令 ipv6 unnumbered 使用示例**

```
Router(config)# interface fastethernet 0/0
Router(config-if)# ipv6 address 2001:0DB8:ABCD:1234::1/64
Router(config)# interface serial 0/1
Router(config-if)# ipv6 unnumbered fastethernet 0/0
```

### 2. 动态配置

全局单播地址还可以采用动态方式进行分配，而无需任何手工配置。动态配置全局单播地址的方法有以下两种。

- **SLAAC**：利用该配置方法时，前缀和前缀长度是通过 ND 路由器宣告（Router Advertisement）消息来确定的，而接口 ID 则是通过 EUI-64 进程创建的。
- **DHCPv6（Dynamic Host Configuration Protocol for IPv6，用于 IPv6 的动态主机配置协议）**：类似于 IPv4 中的 DHCP，设备可以通过 DHCPv6 服务器的相关服务自动获取编址信息。

#### SLAAC

作为全局单播地址的两种自动配置方法之一，SLAAC 定义在 RFC 4862（IPv6 Stateless Address Autoconfiguration）中。图 4-13 列出了 SLAAC 动态地址配置技术。

SLAAC 使用前面所说的 IEEE 改进型 EUI-64 格式。EUI-64 利用以太网 MAC 地址创建 IPv6 地址的接口 ID（主机部分）。结合邻居发现协议，主机就可以确定自己完整的全局单播地址，而无需任何手工配置或 DHCPv6 服务器的服务。有关 ND 及 SLAAC 的

详细信息将在第 5 章进行讨论。为了对 SLAAC 有个基本了解，此处仅对 ND 做简单介绍。

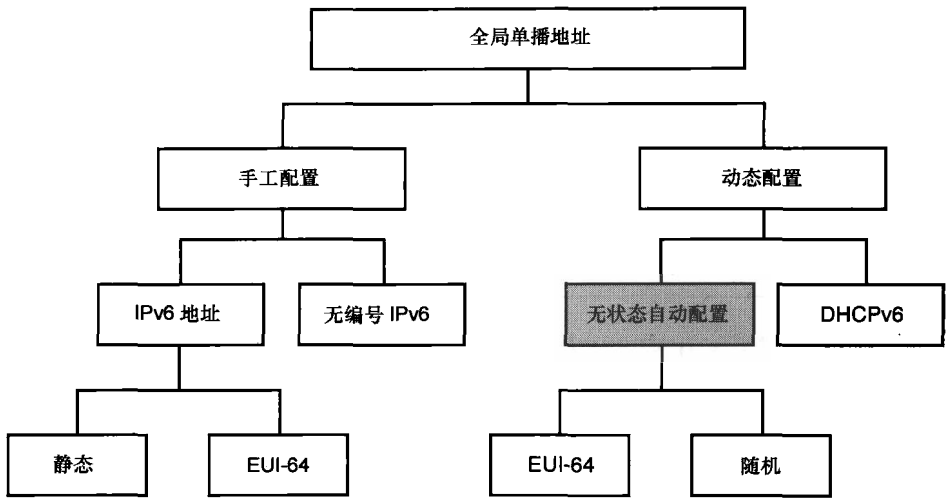



图 4-13 SLAAC

ND 定义在 RFC 4861 “Neighbor Discovery in IPv6” 中。为了实现其功能，ND 利用 ICMPv6 来交换报文消息，5 种新的 ICMPv6 消息分别为：

- RA（Router Advertisement，路由器宣告）消息；
- RS（Router Solicitation，路由器请求）消息；
- NS（Neighbor Solicitation，邻居请求）消息；
- NA（Neighbor Advertisement，邻居宣告）消息；
- 重定向消息。

ICMPv6 不仅仅是 IPv6 版本的 ICMP，而是一个更加健壮的协议。ICMPv6 包含了许多新功能并做了大量改善。只有前 4 种消息与这里要介绍的 SLAAC 有关，如本章所申明的那样，由于这些消息与地址类型相关，因而这里的主要目的就是理解这些消息。有关 ND 与每种消息的详细内容将在第 5 章进行讨论。

顾名思义，IPv6 邻居发现协议的作用就是发现邻居。设备可以接收 ND 消息以自动确定它们的前缀、前缀长度、默认网关以及其他信息。除了接口 ID，网络设备会从路由器发出的这些消息中获得其网络配置信息。这与服务器或路由器被配置为提供 DHCP 服务有何区别呢？DHCP 支持无状态（stateless）和状态化（stateful）服务。状态化服务会记录客户端数据（状态），而无状态服务则不保留任何状态信息。有状态 DHCP 服务器必须维护精确的可用地址列表，因而不会发布重复地址。顾名思义，SLAAC 属于无状态服务，在没有状态化 DHCPv6 服务器提供服务的情况下，主机在创建了自己的唯一地址之后，需要验证该地址的唯一性。

如图 4-14 所示，主机（PC-B）被配置为自动获取其 IPv6 地址。也就是说，PC-B

将从 DHCPv6 服务器或者通过 SLAAC 获取 IPv6 地址。

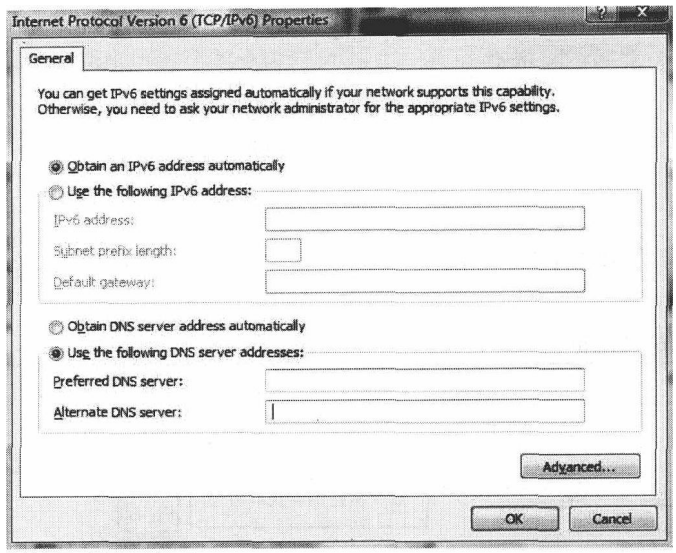



图 4-14　配置 PC-B 自动获取 IPv6 地址

自动配置的（SLAAC）地址会处于以下一种或多种状态。

- **试验（Tentative）状态**：地址的唯一性还处于验证当中。试验地址并不被认为已经分配给了接口。接口会丢弃发送给试验地址的数据包，但是会接收发送给试验地址的与 DAD（Duplicate Address Detection，重复地址检测）相关的邻居发现包。
- **优选（Preferred）状态**：该接口地址的唯一性已经被证实，设备可以利用该地址发送和接收流量。地址能够处于试验和优选状态的时间都包含在 RA 消息中。
- **废除（Deprecated）状态**：分配给接口的该地址仍然有效，但是不建议使用。虽然废除地址不应该被新的通信过程用作源地址，但是来自废除地址或发送给废除地址的数据包仍然能够被正常传送。对于现有通信过程来说，如果某些特定的上层应用（如现有的 TCP 连接）在切换到优选地址时可能会出现问题，那么就可以继续将废除地址用作源地址。
- **有效（Valid）状态**：有效地址是优选地址或废除地址。有效地址可以是数据包的源地址或目的地址。地址能够处于试验和有效状态的时间都包含在 RA 消息中。有效时间必须大于或等于优选时间。有关 RA 消息的详细内容将在后面介绍，有效时间与优选时间则在第 5 章进行详细讨论。
- **无效（Invalid）状态**：当有效时间到期时，有效地址就会变成无效地址。无效地址不应该出现在数据包的源地址或目的地址中。

图 4-15 解释了自动配置地址可能存在的上述 5 种状态。

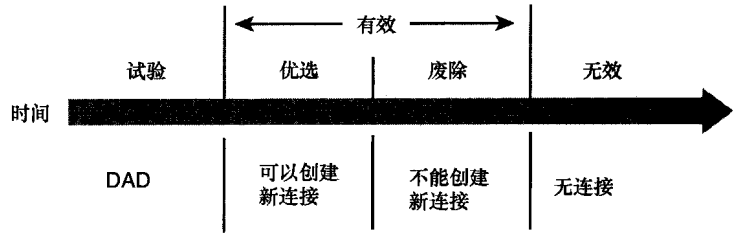



图 4-15 自动配置地址的状态

图 4-16 给出了 SLAAC 进程的简化版本。

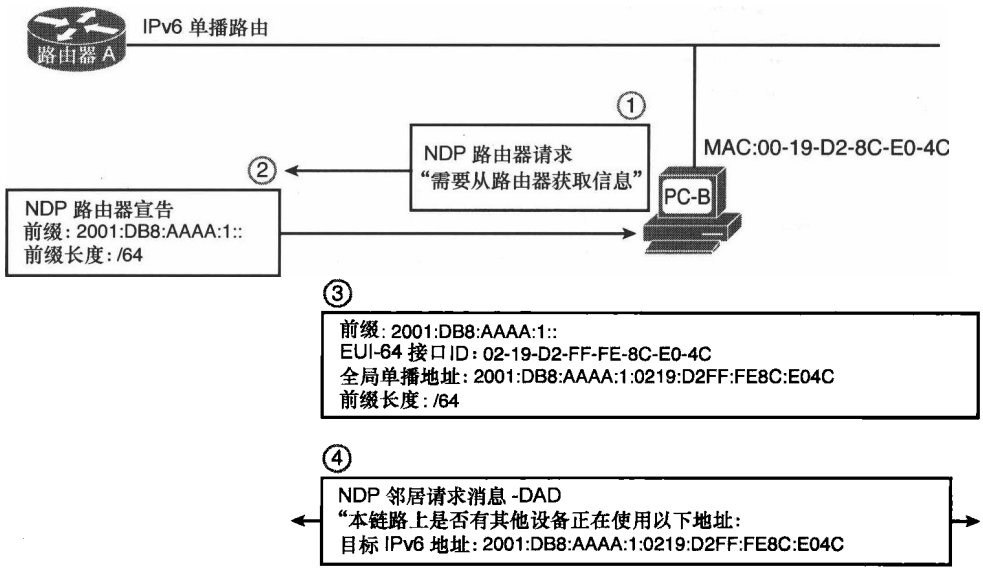



图 4-16 NDP 路由器宣告消息与路由器请求消息

路由器 A 周期性地向外发送 ND RA 消息。由路由器发出的 RA 消息的作用是宣告其存在性以及与链路相关的参数，如链路前缀、前缀长度、默认网关和链路 MTU（Maximum Transmission Unit，最大传输单元）。RA 消息会被发送给全部节点多播（all-nodes multicast）地址（FF02::1），该地址在本质上等同于广播地址。有关多播地址的相关内容将在本章后面介绍。RA 消息是周期性发送的，也可以作为 RS 消息的响应消息。

RS 消息是由主机发送的，作用是请求路由器发送 RA 消息。RS 消息会被发送给全部路由器多播地址（FF02::2）。链路上所有配置了命令 **ipv6 unicast-routing** 的 IPv6 路由器都要处理该消息。

对于路由器来说，如果要发送 RA 消息并运行 IPv6 路由协议（如后所述），都必须配置命令 **ipv6 unicast-routing**：

```
RouterA(config)# ipv6 unicast-routing
```

如图 4-16 所示，SLAAC 进程包括以下步骤。

**第 1 步**：如图 4-14 所示，PC-B 被配置为自动获取 IP 编址信息。启动后，PC-B 没有看到 RA 消息，因而向外发送 RS 消息，以告知本地 IPv6 路由器：它需要 RA 消息。

**第 2 步**：路由器 A 收到 RS 消息后会发出 RA 消息作为响应。RA 消息中包含该链路的前缀与前缀长度，并将自己的地址作为默认网关。路由器 A 向外传播的默认网关地址实际上是其链路本地地址，而不是像大家期待的全局单播地址。有关详细内容将在第 5 章进行讨论。

**第 3 步**：PC-B 收到 RA 消息，该消息中包括了本地网络的前缀与前缀长度信息 2001:DB8:AAAA:1::/64。

PC-B 的以太网 MAC 地址是 00-19-D2-8C-E0-4C。利用改进型 EUI-64 技术，在 MAC 地址的 OUI 和设备标识符之间插入 FF-FE，并将第 7 个比特从 0 反转为 1，从而将第二个十六进制数值从 0 改为 2，因此分配给接口 ID 的 64 比特数值为 02-19-D2-FF-FE-8C-E0-4C。

接着，PC-B 再将 RA 消息中的前缀 2001:DB8:AAAA:1::附加在一起，即可得到全局单播地址 2001:DB8:AAAA:1:02-19-D2-FF-FE-8C-E0-4C。该地址在唯一性被验证（下一步）之前将处于试验状态。

**第 4 步**：由于 SLAAC 属于无状态进程，因而没有任何设备记录链路上的所有全局单播地址以防止地址重复。因此，需要主机自己确定刚才创建的地址没有被其他设备使用。这种 ND 进程就被称为 DAD，有关 DAD 进程的详细内容将在第 5 章进行讨论，不过该进程其实非常简单直观，无外乎就是设备将自己的地址发送给链路上的所有邻居，以查看是否有其他设备正在使用该地址。这一点类似于 IPv4 中的无故 ARP（Gratuitous ARP）进程。PC-B 发送一条将自己的全局单播地址 2001:DB8:AAAA:1:02-19-D2-FF-FE-8C-E0-4C 作为目的地址的 NS 消息（类似于 IPv4 中的 ARP 请求）。如果其他设备也在使用该地址，那么就会响应以 NA 消息（类似于 IPv4 中的 ARP 应答），NS 消息会被发送给请求节点多播（solicited node multicast）地址。请求节点多播地址的作用类似于广播地址，但效率更高。有关请求节点多播地址的相关内容将在本章后面进行讨论。如果 PC-B 没有收到 NS 消息的响应消息（以 NA 消息的形式），那么就可以确认其全局单播地址的唯一性，此时该地址将进入优选状态。

被配置为自动获取编址信息的主机究竟是使用 SLAAC 还是 DHCPv6，取决于路由器 RA 消息中的配置信息。RA 消息可以被配置为指定链路上的设备使用状态化 DHCPv6 自动配置，而不是无状态自动配置。

在没有 DHCPv6 服务器的情况下，设备就能动态确定自己的全局单播地址，这是 IPv6 的一个重要优势。对于物联网来说，网络摄像头和传感器等 IPv6 设备完全可以在开机后从路由器以及 EUI-64 进程获得其全部编址信息。

注：将设备的以太网 MAC 地址用作接口 ID 可能会存在一定的隐私问题。由于通过分析数据包能够追溯到真实的源设备，因此，不是所有的操作系统都使用前面所说的 EUI-64 进程。对 Windows 操作系统来说，Windows XP 和 Windows Server 2003 使用 EUI-64，Windows Vista 和更新的 Windows 版本都不使用 EUI-64。这些操作系统在使用 SLAAC 时，会创建一个随机的 64 比特接口 ID。

### DHCPv6

DHCPv6 类似于 IPv4 的 DHCP，DHCPv6 可以为 IPv6 设备提供状态化自动配置和无状态自动配置能力。设备被配置为自动获得其 IPv6 编址信息后（如图 4-14 所示），将会使用以下方法：

- 使用 RA 消息的无状态编址；
- 使用 DHCPv6 的状态化编址。

图 4-17 列出了动态分配 IPv6 地址的技术之一——DHCPv6。

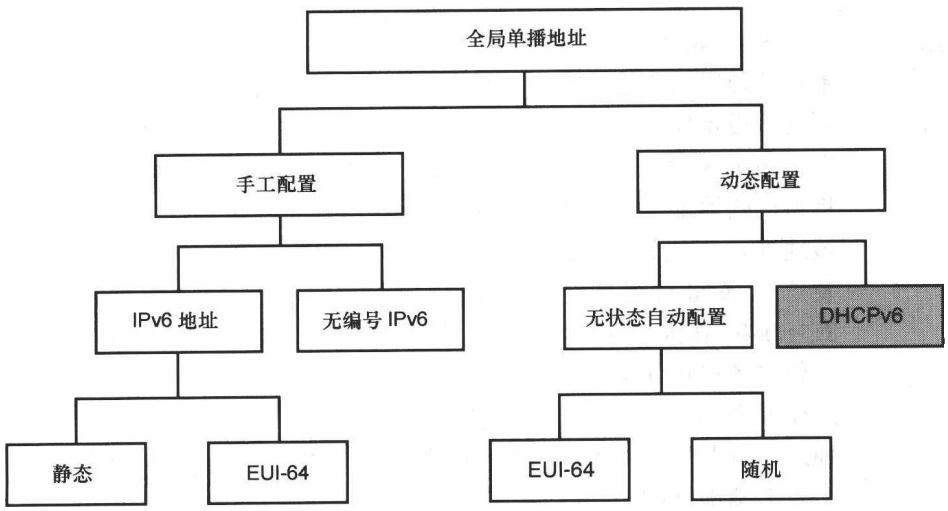



图 4-17 DHCPv6

DHCPv6 定义在 RFC 3315 “Dynamic Host Configuration Protocol for IPv6” 中，有关 DHCPv6 的详细内容将在第 9 章进行讨论。本节将简要解释 DHCPv6 进程并说明设备如何确定使用无状态编址还是状态化编址。

图 4-18 解释了该过程的各个步骤，首先是 PC-B 确定将要使用无状态自动配置还是状态化自动配置。

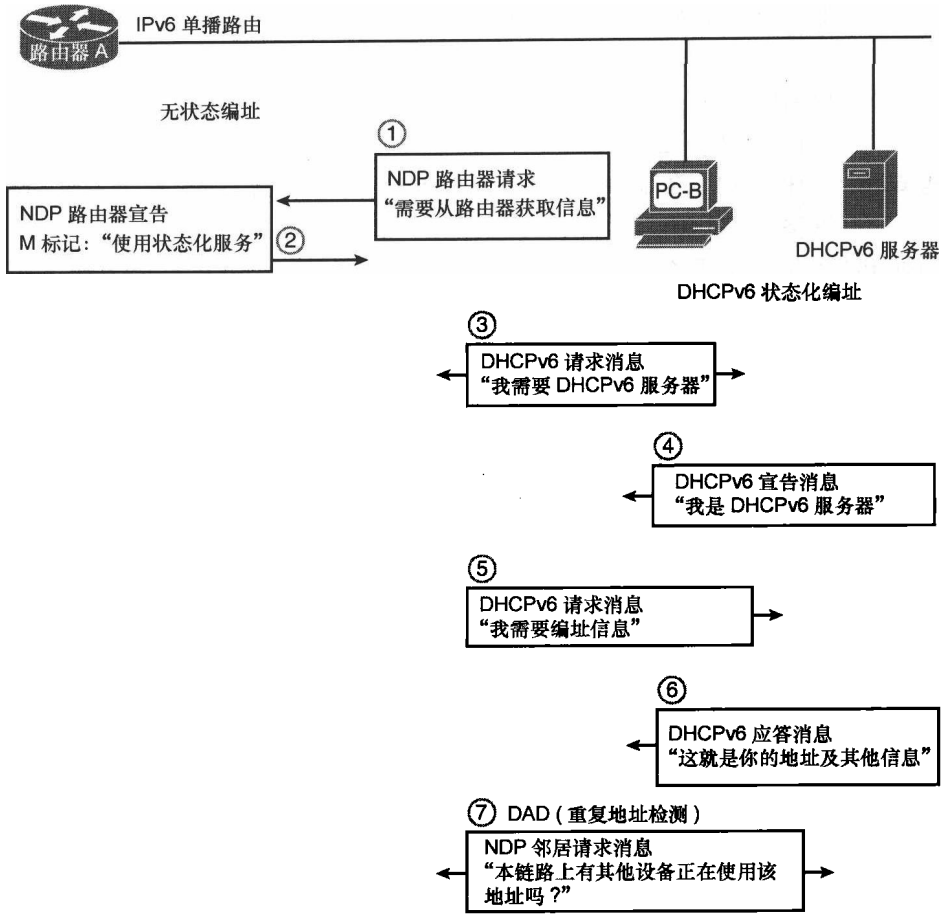



图 4-18　状态化 DHCPv6

**第 1 步**：PC-B 向外发送 RS 消息，除非已经收到了 RA 消息。

**第 2 步**：RA 消息中有一个被称为被管地址配置标记（managed address configuration flag）或 M 标记（M flag）的 1 比特字段。

- M 标记为 0 时，表示该设备使用 SLAAC。
- M 标记为 1 时，表示该设备使用 DHCPv6（如图 4-18 所示）。

由于 PC-B 已经知道了其必须使用状态化自动配置，因而开始发起相应的进程，以便从 DHCPv6 服务器获取编址信息。

**第 3 步**：PC-B 向专用于 DHCPv6 服务器的特定多播地址 FF02::1:2（多播地址将在本章后面讨论）发送 DHCPv6 请求（Solicit）消息。

**第 4 步**：一台或多台 DHCPv6 服务器会返回 DHCPv6 宣告（Advertise）消息，表明它们可以提供 DHCPv6 服务。如果 PC-B 收到多条 DHCPv6 宣告消息，

那么会有相应的进程生成服务器优选值，以便选择合适的 DHCPv6 服务器（将在第 9 章进行详细讨论）。

**第 5 步**：PC-B 通过发送请求（Request）消息来响应选定的 DHCPv6 服务器，以请求包括 IPv6 地址在内的配置参数。

**第 6 步**：DHCPv6 服务器会响应应答（Reply）消息，包含已分配的地址和其他配置参数（类似于 IPv4 DHCP 中用到的参数）。

即使 PC-B 从 DHCPv6 服务器获得了地址（状态化服务），它也要使用 DAD 进程来确保链路上没有其他设备使用该地址；

**第 7 步**：PC-B 使用刚刚获得的全局单播地址向链路上的所有设备发送 NS 消息。由于 PC-B 的全局单播地址在唯一性得到验证之前一直处于试验状态，因而源地址是未指定地址“::”，目的地址是请求节点多播地址。如前所述，请求节点多播地址类似于 IPv4 的广播地址。如果链路上有其他设备使用了相同的地址，那么该设备就会响应以 NA 消息。

### 4.2.2 链路本地单播地址

链路本地地址是仅用于单条链路的单播地址。必须保证这些地址在链路上的唯一性，因为数据包不会被路由到该链路之外。也就是说，路由器不会转发任何以链路本地地址为源地址或目的地址的数据包。图 4-19 标识了作为单播地址的链路本地地址。

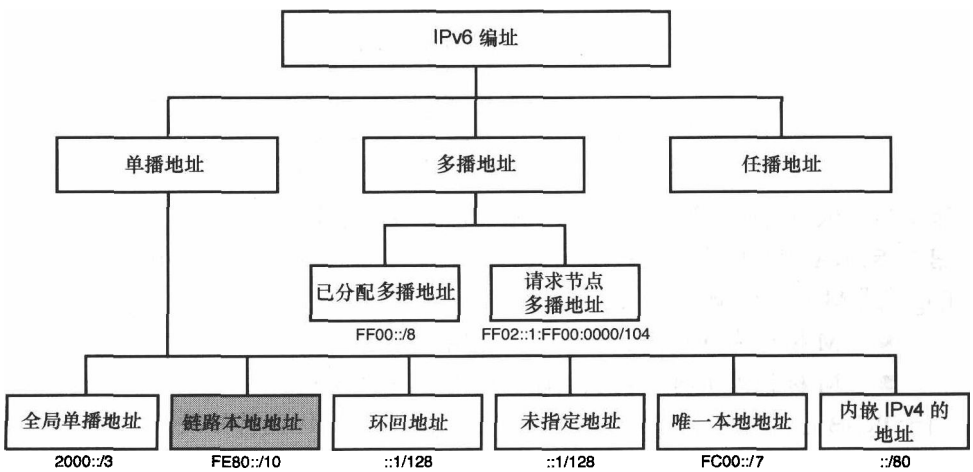



图 4-19 链路本地单播地址

链路本地地址的配置方式有：

- 动态方式，使用 EUI-64；
- 随机生成的接口 ID；
- 静态方式，手工输入链路本地地址。

链路本地地址为 IPv6 提供了一个独一无二的优势。网络设备完全可以自行创建自己的链路本地地址，而无需 DHCPv6 服务器或 RA 消息，这就解决了 IPv4 中的“先有蛋还是先有鸡”的问题，也就是说，“一方面，希望向服务器申请 IP 地址，但另一方面，为了能够与该服务器进行通信，又必须首先有一个 IP 地址。”

图 4-20 显示了链路本地单播地址的格式。链路本地单播地址以 FE80::/10 开头。

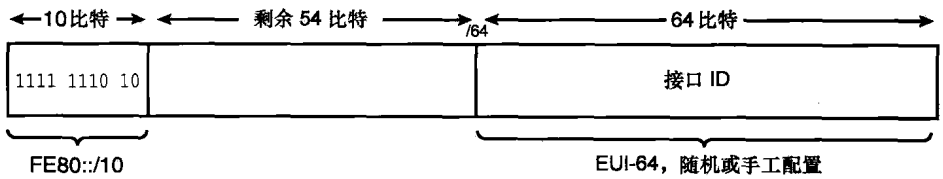



图 4-20　链路本地单播地址格式

从链路本地地址的前缀及前缀长度可以看出，链路本地单播地址的范围是 FE80::/10～FEBF::/10（如表 4-6 所示）。

表 4-6　　链路本地单播地址的范围

| 链路本地单播地址（十六进制） | 第一个十六位组的范围 | 第一个十六位组的范围（二进制） |
|---|---|---|
| FE80::/10 | FE80<br>FEBF | 1111 1110 10 00 0000<br>1111 1110 10 11 1111 |

/10 表示前 10 个比特是有意义的比特（也就是必须匹配该前缀的比特），前缀是 FE80（二进制 1111 1110 10）。因而只要前 10 个比特匹配了，剩余的 54 比特可以是任何值。因此，第一个十六位组的取值范围是 FE80～FEBF。

如前所述，IPv6 链路本地地址用于以下场合：

- 路由器使用链路本地地址作为它们发送的 RA 消息的默认网关；
- 运行路由协议（如用于 IPv6 的 EIGRP 或 OSPFv3）的路由器使用链路本地地址来建立邻接关系；
- IPv6 路由表中的动态路由使用链路本地地址作为下一跳地址。

### 1. 动态链路本地地址：EUI-64

在默认情况下，设备无需 DHCP 服务器或路由器等设备的帮助，就能自动创建自己的链路本地单播地址。如图 4-20 所示，链路本地地址的前缀/前缀长度通常是 FE80::/64。64 比特接口 ID 可以随机生成，也可以使用 EUI-64 格式，Cisco 设备使用 EUI-64 格式。下一节将讨论随机生成的接口 ID。

例 4-9 使用命令 **show ipv6 interface** 显示了路由器 R1 的快速以太网接口 Fast Ethernet 0/0 上的链路本地地址。由于链路本地地址不是在接口上进行手工配置的，因而使用 EUI-64 格式自动创建了该地址。从输出结果可以看出，EUI-64 使用 Fast Ethernet 0/0 的 MAC 地址来生成链路本地地址的接口 ID。

**例 4-9 链路本地单播地址**

```
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  ! Link-local address using EUI-64 format
  IPv6 is enabled, link-local address is FE80::203:6BFF:FEE9:D480
Global unicast address(es):
    2001:DB8:AAAA:1::1, subnet is 2001:DB8:AAAA:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1
    FF02::1:FFE9:D480
<output omitted for brevity>

R1# show interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  ! Ethernet MAC Address
  Hardware is AmdFE, address is 0003.6be9.d480 (bia 0003.6be9.d480)
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
<output omitted for brevity>
```

R1 的快速以太网接口 Fast Ethernet 0/0 的 MAC 地址是 0003.6BE9.D480。使用改进型 EUI-64，在 MAC 地址的 OUI 与设备标识符之间插入 FF-FE，然后将第 7 个比特由 0 反转为 1，从而将第二个十六进制数值由 0 更改为 2，因此接口 ID 被分配的 64 比特值为 02-03-6B-FF-FE-E9-D4-80。将链路本地前缀 FE80::/10 附加到接口 ID 上，即可得到 R1 的 Fast Ethernet 0/0 接口的链路本地地址为 FE80::0203:6BFF:FEE9:D480。

本章在前面已经解释了如何使用 EUI-64 为同一个接口上的全局单播地址创建接口 ID。对比例 4-9 中的链路本地地址与例 4-6 中的全局单播地址，由于这些使用 EUI-64 的地址都用到了相同的 MAC 地址，因而这两个地址的接口 ID 完全相同，只是前缀（前导的 64 比特）有所不同。

### 2. 随机生成的接口 ID

EUI-64 是一种非常方便的从 48 比特 MAC 地址自动创建 64 比特接口 ID 的技术，但是可能会给某些用户带来顾虑：由于 IPv6 地址中包含了用于创建接口 ID 的 48 比特 MAC 地址（用来创建接口 ID），因而能够追踪到真正的设备。为了解决该隐私问题，设备可以使用随机生成的 64 比特接口 ID。

> 注：有关隐私延伸问题的讨论可参考 RFC 5375 “IPv6 Unicast Address Assignment Considerations”和 RFC 4941 “Privacy Extensions for Stateless Address Autoconfiguration in IPv6” 。

设备使用 EUI-64 还是随机生成的接口 ID 取决于操作系统。Cisco 路由器使用 EUI-64，而 Windows 操作系统在 XP 之后都使用随机生成的接口 ID，而之前的 Windows 操作系统则使用 EUI-64。目前的趋势是主机操作系统在默认情况下都随机生成自己的接口 ID。

例 4-10 显示了主机 PC1 和 PC2（是图 4-7 所示拓扑结构中的两台主机）的链路本地地址。这两台 Windows 主机都生成了它们的链路本地地址。由于这些主机运行的 Windows 操作系统都是 XP 之后的版本，因而接口 ID 都是随机生成的，而不是使用 EUI-64 进程。

**例 4-10　主机 PC1 和 PC2 的 IPv6 链路本地地址**

```
PC1> ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2001:db8:aaaa:1::100
   Link-local IPv6 Address . . . . . : fe80::50a5:8a35:a5bb:66e1%11
   Default Gateway . . . . . . . . . : 2001:db8:aaaa:1::1

PC2> ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2001:db8:aaaa:1::200
   Link-local IPv6 Address . . . . . : fe80::1c00:3ea4:74ff:a8cf%8
   Default Gateway . . . . . . . . . : 2001:db8:aaaa:1::1
```

请注意，输出结果中跟在链路本地地址后面的“%n”是 Windows Zone ID，不属于 IPv6 地址。

注：例 4-10 显示的链路本地地址后面还有“%n”，这里的 n 是 Windows Zone ID。该信息与本节讨论的主题关联度不大，如果大家对 Zone ID 感兴趣，可以参考 http://technet.microsoft.com/en-us/library/bb726952.aspx 以及 RFC 4007 “IPv6 Scoped Address Architecture”。

### 3. 静态链路本地地址

对大多数设备（如主机）来说，动态分配链路本地地址都是非常理想的，但缺点是接口 ID 过于冗长，在对网络进行排障或验证过程中难以识别或记忆。由于动态路由协议（如 IPv6 EIGRP 和 OSPFv3）都使用链路本地地址来建立邻接关系和其他消息，因此采取手工方式配置链路本地地址会更易于识别。

配置静态链路本地单播地址的命令如下：

```
Router(config-if)# ipv6 address ipv6-address link-local
```

表 4-7 解释了配置链路本地单播地址的命令。

**表 4-7　　配置链路本地单播地址**

| 命令 | 描述 |
|---|---|
| Router(config)# **interface** *interface-type interface-number* | 指定接口类型和接口号 |
| Router(config-if)# **ipv6 address** *ipv6-address* **link-local** | 指定 IPv6 链路本地地址，需要附加参数 **link-local** |

注：某些 Cisco 文档提到的接口命令是 *ipv6-address/prefix-length* **link-local**，不过不是本案例所用到的 IOS 版本。

图 4-21 显示了为每个接口静态分配链路本地地址的拓扑结构。例 4-11 显示了在路由器 R1、R2 和 R3 上配置这些链路本地地址的配置示例。由于为每台路由器的每个接口都配置了相同的链路本地地址，因而可以很容易地识别出每台路由器上的链路本地地址。R1 所有接口的接口 ID 都是::1，R2 所有接口的接口 ID 都是::2，R3 所有接口的接口 ID 都是::3。请记住，链路本地地址只要在本地链路上唯一即可，因为它们不会被路由到本地链路之外。

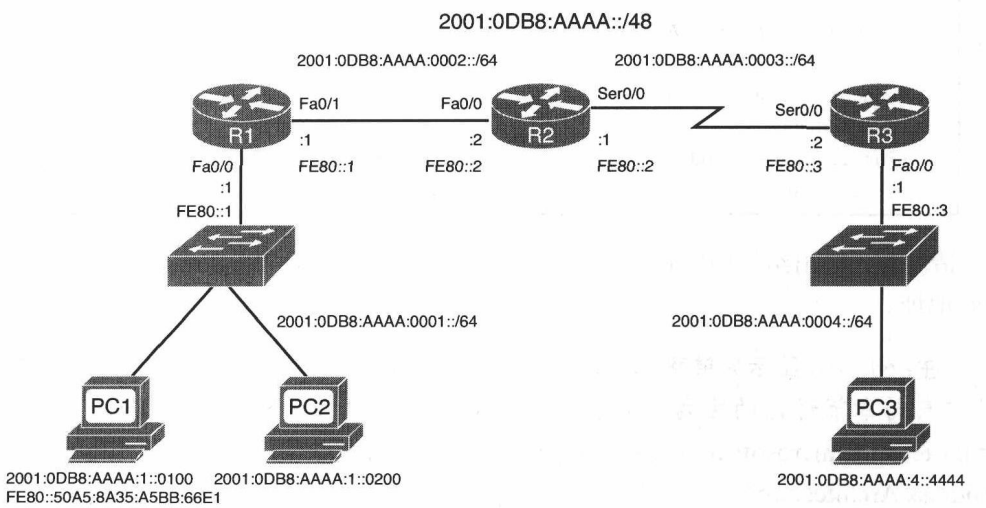



图 4-21　带有静态本地链路地址的 IPv6 拓扑结构

**例 4-11　在路由器 R1、R2 和 R3 上配置链路本地地址**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 address fe80::1 ?
  link-local  Use link-local address

R1(config-if)# ipv6 address fe80::1 link-local
R1(config-if)# exit
R1(config)# interface serial 0/0
R1(config-if)# ipv6 address fe80::1 link-local
```

```
R2(config)# interface fastethernet 0/0
R2(config-if)# ipv6 address fe80::2 link-local
R2(config-if)# exit
R2(config)# interface serial 0/0
R2(config-if)# ipv6 address fe80::2 link-local
```

```
R3(config)# interface fastethernet 0/0
R3(config-if)# ipv6 address fe80::3 link-local
R3(config-if)# exit
R3(config)# interface serial 0/0
R3(config-if)# ipv6 address fe80::3 link-local
```

在例 4-12 中，使用命令 **show ipv6 interface brief** 来验证这些链路本地地址。请注意，对每台路由器来说，必须为每个接口都配置相同的链路本地地址。在本书第 7 章中，给出了 IPv6 路由协议的简单配置方法。

**例 4-12　验证路由器 R1、R2 和 R3 上的链路本地地址**

```
R1# show ipv6 interface brief
! Notice that highlighted link-local unicast addresses are the same
FastEthernet0/0            [up/up]
    FE80::1
    2001:DB8:AAAA:1::1
FastEthernet0/I            [up/up]
    FE80::1
    2001:DB8:AAAA:2::1
R1#
```

```
R2# show ipv6 interface brief
! Notice that highlighted link-local unicast addresses are the same
FastEthernet0/0            [up/up]
    FE80::2
    2001:DB8:AAAA:2::2
```

```
Serial0/0                  [up/up]
    FE80::2
    2001:DB8:AAAA:3::1
R2#
```

```
R3# show ipv6 interface brief
! Notice that highlighted link-local unicast addresses are the same
FastEthernet0/0            [up/up]
    FE80::3
    2001:DB8:AAAA:4::1
Serial0/0                  [up/up]
    FE80::3
    2001:DB8:AAAA:3::2
R3#
```

### 4. 链路本地地址与重复地址检查

在前面讨论全局单播地址时曾经说过，设备通过 DAD（Duplicate Address Detection，重复地址检测）进程来查看链路上是否有其他设备正在使用其将要使用的地址。无论链路本地地址是自动生成的还是手工配置的，设备在使用该链路本地地址之前，都要执行 DAD 以确定该地址在链路上的唯一性（如图 4-22 所示）。

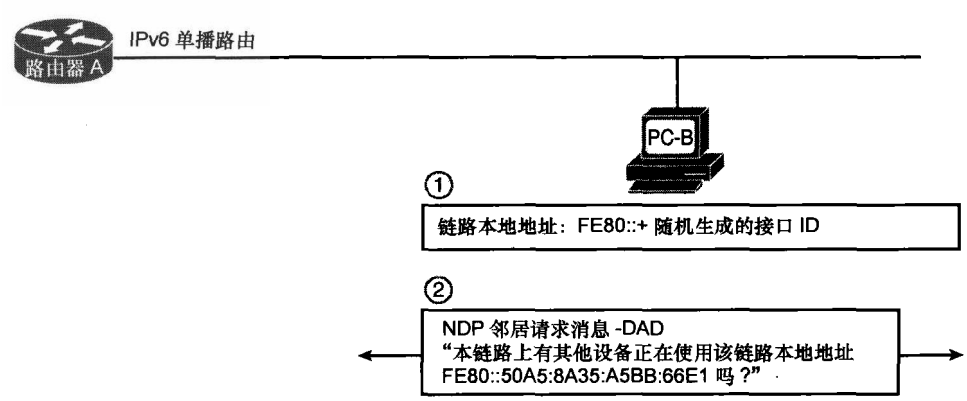



图 4-22 链路本地地址和重复地址检查

例如，如果为某台路由器配置了一个在链路上已经存在的链路本地地址，那么就会收到一条警告消息，表明链路本地地址和接口出现重复地址：

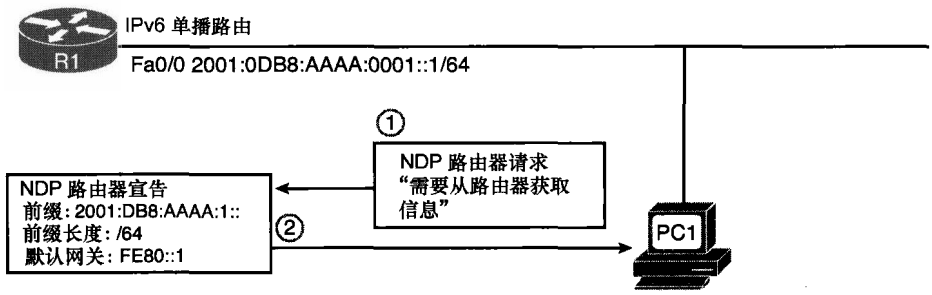
```
*%IPV6-4-DUPLICATE: Duplicate address FE80::3 on Serial0/0
```

虽然接口仍然会接受该重复地址，但接口检测到其将要使用的地址是重复地址时，而会认定该地址不可用。链路本地地址必须在链路上保证其唯一性。

有关 DAD 的详细内容将在第 5 章进行讨论。

### 5. 链路本地地址与默认网关

前面曾经介绍过 DP 或 NDP（Neighbor Discovery Protocol，邻居发现协议）。利用 ND 路由器请求消息和路由器宣告消息，主机能够自动获取 IP 编址信息，如前缀、前缀长度和一个默认网关地址。如图 4-23 所示，拓扑结构中的 PC1 被配置为自动获取其 IP 地址。PC1 向外发送 ND 路由器请求消息，路由器 R1 会响应以路由器宣告消息。由于 PC1 运行的是 Windows Vista，因而 PC1 不使用 EUI-64，而是随机生成其接口 ID，并附加到路由器宣告消息中提供的前缀上。




```
PC1> ipconfig

Windows IP 配置

以太网适配器本地连接：

   连接特定的 DNS 后缀 . :
   IPv6 地址 . . . . . : 2001: db8:aaaa:1:50a5:8a35:a5bb:66e1
   链路本地 IPv6 地址 . . . . . : fe80::50a5:8a35:a5bb:66e1%11
   默认网关 . . . . . : fe80::1% 11
```

图 4-23　Windows ipconfig 以及用作默认网关的链路本地地址

这里需要关注的是默认网关地址。路由器 R 在路由器宣告消息中使用其链路本地地址 FE80::1 作为默认网关，而不是全局单播地址 2001:0DB8:AAAA:0001::1。有关路由器请求消息与路由器宣告消息的详细内容将在第 5 章进行讨论。

### 6. 隔离的链路本地地址

只要在接口上启用了 IPv6，就会自动创建链路本地地址，这是在接口上配置全局单播地址或唯一的本地单播地址或应用命令 **ipv6 enable** 之后的结果。如果有关这些地址的事件都被删除，那么接口的链路本地地址也会被删除。为了保证接口在没有全局单播地址或唯一本地单播地址的情况下，能够拥有链路本地地址，需要按如下方式应用命

令 **ipv6 enable**：

```
Router(config-if)# ipv6 enable
```

例 4-13 解释了该命令是如何在没有单播地址的情况下创建链路本地地址的。在接口上配置了该命令之后，路由器会立即使用 EUI-64 格式创建一个链路本地地址。在例 4-13 中，为路由器的快速以太网接口 Fast Ethernet 0/1 自动创建了链路本地地址 FE80::20C:30FF:FE10:92E1。

**例 4-13 命令 ipv6 enable**

```
Router(config)# interface fastethernet 0/1
Router(config-if)# ipv6 enable
Router(config-if)# end
Router# show ipv6 interface brief
FastEthernet0/1            [up/up]
    FE80::20C:30FF:FE10:92E1
Router#
```

### 4.2.3 环回地址

另一种单播地址是环回地址（如图 4-24 所示）。IPv6 环回地址是除最后一个比特为 1 之外的全 0 比特地址，等同于 IPv4 环回地址 127.0.0.1。

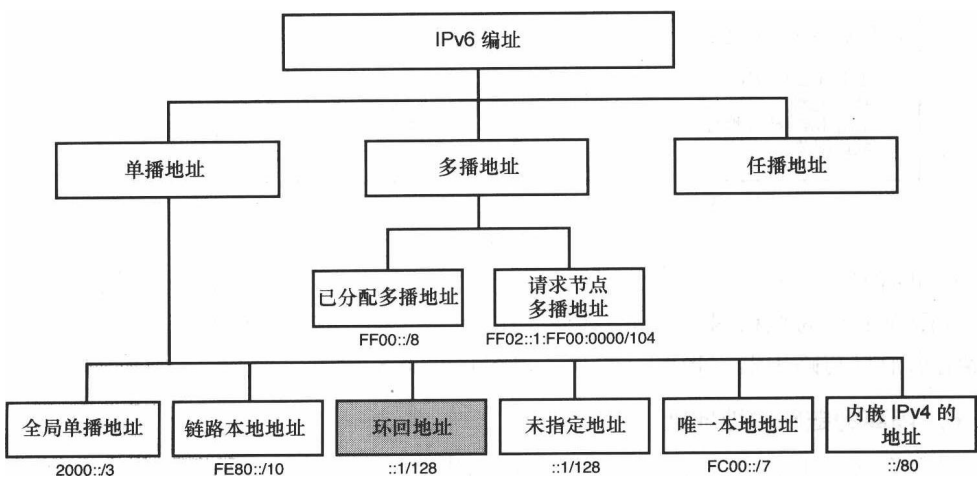



图 4-24 环回地址

表 4-8 给出了不同格式下的 IPv6 环回地址。

表 4-8　　IPv6 环回地址

| 表达格式 | IPv6 环回地址 |
| --- | --- |
| 优选格式 | 0000:0000:0000:0000:0000:0000:0000:0001 |
| 无前导 0 格式 | 0:0:0:0:0:0:0:1 |
| 压缩格式 | ::1 |

节点可以利用环回地址向自己发送 IPv6 数据包，通常用于测试 TCP/IP 协议栈。环回地址等同于 IPv4 中的地址 127.0.0.0/8。环回地址的特性如下：

- 不能将环回地址分配给物理接口；
- 如果数据包不是发送到设备之外，那么环回地址只能作为源地址；
- 如果数据包不是发送到设备之外，那么环回地址只能作为目的地址；
- 路由器不会转发目的地址是环回地址的数据包；
- 设备必须丢弃接口上收到的目的地址为环回地址的数据包。

### 4.2.4　未指定地址

未指定地址是全 0 地址，不能分配给接口。未指定地址被用作源地址以表示接口无地址，图 4-25 列出了另一种单播地址——未指定地址。

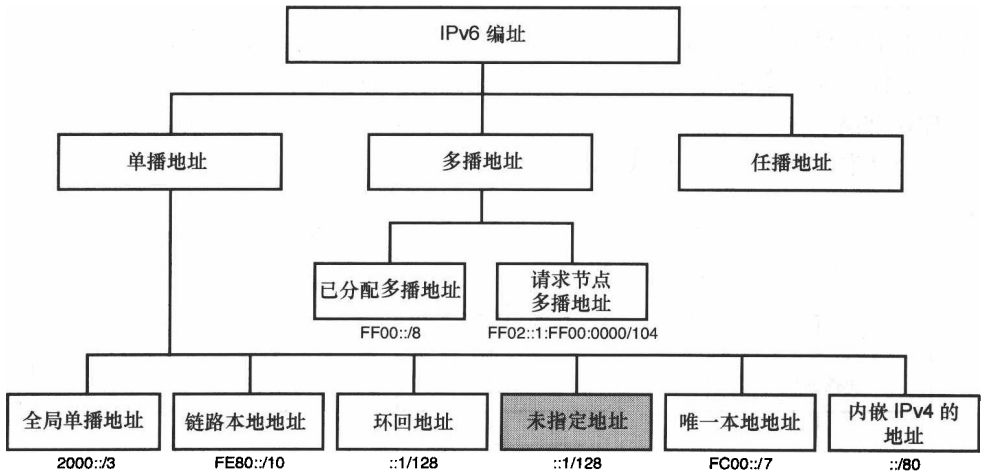



图 4-25　未指定地址

如前所述，DAD 进程是设备向全部节点多播地址（类似于 IPv4 的广播地址）发送消息，也就是向链路上的所有设备发送消息，以确认自己将要使用的地址没有被其他设备所使用。例如，在图 4-22 中，PC1 在使用其最新创建的链路本地地址之前需要发起 DAD 进程。PC1 的源 IPv6 地址的邻居请求（Neighbor Solicitation）消息是一个未指定地址，这是因为此时的 PC1 还没有有效的 IPv6 地址可用。

表 4-9 给出了不同格式下的 IPv6 未指定地址的。

表 4-9 IPv6 未指定地址

| 表达格式 | IPv6 未指定地址 |
|---|---|
| 优选格式 | 0000:0000:0000:0000:0000:0000:0000:0000 |
| 无前导 0 格式 | 0:0:0:0:0:0:0:0 |
| 压缩格式 | ::0 |

未指定地址的特性如下：

- 不能将未指定地址分配给物理接口；
- 源地址为未指定地址表示该接口无地址；
- 未指定地址不能被用作目的地址；
- 路由器不会转发源地址为未指定地址的数据包。

### 4.2.5 唯一本地地址

唯一本地地址是由最初的 IPv6 规范为站点本地地址（site-local address）分配的地址空间，类似于 RFC 1918“Private Address Space in IPv4”分配的私有 IPv4 地址空间。站点本地地址定义在 RFC 3513，前缀为 FEC0::/10（大家可能会在早期文档中看见该前缀）。但问题是术语站点（site）存在歧义，因为没有人能确切地就站点的含义达成一致。另一个问题是不能保证同一个组织内的两个站点会最终使用相同的或重叠的站点本地地址，从而破坏了 IPv6 的本意以及为此专门分配的额外地址空间。因此，站点本地地址已经被废除，取而代之的是唯一本地地址。图 4-26 列出了作为单播地址的唯一本地地址。

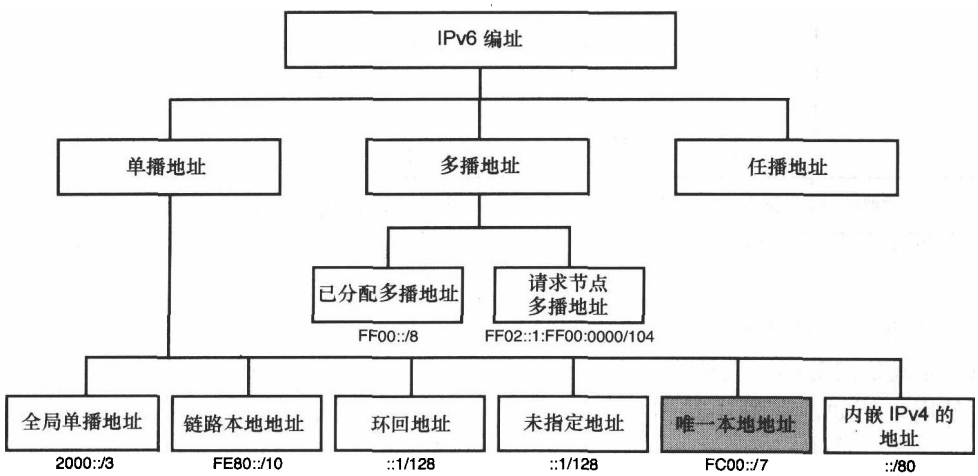



图 4-26 唯一本地地址

ULA（Unique local address，唯一本地地址）定义在 RFC 4193“Unique Local IPv6 Unicast Address”中。图 4-27 解释了唯一本地单播地址的格式。

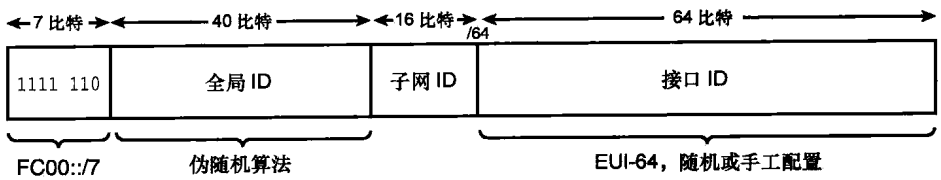



图 4-27　唯一本地单播地址

唯一本地地址的前缀是 FC00::/7，地址范围是 FC00::/7～FDFF::/7（如表 4-10 所示）。

表 4-10　　唯一本地单播地址的范围

| 唯一本地单播地址（十六进制） | 第一个十六位组的范围 | 第一个十六位组的范围（二进制） |
|---|---|---|
| FC00::/7 | FC00<br>FDFF | 1111 110 0 0000 0000<br>1111 110 1 1111 1111 |

唯一本地地址也被称为本地 IPv6 地址（local IPv6 address）。这类地址应该具备全局唯一性，但不应该在全球互联网上进行路由，通常应用于范围有限的区域（如站点内部）或者在数量有限的站点之间进行路由。

唯一本地地址的特性如下：

- 拥有全局唯一的前缀，或者在很大可能上是全局唯一的；
- 允许站点在不出现地址冲突或者重新编址的情况下，能够进行整合或私下互连；
- 能够独立于互联网服务提供商，可以在无任何互联网连接的站点内使用；
- 即使因路由或 DNS 等原因不慎泄露到站点之外，也不会与其他地址相冲突；
- 可以像全局单播地址那样进行使用。

所以很有趣，IPv6 有全局唯一的私有地址。有了唯一本地地址，今后在整合两个使用 ULA 的站点或者被不慎泄露到互联网上时，也不会产生任何冲突。关键在于没有集中式管理机构的情况下仍能保证全局 ID（Global ID）的唯一性。RFC 4193 定义了一个进程，使用伪随机算法生成本地分配的全局 ID。请注意，所有站点都要使用相同的算法来生成全局 ID，以确保最大程度上的唯一性。

实际使用的算法已经超出了本书写作范围，大家可以参考 RFC 4193“Sample Code for Pseudo-Random Global ID Algorithm”。

3.2.2. Sample Code for Pseudo-Random Global ID Algorithm

下面描述的算法主要用于本地分配的全局 ID，每种情况下生成的全局 ID 都会被用在 Section 3.2 定义的合适前缀中。

1. 获得 64 比特 NTP 格式[NTP]的当前时间。

2. 从运行该算法的系统获得 EUI-64 标识符。如果没有 EUI-64，可以按照

[ADDARCH]规定，从 48 比特 MAC 地址创建标识符。如果无法获得或无法创建 EUI-64，那么就应该使用合适的唯一标识符（对节点具有本地意义），如系统序列号。

3. 将当前时间与系统特定的标识符组合在一起，以创建密钥。

4. 按照[FIPS, SHA1]规定计算密钥的 SHA-1 摘要，结果值为 160 比特。

5. 使用最低阶 40 比特作为全局 ID。

6. 将 FC00::/7（L 比特置 1）和 40 比特全局 ID 组合在一起，即可创建本地 IPv6 地址前缀。

该算法能够得到唯一的全局 ID，可用于创建本地分配的本地 IPv6 地址前缀。大家可以在网站 www.sixxs.net/tools/grh/ula 生成并注册自己的 ULA 空间。

### 4.2.6 内嵌 IPv4 的地址

最后一种单播地址就是内嵌 IPv4 的地址（如图 4-28 所示）。内嵌 IPv4 的地址是用来帮助从 IPv4 迁移到 IPv6 的 IPv6 地址。内嵌 IPv4 的地址在低阶 32 比特中承载 IPv4 地址，这类地址表示在 IPv6 地址中内嵌了一个 IPv4 地址。RFC 4291 定义了两种内嵌 IPv4 的地址：

- 兼容 IPv4 的 IPv6 地址（已废除）；
- 映射 IPv4 的 IPv6 地址。

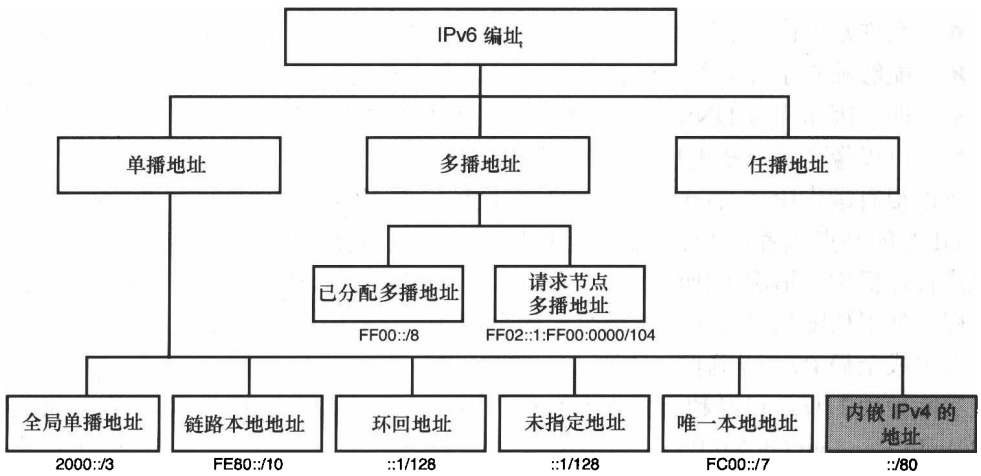



图 4-28 内嵌 IPv4 的地址

可以利用隧道等技术为 IPv6 孤岛内的设备经纯 IPv4-noly 网络进行通信。为了支持该兼容性，可以将 IPv4 地址内嵌到 IPv6 地址中。做法很简单，由于 128 比特的 IPv6 地址拥有足有的空间容纳 32 比特的 IPv4 地址，因而 IPv6 直接将 IPv4 地址放在 IPv6 地址的后端，并在前端进行填充。由于 IPv4 包和 IPv6 包是不兼容的，因而 NAT-PT

和 NAT64 等功能特性需要在两个地址族之间进行转换。相关内容将在第 10 章进行详细讨论。

### 1. 兼容 IPv4 的 IPv6 地址

兼容 IPv4 的 IPv6 地址（IPv4-compatible IPv6 address）用于同时支持 IPv4 和 IPv6 的双栈设备。如图 4-29 所示，前 96 个比特均被置为 0，其中包含一个 16 比特段，用来区分映射 IPv4 的 IPv6 地址（IPv4-mapped IPv6 address），最后 32 比特是以点分十进制方式表示的 IPv4 地址，因此表达形式是前 96 比特采用十六进制，后面包含 IPv4 地址的 32 比特采用点分十进制形式。

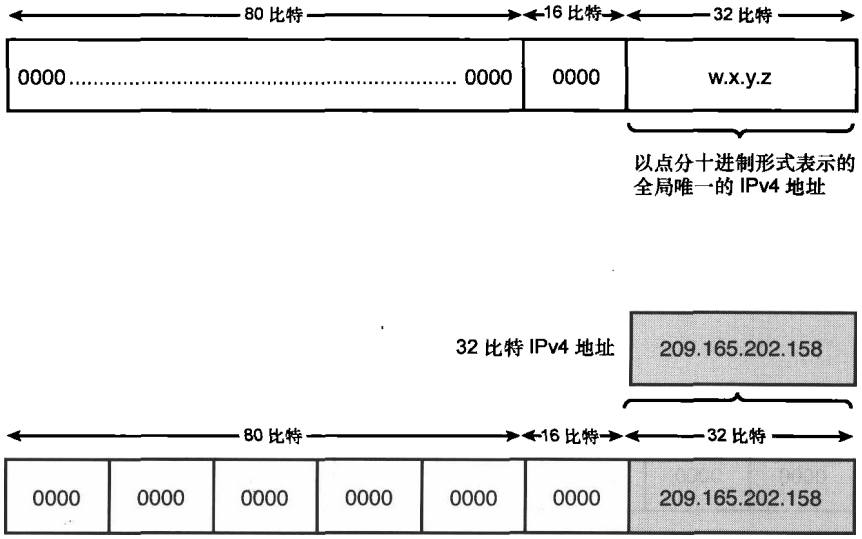



图 4-29　兼容 IPv4 的 IPv6 地址（已废除）

表 4-11 以 IPv4 地址 209.165.202.158 为例，给出了兼容 IPv4 的 IPv6 地址的不同表达格式。

**表 4-11　　兼容 IPv4 的 IPv6 地址**

| 表达格式 | 兼容 IPv4 的 IPv6 地址 |
|---|---|
| 优选格式 | 0000:0000:0000:0000:0000:0000:0000:209.165.202.158 |
| 无前导 0 格式 | 0:0:0:0:0:0:0:209.165.202.158 |
| 压缩格式 | ::209.165.202.158 |

“兼容 IPv4 的 IPv6 地址”中用到的 IPv4 地址必须是全局唯一的 IPv4 单播地址。不过大家很少使用“兼容 IPv4 的 IPv6 地址”，该地址已被废除，当前的 IPv6 过渡机制都不再使用该地址类型。

### 2. 映射 IPv4 的 IPv6 地址

映射 IPv4 的 IPv6 地址与兼容 IPv4 的 IPv6 地址类似。映射 IPv4 的 IPv6 地址用来表示 IPv4-only 设备的地址，IPv6 设备可以利用该地址向纯 IPv4 设备发送数据包。

映射 IPv4 的 IPv6 地址与兼容 IPv4 的 IPv6 地址几乎完全相同，唯一的区别就是 32 比特 IPv4 地址前面的 16 比特段为 1。图 4-30 显示了映射 IPv4 的 IPv6 地址的结构，此处的 IPv4 地址不需要具有全局唯一性。

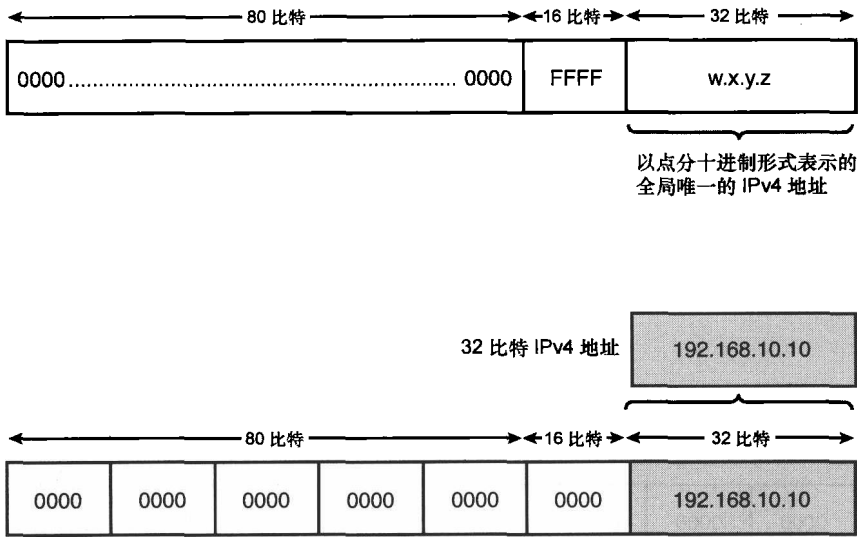



图 4-30 映射 IPv4 的 IPv6 地址

表 4-12 以 IPv4 地址 192.168.10.10 为例，给出了映射 IPv4 的 IPv6 地址的不同表达格式。

表 4-12 映射 IPv4 的 IPv6 地址

| 表达格式 | 映射 IPv4 的 IPv6 地址 |
|---|---|
| 优选格式 | 0000:0000:0000:0000:0000:0000:FFFF:192.168.10.10 |
| 无前导 0 格式 | 0:0:0:0:0:0:FFFF:192.168.10.10 |
| 压缩格式 | ::FFFF:192.168.10.10 |

对于在主机和路由器上创建 IPv4 隧道，以通过 IPv4 网络传送 IPv6 数据包的过渡机制来说，需要使用兼容 IPv4 的 IPv6 地址和映射 IPv4 的 IPv6 地址。有关这些地址以

及使用这些地址的过渡机制将在第 10 章进行详细讨论。

> 注：第 10 章将讨论 IPv4 与 IPv6 的过渡和共存策略，以及 6to4 和 ISATAP 等较为常见的隧道机制。

## 4.3　多播地址

接下来将要讨论的地址类型就是多播地址（如图 4-31 所示）。单播地址是将单个数据包发送给单个目的端（一对一），而多播则是一种设备将单个数据包同时发送给多个目的端（一对多）的技术。虽然多个目的端可以是同一台设备上的多个接口，但通常都是不同设备。

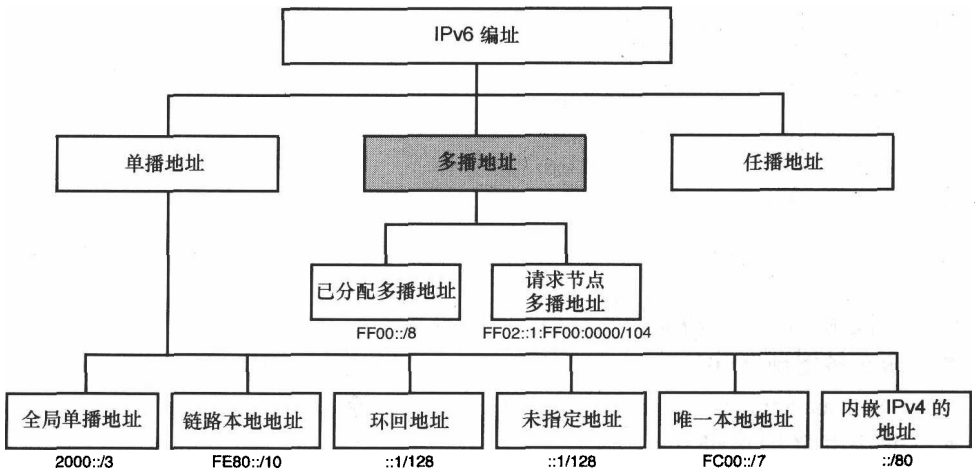



图 4-31　多播地址

IPv6 多播地址定义了一组设备，被称为多播组（multicast group）。与 IPv4 多播地址 224.0.0.0/4 相似，发送给多播组的数据包必须有一个单播源地址。多播地址不能用作源地址。与 IPv4 不同的是，IPv6 没有广播地址。

IPv6 多播地址的前缀是 FF00::/8。表 4-13 给出了多播地址的不同表达格式。

**表 4-13　　IPv6 多播地址**

| 表达格式 | 映射 IPv4 的 IPv6 地址 |
|---|---|
| 优选格式 | FF00:0000:0000:0000:0000:0000:0000:0000/8 |
| 无前导 0 格式 | FF00:0:0:0:0:0:0:0/8 |
| 压缩格式 | FF00::/8 |

图 4-32 给出了 IPv6 多播地址的结构。前 8 比特都是 1（FF），然后是 4 比特标志（Flag）和 4 比特范围（Scope），接下来的 112 比特表示组 ID（Group ID）。

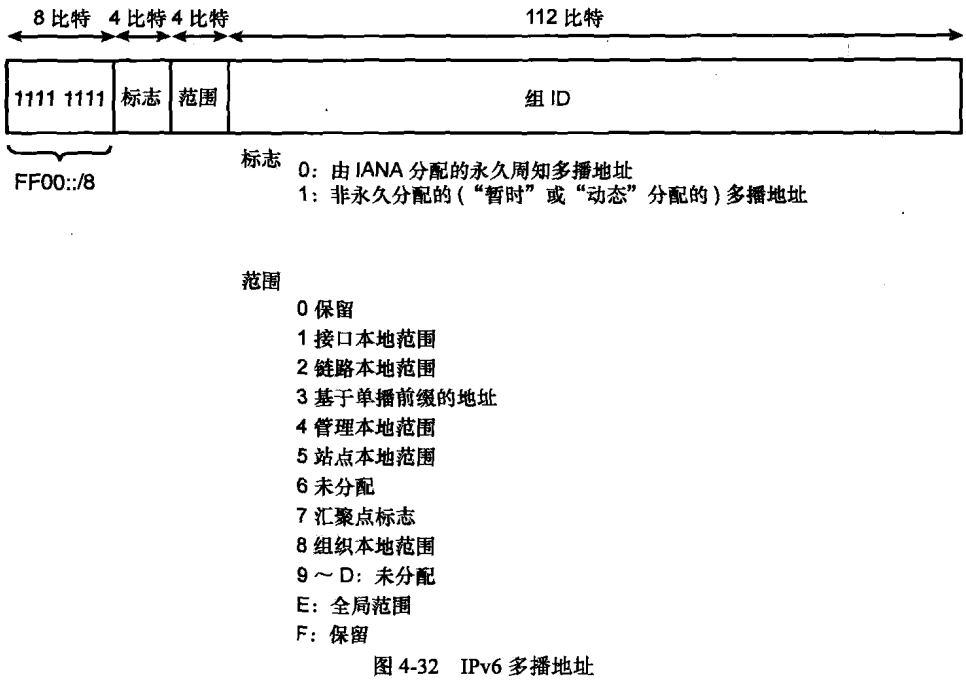



图 4-32 IPv6 多播地址

标志字段用于表示多播地址的类型，多播地址有两类。

- **永久多播地址（0）**：这类多播地址是由 IANA 分配的周知多播地址（well-known multicast address），相关内容将在下一节讨论。
- **非永久多播地址（1）**：这类多播地址是“暂时”或“动态”分配的多播地址。

范围（Scope）是一个 4 比特字段，用于定义多播包的范围，其可能的取值如下所示。

- 0：保留。
- 1：接口本地范围（Interface-Local scope）。
- 2：链路本地范围（Link-Local scope）。
- 3：基于单播前缀的地址（Unicast-Prefix-based address）。
- 4：管理本地范围（Admin-Local scope）。
- 5：站点本地范围（Site-Local scope）。
- 6：未分配。
- 7：汇聚点标记（Rendezvous Point flag）。
- 8：组织本地范围（Organization-Local scope）。

- 9：未分配。

- A：未分配。
- B：未分配。
- C：未分配。
- D：未分配。
- E：全局范围（Global scope）。
- F：保留。

RFC 4007“IPv6 Scoped Address Architecture”规定了不同范围的 IPv6 地址的特性、期望行为以及使用方式。图 4-33 以图形化方式表示了这些范围。利用范围字段，设备可以定义多播包的范围。路由器能够即刻确定在多大范围内传播多播包，因此可以避免将流量发送到目的区域之外，从而大大提高的发送效率。

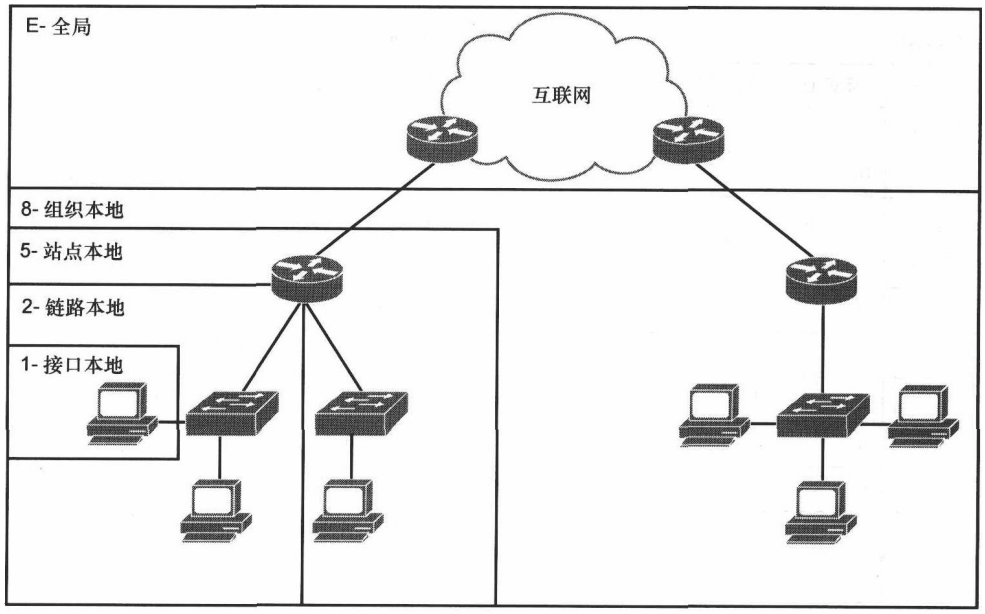



图 4-33　多播范围

### 4.3.1　已分配的多播地址

RFC 2375“IPv6 Multicast Address Assignments”定义了最初分配的 IPv6 多播地址，都拥有永久分配的全局 ID。也就是说，这些保留的多播地址被用于预定义的设备组，已分配的多播地址的前缀是 FF00::/8（如图 4-34 所示）。

表 4-14 给出了一些已分配的或周知的多播地址的格式以及示例。

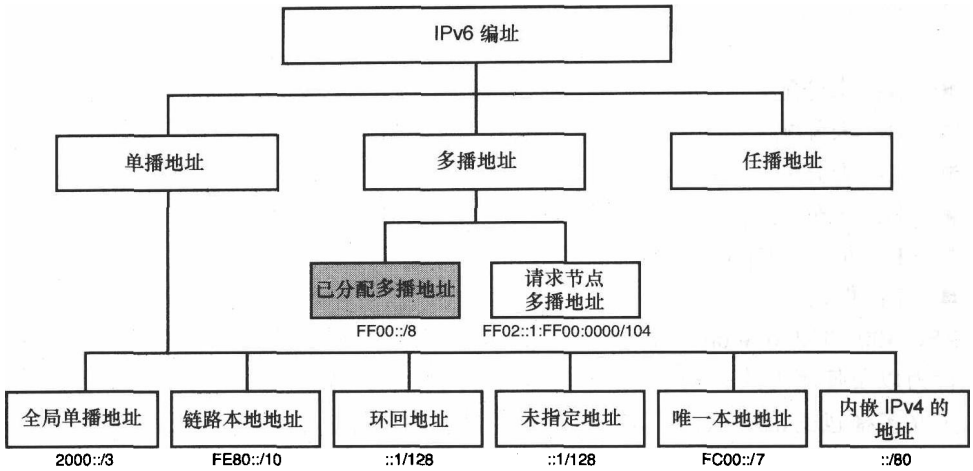



图 4-34 已分配的多播地址

表 4-14 已分配的多播地址

| /8 前缀 FF | 标记 0 | 范围（0～F） | 预定义的组 ID | 压缩格式 | 描述 |
|---|---|---|---|---|---|
| 接口本地范围 | | | | | |
| FF | 0 | 1 | 0:0:0:0:0:0:1 | FF01::1 | 全部节点（all-nodes） |
| FF | 0 | 1 | 0:0:0:0:0:0:2 | FF01::2 | 全部路由器（all-routers） |
| 链路本地范围 | | | | | |
| FF | 0 | 2 | 0:0:0:0:0:0:1 | FF02::1 | 全部节点 |
| FF | 0 | 2 | 0:0:0:0:0:0:2 | FF02::2 | 全部路由器 |
| FF | 0 | 2 | 0:0:0:0:0:0:5 | FF02::5 | OSPF 路由器 |
| FF | 0 | 2 | 0:0:0:0:0:0:6 | FF02::6 | OSPF 指派路由器 |
| FF | 0 | 2 | 0:0:0:0:0:0:9 | FF02::9 | RIP 路由器 |
| FF | 0 | 2 | 0:0:0:0:0:0:A | FF02::A | EIGRP 路由器 |
| FF | 0 | 2 | 0:0:0:0:0:1:2 | FF02::1:2 | 全部 DHCP 代理 |
| 站点本地范围 | | | | | |
| FF | 0 | 5 | 0:0:0:0:0:0:2 | FF05::2 | 全部路由器 |
| FF | 0 | 5 | 0:0:0:0:0:1:3 | FF05::1:3 | 全部 DHCP 服务器 |

从表 4-14 可以看出，相同的组 ID 可以有不同范围。根据不同的范围，发送到全部路由器组 ID 为 0:0:0:0:0:0:2 的数据包可能会被限制在单条链路范围（FF02::2）或整个站点范围（FF05::2）。

已分配的多播地址用于各种特定协议中，如 NDP 和用于 IPv6 的 EIGRP，相关内

容将在后续章节进行讨论。利用命令 **show ipv6 interface fastethernet 0/0** 即可检查路由器 R1 是哪些多播组的成员。例 4-14 显示了路由器 R1 的快速以太网接口 Ethernet 0/0 是 4 个多播组的成员。路由器 R1 将侦听并处理目的地址是下述已分配多播地址的数据包。

- **FF02::1**：该链路的全部节点（all-nodes）多播组。
- **FF02::2**：该链路的全部路由器（all-routers）多播组。
- **FF02::1:FF00:1**：这是路由器 R1 在该接口上的全局单播地址的请求节点（solicited-node）多播地址。有关请求节点多播地址的内容将在下一节讨论。
- **FF02::1:FFE9:D480**：这是路由器 R1 在该接口上的链路本地地址的请求节点多播地址。

**例 4-14　R1 的多播组**

```
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::203:6BFF:FEE9:D480
  Global unicast address(es):
    2001:DB8:AAAA:1::1, subnet is 2001:DB8:AAAA:1::/64
  Joined group address(es):  ! Multicast Groups
    FF02::1                  ! All-nodes on this link
    FF02::2                  ! All-routers on this link
    FF02::1:FF00:1           ! Solicited-node multicast address
                             ! for Global Unicast Address
    FF02::1:FFE9:D480        ! Solicited-node multicast address for
                             ! Link-local Address
<output omitted for brevity>
```

注：链路本地单播地址与链路本地范围的多播地址很容易被混淆。链路本地单播地址用于标识一台设备或一个接口，并且仅在链路上唯一。目的地址为链路本地单播地址的 IPv6 数据包仅发送给该链路上的单台设备，而目的地址为链路本地范围的多播地址的 IPv6 数据包则发送给该链路上的一台或多台设备。

### 4.3.2　请求节点多播地址

除了分配给接口的各种单播地址之外，每台设备还都有一个被称为请求节点多播地址的特殊多播地址（如图 4-35 所示）。这类多播地址是利用设备单播地址的特定映射以及请求节点多播前缀 FF02:0:0:0:0:1:FF00::/104 自动创建而成的。

与 IPv4 不同，IPv6 没有广播地址。对 IPv4 来说，当设备希望到达某台设备（拥有

目的 IPv4 地址）时，会利用 ARP 进程向网络上的所有设备发送二层广播包。虽然 IPv6 中的全部节点多播地址在本质上也完成了相同的功能，但其目的是让 IPv6 成为更有效的协议。在只有一台设备需要对 ARP 请求进行回应的情况下，为何要让网络中的所有设备都处理该 ARP 请求呢？

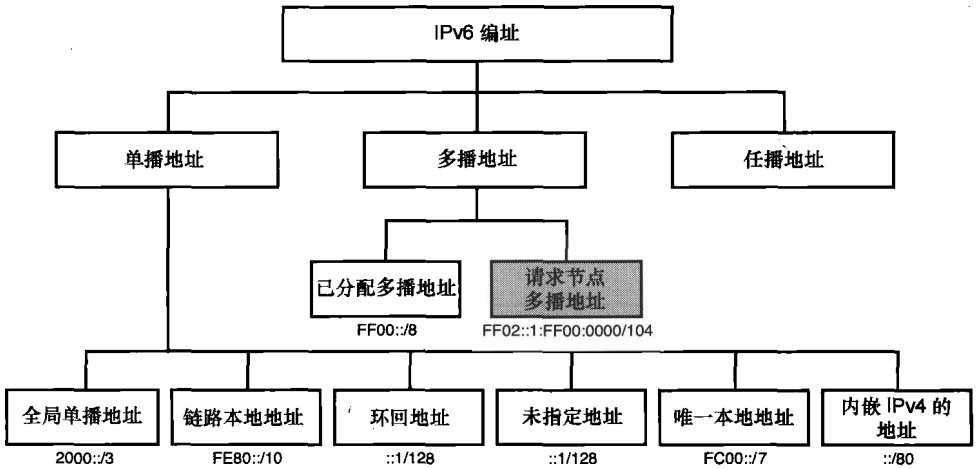



图 4-35 请求节点多播地址

IPv6 的请求节点多播地址提供了一种更为有效的解决方案。请求节点多播地址能够到达链路上的每台设备，但不需要绝大多数设备都处理数据包的内容。如图 4-36 所示，请求节点多播地址通常被用来实现以下两种基本的 IPv6 机制（都是 NDP 的一部分）。

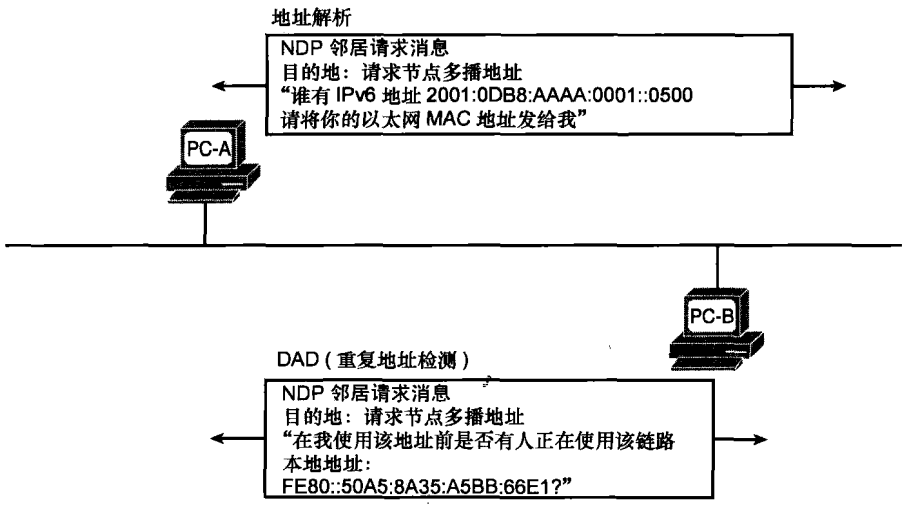



图 4-36 请求节点多播地址的使用与地址解析和 DAD

- **地址解析**：等同于 IPv4 中的 ARP，IPv6 设备会向请求节点多播地址发送邻居请求消息，以学习同一链路上设备的链路层（通常是以太网）地址。设备除了要知道该链路上目的端的 IPv6 地址之外，还要知道其数据链路层（以太网）地址。
- **DAD**：DAD 允许设备验证其通过 SLAAC 创建的单播（或任播）地址在链路上的唯一性。设备会向自己的请求节点多播地址发送邻居请求消息，以确定链路上是否还有其他设备也在使用该地址。

有关 ND、DAD 以及地址解析的详细内容将在第 5 章进行讨论。

如表 4-15 所示，IPv6 请求节点多播地址拥有前缀 FF02:0:0:0:0:1:FF00::/104。图 4-37 显示了请求节点多播地址的结构。

**表 4-15　IPv6 请求节点多播地址**

| 表达格式 | 映射 IPv4 的 IPv6 地址 |
|---|---|
| 优选格式 | FF02:0000:0000:0000:0000:0001:FF00::/104 |
| 压缩格式 | FF02:0:0:0:0:1:FF00::/104 |

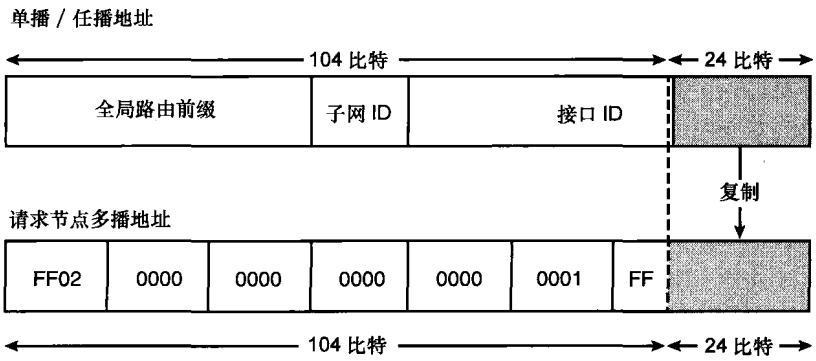



图 4-37　请求节点多播地址

如前所述，请求节点多播地址是为设备上的每个单播地址自动创建的。将请求节点多播前缀 FF02:0:0:0:0:1:FF00::/104 附加到单播地址的低阶 24 比特上即可生成请求节点多播地址（如图 4-37 所示）。

例 4-14 给出了命令 **show ipv6 interface** 的输出结果。路由器 R1 有两个请求节点多播地址：一个用于全局单播地址，另一个用于链路本地单播地址。表 4-16 不但显示了这两个请求节点多播地址通过各自单播地址的低阶 24 比特（以粗体表示）的创建过程，而且还包含了为 PC1 和 PC2 的全局单播地址和链路本地单播地址生成的请求节点多播地址。

表 4-16 从单播地址生成请求节点多播地址

| 单播地址 | | 请求节点多播地址 |
|---|---|---|
| 路由器 R1 | | |
| 全局单播地址 | 2001:DB8:AAAA:1:: 1 | FF02::1:FF00: 1 |
| 链路本地地址 | FE80::203:6BFF:FE E9:D480 | FF02::1:FF E9:D480 |
| PC1 | | |
| 全局单播地址 | 2001:DB8:AAAA:1:: 100 | FF02::1:FF00: 100 |
| 链路本地地址 | FE80::50A5:8A35:A5 BB:66E1 | FF02::1:FF BB:66E1 |
| PC2 | | |
| 全局单播地址 | 2001:DB8:AAAA:1:: 200 | FF02::1:FF00: 200 |
| 链路本地地址 | FE80::1C00:3EA4:74 FF:A8CF | FF02::1:FF FF:A8CF |

设备不仅要处理目的地址为其单播地址的数据包，而且还要处理目的地址为这些单播地址相对应的请求节点多播地址的数据包，这是什么意思呢？正如将在第 5 章所描述的那样，NDP 主要利用这些请求节点多播地址进行地址解析和 DAD。第 5 章将会说明主机启动后的处理过程以及这些地址和进程是如何工作的。

## 4.4 任播地址

本章将要讨论的最后一种 IPv6 地址是任播地址（如图 4-38 所示）。IPv6 任播地址是可以分配给多个接口（通常是不同设备）的地址。也就是说，多台设备可以拥有相同的任播地址，发送给任播地址的数据包会根据路由器的路由表被路由到“最近”的拥有该地址的接口。

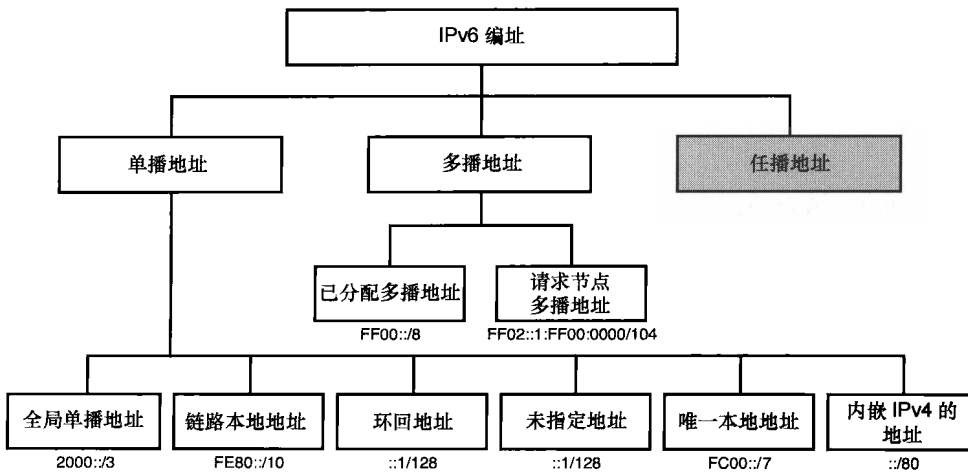



图 4-38 任播地址

在最初的 RFC 1546“Host Anycasting Service”中，任播地址可用于 IPv4 和 IPv6，主要为 DNS 和 HTTP 提供服务，不过业界从来没有实施过该设计方案。

IPv6 任播地址没有特定的前缀。IPv6 任播地址与全局单播地址共用相同的地址空间。每台参与设备都会被配置相同的任播地址（如图 4-39 所示）。

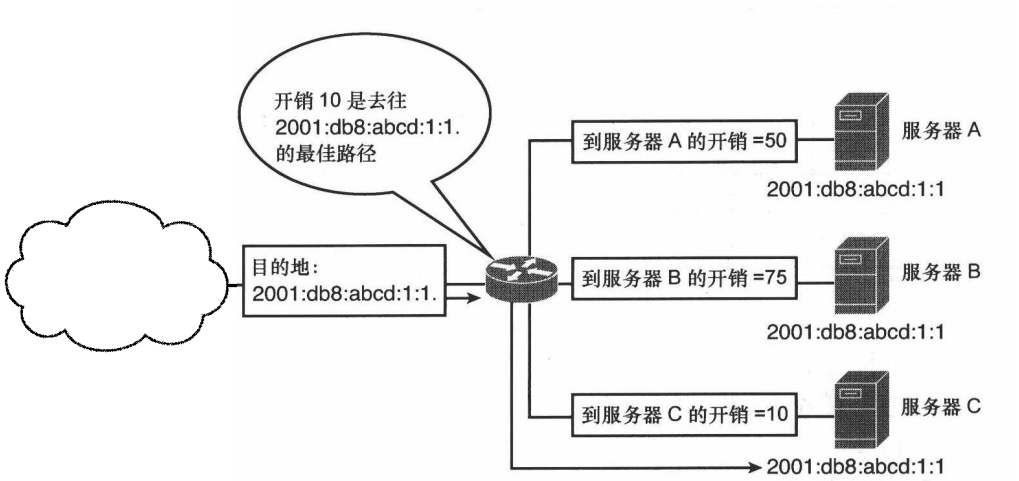



图 4-39　任播编址示例

RFC 4291 和 RFC 2526 定义了一些保留的任播地址格式，如子网路由器（subnet-router）任播地址。IPv6 任播编址目前还处于试验阶段，其详细内容已超出了本书写作范围。

## 4.5　本章小结

如图 4-40 所示，本章介绍了三类 IPv6 地址：

- 单播地址；
- 任播地址；
- 多播地址。

本章讨论的各类地址的主要特性如下。

- **单播地址**：单播地址可以唯一地表示 IPv6 设备上的接口。源 IPv6 地址必须是单播地址；
  - **全局单播地址**：全局单播地址也被称为可聚合全局单播地址。这些地址是在 IPv6 互联网上可路由且可达的地址，等同于 IPv4 公有地址。目前 IANA 已分配的全局单播地址以二进制 001 开头，即前缀为 2000::/3。

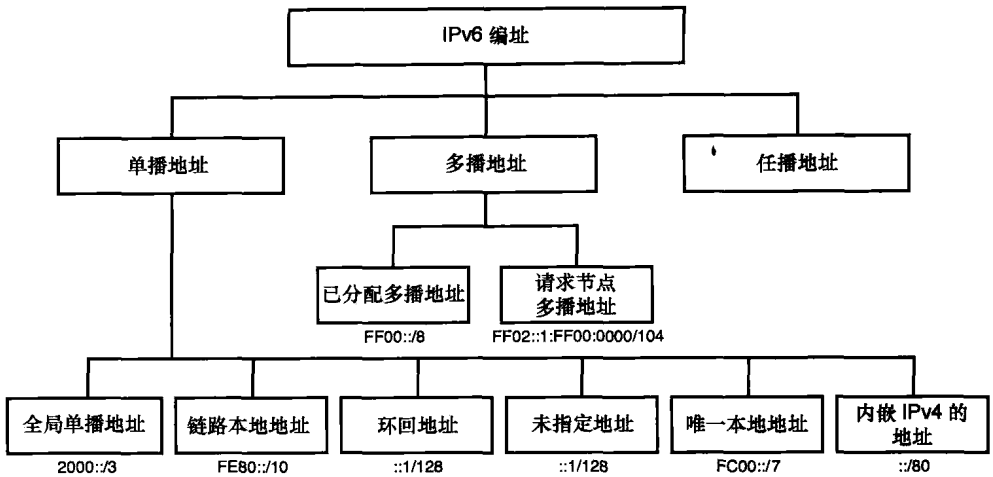



图 4-40 IPv6 地址类型

全局单播地址可以采取以下手工配置方式。

* **静态配置方式**：静态配置类似于静态 IPv4 地址的配置，会在接口上配置 IPv6 地址和前缀长度。

* **EUI-64 配置方式**：该配置方式只要配置 IPv6 地址前缀和前缀长度，接口 ID 是自动创建的。改进型 EUI-64 是由 24 比特 OUI、求反后的 U/L 比特、16 比特值 FFFE 以及 24 比特设备标识符组成的。

* **无编号 IPv6 配置方式**：IPv6 中的无编号 IP 配置方式与 IPv4 相同，是在同一台设备上使用其他接口的 IP 地址。

全局单播地址也可以采取以下动态配置方式。

* **SLAAC**：该配置方式下的接口 ID 使用 EUI-64 格式，而地址前缀和前缀长度则是通过来自路由器的 ND 路由器宣告消息确定的。

* **DHCPv6（状态化）**：DHCPv6 类似于 IPv4 中的 DHCP。利用 DHCPv6 服务器提供的服务，设备能够自动获得其编址信息。

- **链路本地地址**：链路本地地址是应用于单条链路的单播地址。由于拥有链路本地地址的数据包不会被路由到链路之外，因而只要保证其在链路上的唯一性即可。链路本地地址的配置方式有下面两种。
  * **动态配置方式**：使用 EUI-64（或随机生成的接口 ID）。
  * **静态配置方式**：手工输入链路本地地址。
- **环回地址**：环回地址是除最后一个比特为 1 之外的全 0 地址，等同于 IPv4 的环回地址 127.0.0.1。
- **未指定地址**：未指定地址是全 0 地址，不能分配给任何接口。未指定地址作为源地址时表示该接口无地址。

- **唯一本地地址**：唯一本地地址类似于 IPv4 中的 RFC 1918 私有地址空间。RFC 3513 定义了站点本地地址并分配了前缀 FEC0::/10，但是后来被废除了。唯一本地地址通常也是全局唯一的地址，但不能在全球互联网上进行路由，通常应用于非常有限的区域，如站点内部或者在数量有限的站点之间进行路由。
- **内嵌 IPv4 的地址**：内嵌 IPv4 的 IPv6 地址用于帮助从 IPv4 迁移到 IPv6。内嵌 IPv4 的地址在低阶 32 比特中承载 IPv4 地址。这类地址表示在 IPv6 地址中内嵌了一个 IPv4 地址。RFC 4291 定义了两种内嵌 IPv4 的地址：
  * 兼容 IPv4 的 IPv6 地址（已废除）；
  * 映射 IPv4 的 IPv6 地址。

- **多播地址**：多播是一种设备将单个数据包同时发送给多个目的端的技术。
  - **已分配的多播地址**：这些保留的多播地址用于预定义的设备组（如全部节点和全部路由器）。多播地址中的范围字段允许设备定义多播包的范围，并允许路由器立即确定传播范围。由于避免了将流量发送到期望区域之外，因而大大提高了发送效率。
  - **请求节点多播地址**：分配给接口的每个单播地址都会有一个被称为请求节点多播地址的特殊多播地址。这类多播地址是利用特定映射自动创建的，也就是将请求节点多播地址 FF02:0:0:0:0:1:FF00::/104 与单播地址的最后 24 比特组合在一起。IPv6 的请求节点多播地址可以让数据包到达链路上的每台设备，而无需这些设备都处理该数据包的内容。
- **任播地址**：IPv6 任播地址是可以分配多个接口（通常是不同设备）的地址。也就是说，多台设备可以拥有相同的任播地址，发送给任播地址的数据包会根据路由器的路由表被路由到“最近”的拥有该地址的接口。

## 4.6 参考文献

RFC：

RFC 1546, Host Anycasting Service , C. Partridge, BBN, IETF, www.ietf.org/rfc/rfc1546.txt , November 1993

RFC 1918, Address Allocation for Private Internets , Y. Rekhter, Cisco Systems, IETF, www.ietf.org/rfc/rfc1918.txt , February 1996

RFC 2373, IP Version 6 Addressing Architecture , R. Hinden, Nokia, IETF, www.ietf.org/rfc/rfc2373.txt , July 1998

RFC 2374, An IPv6 Aggregatable Global Unicast Address Format , R. Hinden, Nokia,

IETF, www.ietf.org/rfc/rfc2374.txt , July 1998

RFC 2375, IPv6 Multicast Address Assignments , R. Hinden, Ipsilon Networks, IETF, www.ietf.org/rfc/rfc2375.txt , July 1998

RFC 3306, Unicast-Prefix-Based IPv6 Multicast Addresses , B. Haberman, Consultant, IETF, www.ietf.org/rfc/rfc3306.txt , August 2002

RFC 3956, Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address , P. Savola, CSC/FUNET, IETF, www.ietf.org/rfc/rfc3956.txt , November 2004

RFC 3315, Dynamic Host Configuration Protocol for IPv6 (DHCPv6) , R. Droms, Cisco Systems, IETF, www.ietf.org/rfc/rfc3315.txt , July 2003

RFC 3513, Internet Protocol Version 6 (IPv6) Addressing Architecture , R. Hinden, Nokia, IETF, www.ietf.org/rfc/rfc3513.txt , April 2003

RFC 3484, Default Address Selection for Internet Protocol version 6 (IPv6) , R. Draves, Microsoft Research, IETF, www.ietf.org/rfc/rfc3484.txt , February 2003

RFC 3587, IPv6 Global Unicast Address Format , R. Hinden, Nokia, IETF, www.ietf.org/rfc/rfc3587.txt , August 2003

RFC 3972, Cryptographically Generated Addresses (CGA) , T. Aura, Microsoft Research, www.ietf.org/rfc/rfc3972.txt , March 2005

RFC 4007, IPv6 Scoped Address Architecture , S. Deering, Cisco Systems, IETF, www.ietf.org/rfc/rfc4007.txt , March 2005

RFC 4193, Unique Local IPv6 Unicast Addresses , R. Hinden, Nokia, IETF, www.ietf.org/rfc/rfc4193.txt , October 2005

RFC 4291, IP Version 6 Addressing Architecture , R. Hinden, Nokia, IETF, www.ietf.org/rfc/rfc4291.txt , February 2006

RFC 4861, Neighbor Discovery for IP version 6 (IPv6) , Y. Narten, IMB, IETF, www.ietf.org/rfc/rfc4861.txt , September 2007

RFC 4862, IPv6 Stateless Address Autoconfiguration , S. Thomon, Cisco Systems, IETF, www.ietf.org/rfc/rfc4862.txt , September 2007

网站：

Internet Protocol Version 6 Address Space: www.iana.org/assignments/ipv6-addressspace/ipv6-address-space.txt

IPv6 Global Unicast Address Assignments: www.iana.org/assignments/ipv6- unicastaddress-assignments/ipv6-unicast-address-assignments.xml

# 第5章 ICMPv6与邻居发现协议

如果熟悉IPv4的ICMP（Internet Control Message Protocol，互联网控制消息协议），那么就会发现ICMPv6也很相似。不过ICMPv6并不仅仅是IPv6的ICMP，而是一种更加健壮的协议，它不但包含了很多新特性，而且还改善了ICMPv4中的大量功能。

ICMP是TCP/IP协议栈的一个核心协议。操作系统利用该协议在设备之间发送消息。消息类型有通知型和/或差错型消息，如**ping**命令使用的回显请求（Echo Request）消息或者向发送端通知路由器无法转发数据包的消息。ICMP与ping和traceroute等应用配合，可以测试两台设备之间的网络连接性。

ICMPv6定义在RFC 4443“Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification”中。ICMPv6比ICMPv4更健壮，而且还包含了很多新特性并做了很多功能改善。

本章将介绍与ICMPv4使用的类型（Type）字段和代码（Code）字段相类似的ICMPv6消息格式，包括差错消息和通知信息两类ICMPv6消息。

本章将要讨论的ICMPv6差错消息有：

- 目的地不可达（Destination Unreachable）消息；
- 数据包超大（Packet Too Big）消息；
- 超时（Time Exceeded）消息；
- 参数问题（Parameter Problem）消息。

本章将讨论**ping**命令使用的以下两种ICMPv6通知消息：

- 回显请求（Echo Request）消息；
- 回显应答（Echo Reply）消息。

接着将讨论多播侦听发现（Multicast Listener Discovery，RFC 2710和RFC 3810）使用的以下ICMPv6通知消息：

- 多播侦听查询（Multicast Listener Query）消息；

- 多播侦听报告（Multicast Listener Report）消息；
- 多播侦听完成（Multicast Listener Done）消息。

最后，本章将讨论 ICMPv6 通知消息以及邻居发现（RFC 4861）使用的范围消息：

- 路由器请求（Router Solicitation）消息；
- 路由器宣告（Router Advertisement）消息；
- 邻居请求（Neighbor Solicitation）消息；
- 邻居宣告（Neighbor Advertisement）消息；
- 重定向（Redirect）消息。

为了更清楚地说明这么多消息，本章将以一定的拓扑结构为例并使用 Wireshark。在邻居发现协议一节，将讨论地址解析（类似于 IPv4 中的 ARP）、DAD（Duplicate Address Detection，重复地址检查）和 NUD（Neighbor Unreachability Detection，邻居不可达性检测）等内容。

## 5.1 通用消息格式

ICMPv6 的通用格式类似于 ICMPv4。如图 5-1 所示，每条 ICMPv6 消息的前面都有一个下一报头（Next Header）值为 58 的 IPv6 报头（IPv4 通过将协议[Protocol]字段置 1 表示后面是一条 ICMPv4 消息）。但前面的 IPv6 报头不必是 IPv6 基本报头，可以是第 3 章中讨论过的任何一种 IPv6 扩展报头。

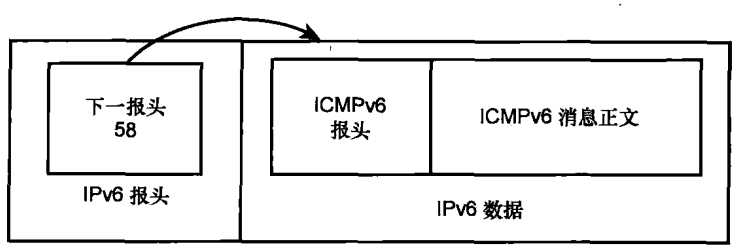



图 5-1 ICMPv6 下一报头值为 58

所有的 ICMPv6 消息都有相同的通用格式（如图 5-2 所示）。

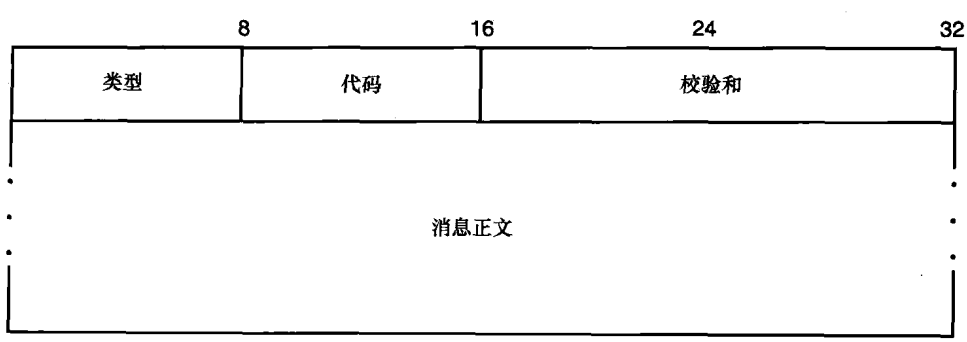



图 5-2 ICMPv6 通用消息格式

ICMPv6 消息中的三个字段如下所示。

- **类型（Type，8 比特）**：表示 ICMPv6 消息的类型，如回显请求、目的地不可达或数据包超大消息。
- **代码（Code，8 比特）**：为类型字段提供更精确的说明，其含义取决于消息类型。例如，如果消息类型是目的地不可达，那么代码字段就会说明数据包为何无法到达其目的地，如主机不可达或路由器的路由表中没有去往主机所在网络的路由。
- **校验和（Checksum，16 比特）**：用于检测 ICMPv6 消息以及部分 IPv6 报头中的数据损坏。

类型字段将 ICMPv6 消息分为两类：

- 差错消息（类型=0～127）；
- 通知消息（类型=128～255）。

ICMPv6 消息中的类型字段的高阶比特为 0（0xxxxxx）表示差错消息，因此差错消息的类型值为 0～127。与此相对应，通知消息中的类型字段的高阶比特为 1，其类型值为 128～255。

ICMPv6 差错消息的作用是告诉设备其发送的数据包无法被正确传送的原因，如已达到跳数限制（递减到 0）并被路由器丢弃。

ICMP 通知消息的作用不是报告差错，而是为各种测试、诊断和支撑功能提供必需的信息。IPv4 和 IPv6 都有的两条常用通知消息就是 **ping** 命令使用的回显请求消息和回显应答消息。

表 5-1 和表 5-2 分别概况了不同类型的 ICMPv6 差错消息和 ICMPv6 通知消息。

**表 5-1　　ICMPv6 差错消息**

| 类型 | 类型描述 | 代码和代码描述 |
|---|---|---|
| 1 | 目的地不可达 | 0：无去往目的地的路由；<br>1：与目的地的通信被管理性地禁止<br>2：超出了源地址范围<br>3：地址不可达<br>4：端口不可达<br>5：源地址与入站/出站策略相抵触<br>6：拒绝路由到目的地 |
| 2 | 数据包超大 | 0：被接收端忽略 |
| 3 | 超时 | 0：传送过程中超出了跳数限制<br>1：分段重组超时 |
| 4 | 参数问题 | 0：遇到了错误的报头字段<br>1：遇到了无法识别的下一报头类型<br>2：遇到了无法识别的 IPv6 选项 |
| 101 | 私有试验 | - |
| 107 | 私有试验 | - |
| 127 | 保留用作扩展 ICMPv6 差错消息 | - |

表 5-2 ICMPv6 通知消息

| 类型 | 类型描述 | 代码和代码描述 |
|---|---|---|
| ping 命令使用的消息（RFC 4443） | | |
| 128 | 回显请求 | 0：被接收端忽略 |
| 129 | 回显应答 | 0：被接收端忽略 |
| 多播侦听发现使用的消息（RFC 2710） | | |
| 130 | 多播侦听查询 | 0：被接收端忽略 |
| 131 | 多播侦听报告 | 0：被接收端忽略 |
| 132 | 多播侦听完成 | 0：被接收端忽略 |
| 邻居发现使用的消息（RFC 4861） | | |
| 133 | 路由器请求消息 | 0：被接收端忽略 |
| 134 | 路由器宣告消息 | 0：被接收端忽略 |
| 135 | 邻居请求消息 | 0：被接收端忽略 |
| 136 | 邻居宣告消息 | 0：被接收端忽略 |
| 137 | 重定向消息 | 0：被接收端忽略 |

表 5-3 列出的 ICMPv6 通知消息已经超出了本书写作范围，列在这里主要是为了完整性，以便读者更好地理解 ICMPv6 的使用方式。

表 5-3 ICMPv6 通知消息

| 类型 | 描述 |
|---|---|
| **类型 138**<br>路由器重新编址 | 用于路由器重新编址，其作用是通知一组路由器，它们即将执行重新编址操作（RFC 2894） |
| **类型 139**<br>节点信息查询<br>**类型 140**<br>节点信息应答 | 向 IPv6 节点请求特定的网络信息，如其主机名或者已完全授权的域名（RFC 4620） |
| **类型 141**<br>反向邻居发现请求消息<br>**类型 142**<br>反向邻居发现宣告消息 | 允许节点确定并宣告与特定链路层地址相对应的 IPv6 地址，类似于 IPv4 用于帧中继的反向 ARP（RFC 3122） |
| **类型 143**<br>版本 2 的多播侦听报告消息 | MLDv2 增加了一项功能，允许节点报告自己期望侦听特定多播地址的数据包，这里所说的多播地址可以是仅来自特定源地址，也可以是来自除特定源地址之外的全部源地址（RFC 3810） |
| **类型 144**<br>ICMP 家乡代理地址发现请求消息<br>**类型 145**<br>ICMP 家乡代理地址发现应答消息<br>**类型 146**<br>移动前缀请求消息格式<br>**类型 147**<br>ICMP 移动前缀宣告消息格式 | 支持移动 IPv6（RFC 3775） |

续表

| 类型 | 描述 |
| --- | --- |
| **类型 148**<br>证书路径请求消息格式<br>**类型 149**<br>证书路径宣告消息格式 | 为邻居发现协议提供安全机制 |
| **类型 151**<br>宣告分组格式<br>**类型 152**<br>请求分组格式<br>**类型 153**<br>终结分组格式 | 允许发现多播路由器（RFC 4286） |
| **类型 200**<br>**类型 201** | 用于私有试验 |
| **类型 255** | 保留用作扩展 ICMPv6 通知消息（RFC 4443） |

## 5.2　IGMP 差错消息

三层设备（如主机和路由器）使用 ICMPv6 差错消息通知发送端为何无法传送数据包。如表 5-1 所示，差错消息分为 4 类：

- 目的地不可达消息；
- 数据包超大消息；
- 超时消息；
- 参数问题消息。

注：设备永远也不会因为前一条 ICMPv6 差错消息而发送另一条 ICMPv6 差错消息，否则这将会导致永无止境的差错消息循环。

### 5.2.1　目的地不可达

当数据包因拥塞之外的其他原因而无法被传送到目的地时，就会发送 ICMPv6 目的地不可达消息。该反馈消息的作用不是让发送端不知其所以然的重发数据包，而是为发送端提供这么做的原因。路由器和防火墙通常都会生成这类目的地不可达消息。

注：有关防火墙或其他安全设备是否不应该发送目的地不可达消息还存在很大的争议。

图 5-3 显示了目的地不可达消息的格式。请注意，类型字段被置为 1。

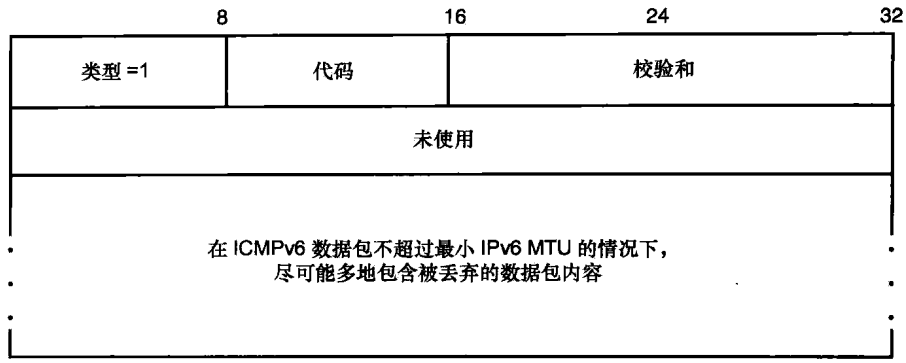



图 5-3 ICMPv6 目的地不可达消息

目的地不可达的原因有很多，代码字段可以提供目的地不可达的更准确原因。一共有 7 种代码。

- **代码=0，无去往目的地的路由**：数据包无法被传送的原因是路由器没有去往目的地的路由，这仅发生在路由器的路由表中无默认路由的场合。该消息等同于 ICMPv4 中的网络不可达消息。
- **代码=1，与目的地的通信被管理性地禁止**：数据包被阻塞的原因是访问控制列表或其他包过滤机制。
- **代码=2，超出了源地址范围**：当源地址是链路本地地址且目的地址是全局单播地址时，就会生成该差错消息。
- **代码=3，地址不可达**：该差错消息表明数据包在发送过程中出现问题的原因是目的地址中指定的主机不可达，这仅发生在目的地址无法被解析为相应的链路层地址（LAN 中的 MAC 地址）或目的地址错误时。该消息等同于 ICMPv4 中的主机不可达消息。
- **代码=4，端口不可达**：该差错消息的原因是 TCP 或 UDP 报头中指定的目的端口不存在或目的地不在该端口上进行侦听。例如，如果发送的数据包的 TCP 目的端口为 80，但接收主机却没有运行 HTTP Web 服务，那么就会发送端口不可达消息.
- **代码=5，源地址与入站/出站策略相抵触**：该差错消息表明具有该源地址的数据包因为访问控制列表或其他包过滤机制而被阻塞了。代码 5 是代码 1 的子集。
- **代码=6，拒绝路由到目的地**：该差错消息出现的原因是带有指定前缀的数据包被访问控制列表或其他包过滤机制阻塞了。代码 6 是代码 1 的子集。

### 5.2.2 数据包超大

IPv6 的一个重大变化就是数据包的分段和重组。对 IPv4 来说，当出链路的 MTU

（Maximum Transmission Unit，最大传输单元）小于数据包的尺寸时，路由器就会执行分段操作。目的端设备就要对分段后的数据包进行重组。

路由器可以根据需要对数据包进行分段，这看起来比较方便，但实际上也会影响路由器的效率。因此，IPv6 删除了路由器的该功能，仅允许数据包的源端执行分段操作。当 IPv6 路由器收到的数据包大于出站接口的 MTU 时，就会丢弃该数据包并向源端发送 ICMPv6 数据包超大消息。数据包超大消息中包含了该链路的 MTU 值（以字节为单位），因此源端能够在调整数据包大小后再重发数据包。

注：如果路由器是 IPv6 数据包的源端，那么也可以执行分段操作。

如图 5-4 所示，ICMPv6 数据包超大消息中的类型字段为 2 且代码字段为 0，MTU 是下一跳链路的最大传输单元，该 ICMPv6 差错消息也是路径 MTU 发现（Path MTU Discovery）的一部分。

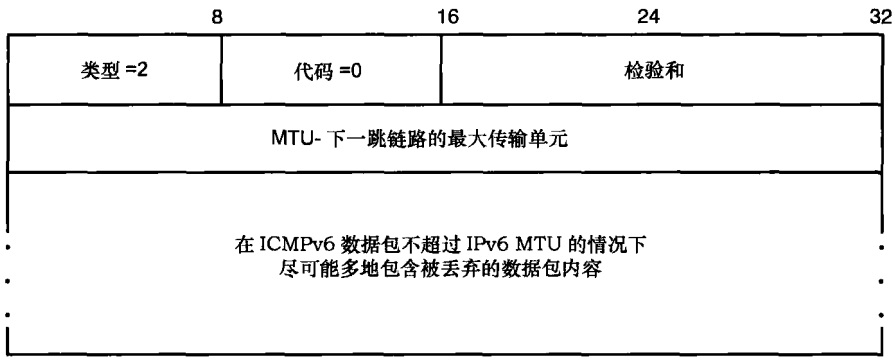



图 5-4　ICMPv6 数据包超大消息

### 路径 MTU 发现

路径 MTU 发现定义在 RFC 1981“Path MTU Discovery for IP version 6”中。当设备需要发送大量数据包时，最好的方式是数据包越大越好，从而减少数据包数量。这就需要设备了解去往目的地的路径上的所有链路的最小链路 MTU（也就是最小的 MTU），这样发送端就可以发送最大的数据包，而无需担心数据包在途中会因出站链路 MTU 过小而被丢弃。此时的数据包大小就被称为 PMTU（Path MTU，路径 MTU）。

注：与 IPv4 的 68 字节相比，IPv6 要求互联网上所有链路的最小 MTU 为 1280 字节，从而大大提高了净荷与报头的比率，并减少了源端的分段需求。

路径 MTU 发现进程的工作方式如图 5-5 所示，下面列出了相应的步骤。

**第 1 步：**设备假设数据包的 PMTU 值等于其去往第一跳路由器的出站链路的 MTU。

在图 5-5 中，PC-A 将 PMTU 值设为 1500 字节，也就是以太网链路的 MTU。

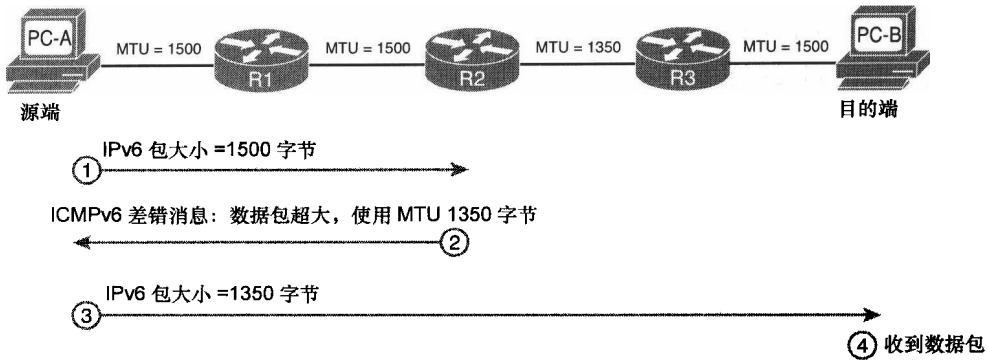



图 5-5 路径 MTU 发现

**第 2 步**：如果数据包尺寸大于该路由器的下一跳链路的 MTU，就会被路由器丢弃并向源端发送 ICMPv6 数据包超大消息。消息中包含了下一跳链路的 MTU 值。在图 5-5 中，路由器 R2 丢弃了数据包并向 PC-A 发送了数据包超大消息，消息中包含的 MTU 值为 1350 字节。

**第 3 步**：源端设备根据数据包超大消息中的信息，减小数据包尺寸以匹配消息中所包含的 MTU 值，然后源端设备再使用较小的 MTU 发送数据包。在图 5-5 中，PC-A 以新 MTU 值 1350 字节再次发送数据包。请注意，MTU 值永远也不会小于 1280 字节，因为这是 IPv6 规定的最小链路 MTU。

**第 4 步**：路由器会不断发送 ICMPv6 数据包超大消息，并且源端设备也会不断减小其数据包尺寸，直至数据包最终到达目的地。在图 5-5 中，新的 MTU 能够将数据包发送给目的端 PC-B。

由于从特定源端到给定目的端的路径可能会发生变化，使得 PMTU 也会随之发生变化，因此源端设备可能需要动态调整数据包的 PMTU。虽然并不强制要求设备实施路径 MTU 发现机制，但 RFC 4443 仍建议这么做。路径 MTU 发现既支持多播目的端，也支持单播目的端。

### 5.2.3 超时

路由器在转发数据包之前，会将跳数限制字段递减 1，这一点与 IPv4 的 TTL 字段相同，只是在字段名称上更能明确地反映该功能。当跳数限制字段递减到 0 时，该数据包就会被丢弃，并向源端发送 ICMPv6 超时消息。对于 IPv4 和 IPv6 来说，这是一种避免数据包在网络中被无休止传送的保证机制。

ICMPv6 超时消息的类型字段为 3，代码字段可以是 0 或 1。当数据包在传送途中就已经超出了跳数限制时，跳数限制字段会最终递减至 0 并被路由器丢弃，此时代码为 0，主要原因是可能存在路由环路或者最初设定的跳数限制值过小。traceroute 工具就是

利用超时消息来确定去往目的地的路由器路径。

在最初的 traceroute 工具实现中，程序会发送一串跳数限制字段从 1 开始的 ICMPv6 回显请求消息。每接收到一条 ICMPv6 超时消息，就会显示数据包的源地址并将跳数限制值递增 1，然后再发送另一条回显请求消息。以此往复，直至回显请求消息到达目的端。

Cisco IOS **traceroute** 命令使用初始端口号为 33434 或者在扩展的 **traceroute** 命令中指定端口号的 UDP 报文段，第一个数据包的 IPv6 跳数限制是 1。只要源端收到了 ICMP 超时消息，那么跳数限制以及目的端口号递增的循环就会持续。当源端收到 ICMPv6 端口不可达消息时，就是告诉源端已经到达目的端。

> 注：RFC 1393 “Traceroute Using an IP Option” 定义了一种更有效的 traceroute 执行方式。源端设备向目的地发送一个包含特殊的 Traceroute IP 选项的数据包，路径上的每台路由器都能将该消息识别为测试消息，并以 ICMP Traceroute 消息向原始源端发送响应消息。

当目的端设备在重组原始消息时发现没有接收到全面分段后的数据包时，代码字段将等于 1。收到第一个分段之后就会启动定时器（而不是等待一个不确定时间），并为全部数据包到达目的地分配特定的时间量。如果这些数据包未在特定时间内全部到达，那么目的端就会向源端发送 ICMPv6 超时消息。

### 5.2.4　参数问题

设备在处理数据包时，如果发现 IPv6 基本报头或扩展报头中存在字段问题，那么就会生成 ICMPv6 参数问题差错消息，这就意味着接收端无法理解 IPv6 报头中的信息，因而必须丢弃该数据包。出现该问题的原因是扩展报头中存在争议的字段是无效的，或者该设备不支持扩展报头。

## 5.3　ICMP 通知消息

设备使用 ICMPv6 差错消息告诉发送端数据包无法发送的原因。ICMPv6 通知消息用来帮助设备发现原因并在设备之间共享信息。如表 5-2 所示，ICMPv6 拥有以下通知消息：

ping 命令使用的消息如下（RFC 4443）：

- 回显请求消息；
- 回显应答消息。

多播侦听发现使用的消息如下（RFC 2710 和 RFC 3810）：

- 多播侦听查询消息；
- 多播侦听报告消息；
- 多播侦听完成消息。

邻居发现使用的消息如下（RFC 4861）：

- 路由器请求消息；
- 路由器宣告消息；
- 邻居请求消息；
- 邻居宣告消息；
- 重定向消息。

### 5.3.1 回显请求与回显应答

回显请求与回显应答是 ping（是最常见的 TCP/IP 工具）使用的两种 ICMP 消息。ping 常用来测试两台设备之间的网络连接性，其名称来源于主动声纳术语。

设备发出回显请求消息后，要求目的端返回回显应答消息以验证网络层的连接性。如果发送端没有收到相对应的回显应答消息，那么并不意味着目的端不可达，有可能是路径上的网络设备丢弃了回显请求或回显应答消息，也有可能是目的端本身不接受或不响应该回显请求消息。

图 5-6 显示了回显请求和回显应答消息的格式。回显请求和回显应答消息的结构完全相同，只是类型字段值不同。回显请求消息的类型字段值为 128，而回显应答消息的类型字段值为 129，代码字段始终为 0，其余字段的情况如图 5-6 所示。

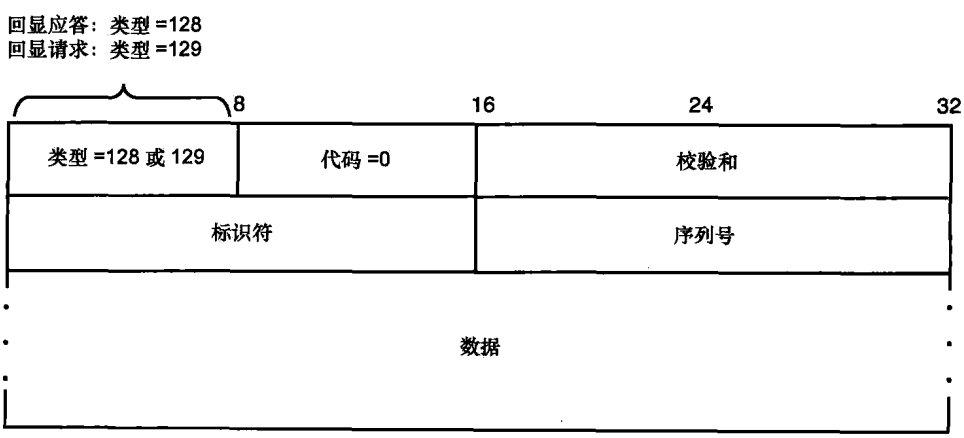



图 5-6 ICMPv6 回显请求和回显应答消息

图 5-7 给出了前面曾经用过的拓扑结构，其中标出了 PC1 和路由器 R1 的全局单播

地址和链路本地地址。利用数据包分析仪 Wireshark，通过对全局单播地址和链路本地地址执行 ping 操作，即可检验 ICMPv6 回显请求和回显应答消息。

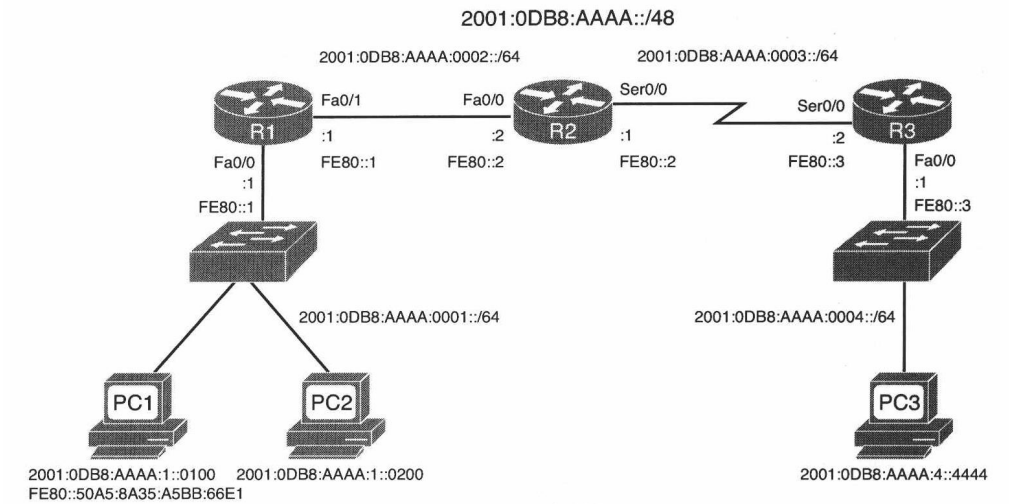



图 5-7　IPv6 拓扑结构

### 1. ping 全局单播地址

如例 5-1 所示，从主机 PC1 向路由器 R1 的全局单播地址发起 ping 操作。例中显示了从 PC1 发起的 **ping** 命令。除了 IPv6 地址之外，其余的都与 IPv4 地址的 ping 操作完全相同。

**例 5-1　从 PC1 向路由器 R1 的全局单播地址发起 ping 操作**

```
PC1> ping 2001:db8:aaaa:1::1

Pinging 2001:db8:aaaa:1::1 from 2001:db8:aaaa:1:1c3f:114a:4bcb:1b9c with 32 bytes of
data:

Reply from 2001:db8:aaaa:1::1: time=1ms
Reply from 2001:db8:aaaa:1::1: time=1ms
Reply from 2001:db8:aaaa:1::1: time=1ms
Reply from 2001:db8:aaaa:1::1: time=1ms

Ping statistics for 2001:db8:aaaa:1::1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 1ms, Average = 1ms

PC1>
```

例 5-2 显示了 PC1 向路由器 R1 发送 ICMPv6 回显请求消息的详细情况。

**例 5-2 PC1 发送给 R1 的回显请求消息**

```
Ethernet II, Src: 00:21:9b:d9:c6:44, Dst: 00:03:6b:e9:d4:80

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 40
     Next header: ICMPv6 (0x3a)
    Hop limit: 128
    Source: 2001:db8:aaaa:1:1c3f:114a:4bcb:1b9c
    Destination: 2001:db8:aaaa:1::1

Internet Control Message Protocol v6
     Type: 128 (Echo (ping) request)
     Code: 0 (Should always be zero)
    Checksum: 0x8f38 [correct]
    ID: 0x0001
     Sequence: 0
    Data (32 bytes)
```

请注意第 3 章讨论过的所有字段（以 IPv6 报头开头），包括版本以及源地址和目的地址。由于 **ping** 命令中的目的地址是全局单播地址 2001:db8:aaaa:1::1，因而源地址也是全局单播地址 2001:db8:aaaa:1:1c3f:114a:4bcb:1b9c。例 5-3 给出的 ICMPv6 回显应答消息与此相似。在这两个数据包中，下一报头字段值（十六进制 3A，十进制 58）表明 IPv6 报头后面是一个 ICMPv6 报头。

**例 5-3 R1 发送给 PC1 的回显应答消息**

```
Ethernet II, Src: 00:03:6b:e9:d4:80, Dst: 00:21:9b:d9:c6:44

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 40
     Next header: ICMPv6 (0x3a)
    Hop limit: 64
    Source: 2001:db8:aaaa:1::1
    Destination: 2001:db8:aaaa:1:1c3f:114a:4bcb:1b9c
```

```
Internet Control Message Protocol v6
    Type: 129 (Echo (ping) reply)
    Code: 0 (Should always be zero)
    Checksum: 0x8e38 [correct]
    ID: 0x0001
    Sequence: 0
    Data (32 bytes)
```

通过分析例 5-2 和例 5-3 的 ICMPv6 报头信息，可以更好地理解这些消息。

- **类型（Type）**：回显请求消息中的类型字段值为 128，回显应答消息中的类型字段值为 129。
- **代码（Code）**：被接收端忽略，回显请求和回显应答消息的代码字段始终为 0。
- **校验和（Checksum）**：校验和验证了 ICMPv6 报头。
- **标识符（Identifier）**：该字段用于匹配回显请求消息与其相对应的回显应答消息。请注意，回显请求与回显应答消息的标识符值相同，由该 **ping** 命令生成的所有回显请求和回显应答消息串的标识符值均相同。对例 5-3 来说，该串 ICMPv6 消息的标识符被设置为 1。
- **序列号（Sequence）**：该字段也用于匹配回显请求消息与其相对应的回显应答消息，而且提供了更为精细化的匹配信息。回显请求消息中会包含一个序列号，与其相对应的回显应答消息中则包含相同的序列号，下一条回显请求消息的序列号会递增 1，而且接收端在其返回的回显应答消息中也使用相同的序列号。对例 5-3 来说，回显请求消息和回显应答消息的序列号都是 0，由同一条 **ping** 命令产生的下一条回显请求消息和回显应答消息的序列号都将递增为 1。
- **数据（Data）**：回显请求消息会增加零个或多个字节的任意数据，而接收端设备则将这些数据都复制到返回的回显应答消息中。

### 2. ping 链路本地地址

例 5-4 显示了另一个 **ping** 命令，这次是从路由器 R1 向 PC1 的链路本地地址发起的 ping 操作。请记住，链路本地地址只要在链路上唯一即可，因为其不会被路由到本地链路之外。

**例 5-4　从路由器 R1 向 PC1 的链路本地地址发起 ping 操作**

```
R1# ping fe80::50a5:8a35:a5bb:66e1
Output Interface: fastethernet 0/0
% Invalid interface. Use full interface name without spaces (e.g. Serial0/1)
R1# ping fe80::50a5:8a35:a5bb:66e1
```

```
Output Interface: fastethernet0/0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FE80::50A5:8A35:A5BB:66E1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
R1#
```

从例 5-4 中可以看出，如果事先没有指定路由器的出接口（exit interface）或输出接口（output interface），那么就无法 ping PC1 的链路本地地址。如后所述，去往链路本地地址的目的地不在路由器的路由表中，因而路由器不知道应该使用哪个出接口。因此第一次向链路本地地址发起的 **ping** 命令失败了，Cisco IOS 要求使用无空格的完整接口名称。

**例 5-5　R1 向 PC1 的链路本地地址发送的回显请求消息**

```
Ethernet II, Src: 00:03:6b:e9:d4:80, Dst: 00:21:9b:d9:c6:44

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 60
    Next header: ICMPv6 (0x3a)
    Hop limit: 64
    Source: fe80::1
    Destination: fe80::50a5:8a35:a5bb:66e1

Internet Control Message Protocol v6
    Type: 128 (Echo (ping) request)
    Code: 0 (Should always be zero)
    Checksum: 0x0444 [correct]
    ID: 0x0a24
    Sequence: 0
    Data (52 bytes)
```

**例 5-6　从 PC1 的链路本地地址向 R1 发送的回显应答消息**

```
Ethernet II, Src: 00:21:9b:d9:c6:44, Dst: 00:03:6b:e9:d4:80

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
```

```
    Payload length: 60
    Next header: ICMPv6 (0x3a)
    Hop limit: 64
    Source: fe80::50a5:8a35:a5bb:66e1
    Destination: fe80::1

Internet Control Message Protocol v6
    Type: 129 (Echo (ping) reply)
    Code: 0 (Should always be zero)
    Checksum: 0x0344 [correct]
    ID: 0x0a24
    Sequence: 0
     Data (52 bytes)
```

请注意，这两台设备都将自己的链路本地地址用作数据包的 IPv6 源地址。该消息的其余内容与前一条 ICMPv6 消息完全相同。同样，链路本地地址只在本地链路上有意义，因此 PC1 无法 ping PC3 的链路本地地址，因为它们位于不同的网络或链路上。

注：如果从主机 PC1 ping 链路本地地址，由于该主机只有一个接口，因而无需指定出接口。

### 5.3.2　多播侦听者发现

多播对 IPv6 来说并不是什么新鲜事物。多播自 1988 年开始就已经出现在 IPv4 中了。多播地址用于将单个数据包或更为常见的分组流同时发送给多台设备，其效率显然要比单播传送模式下复制这些数据包并分别发送给每个目的地要高得多。

IPv4 中的多播组管理工作是由 IGMP（Internet Group Management Protocol，互联网组管理协议）来完成的。主机利用 IGMP 注册到特定网络的多播组，方法是由主机向本地多播路由器发送 IGMP 消息，告诉路由器自己希望接收哪个多播地址的流量。路由器被配置为侦听来自主机的 IGMP 消息。路由器会周期性地发出查询消息以发现哪些多播组仍然有效。也就是说，发现哪些拥有主机的多播组仍然希望接收来自多播地址的流量，使得路由器能够确定哪些多播地址已经无效，不再有主机希望接收其流量。对于第一个版本的 IGMP 来说，主机无法显式地离开多播组，告诉路由器自己希望离开多播组。IGMPv2 为主机增加了离开机制，主机可以通知路由器自己希望离开多播组。

IPv6 利用 ICMPv6 MLD（Multicast Listener Discovery，多播侦听者发现）来完成相同的服务能力，其功能完全基于 IGMPv2。因此，如果大家熟悉 IGMP，那么就会发现 MLD 与其非常相似。MLD 定义在 RFC 2710“Multicast Listener Discovery for IPv6”

中，MLDv2 定义在 RFC 3810“Multicast Listener Discovery Version 2 (MLDv2) for IPv6”中。MLDv2 基于 IGMPv2，扩展了 MLDv1 的功能，以支持 SSM（Source Specific Multicast，指定源多播）并后向兼容 MLDv1。SSM 为主机提供了请求多播包的能力，不仅可以向目的多播地址请求，也可以从指定源地址发出请求。MLDv2 是 Cisco IOS 的默认版本。

注：MLD 消息利用 IPv6 逐跳报头选项通知路由器更精确地检查数据包，目的是提供一种有效机制，让路由器知道何时拦截不是发给自己的数据报，而无需逐一检查每一个数据报。

注：如同所有的 IPv4 和 IPv6 数据包，源地址必须是单播地址。

MLD 有三种消息。

- **多播侦听者查询消息（Multicast Listener Query，类型=十进制 130）**：路由器周期性地发送主机成员关系查询消息，以确定哪些多播组仍然有成员在路由器直连的网络上。多播侦听查询消息有两个子类型。
  - **通用查询（General Query）**：该消息用于学习直连链路上哪些多播地址有侦听者。通用查询消息发送给链路范围的全部节点多播地址 FF02::1，该链路上的所有 IPv6 设备都能收到该消息。
  - **特定多播地址查询（Multicast-Address-Specific Query）**：该消息用于学习直连链路上是否有特定多播地址（多播组）的侦听者，会向被查询的多播地址发送指定地址查询（Address-Specific Query）消息。
- **多播侦听者报告（Multicast Listener Report，类型=十进制 131）**：侦听者通过发送该消息向多播组进行注册。侦听者不但可以发送该消息作为查询的响应消息，而且还可以自主发送该消息，而无需等待路由器发送来的查询消息。如果作为查询的响应消息，那么只有多播组中的一个成员需要发送多播侦听者报告消息。在 MLDv1 中，这些多播侦听者报告会发送给被报告的多播地址，但是 MLDv2 改变了这一做法，将这些多播侦听者报告发送给指定的多播地址 FF02::16，即发送给全部支持 MLDv2 的路由器（all-MLDv2-capapable routers）。
- **多播侦听者完成（Multicast Listener Done，类型=十进制 132）**：当侦听者不希望接收某特定多播组的流量时，就会发送一条多播侦听者完成消息，以通知路由器其将要离开该多播组。多播侦听者完成消息会被发送给链路范围的全部路由器（all-routers）多播地址 FF02::2。

图 5-8 给出了 MLDv2 通用查询消息和多播侦听者报告消息的示例，并在后面列出了相应的步骤。

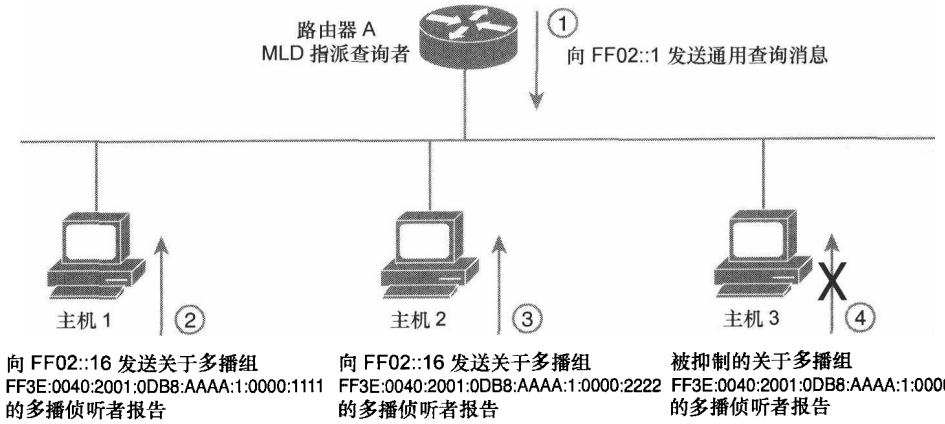



图 5-8　MLDv2 通用查询消息和多播侦听者报告消息

**第 1 步**：路由器 A 是网络上的 MLD 指派查询者（MLD-designated querier）。路由器 A 会周期性地向链路范围的全部节点多播地址 FF02::1 发送通用查询消息，以发现哪些多播组主机希望接收多播组的流量。

**第 2 步**：主机 1 是多播组 FF3E:0040:2001:0DB8:AAAA:1:0000:1111 的成员。主机 1 收到通用查询消息后，在发送多播侦听者报告消息之前会等待一个随机延时间隔。主机 1 没有看见该多播组的其他主机发送的报告消息，因此向 FF02::16（全部支持 MLDv2 的路由器）发送多播侦听者报告消息，用于告诉路由器 A，其仍然希望接收该多播地址的流量。

**第 3 步**：主机 2 是多播组 FF3E:0040:2001:0DB8:AAAA:1:0000:2222 的成员。主机 2 向 FF02::16 发送了一条独立的多播侦听者报告消息，用于告诉路由器 A，其仍然希望接收该多播地址的流量；

**第 4 步**：主机 3 是多播组 FF3E:0040:2001:0DB8:AAAA:1:0000:1111 的成员。在等待了一个随机延时间隔之后，主机 3 看见主机 1 已经向路由器 A 发送了一条针对该多播组的多播侦听者报告消息。由于路由器 A 只要收到该多播组的任一个成员发来的报告消息即可，因而主机 3 不再发送报告消息。

当主机不希望再接收多播组的流量时，可以通过发送多播侦听者完成消息来通知路由器。图 5-9 解释了主机离开多播组的过程，并在后面列出了相应的步骤。

**第 1 步**：主机 1 不希望再接收来自多播组 FF3E:0040:2001:0DB8:AAAA:1:0000:1111 的流量，因而向链路范围的全部路由器多播地址 FF02::2 发送多播侦听者完成消息，以告诉路由器其希望离开该多播组。

**第 2 步**：路由器 A 是网络上的 MLD 指派查询者，接收多播侦听者完成消息。由于路由器仅维护网络上的多播组列表，而不维护这些多播组的成员设备列表，因而路由器收到多播侦听者完成消息之后，必须发送特定多播地址查询消息，以确定是否有其他设备仍然需要接收该多播组的流量。即路由器

A 会周期性地向多播组 FF3E:0040:2001:0DB8:AAAA:1:0000:1111 发送特定多播地址查询消息。

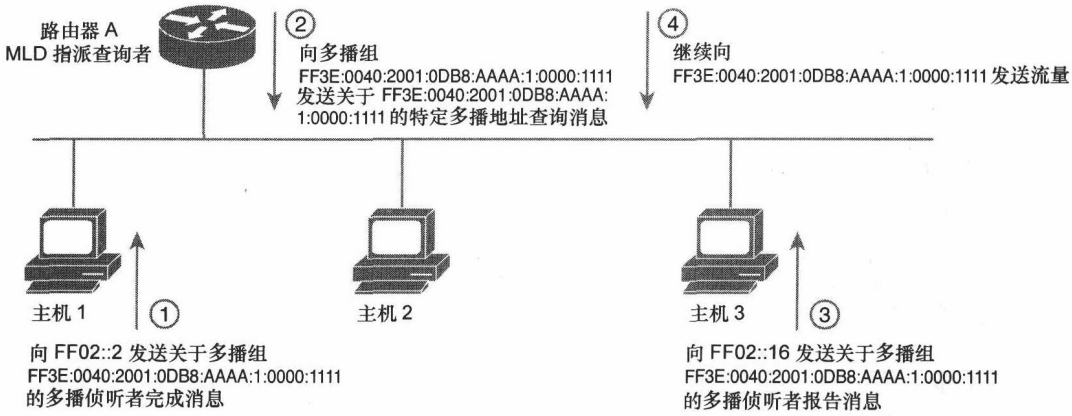



图 5-9 MLDv2 多播侦听者完成和多播侦听者报告消息

**第 3 步**：主机 3 是多播组 FF3E:0040:2001:0DB8:AAAA:1:0000:1111 的成员。主机 3 向该多播组发送多播侦听者报告消息作为应答，以告诉路由器，其仍然希望接收该多播组的流量。多播侦听者报告消息会被发送给 FF02::16（全部支持 MLDv2 的路由器）。

**第 4 步**：路由器 A 收到主机 3 发送来的多播侦听者报告消息后，继续向该多播地址发送流量。但是，如果路由器 A 在等待了已配置的时间段之后，仍然没有收到该多播组任何主机的报告消息，那么就会停止为该多播组转发流量。

虽然主机可以通过这些与路由器之间的通信机制来加入和离开多播组，但是路由器之间也应该具有某种通信机制以路由多播流量，此时就要用到 IPv4 的专用协议 PIM（Protocol Independent Multicast，协议无关多播）和 IPv6 的 PIM6。有关 MLD 和 PIM6 的配置信息不在本书写作范围之内，如果对此感兴趣，可以参考以下 Cisco 网站：

- www.cisco.com/en/US/docs/switches/datacenter/sw/5_x/nx-os/multicast/configuration/guide/ mld.html
- www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-multicast.html
- www.cisco.com/en/US/technologies/tk648/tk872/technologies_white_paper0900aecd80260049.pdf

## 5.4 邻居发现协议

ND 或 NDP（Neighbor Discovery Protocol，邻居发现协议）定义在 RFC 4861“Neighbor

Discovery for IPv6”中。邻居发现协议中包含了与 IPv4 的 ARP、ICMP 路由器发现和重定向等相似进程，但是也存在很多重要差异。此外，ND 还增加了很多新功能，如 DAD 和 NUD（Neighbor Unreachability Detection，邻居不可达性检测）。如第 4 章（IPv6 地址类型）所述，邻居发现在 IPv6 地址的自动配置机制中扮演了非常重要的角色。

设备（主机和路由器）使用邻居发现协议的原因如下所述。

- SLAAC，自动确定网络前缀、默认网关及其他配置信息。
- 确定自己将要使用的链路本地地址或全局单播地址是否被其他设备所使用（DAD）。
- 识别出目的 IPv6 地址之后，确定网络上的设备的二层数据链路地址（通常是以太网）。
- 了解哪些邻居可达以及哪些邻居不可达（NUD）。
- 当路由器或去往路由器的路径出现故障后，主机主动寻找相应的替代设备或路径。

邻居发现协议用到了以下 5 种 ICMPv6 消息：

- 路由器请求（RS）消息；
- 路由器宣告（RA）消息；
- 邻居请求（NS）消息；
- 邻居宣告（NA）消息；
- 重定向消息。

下面各节将逐一解释这些消息以及路由器使用这些消息的方式，并介绍前缀发现、地址解析和 DAD 等进程。

### 5.4.1　路由器请求消息和路由器宣告消息

可以将 IPv6 设备分为两大类：路由器和主机。路由器请求消息和路由器宣告消息用于主机与路由器之间的通信过程。路由器周期性地发送路由器宣告消息或者响应链路上的主机发送的路由器请求消息。主机通过发送路由器请求消息，要求路由器立即发送路由器宣告消息。

当主机需要前缀、前缀长度、默认网关以及其他用于 SLAAC 进程的相关信息时，就会发送路由器请求（RS）消息。这通常发生于主机刚刚加电并被配置为自动获取其 IP 地址的场合下。第 4 章解释了 SLAAC 可以为主机的单播地址使用 IEEE 改进型 EUI-64 格式或随机生成的 64 比特接口 ID。主机从路由器宣告（RA）消息中可以获得单播地址的前缀和前缀长度。主机既可以等待下一周期的 RA 消息，也可以主动发送 RS 消息以要求路由器发送一条 RA 消息。下面将详细讨论 RS 消息和 RA 消息的各个字段信息。

注：为了完整起见，本书描述了路由器请求和路由器宣告消息的所有字段信息。

注：由于某些字段的重要性远远高于其他字段，因而大家不要被这些消息中描述的大量信息所迷惑。在学习这些消息描述后面的案例时，请大家格外注意那些与IPv6相关的字段。

图 5-10 显示了 ND 路由器请求消息的格式，具体字段信息如下。

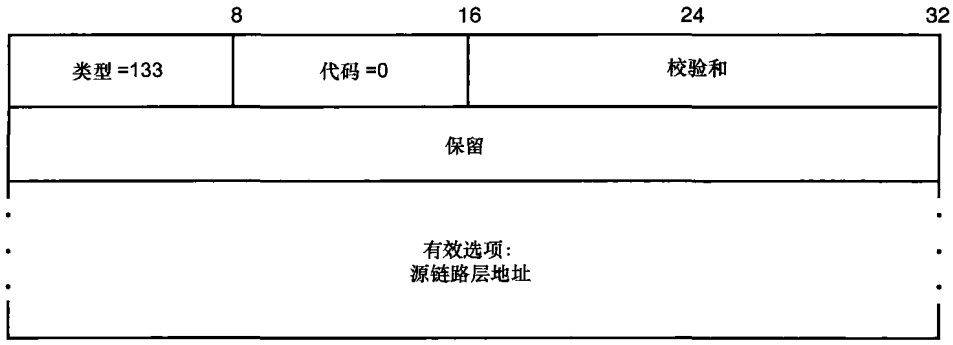



图 5-10 ICMPv6 ND 路由器请求消息

- **IPv6 报头：**
  - **源地址：**该字段既可以是已分配给发送接口的 IPv6 地址，也可以是未指定地址（如果还未分配地址）。仔细研究 SLAAC，可以发现该地址通常是主机的链路本地地址。请记住，链路本地地址可以采取随机生成方式，也可以由设备利用前缀 FE80::/10 以及由 EUI-64 生成的接口 ID 自动创建而成。
  - **目的地址：**目的地址通常是全部路由器多播地址 FF02::2。
  - **跳数限制：**跳数限制总是被设置为 255。
- **ICMPv6 字段：**
  - **类型（Type）：**该字段被设置为 133，表示这是一条路由器请求消息。
  - **代码（Code）：**该字段被设置为 0，表示被接收端忽略。
  - **校验和（Checksum）：**用于验证 ICMPv6 报头。
  - **保留（Reserved）：**该字段未使用。
  - **源链路层地址（Source Link Layer Address）：**该字段是发送端的二层（链路层）地址。

路由器宣告消息是周期性发送的，也可以作为路由器宣告消息的响应消息，它

主要为主机提供编址信息以及其他配置信息，并且是 SLAAC 的重要组成部分。路由器宣告消息中包含了很多信息，大家不要被大量的字段信息所迷惑。例 5-7 解释了其中最重要的一些信息。路由器宣告消息的格式如图 5-11 所示，后面还给出了进一步的解释。

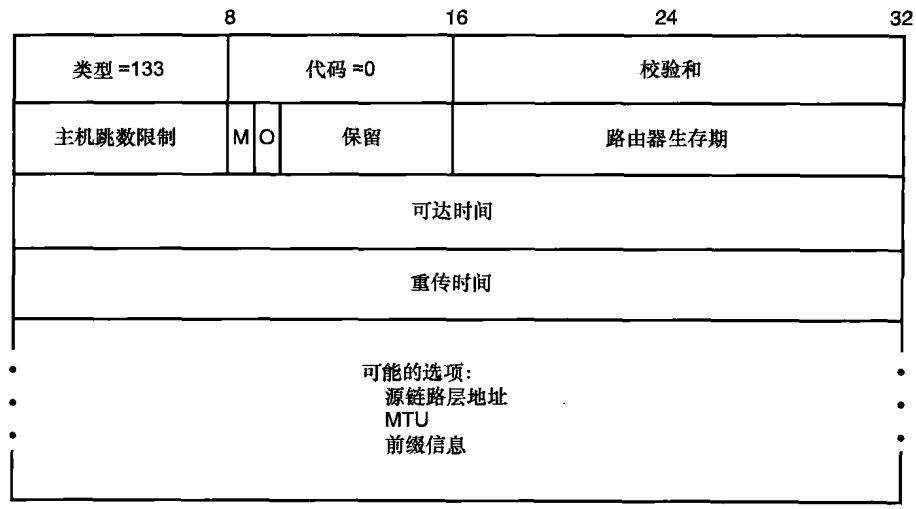



图 5-11　ICMPv6 ND 路由器宣告消息

- **IPv6 报头：**
  - **源地址：**源地址是路由器分配给该接口的链路本地地址。
  - **目的地址：**目的地址通常是全部节点多播地址 FF02::1。
  - **跳数限制：**跳数限制总是被设置为 255。
- **ICMPv6 字段：**
  - **类型（Type）：**该字段被设置为 134，表示这是一条路由器宣告消息。
  - **代码（Code）：**该字段被设置为 0，表示被接收端忽略。
  - **校验和（Checksum）：**用于验证 ICMPv6 报头。
  - **主机跳数限制（Cur Hop Limit）：**是路由器建议网络上的主机在各自 IP 包的跳数限制字段中使用的数值。该字段值为 0 时，表示路由器不推荐跳数限制值，由主机自己决定各自的跳数限制字段值。
  - **M 标记（M Flag）：**该字段是被管地址配置（Managed Address Configuration）标记。M 标记为 0 时，网络上的主机使用 SLAAC；M 标记为 1 时，网络上的主机使用 DHCPv6。
  - **O 标记（O Flag）：**该字段是其他配置（Other Configuration）标记。O 标记为 0 时，表示 DHCPv6 服务器没有其他可用信息；O 标记为 1 时，告诉主机可以从 DHCPv6 服务器获得额外信息，如与 DNS 相关的信息。

注：如果既不设置 M 标记也不设置 O 标记，那么就表明无法通过 DHCPv6 服务器获得可用信息。

注：除了 M 标记和 O 标记之外，路由器宣告消息还包含了 A 标记（autonomous address-configuration flag，自主地址配置标记）。A 标记表示 RA 中的前缀是否可以被 SLAAC 使用。默认情况下，A 标记被设置为 1，即前缀可以被 SLAAC 使用。A 标记提供了额外的地址配置选项，但相关内容已超出了本书写作范围，大家可以参考 http://blogs.cisco.com/borderless/ipv6-automatic-addressing 和 RFC 4861 “Neighbor Discovery for IP version 6 (IPv6)”。

- **保留（Reserved）**：该字段未使用。
- **路由器生存期（Router Lifetime）**：告诉主机该路由器能够被用作默认网关的持续时间（以秒为单位）。生存期为 0 表示路由器不是默认网关。路由器生存期字段仅适用于路由器充当默认网关功能的场合，并不适用于其他消息字段或选项中包含的其他信息，如前缀和前缀长度。主机每次收到路由器宣告消息之后，都会刷新自己的定时器。
- **可达时间（Reachable Time）**：告诉主机在收到可达性确认消息之后，可以在多长时间内假定邻居可达（以毫秒为单位）。该信息用于 NUD（邻居不可达性检测）机制，详细内容将在本章后面进行讨论。该字段值为 0 时表示路由器未指定该值。
- **重传时间（Retrans Time）**：告诉主机在重传邻居请求消息之前应该等待的时间（以毫秒为单位）。该字段被用于地址解析和 NUD 进程。有关邻居请求消息的详细内容将在本章后面进行讨论。
- **源链路层地址（Source Link Layer Address）**：该字段是发送端的二层（链路层）地址。在后面的例 5-9 中，该地址就是发送端的以太网 MAC 地址。
- **前缀信息（Prefix Information）**：告诉主机网络的前缀（地址的网络部分）和前缀长度（与 IPv4 的子网掩码相似）。

竟然包含了这么多信息！下面来看一下其中最重要的字段。首先，对于发送路由器宣告消息的路由器或实施 IPv6 路由协议的路由器来说，必须配置命令 **ipv6 unicast-routing**：

```
R1(config)# ipv6 unicast-routing
```

图 5-12 重点关注路由器 R1 的 LAN 和 PC1 的拓扑结构。PC1 被配置为自动获取其 IP 编址信息，这部分 SLAAC 用于分析路由器请求消息和路由器宣告消息（使用

Wireshark）。本章将在后面讨论完整的 SLAAC 进程。

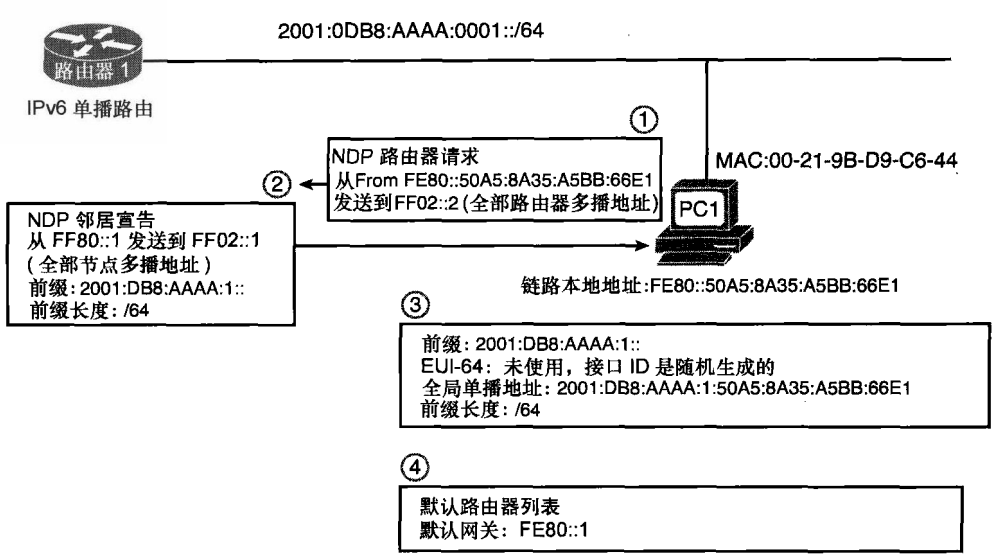



图 5-12　ND 路由器请求消息和路由器宣告消息

图 5-12 解释了路由器请求和路由器宣告消息的处理过程，步骤如下。

**第 1 步：** PC1 发出路由器请求消息，以请求路由器 R1 发送路由器宣告消息。该消息被发送给全部路由器多播地址 FF02::2。

**第 2 步：** 路由器 R1 以路由器宣告消息作为响应。该消息包含了 PC1 使用 SLAAC 配置其地址所需的前缀和前缀长度信息。路由器宣告消息并不是专门发送给 PC1 的，而是发送给全部节点多播地址 FF02::1。

**第 3 步：** PC1 收到路由器宣告消息之后，使用其中的前缀和前缀长度信息自动配置其全局单播地址。

**第 4 步：** PC1 还会使用路由器宣告消息中的信息以该路由器的链路本地地址更新其默认路由器列表（Default Router List），并将该信息用作其默认网关地址。其中，默认路由器列表是能够用来到达网络上除自身之外的其他设备的默认网关列表。

例 5-7 显示了 PC1 发出的路由器请求消息的内容。

**例 5-7　PC1 发出的路由器请求消息**

```
Ethernet II, Src: 00:21:9b:d9:c6:44, Dst: 33:33:00:00:00:02

Internet Protocol Version 6
    0110 .... = Version: 6
```

```
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 16
    Next header: ICMPv6 (0x3a)              ! Next header is an ICMPv6 header
    Hop limit: 255
    Source: fe80::50a5:8a35:a5bb:66e1       ! Link-local address
    Destination: ff02::2                    ! All-routers multicast address


Internet Control Message Protocol v6
    Type: 133 (Router solicitation)         ! Router Solicitation message
    Code: 0
    Checksum: 0x3277 [correct]
    ICMPv6 Option (Source link-layer address)
        Type: Source link-layer address (1)
        Length: 8
        Link-layer address: 00:21:9b:d9:c6:44   ! MAC address of PC1
```

首先看 IPv6 报头，请注意，源地址是 PC1 的链路本地地址，目的地址是全部路由器多播地址 FF02::2，意思是希望从链路上的路由器请求配置信息。下一报头字段表明紧跟在后面的是 ICMPv6 报头。

接着看 ICMPv6 报头，类型字段为 133，表明该 ICMPv6 消息是一条路由器请求消息，源链路层地址是 PC1 的 MAC 地址。

下面再来分析路由器宣告消息。路由器 R1 通过命令 **ipv6 unicast-routing** 被配置为 IPv6 路由器。利用命令 **show ipv6 interface fastethernet 0/0** 即可加以验证（如例 5-8 所示）。

**例 5-8　验证 R1 已被配置 IPv6 路由器**

```
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::1
  Global unicast address(es):
    2001:DB8:AAAA:1::1, subnet is 2001:DB8:AAAA:1::/64
  Joined group address(es):
    FF02::1
    FF02::2       ! All-routers multicast group
    FF02::1:FF00:1
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is enabled, number of DAD attempts: 1
```

```
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  Hosts use stateless autoconfig for addresses.
R1#
```

R1 是全部路由器多播组 FF02::2 的成员。其中，FF02::2 属于已加入多播组地址(joined group address(es))，在这之前必须首先在 R1 上应用命令 **ipv6 unicast-routing** 以启用 IPv6 功能。作为 IPv6 路由器，R1 会发送路由宣告消息。请注意，R1 在默认情况下每 200 秒发送一条 RA 消息。路由器生存期字段决定了该路由器充当默认网关的时间为 1800 秒。如果主机在下一个 1800 秒周期内不再从该路由器接收 RA 消息，那么就应该从其默认网关列表中删除该路由器。

注：如果没有通过命令 **ipv6 unicast-routing** 将路由器启用为 IPv6 路由器，那么路由器就不会发出 ND 路由器宣告消息，但是仍然能够配置 IPv6 接口。从本质上来看，这样做的结果就是将路由器变为 IPv6 主机，此时的路由器能够发送和接收 IPv6 数据包，但无法路由这些 IPv6 数据包。命令 **show ipv6 interface** 的输出结果将不会显示“ND advertised retransmit interval”或“ND router advertisements”信息。

例 5-9 分析了路由器 R1 的路由器宣告消息的内容。

**例 5-9　路由器 R1 发出的 ND 路由器宣告消息**

```
Ethernet II, Src: 00:03:6b:e9:d4:80, Dst: 33:33:00:00:00:01
Internet Protocol Version 6
    0110 .... = Version: 6
    .... 1110 0000 .... .... .... .... .... = Traffic class: 0x000000e0
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 64
    Next header: ICMPv6 (0x3a) ! Next header is an ICMPv6 header
    Hop limit: 255
    Source: fe80::1            ! Link-local address
                               ! Added to the Default Router List and
                               ! is the address hosts will use as
                               ! their defa`ult gateway
    Destination: ff02::1       ! All-nodes multicast group

Internet Control Message Protocol v6
    Type: 134 (Router advertisement) ! Router advertisement message
```

```
Code: 0
Checksum: 0x04d2 [correct]
Cur hop limit: 64          ! Recommended Hop Limit value for hosts
Flags: 0x00                ! M and O flags indicate that no
                           ! information is available via DHCPv6
Router lifetime: 1800      ! This router is a valid gateway for the
                           ! next 1800 seconds (3 minutes)
Reachable time: 0          ! No value specified
Retrans timer: 0           ! No value specified
ICMPv6 Option (Source link-layer address)
    Type: Source link-layer address (1)
    Length: 8
    Link-layer address: 00:03:6b:e9:d4:80  ! R1's MAC address
ICMPv6 Option (MTU)
    Type: MTU (5)
    Length: 8
    MTU: 1500              ! MTU of the link:
ICMPv6 Option (Prefix information)
    Type: Prefix information (3)
    Length: 32
    Prefix Length: 64      ! Prefix-length (/64) to be used for
                           ! autoconfiguration
    Flags: 0xc0
    Valid lifetime: 2592000
    Preferred lifetime: 604800
    Reserved
    Prefix: 2001:db8:aaaa:1::   ! Prefix of this network to
                                ! be used for autoconfiguration
```

从 IPv6 报头可以看出，下一报头字段表明紧跟在后面的是一个 ICMPv6 报头。源地址是接口的链路本地地址。需要引起注意的是，主机将使用该地址作为自己的默认网关。目的地址是全部节点多播地址 FF02::1。虽然该路由器宣告消息可以作为特定路由器请求消息的响应，但始终发送给全部节点多播地址。

> 注：非路由器的设备都要维护一个默认路由器列表（Default Router List）。设备收到路由器宣告消息后，会将数据包的链路本地源地址加入列表中，作为可用的默认网关地址。每个表项都有一个失效定时器——路由器生存期（Router Lifetime）。该信息是从路由器宣告消息中提取的，用于删除不再被宣告的表项。

ICMPv6 报头提供了 PC1 和其他主机所需的编址和配置信息。类型字段值为 134，表明这是一条路由器宣告消息。主机跳数限制字段建议主机在通过该路由器发送数据包时，将 IPv6 报头中的跳数限制字段值设置为 64。标记字段（包括 M 标记和 O 标记）

都被设置为 0，表示使用 SLAAC，并且不能从 DHCPv6 服务器获得额外信息。路由器生存期字段值为 1800，表示该路由器在接下来的 1800 秒钟（即 30 分钟）内可作为有效默认网关，除非该值被其他路由器宣告消息所更新。可达时间和重传定时器都被设置为 0，表示该路由器没有为 NUD 指定这些值。

路由器宣告消息包含很多选项。源链路层地址是 R1 的 MAC 地址。MTU 选项表明最大传输单元是 1500 字节。前面在讨论路径 MTU 发现时曾经说过，主机将使用该值作为其数据包大小的 PMTU，除非该主机收到去往目的地的路径上的其他路由器发来的 ICMPv6 数据包超大消息。前缀选项提供多种信息，主要是前缀长度 64（/64）和前缀本身 2001:DB8:AAAA:1::。使用自动配置机制的主机利用这些信息即可生成其全局单播地址。在图 5-12 中，第 3 步解释了 PC1 使用这些信息创建其全局单播地址的过程，不过并没有使用 EUI-64 进程，而是采取随机方式生成接口 ID。请注意，PC1 的默认网关是路由器 R1 的链路本地地址。

例 5-9 中的路由器宣告消息还包含了两个其他生存期字段：优选生存期（Preferred Lifetime）和有效生存期（Valid Lifetime）。第 4 章讨论了自动配置方式下的地址状态。图 5-13 给出了与优选生存期和有效生存期相关联的自动配置的地址状态。优选生存期是设备应该将通过 RA 的前缀与 SLAAC 生成的地址视为优选地址的时间（以秒为单位）。优选地址表示设备可以使用该地址发送和接收流量。优选生存期到期之后，设备可以不再使用该地址创建新连接。有效生存期是设备应该将路由器宣告消息中的前缀视为有效的时间（以秒为单位），有效生存期大于优选生存期。虽然优选生存期到期后，设备无法再使用优选地址创建新连接，但是在有效生存期到期之前，现有的所有连接都仍然有效。对这两个生存期来说，全 1（即 0xFFFFFFFF）表示“无穷大”。设备每收到一条新的路由器宣告消息，都要复位优选生存期和有效生存期定时器。

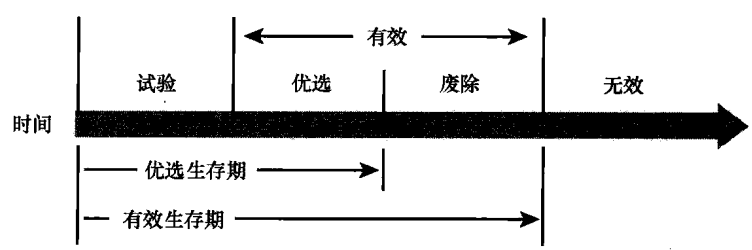



图 5-13　RA 的有效生存期和优选生存期

## 5.4.2　邻居请求消息和邻居宣告消息

ICMPv6 邻居发现协议使用的另外两种消息是邻居请求消息和邻居宣告消息。设备利用这些消息向同一网络上的其他设备请求二层（即链路层）地址或者向请求设备提供

该信息。邻居请求和邻居宣告消息是以下三个重要进程的组成部分：

- 地址解析；
- DAD（Duplicate Address Detection，重复地址检测）；
- NUD（Neighbor Unreachability Detection，邻居不可达性检测）。

邻居请求和邻居宣告消息非常类似于 IPv4 中的 ARP 请求和 ARP 应答消息。设备通过发送邻居请求消息来请求目标设备（target device）的二层（链路层）地址，同时也向目标设备提供自己的链路层地址。这些链路层地址通常是以太网 MAC 地址。

邻居宣告消息既可以作为邻居请求消息的响应消息，也可以根据需要向外发送以快速传播新信息。下面将分析邻居请求消息和邻居宣告消息的格式，并说明它们如何用于前面所说的各种进程。

注：这两个消息中也同样包含了大量信息，大家无需彻底理解每个字段的内容。这里描述所有字段信息的目的是为了完整起见，下面将重点介绍其中的一些关键字段。

图 5-14 给出了 ND 邻居请求消息的格式。

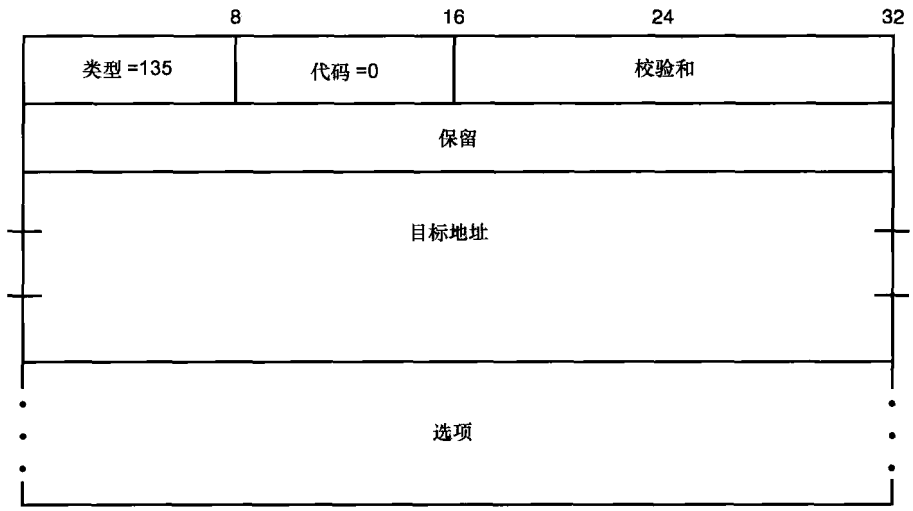



图 5-14 ICMPv6 ND 邻居请求消息

下面将介绍封装 ICMPv6 邻居请求消息的 IPv6 报头和 ICMPv6 报头，大家同样不必深究每个字段的细节信息。

- **IPv6 报头：**
  - **源地址：**该字段既可以是已分配给发送接口的 IPv6 地址，也可以是未指定地址（如果发送的该消息是 DAD 进程的一部分）。
  - **目的地址：**目的地址可以是与目标设备相关的请求节点多播地址，也可以

是目标地址本身。

- **跳数限制**：跳数限制总是被设置为 255。

- **ICMPv6 字段**：
  - **类型（Type）**：该字段被设置为 135，表示这是一条邻居请求消息。
  - **代码（Code）**：该字段被设置为 0，表示被接收端忽略。
  - **校验和（Checksum）**：用于验证 ICMPv6 报头。
  - **保留（Reserved）**：该字段未使用。
  - **目标地址（Target Address）**：该字段是目标设备的 IPv6 地址，发送端知道其 IPv6 地址，但是不知道二层（链路层）地址。目标地址不能是多播地址。
  - **源链路层地址（Source Link Layer Address）**：该字段是发送端的二层（链路层）地址。当 IPv6 源地址是未指定地址时，就不能包含该地址。

邻居宣告消息既可以作为邻居请求消息的响应消息，也可以根据需要自主发送，以快速传播新信息，图 5-15 显示了 ND 邻居宣告消息的格式。

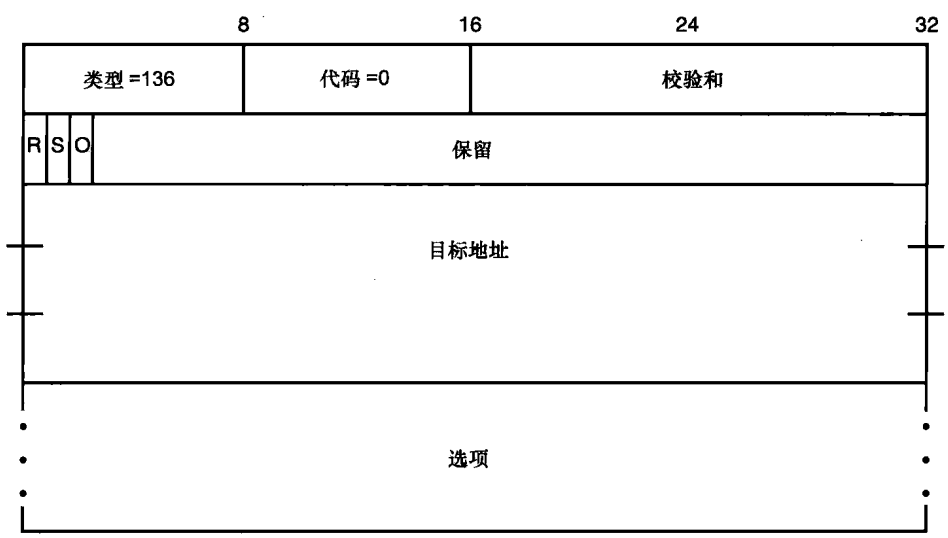



图 5-15　ICMPv6 ND 邻居宣告消息

下面将介绍封装 ICMPv6 邻居宣告消息的 IPv6 报头和 ICMPv6 报头，大家同样不必深究每个字段的细节信息。

- **IPv6 报头**：
  - **源地址**：是分配给发送接口的 IPv6 地址。
  - **目的地址**：如果相应的邻居请求消息的源地址是未指定单播地址，那么目的地址就是全部节点多播地址 FF02::1。否则，目的地址就是邻居请求消

息中的源地址。

- **跳数限制**：跳数限制总是被设置为 255。

- **ICMPv6 字段**：
  - **类型（Type）**：该字段被设置为 136，表示这是一条邻居宣告消息。
  - **代码（Code）**：该字段被设置为 0，表示被接收端忽略。
  - **校验和（Checksum）**：用于验证 ICMPv6 报头。
  - **R（R 比特或路由器标记，R-bit or Router flag）**：R 比特为 1 时，表示发送端是路由器。邻居不可达性检测机制利用 R 比特来检测变更为主机的路由器。
  - **S（S 比特或请求标记，S-bit or Solicited flag）**：S 比特为 1 时，表示所发送的该邻居宣告消息是作为邻居请求消息的响应消息。S 比特在邻居可达性检测阶段被用作可达性确认。
  - **O（O 比特或覆盖标记，O-bit or Override flag）**：O 比特为 1 时，表示该邻居宣告消息应该通过更新已缓存的 IPv6 地址对应的二层地址来覆盖现有的邻居缓存表项（相当于 IPv4 ARP 缓存表）。如果 O 比特为 0，表示该邻居宣告消息将不更新已缓存的链路层地址，而是创建一个链路层地址（如果没有的话）。
  - **保留（Reserved）**：该字段未使用。
  - **目标地址（Target Address）**：如果邻居宣告消息是邻居请求消息的响应消息，那么目标地址就是邻居请求消息中的目标地址字段的目标地址。也就是说，目标地址是发送该宣告消息的设备的 IPv6 地址。对于自主发送的宣告消息来说，目标地址是链路层地址发生变化的 IPv6 地址。
  - **源链路层地址（Source Link Layer Address）**：该字段是发送端的二层（链路层）地址。当 IPv6 源地址是未指定地址时，就不能包含该地址。

在使用这些消息之前，先来讨论两种重要的数据结构——邻居缓存表（Neighbor Cache）和目的地缓存表（Destination Cache）。

### 1. 邻居缓存表和目的地缓存表

主机需要为每个接口维护两张表或缓存表：

- 邻居缓存表；
- 目的地缓存表。

邻居缓存表等同于 IPv4 的 ARP 缓存表。邻居缓存表负责维护最近流量所送往邻居的信息列表，表项中包含了 IPv6 单播地址及其相对应的二层地址（通常是以太网 MAC 地址）。设备通过收到的邻居宣告消息中的信息来维护该缓存表。邻居宣告消息中的 R 比特会表明该设备是否是路由器。邻居缓存表项一共有 5 种状态（如图 5-16 所示的邻

居缓存表状态图）。虽然本书还没有深入分析邻居请求和邻居宣告消息，但这里为了便于大家更好地理解邻居缓存表，可以先将邻居请求和邻居宣告消息分别视为 IPv4 中的 ARP 请求和 ARP 应答消息。

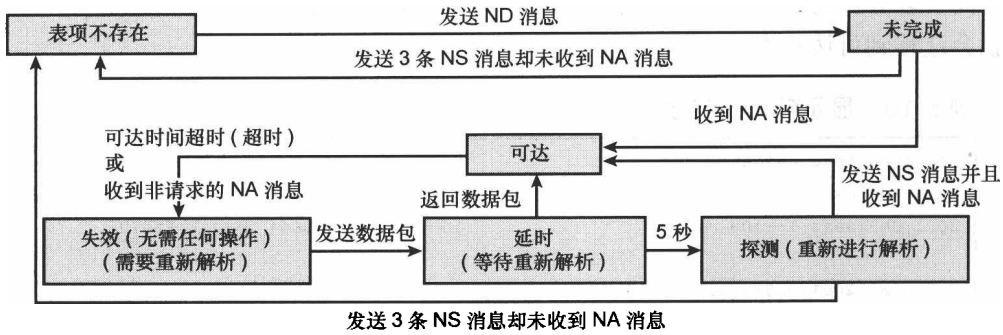



图 5-16　邻居缓存表的状态

图 5-16 给出的邻居缓存表状态如下所示。

- **表项不存在（No Entry Exists）**：这不是邻居缓存表的状态，表示邻居缓存表中不存在 IPv6 地址及其相对应的 MAC 地址表项。
- **未完成（Incomplete）状态**：此时地址解析过程还在进行之中，还不知道链路层地址。发出邻居请求消息以请求目标 IPv6 地址的 MAC 地址，如果发出三条邻居请求消息都没有收到相应的邻居宣告应答消息，就表明表项不存在，从而删除处于未完成状态的表项。
- **可达（Reachable）状态**：此时已经收到确认该设备可达的数据包。在已收到的邻居宣告消息中指定了与目标 IPv6 地址（在先前发出的邻居请求消息中指定）相对应的 MAC 地址；
- **失效（Stale）状态**：该状态是设备从该地址收到数据包开始已消逝的时间段（该时间段称为可达时间[reachable time]）。如果收到自主发送的邻居宣告消息，邻居缓存表会迁移到该状态，此时设备不需要任何操作。
- **延时（Delay）状态**：此时邻居处于等待重新解析状态，而流量可能会流经该邻居。设备已发出数据包并等待相应的确认消息（以某种形式的返回数据包进行确认，如三方握手的 TCP Syn/Ack），如果在 5 秒钟内收到了数据包，那么该表项就会返回到可达状态，否则就会迁移到探测状态；
- **探测（Probe）状态**：此时邻居处于重新解析状态，而流量可能会流经该邻居。该状态类似于未完成状态，设备会再次发送邻居请求消息以请求与目标 IPv6 地址相对应的 MAC 地址。如果邻居宣告消息是由被请求设备回送的，那么邻居缓存表项就会返回到可达状态，如果发送了三条邻居请求消息而没有收到相对应的邻居宣告消息，那么就删除该表项。

可以使用命令 **show ipv6 neighbors** 来显示路由器的邻居缓存表。如例 5-10 所示，路由器 R1 的邻居缓存表中有一条关于 2001:db8:aaaa:1::100（PC1 的地址）的表项，由于可达时间已到期，因而该表项当前正处于失效状态。**Age** 列显示了表项处于当前状态的时间（以分钟为单位）。使用 **ping** 命令与 PC1 重新建立通信之后，邻居缓存表的状态就会迁移到可达状态。

**例 5-10　显示邻居缓存表**

```
R1# show ipv6 neighbors
IPv6 Address                              Age Link-layer Addr State Interface
2001:db8:aaaa:1::100                       16 0021.9bd9.c644  STALE Fa0/0

R1# ping 2001:db8:aaaa:1::100

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:AAAA:1::100, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R1# show ipv6 neighbors
IPv6 Address                              Age Link-layer Addr State Interface
2001:DB8:AAAA:1::100                        0 0021.9bd9.c644  REACH Fa0/0

R1#
```

另一种数据结构是目的地缓存表（Destination Cache）。与邻居缓存表相似，目的地缓存表维护的是流量最近被发送的目的地列表，包括其他链路或其他网络上的目的地，此时的表项是下一跳路由器的二层地址。目的地缓存表可以是邻居缓存表的子集。

### 2. 地址解析

地址解析是使用邻居请求消息和邻居宣告消息的进程之一，该进程类似于 IPv4 中的 ARP 进程。如图 5-17 所示，PC1 正在 ping PC2。

PC1 在发送回显请求消息之前，必须首先确定 PC2 的二层地址（链路层地址），从而能够将 IPv6 数据包封装到以太网帧中，步骤如下。

**第 1 步**：PC1 向 PC2 的地址 2001:DB8:AAAA:1::200 发起 **ping** 命令。

**第 2 步**：PC1 需要将包含回显请求消息的 IPv6 包封装到以太网帧中。PC1 已经知道 PC2 的 IPv6 地址，因而检查其邻居缓存表以获得相应的二层 MAC 地址，但此时缓存表中还没有关于 2001:DB8:AAAA:1::200 的表项。

**第 3 步**：PC1 暂停发送该 IPv6 包，而是向请求节点多播地址（曾在第 4 章讨论过）发送一条邻居请求消息。邻居请求消息中的目标地址是 **ping** 命令中的 IPv6

地址。稍后将详细讨论该消息以及请求节点多播地址。

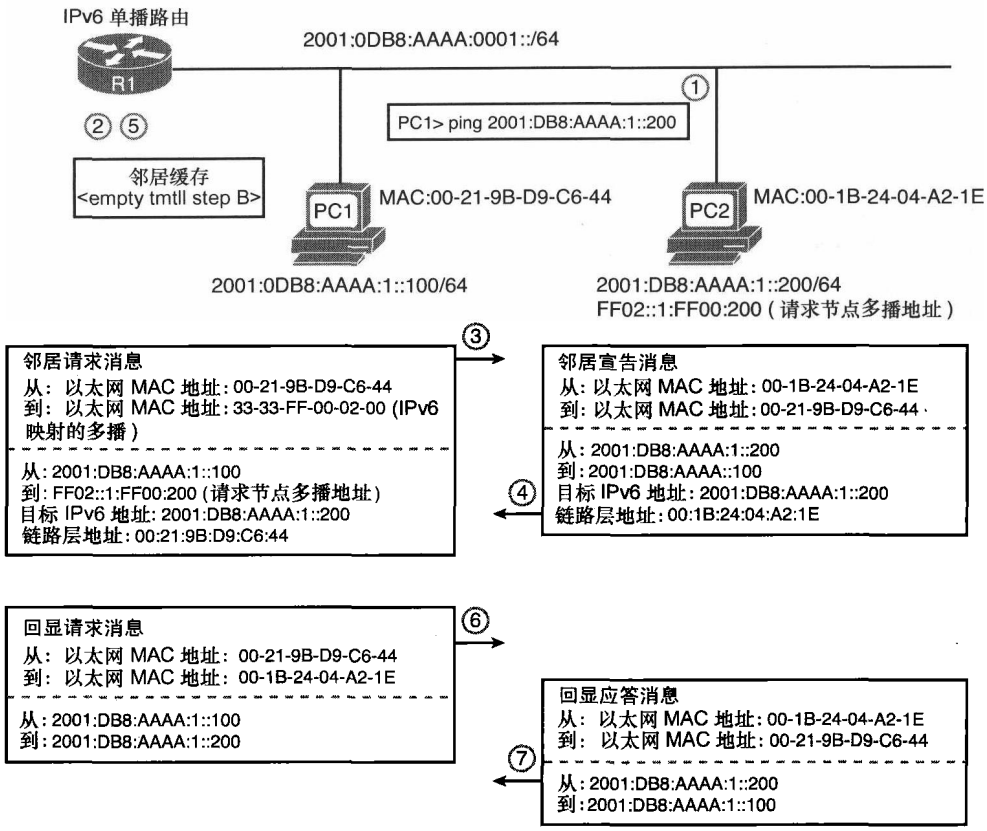



图 5-17　地址解析

**第 4 步**：PC2 收到邻居请求消息之后，向 PC1 响应邻居宣告消息，在消息中提供了其链路层 MAC 地址。邻居宣告消息是以单播方式发送给 PC1 的，大家很快就会知道 PC2 是如何从邻居请求消息中提取 MAC 地址，从而能够快速返回邻居宣告消息的。PC2 将 PC1 的 IPv6 地址和 MAC 地址加入到自己的邻居缓存表中。

**第 5 步**：PC1 收到邻居宣告消息，并以 IPv6 地址 2001:DB8:AAAA:1::200 以及刚刚发现的 PC2 的 MAC 地址 00-1B-24-04-A2-1E 更新其邻居缓存表。

**第 6 步**：PC1 此时已经有了将包含回显请求消息的 IPv6 包封装到以太网帧中的所有信息，因而将以太网帧发送给 PC2。

**第 7 步**：PC2 以回显应答消息响应。

在例 5-11 中，将详细讲解邻居请求信息。

### 例 5-11 PC1 发出的 ND 邻居请求消息

```
! Mapped multicast address for PC2
Ethernet II, Src: 00:21:9b:d9:c6:44, Dst: 33:33:ff:00:02:00

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 32
    Next header: ICMPv6 (0x3a)      ! Next header is an ICMPv6 header
    Hop limit: 255
    Source: 2001:db8:aaaa:1::100    ! Global unicast address
    Destination: ff02::1:ff00:200   ! Solicited-node multicast address

Internet Control Message Protocol v6
    Type: 135 (Neighbor solicitation) ! Neighbor Solicitation message
    Code: 0
    Checksum: 0xbbab [correct]
    Reserved: 0 (Should always be zero)
    Target: 2001:db8:aaaa:1::200  ! Solicited IPv6 address,
                                  ! needing MAC address
    ICMPv6 Option (Source link-layer address)
        Type: Source link-layer address (1)
        Length: 8
        Link-layer address: 00:21:9b:d9:c6:44  ! MAC address of the
                                               ! sender PC1
```

从例 5-11 的 IPv6 报头可以看出，下一报头字段值是十进制 58（十六进制 3A）。这表明紧跟该 IPv6 报头之后的是 ICMPv6 报头。源 IPv6 地址是 PC1 的全局单播地址，目的地址可能有点儿难以识别，它是 PC2 的请求节点多播地址。在 ICMPv6 消息中，类型字段表明这是一条邻居请求消息。目标地址是 PC1 正试图解析（查找相应的 MAC 地址）的 IPv6 地址。链路层地址是 PC1 的 MAC 地址，因而接收端 PC2 有了在以太网帧中封装邻居宣告消息的相关信息。

例 5-12 显示了 PC2 发送给 PC1 的邻居宣告消息的内容。

### 例 5-12 PC2 发出的 ND 邻居宣告消息

```
! Unicast MAC address of PC2
Ethernet II, Src: 00:1b:24:04:a2:1e, Dst: 00:21:9b:d9:c6:44

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
```

```
    Payload length: 32
    Next header: ICMPv6 (0x3a)     ! Next header is and ICMPv6 header
    Hop limit: 255
    Source: 2001:db8:aaaa:1::200        ! Global unicast address
    Destination: 2001:db8:aaaa:1::100   ! Global unicast address

Internet Control Message Protocol v6
    Type: 136 (Neighbor advertisement) ! Neighbor advertisement
                                       ! message
    Code: 0
    Checksum: 0x1b4d [correct]
    Flags: 0x60000000  ! (110) Router Flag = 1,
                       ! Solicitation Flag = 1, Override Flag = 0
    Target: 2001:db8:aaaa:1::200  ! IPv6 address of the sender, PC2
    ICMPv6 Option (Target link-layer address)
        Type: Target link-layer address (2)
        Length: 8
        Link-layer address: 00:1b:24:04:a2:1e ! MAC address of the
                                              ! sender PC2
```

从例 5-12 的 IPv6 报头可以看出，下一报头字段值是十进制 58（十六进制 3A）。这表明紧跟该 IPv6 报头之后的是 ICMPv6 报头。源 IPv6 地址是 PC2 的全局单播地址，目的地址是 PC1 的全局单播地址。在 ICMPv6 消息部分，类型字段表明这是一条邻居宣告消息。标记表明该消息来自于路由器，而且该路由器宣告消息是邻居请求消息的响应消息。目标地址是 PC2 的 IPv6 地址，即邻居宣告消息的源地址。链路层地址是 PC2 的 MAC 地址。PC1 利用这些信息就可以将 IPv6 包封装到以太网帧中。

既然 PC1 有了 PC2 的以太网 MAC 地址，那么就可以发送回显请求消息了（如例 5-13 所示）。例 5-14 给出了相应的由 PC2 发送的回显应答消息。

**例 5-13　PC1 发出的回显请求消息**

```
Ethernet II, Src: 00:21:9b:d9:c6:44, Dst: 00:1b:24:04:a2:1e

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 40
    Next header: ICMPv6 (0x3a)
    Hop limit: 128
    Source: 2001:db8:aaaa:1::100
    Destination: 2001:db8:aaaa:1::200
```

```
Internet Control Message Protocol v6
    Type: 128 (Echo (ping) request)
    Code: 0 (Should always be zero)
    Checksum: 0x7b37 [correct]
    ID: 0x0001
    Sequence: 13
    Data (32 bytes)
```

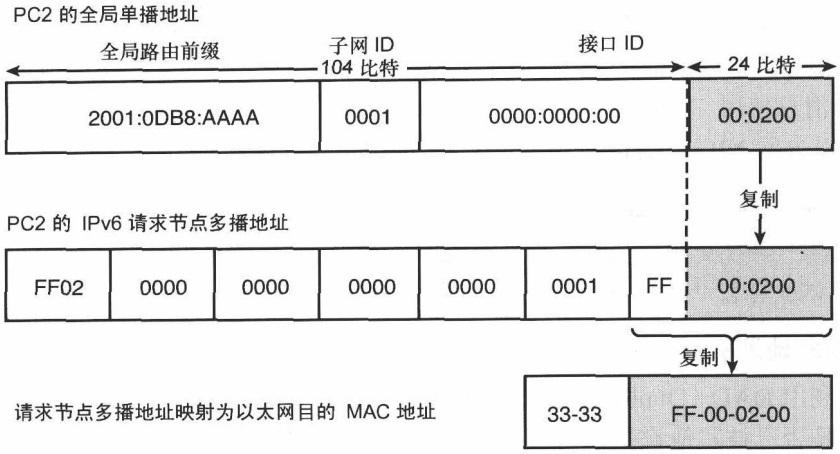
**例 5-14　PC2 发出的回显应答消息**

```
Ethernet II, Src: 00:1b:24:04:a2:1e, Dst: 00:21:9b:d9:c6:44

Internet Protocol Version 6
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
    .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
    Payload length: 40
    Next header: ICMPv6 (0x3a)
    Hop limit: 64
    Source: 2001:db8:aaaa:1::200
    Destination: 2001:db8:aaaa:1::100

Internet Control Message Protocol v6
    Type: 129 (Echo (ping) reply)
    Code: 0 (Should always be zero)
    Checksum: 0x7a37 [correct]
    ID: 0x0001
    Sequence: 13
    Data (32 bytes)
```

### 有关请求节点多播地址的详细信息

第 4 章曾经介绍过请求节点（solicited-node）多播地址。IPv6 接口会接收发往其单播地址的数据包以及发往其所属多播组的数据包。除了常见的多播地址（如全部节点多播地址 FF02::1）之外，IPv6 接口还会接收发往其请求节点多播地址的数据包。

设备会利用特殊的映射技术为其每个单播地址自动创建一个请求节点多播地址。请求节点多播地址是通过将前缀 FF02:0:0:0:0:1:FF00::/104 附加到单播地址最后 24 比特之上创建而成的。如图 5-18 所示，PC2 的请求节点多播地址是前缀 FF02:0:0:0:0:1:FF00::/104 加上其全局单播地址的最后 24 比特 00:0200，两者组合成请求节点多播地址 2001:DB8:AAAA:1::200。此外，PC2 还为其链路本地单播地址生成了一个请求节点多播地址。




PC2 的 IPv6 请求节点多播地址：FF02::1:FF00:200
PC2 映射的请求节点以太网 MAC 地址：33-33-FF-00-02-00

图 5-18　PC2 的请求节点多播地址

回顾例 5-11 的邻居请求消息。注意到目的 IPv6 地址是 PC2 的请求节点多播地址 FF02::1:FF00:200。当 PC2 收到该数据包后，认出该地址是接口的一个请求节点多播地址，因此 PC2 处理该数据包并认出该 ICMPv6 消息的目标地址 2001:DB8:AAAA:1::200 与其全局单播地址相匹配，因而 PC2 知道该邻居请求消息是发送给自己的，并且必须用邻居宣告消息进行响应。

注：为何目的 IPv6 地址是 PC2 的请求节点多播地址而不是其单播地址？这是因为 PC1 虽然认识 PC2 的 IPv6 地址，但是其邻居缓存表中的 2001:db8:aaaa:1::200 表项正处于未完成状态。RFC 2461 和 RFC 4861 详细解释了这个问题。

虽然几率很小，但其他设备的接口 ID 仍然有可能拥有相同的最后 24 比特。也就是说这些接口拥有与 PC2 相同的请求节点多播地址，不过这并不会给处理数据包和检查 ICMPv6 目标地址带来问题。因为在这过程中，其他设备能够确定该目标地址与自己的单播地址并不匹配，因而也就不会用邻居宣告消息进行响应。

至此已经分析了三层 IPv6 报头，接下来分析二层以太网报头。在 IPv4 中，ARP 通常以二层广播的形式发送给目的 MAC 地址 FF-FF-FF-FF-FF-FF。虽然 ARP 仅查找一台设备，但是网络上的每台设备（以太网 NIC）都要在以太网帧中进行复制并将内容传送给各自的 ARP 进程，直至检查 ARP 请求中目标地址的 ARP 进程确定自己是否是 ARP 请求中的目标为止。与之相比较，IPv6 实现该进程的效率则要高得多。

RFC 2464“IPv6 Packets over Ethernet”规定使用以太网多播地址 33-33-xx-xx-xx-xx 来完成该功能。以太网多播地址的最后 4 字节（xx）是目的地址的最后 4 字节（32 比特）。在例 5-11 中，PC1 用于到达 PC2 的目的 MAC 地址是 33-33-FF-00-02-00，也就是将 33-33 附加到请求节点多播地址的最后 32 比特上（如图 5-18 所示）。

PC2 的以太网 NIC 将会接收目的 MAC 地址为 33-33-FF-00-02-00 的所有帧，因为该 MAC 地址是与其请求节点多播地址 FF02::1:FF00:200 相对应的 MAC 地址。与 IPv4 ARP 请求消息使用二层广播地址不同，IPv6 邻居请求消息使用多播地址，因而只有非常有限的以太网 NIC 需要将其复制到以太网帧中。

注：如前所述，网络中多台设备可以拥有相同的请求节点多播地址。也就是说它们的 NIC 会接受相同的以太网多播地址，但是只有 IPv6 地址与作为响应消息的 ICMPv6 邻居宣告中目标地址相匹配的设备才进行处理。

### 3. 重复地址检测

设备利用 DAD（Duplicate Address Detection，重复地址检测）机制来确定其希望使用的地址是否已被其他设备使用。除了某些例外，RFC 4861 建议在将单播地址分配给接口之前，要对每个单播地址（链路本地单播地址或全局单播地址）都执行 DAD 进程，而不管采取何种地址配置方式（包括 SLAAC、DHCPv6 或手工配置方式）。

如果在 DAD 进程中发现了重复地址，那么接口就不能使用该地址。检测重复地址的进程需要用到邻居请求和邻居宣告消息。图 5-19 以 PC1 及其链路本地单播为例解释了该检测进程，相应的步骤如下。

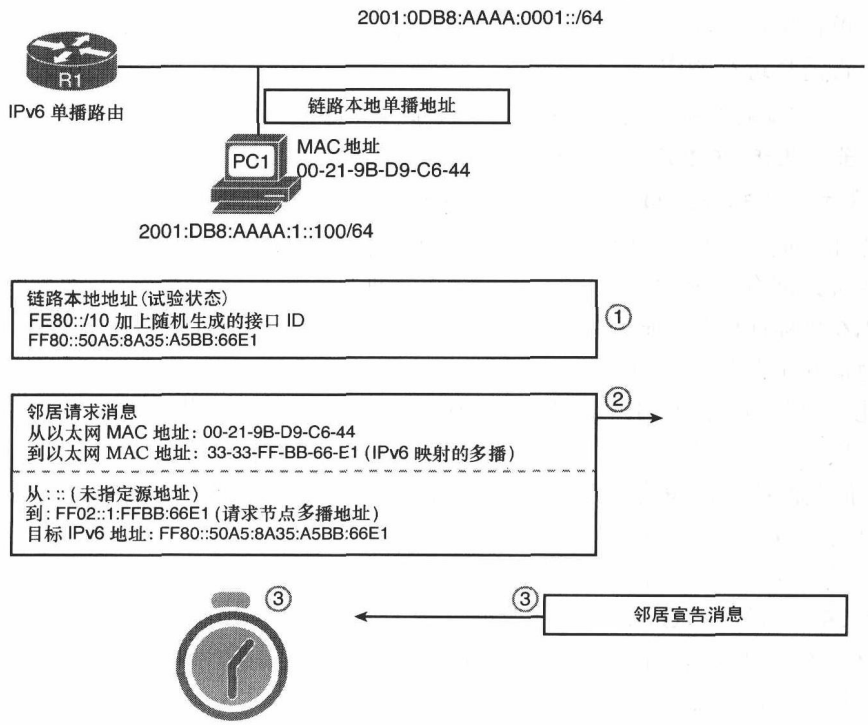



图 5-19 DAD

**第 1 步**：PC1 为其以太网接口自动创建其链路本地单播地址，将前缀 FE80::/10 附加到随机生成的 64 比特接口 ID 上，创建出链路本地地址 FE80::50A5:8A35:A5BB:66E1（由于 PC1 使用的是较新的 Windows 版本，因而不使用 EUI-64 创建接口 ID）。在使用该链路本地地址之前，必须先进行 DAD 检测以确保该地址在链路上的唯一性。在 DAD 进程完成之前，该地址将处于试验（tentative）状态。

**第 2 步**：PC1 发出邻居请求消息以确定链路上是否有其他设备正在使用该链路本地地址，从图 5-19 的邻居请求消息中可以看出，以太网报头信息如下。

- **源 MAC 地址（Source MAC address）**：PC1 的源 MAC 地址是 00-21-9B-D9-C6-44。
- **目的 MAC 地址（Destination MAC address）**：是一个 33-33 多播地址，并且最后 32 比特是从 IPv6 目的地址的低阶 32 比特（BB-66-E1）映射而来，因此目的 MAC 地址是 33-33-BB-66-E1。

IPv6 报头如下。

- **源 IPv6 地址（Source IPv6 address）**：源 IPv6 地址是未指定源地址::。PC1 使用未指定地址的原因是没有有效的 IPv6 地址(DAD 始终将::用作源地址)。
- **目的 IPv6 地址（Destination IPv6 address）**：目的 IPv6 地址是与 PC1 的链路本地地址相对应的请求节点多播地址 FF02::1:FFBB:66E1。

ICMPv6 报头（邻居请求消息）如下。

- **目标 IPv6 地址（Target IPv6 address）**：是 PC1 自己的链路本地地址 FE80::50A5:8A35:A5BB:66E1，如果其他设备也在使用该地址，那么需要用邻居宣告消息进行响应。

**第 3 步**：PC1 设置一个定时器，让正在使用其试验状态的链路本地地址的设备有机会用邻居宣告消息响应。如果在设定时间内未收到响应消息，那么该链路本地地址就会从试验状态迁移到已分配（assigned）状态。如果链路上有其他设备拥有相同的链路本地地址，那么就必须用邻居宣告消息进行响应，以告诉 PC1 知道该地址已被占用，不能再使用该地址，那么 PC1 将会中止使用该地址。首先获取该地址的设备就是保持该地址的设备。

#### 4. 邻居不可达性检测

NUD（Neighbor Unreachability Detection，邻居不可达性检测）定义在 4861 中。由于两台 IPv6 设备之间出现通信故障的原因可能非常多，如主机掉电或电缆故障，因此设备需要主动跟踪数据包将要发往的邻居的可达性状态。

确认可达性的方式有两种：

- 响应邻居请求消息的邻居宣告消息；

■ 上层进程指示连接成功，如活动 TCP 连接中的确认消息。

当去往某邻居的路径可能出现故障时，特定的恢复进程取决于邻居被使用的方式。如果该邻居是最终目的地，那么就需要重新执行地址解析进程，包括发送邻居请求消息并等待响应来的邻居宣告消息；如果该邻居是路由器，那么对该设备来说，较恰当的就是使用其他默认网关。设备仅对单播数据包所送往的邻居执行邻居可达性检测。

本章前面已经讨论了邻居缓存表的状态。图 5-16 解释了各种状态以及在这些状态之间发生的事件。邻居可达性检测机制也利用这些状态和事件来检测与解析邻居的可达性问题。

### 5. 无状态地址自动配置

SLAAC（Stateless Address Autoconfiguration，无状态地址自动配置）是一种允许主机通过组合本地可用信息与路由器宣告的信息生成自己的单播地址的机制。自动配置进程包括通过 SLAAC 生成链路本地地址和全局地址，还包括重复地址检测的步骤，以用来验证地址在链路上的唯一性。

图 5-20 解释了 SLAAC 进程（分为 4 个步骤），其中用到了很多前面已经讨论过的进程和消息。

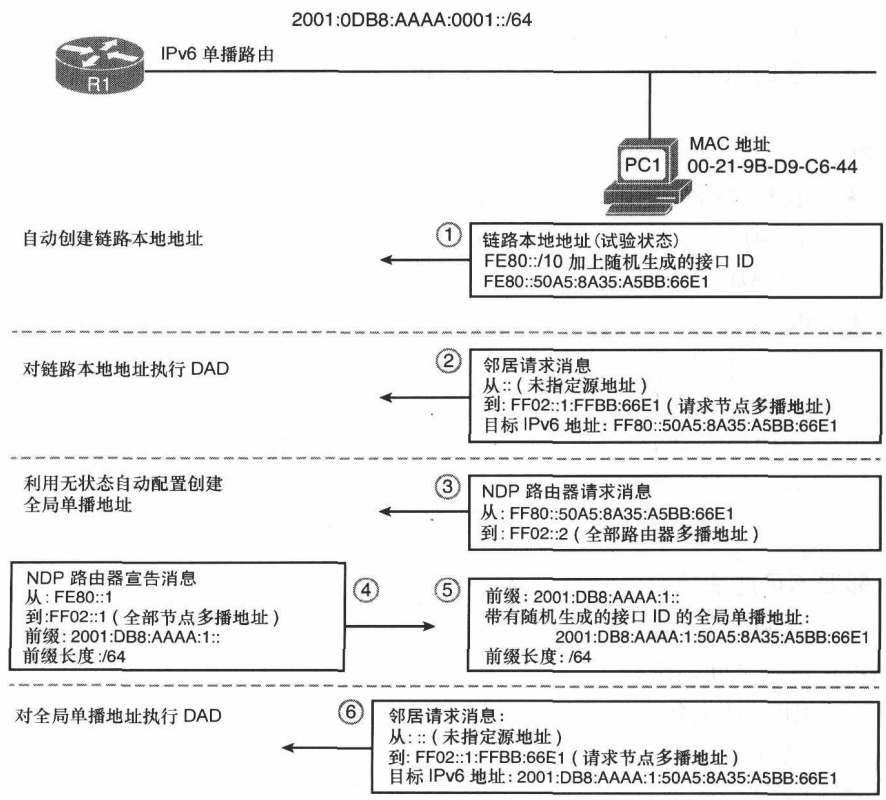



图 5-20 SLAAC

**第 1 步：创建链路本地单播地址**：主机 PC1 创建自己的链路本地单播地址，无需手工配置或借助 DHCPv6 服务器。链路本地地址前缀是 FE80::/10，附加到 64 比特接口 ID 上即可。接口 ID 通过 EUI-64 格式或随机生成方式进行创建（对本案例中的 PC1 而言），然后该地址进入试验状态，并等待 DAD 进程验证其唯一性。

**第 2 步：对链路本地地址执行 DAD**：在使用该链路本地地址之前，主机必须对该链路本地地址执行 DAD，以确保该地址的唯一性。PC1 发出一条路由器请求消息。请注意，该消息的源地址是未指定地址::，目的地址是自己的与该链路本地地址相关联的请求节点多播地址。ICMPv6 消息中的目标 IPv6 地址是自己的链路本地地址，如果有其他设备正在使用该地址，那么就会响应以邻居宣告消息。如果在等待了特定时间间隔之后，PC1 仍未收到关于该地址的路由器宣告消息，那么该地址就会成为有效的链路本地地址。如果发送了路由器宣告消息，那么就会使用全部节点多播地址 FF02::1 作为目的地址。

**第 3 步：路由器宣告消息提供地址配置信息**：PC1 需要从路由器宣告消息获取其全局单播地址信息和其他配置信息。如果 PC1 未收到路由器宣告消息，那么就会向路由器发送路由器请求消息。路由器请求消息是从上一步创建的 PC1 的链路本地地址发出的，且发送给全部路由器多播地址 FF02::2。

路由器 R1 周期性地发送路由器宣告消息，或根据路由器请求消息发送相对应的路由器宣告消息。无论是哪一种情况，路由器宣告消息都是发送给全部节点多播地址 FF02::1，且都是从本地链路地址发出的。路由器宣告消息中包含了前缀、前缀长度及链路 MTU 等信息。

利用路由器宣告消息提供的信息，PC1 将前缀附加到随机生成的 64 比特接口 ID，即可创建其全局单播地址（PC1 生成接口 ID 的另一种选项是使用 EUI-64 格式）。PC1 还从路由器宣告消息中获得了前缀长度、MTU 以及其他配置信息，默认网关就是路由器 R1 的链路本地地址（即路由器宣告消息的源地址），该地址会被加入 PC1 的默认网关列表中。

**第 4 步：对全局单播地址执行 DAD**：将全局单播地址分配给接口之前，PC1 需要执行 DAD 以验证该地址的唯一性。PC1 发出一条邻居请求消息，该消息的源地址是未指定地址::，目的地址是自己的与该全局单播 IPv6 地址相对应的请求节点多播地址。ICMPv6 消息中的目标 IPv6 地址是自己的全局单播地址。请注意，Windows 主机为链路本地地址和全局单播地址的接口 ID 使用相同的随机生成的 64 比特值。如果有其他设备正在使用该全局单播地址，那么就会响应以邻居宣告消息。如果在等待了特定时间间隔之后，PC1 仍未收到关于该地址的路由器宣告消息，那么该地址就会成为有效的

全局单播地址。

### 5.4.3 重定向消息

ICMPv6 重定向消息的作用是通知设备有更优的下一跳路由器，其工作方式与 IPv4 中的重定向消息相同。图 5-21 显示了 ND ICMPv6 重定向消息的格式。

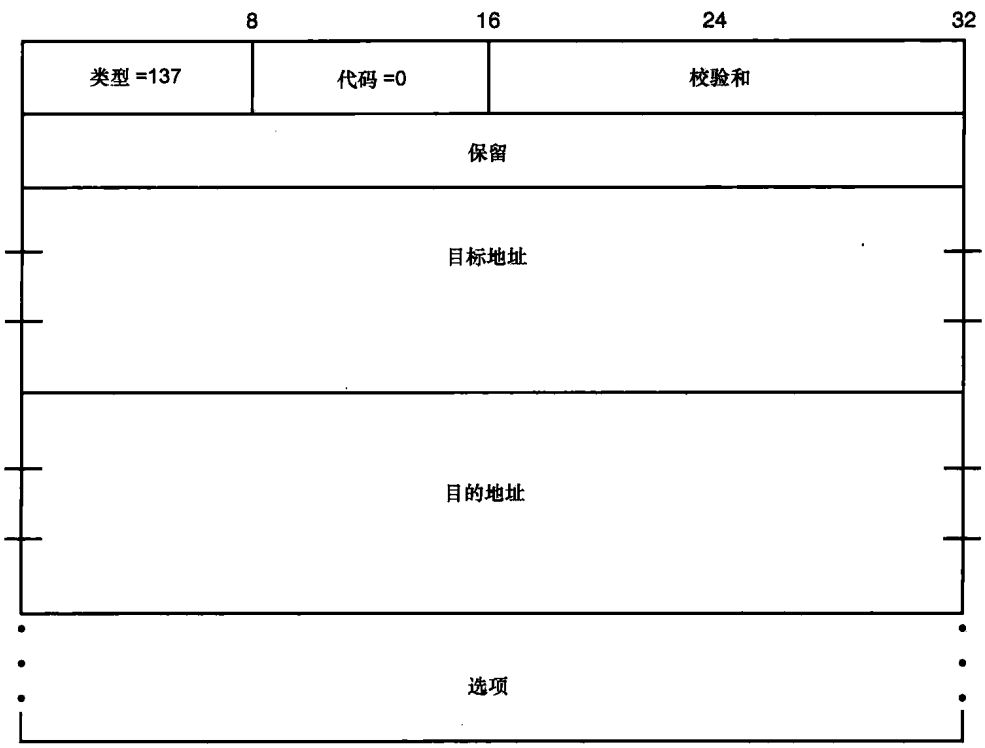



图 5-21 ICMPv6 重定向消息

IPv6 报头和 ICMPv6 报头中的字段信息如下。

- **IPv6 报头：**
  - **源地址：**分配给接口的链路本地地址。
  - **目的地址：**是触发该重定向消息的数据包的源地址。
  - **跳数限制：**跳数限制总是被设置为 255。
- **ICMPv6 字段：**
  - **类型（Type）：**该字段被设置为 137，表示这是一条重定向消息。
  - **代码（Code）：**该字段被设置为 0，表示被接收端忽略。
  - **校验和（Checksum）：**用于验证 ICMPv6 报头。
  - **保留（Reserved）：**该字段未使用。

- **目标地址（Target Address）**：是将要使用的更优下一跳的 IPv6 地址。
- **目的地址（Destination Address）**：是被重定向到目标地址的目的 IP 地址。
- **可能的选项—目标链路层地址（Possible Option–Target Link Layer Address）**：是目标地址（建议的下一跳路由器）的链路层地址，该选项可以避免主机解析其他路由器的链路层地址。

图 5-22 解释了重定向进程。

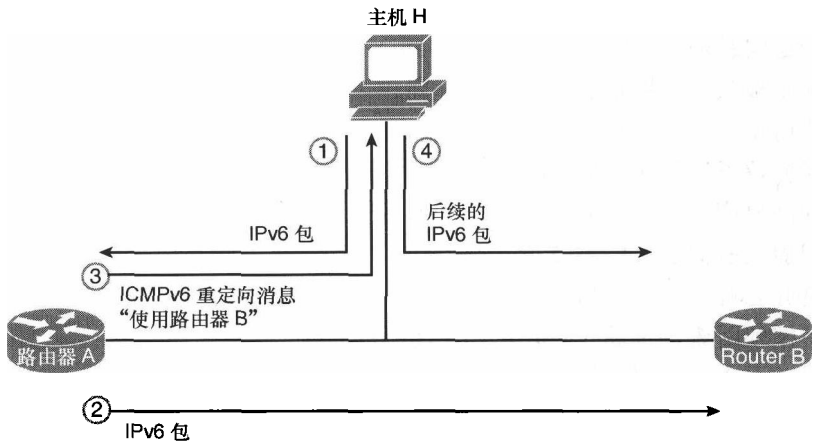



图 5-22　ICMPv6 重定向进程

图 5-22 中的步骤如下。

**第 1 步**：主机 H 向路由器 A 发送数据包，路由器 A 是其目的 IPv6 地址的默认网关。

**第 2 步**：路由器 A 将数据包转发给路由器 B。

**第 3 步**：路由器 A 发现路由器 B 与接收到的数据包位于同一个网络中，因而向主机 H 发送 ICMPv6 重定向消息，以通知主机 H 有更优的下一跳路由器。

**第 4 步**：主机 H 收到重定向消息后，将后续数据包直接发送给路由器 B。

## 5.5　本章小结

本章主要讨论了 ICMPv6。虽然 ICMPv6 在很多方面与 ICMPv4 相同，但是 ICMPv6 是一种更为健壮的协议，包含了很多新功能，并做了大量的功能改善。

本章讨论了两类 ICMPv6 消息：差错消息和通知消息。

本章讨论的 ICMPv6 差错消息如下所示。

- **目的地不可达消息（Destination Unreachable）**：当数据包因拥塞之外的原因

无法被传送时就会发送该消息。

- **数据包超大消息（Packet Too Big）**：当路由器收到的数据包大于其出接口的 MTU 时，就会向源端发送 ICMPv6 数据包超大消息，并且路由器会丢弃该数据包。该消息还被用于路径 MTU 发现进程。
- **超时消息（Time Exceeded）**：路由器在转发 IPv6 包之前会将跳数限制字段（类似于 IPv4 的 TTL 字段）递减 1。当跳数限制字段递减至 0 时，路由器就会丢弃该数据包并向源端发送 ICMPv6 超时消息。
- **参数问题消息（Parameter Problem）**：设备在处理数据包时，如果发现 IPv6 基本报头或扩展报头的字段存在问题，那么就会生成该消息并丢弃该数据包。

本章还讨论了 ICMPv6 通知消息，首先是 ping 命令使用的两个最常见的通知消息。

- **回显请求消息（Echo Request）**：设备通过发送回显请求消息，要求目的端返回回显应答消息以确认网络层的连接性。
- **回显应答消息（Echo Reply）**：回显应答消息是回显请求消息的响应消息。

多播侦听发现协议在主机和路由器之间提供了通信机制，允许主机加入和离开多播组。

- **多播侦听查询消息（Multicast Listener Query）**：路由器周期性地发送主机成员关系查询消息，以确定哪些多播组仍然有成员在路由器直连的网络上。
- **多播侦听者报告（Multicast Listener Report）**：该消息由侦听者发送，其作用是向多播组进行注册。
- **多播侦听者完成（Multicast Listener Done）**：当侦听者不希望接收某特定多播组的流量时，就会发送一条多播侦听者完成消息，以通知路由器其将要离开该多播组。

邻居发现协议包含了与 IPv4 的 ARP、ICMP 路由器发现和重定向等相似的进程，但是也存在很多重要差异。此外，ND 还增加了很多新功能，如 DAD 和 NUD。邻居发现协议用到了以下 5 种 ICMPv6 消息。

- **路由器请求（RS）消息**：当主机需要 SLAAC 所需的前缀、前缀长度、默认网关以及其他信息时，就会发送路由器请求消息。
- **路由器宣告（RA）消息**：路由器宣告消息由路由器周期性地发送，它也可以作为路由器请求消息的响应消息，其作用是为主机提供编址及其他配置信息，是 SLAAC 的重要组成部分。
- **邻居请求（NS）消息**：邻居请求消息和邻居宣告消息非常类似于 IPv4 中的 ARP 请求消息和 ARP 应答消息。邻居请求消息的作用是请求目标设备的二层地址（通常是以太网 MAC 地址），同时也向目标设备提供了自己的链路层地址。
- **邻居宣告（NA）消息**：邻居宣告消息是邻居请求消息的响应消息，作用是提供与 IPv6 地址相对应的二层地址（通常是以太网 MAC 地址）。
- **重定向消息**：ICMPv6 重定向消息的作用是通知设备有更优的下一跳路由器，

其工作方式与 IPv4 的重定向消息相同。

这些 ND 消息被多种表及进程所使用。

- **邻居缓存表（Neighbor Cache）**：等同于 IPv4 中的 ARP 缓存表。邻居缓存表负责维护最近流量所送往邻居的信息列表。
- **目的地缓存表（Destination Cache）**：与邻居缓存表相似，目的地缓存表维护的是流量最近被发送的目的地列表，包括其他链路或其他网络上的目的地。
- **地址解析（Address resolution）**：地址解析使用邻居请求和邻居宣告消息来解析与三层地址相对应的二层（MAC）地址。它与 IPv4 中的 ARP 机制相似。
- **DAD（Duplicate Address Detection，重复地址检测）**：设备利用 DAD 进程来确定其希望使用的地址是否已被其他设备所使用。DAD 以设备自身的地址使用邻居请求消息，如果其他设备正在使用该地址，那么就会响应以邻居宣告消息。
- **NUD（Neighbor Unreachability Detection，邻居不可达性检测）**：设备需要主动跟踪数据包将要发往的邻居的可达性状态，为此使用邻居请求消息和邻居宣告消息来确定邻居的可达性。
- **SLAAC（Stateless Address Autoconfiguration，无状态地址自动配置）**：SLAAC 是一种允许主机通过组合本地可用信息与路由器宣告的信息生成自己的单播地址的机制。自动配置进程包括：
  - 采取随机生成方式或 EUI-64 格式生成的接口 ID 创建自己的链路本地地址；
  - 使用 DAD 进程来确保链路本地地址的唯一性；
  - 通过路由器宣告消息中提供的前缀和前缀长度信息来创建自己的全局单播地址，接口 ID 可以随机生成，也可以采取 EUI-64 格式，路由器宣告消息还提供了默认网关等其他配置信息；
  - 使用 DAD 进程来确保全局单播地址的唯一性。

## 5.6 参考文献

RFC：

RFC 1393, Traceroute Using an IP Option , G. Malkin, Xylogics, Inc., IETF, www.ietf.org/rfc/rfc1393.txt , January 1963

RFC 1981, Path MTU Discovery for IP version 6 , J. McCann, DEC, IETF, www.ietf.org/rfc/rfc1918.txt , August 1996

RFC 2464, Transmission of IPv6 Packets over Ethernet Networks , M. Crawford,

Fermilab, IETF, www.ietf.org/rfc/rfc2464.txt , December 1998
RFC 2710, Multicast Listener Discovery (MLD) for IPv6 , S. Deering, IBM, IETF, www.ietf.org/rfc/rfc2710.txt , October 1999
RFC 2711, IPv6 Router Alert Option , C. Partridge, BBN, IETF, www.ietf.org/rfc/rfc2711.txt , October 1999
RFC 2894, Router Renumbering for IPv6 , M. Crawford, Fermilab, IETF, www.ietf.org/rfc/rfc2894.txt , August 2000
RFC 3122, Extensions to IPv6 Neighbor Discovery for Inverse Discovery Specification , A. Conta, Transwitch Corp., IETF, www.ietf.org/rfc/rfc3122.txt , June 2001
RFC 3775, Mobility Support in IPv6 , D. Johnson, Nokia, IETF, www.ietf.org/rfc/rfc3775.txt , June 2004
RFC 3810, Multicast Listener Discovery Version 2 (MLDv2) for IPv6 , R. Vida, LIP6, IETF, www.ietf.org/rfc/rfc3810.txt , June 2004
RFC 4286, Multicast Router Discovery , B. Haberman, JHU APL, IETF, www.ietf.org/rfc/rfc4286.txt , December 2005
RFC 4443, Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification , A. Conta, Transwitch, IETF, www.ietf.org/rfc/ rfc4443.txt , March 2006
RFC 4861, Neighbor Discovery for IP version 6 (IPv6) , T. Narten, IBM, IETF, www.ietf.org/rfc/rfc4861. txt , September 2007

# 第 6 章　IPv6 配置

前面各章已经讨论了各种类型的 IPv6 地址，如全局单播地址和链路本地地址。还讨论了邻居发现协议以及与地址解析相关的各种消息、DAD 和 SLAAC 等机制。由于 IPv6 不仅仅是更长的 IP 地址，因而全面掌握前面各章讨论过的信息对于理解与 IPv6 网络实施相关的进程和协议来说是至关重要的。不过，有时在大量新信息面前，极有可能会忽略配置 IPv6 所需的各种基本命令，因此，本章将开始介绍与 IPv6 配置相关的命令。

本章将详细解释如何为特定拓扑结构（后面几章也要用到该拓扑结构）配置 IPv6 地址，并介绍 IPv6 路由以及 IPv6 IGP 路由协议，其中很多命令都已经讨论过了。本章也会介绍一些必需的新命令。

接下来，本章将讨论 IPv6 ACL（访问控制列表）。IPv6 ACL 的配置方式与 IPv4 相似，如果大家熟悉 IPv4 ACL 的配置，那么就会发现 IPv6 ACL 的配置几乎完全相同。

图 6-1 给出了本章将要用到的拓扑结构。假设您是当地一家咖啡馆和咖啡烘焙公司 Rick's Café 的网络管理员，并且从提供商那里获得了 IPv6 地址 2001:0db8:cafe::/48。

该网络包含 3 台路由器 R1、R2 和 R3，每台路由器的 Fast Ethernet 0/0 接口上都连接了一个 LAN。

- R1: 2001:0db8:cafe:0001::/64
- R2: 2001:0db8:cafe:0002::/64
- R3: 2001:0db8:cafe:0003::/64

注：IPv6 使用十六进制数字来表示 IPv6 地址，因而大家有机会得到更具描述性的地址。例如，Facebook 的 IPv6 地址是 2620:0:1cfe: face:b00c ::3。

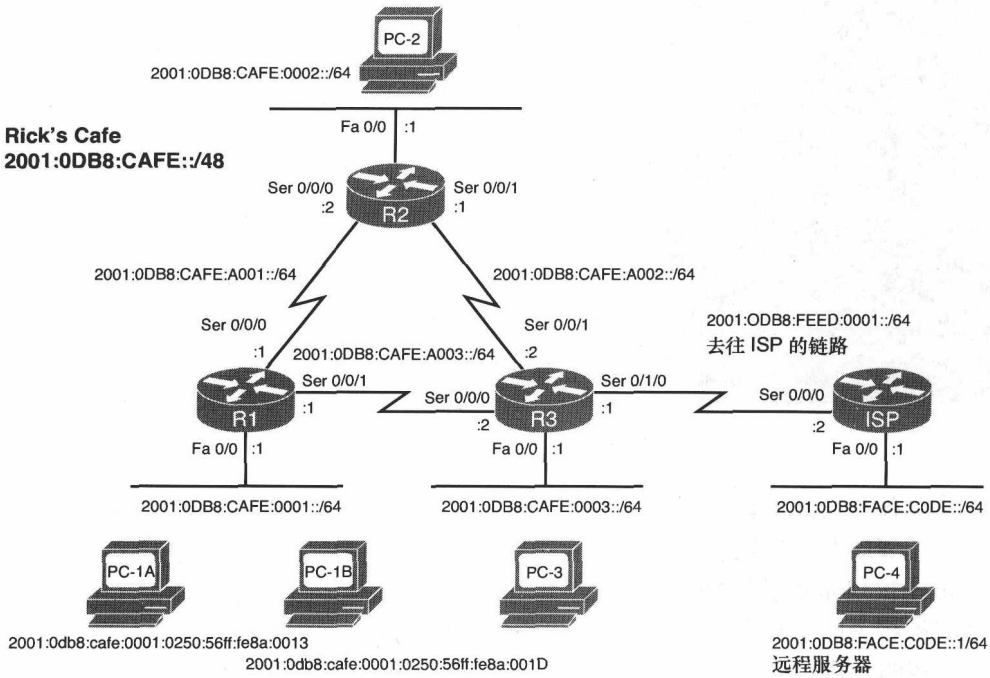



图 6-1 Rick's Café 公司的网络拓扑结构

每台路由器都通过一条点到点串行链路连接在一起。为了更好地标识这些串行连接，设置子网 ID 都以 a 开头，即三个内部串行网络分别如下所示。

- R1 与 R2 之间：2001:0db8:cafe:a001:/64
- R2 与 R3 之间：2001:0db8:cafe:a002:/64
- R3 与 R1 之间：2001:0db8:cafe:a003:/64

Rick's Café 公司通过网络 2001:0db8:feed:0001:/64 连接到 ISP。作为一台示例用的远程服务器，主机 2001:0db8:face:c0de::1/64 连接在 ISP 路由器的快速以太网接口 Fast Ethernet 0/0 上。图 6-1 标示的地址都是全局单播地址。

> 注：该拓扑结构使用此编址方案的目的只是简化对图中网络的标识，并不意味着就是一种正确的编址方案。最佳编址做法取决于网络的复杂度和企业的需求。SURFNET 和 RIPE 就如何制定有效的 IPv6 编址方案提供了一些有用的参考指南：https://labs.ripe.net/Members/steffann/preparing-an-ipv6-addressing-plan。

## 6.1 配置全局单播地址

本节首先讨论在每台路由器上配置全局单播地址。与 IPv4 网络编址相似，最佳做

法是以手工方式配置路由器、服务器和其他网络设备的 IPv6 地址，因为这有助于网络实施和故障排查。

例 6-1 给出了为路由器 R1 配置全局单播地址的示例，使用的是接口级命令 **ipv6 address** *ipv6-address/prefix-length*。

**例 6-1 配置 R1 的全局单播地址**

```
R1# conf t
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 address 2001:0db8:cafe:0001::1/64
R1(config-if)# exit
R1(config)# interface serial 0/0/0
R1(config-if)# ipv6 address 2001:0db8:cafe:a001::1/64
R1(config-if)# exit
R1(config)# interface serial 0/0/1
R1(config-if)# ipv6 address 2001:0db8:cafe:a003::1/64
R1(config-if)# end
R1#

R1# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::21B:CFF:FEC2:82D8
    2001:DB8:CAFE:1::1
Serial0/0/0                [up/up]
    FE80::21B:CFF:FEC2:82D8
    2001:DB8:CAFE:A001::1
Serial0/0/1                [up/up]
    FE80::21B:CFF:FEC2:82D8
    2001:DB8:CAFE:A003::1
R1#
```

例 6-1 还以命令 **show ipv6 interface brief** 验证了这些地址的配置情况，显示了每个接口的链路本地地址和全局单播地址。请注意，只要接口分配了全局单播地址，就会利用 EUI-64 自动创建一个相应的链路本地地址。

例 6-2、例 6-3 和例 6-4 分别给出了路由器 R2、R3 以及 ISP 的配置情况和验证情况。

**例 6-2 配置 R2 的全局单播地址**

```
R2(config)# interface fastethernet 0/0
R2(config-if)# ipv6 address 2001:0db8:cafe:0002::1/64
R2(config-if)# exit
R2(config)# interface serial 0/0/0
R2(config-if)# ipv6 address 2001:0db8:cafe:a001::2/64
```

```
R2(config-if)# exit
R2(config)# interface serial 0/0/1
R2(config-if)# ipv6 address 2001:0db8:cafe:a002::1/64
R2(config-if)# end
R2#

R2# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::21B:53FF:FE87:C050
    2001:DB8:CAFE:2::1
Serial0/0/0                [up/up]
    FE80::21B:53FF:FE87:C050
    2001:DB8:CAFE:A001::2
Serial0/0/1                [up/up]
    FE80::21B:53FF:FE87:C050
    2001:DB8:CAFE:A002::1
R2#
```

## 例 6-3 配置 R3 的全局单播地址

```
R3(config)# interface fastethernet 0/0
R3(config-if)# ipv6 address 2001:0db8:cafe:0003::1/64
R3(config-if)# exit
R3(config)# interface serial 0/0/0
R3(config-if)# ipv6 address 2001:0db8:cafe:a003::2/64
R3(config-if)# exit
R3(config)# interface serial 0/0/1
R3(config-if)# ipv6 address 2001:0db8:cafe:a002::2/64
R3(config-if)# exit
R3(config)# interface serial 0/1/0
R3(config-if)# ipv6 address 2001:0db8:feed:0001::1/64
R3(config-if)# end

R3# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::226:99FF:FE89:88C8
    2001:DB8:CAFE:3::1
Serial0/0/0                [up/up]
    FE80::226:99FF:FE89:88C8
    2001:DB8:CAFE:A003::2
Serial0/0/1                [up/up]
    FE80::226:99FF:FE89:88C8
    2001:DB8:CAFE:A002::2
Serial0/1/0                [up/up]
```

```
    FE80::226:99FF:FE89:88C8
    2001:DB8:FEED:1::1
R3#
```

**例 6-4　配置 ISP 的全局单播地址**

```
ISP(config)# interface fastethernet 0/0
ISP(config-if)# ipv6 address 2001:0db8:face:c0de::1/64
ISP(config-if)# exit
ISP(config)# interface serial 0/0/0
ISP(config-if)# ipv6 address 2001:0db8:feed:0001::2/64
ISP(config-if)# end

ISP# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::226:99FF:FED1:6E90
    2001:DB8:FACE:C0DE::1
Serial0/0/0                [up/up]
    FE80::226:99FF:FED1:6E90
    2001:DB8:FEED:1::2
ISP#
```

## 6.2　配置链路本地地址

为接口分配了全局单播地址之后，就会自动创建相应的链路本地地址。除非采取手工配置方式，否则链路本地地址都是以前缀 FE80::/10 加上接口 ID（使用 EUI-64 或随机生成的值）创建而成的。Cisco IOS 使用的是 EUI-64 格式。如前所述，EUI-64 使用 48 比特以太网 MAC 地址，在中间插入 FFFE 并反转第 7 个比特。对于串行接口来说，Cisco 使用快速以太网接口的 MAC 地址。由于路由器有一个快速以太网接口和至少一个串行接口，因此会出现多个接口共用同一个链路本地地址的情况。由于链路本地地址只要求在链路上具有唯一性即可，因而这样做是可以接受的。

链路本地地址在 IPv6 中起了非常重要的作用。如第 5 章所述，邻居发现协议的路由器请求消息和路由器宣告消息使用的都是链路本地地址。在使用 SLAAC 时，主机也要将路由器的链路本地地址作为其默认网关。后面的章节还会讨论链路本地地址在 IPv6 路由协议中的使用情况。对 IPv6 接口来说，虽然并不强制要求配置全局单播地址，但链路本地地址却是必不可少的。

由于使用 EUI-64 格式的链路本地地址，会出现难以识别数据包的源端和目的端的情况，或者不易验证 IPv6 路由协议中的邻居邻接的情况，因此，一种好的方式是采取

手工方式配置路由器的链路本地地址，让其易于识别也易于记忆。

例 6-5 给出了为 R1 接口配置链路本地地址的示例。利用以下命令为 R1 的每个接口都配置了相同的链路本地地址。

```
Router(config-if)# ipv6 address ipv6-address link-local
```

**例 6-5 配置 R1 的链路本地地址**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 address fe80::1 link-local
R1(config-if)# exit
R1(config)# interface serial 0/0/0
R1(config-if)# ipv6 address fe80::1 link-local
R1(config-if)# exit
R1(config)# interface serial 0/0/1
R1(config-if)# ipv6 address fe80::1 link-local
R1(config-if)# end
R1#
R1# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::1
    2001:DB8:CAFE:1::1
Serial0/0/0                [up/up]
    FE80::1
    2001:DB8:CAFE:A001::1
Serial0/0/1                [up/up]
    FE80::1
    2001:DB8:CAFE:A003::1
R1#
```

利用命令 **show ipv6 interface brief** 可以验证这些地址的配置情况。例 6-6、例 6-7 以及例 6-8 分别给出了路由器 R2、R3 以及 ISP 的链路本地地址的配置情况和验证情况（示例使用了易于识别每台路由器的接口 ID）。

- R1（所有接口）：FE80::1；
- R2（所有接口）：FE80::2；
- R3（所有接口）：FE80::3；
- ISP（所有接口）：FE80::FEED。

**例 6-6 配置 R2 的链路本地地址**

```
R2(config)# interface fastethernet 0/0
R2(config-if)# ipv6 address fe80::2 link-local
R2(config-if)# exit
R2(config)# interface serial 0/0/0
```

```
R2(config-if)# ipv6 address fe80::2 link-local
R2(config-if)# exit
R2(config)# interface serial 0/0/1
R2(config-if)# ipv6 address fe80::2 link-local
R2(config-if)# end
R2#
R2# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::2
    2001:DB8:CAFE:2::1
Serial0/0/0                [up/up]
    FE80::2
    2001:DB8:CAFE:A001::2
Serial0/0/1                [up/up]
    FE80::2
    2001:DB8:CAFE:A002::1
R2#
```

## 例 6-7　配置 R3 的链路本地地址

```
R3(config)# interface fastethernet 0/0
R3(config-if)# ipv6 address fe80::3 link-local
R3(config-if)# exit
R3(config)# interface serial 0/0/0
R3(config-if)# ipv6 address fe80::3 link-local
R3(config-if)# exit
R3(config)# interface serial 0/0/1
R3(config-if)# ipv6 address fe80::3 link-local
R3(config-if)# exit
R3(config)# interface serial 0/1/0
R3(config-if)# ipv6 address fe80::3 link-local
R3(config-if)# end
R3#
R3# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::3
    2001:DB8:CAFE:3::1
Serial0/0/0                [up/up]
    FE80::3
    2001:DB8:CAFE:A003::2
Serial0/0/1                [up/up]
    FE80::3
    2001:DB8:CAFE:A002::2
Serial0/1/0                [up/up]
```

```
    FE80::3
    2001:DB8:FEED:1::1
R3#
```

例 6-8 配置 ISP 的链路本地地址

```
ISP(config)# interface fastethernet 0/0
ISP(config-if)# ipv6 address fe80::feed link-local
ISP(config-if)# exit
ISP(config)# interface serial 0/0/0
ISP(config-if)# ipv6 address fe80::feed link-local
ISP(config-if)# end
ISP#
ISP# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::FEED
    2001:DB8:FACE:C0DE::1
FastEthernet0/1            [administratively down/down]
Serial0/0/0                [up/up]
    FE80::FEED
    2001:DB8:FEED:1::2
Serial0/0/1                [administratively down/down]
ISP#
```

## 6.3 ipv6 enable 命令

为接口分配了全局单播地址之后，就会自动创建相应的链路本地地址。即使接口上没有全局单播地址和唯一本地单播地址，链路本地地址也是必不可少的 IPv6 地址。大家可能会经常遇到在没有指定全局单播地址或唯一本地单播地址的情况下，需要在接口上启用 IPv6，此时可以按照如下方式应用 **ipv6 enable** 命令：

```
Router(config-if)# ipv6 enable
```

例 6-9 给出了在 R1 的快速以太网接口 Fast Ethernet 0/1 上使用该命令的示例。第一条 **show ipv6 interface fastethernet 0/1** 命令无输出结果，表明还未在该接口上启用 IPv6，此时还没有为接口分配任何 IPv6 地址。

例 6-9 在 R1 的快速以太网接口 Fast Ethernet 0/1 配置 ipv6 enable 命令

```
R1# show ipv6 interface fastethernet 0/1

R1# conf t
```

```
R1(config)# interface fastethernet 0/1
R1(config-if)# ipv6 enable
R1(config-if)# end

R1# show ipv6 interface fastethernet 0/1
FastEthernet0/1 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::21B:CFF:FEC2:82D9
  No Virtual link-local address(es):
  No global unicast address is configured
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FFC2:82D9
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ICMP unreachables are sent
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  ND advertised default router preference is Medium
  Hosts use stateless autoconfig for addresses.
R1#
```

从例 6-9 的输出结果可以看出，命令 **ipv6 enable** 为接口自动创建了一个链路本地地址。此时再使用命令 **show ipv6 interface fastethernet 0/1** 即可看出，在该接口没有全局单播地址的情况下，已经有了链路本地地址。利用命令 **show ipv6 interface brief** 即可确认，快速以太网接口 Fast Ethernet 0/1 只有链路本地地址，而没有全局单播地址（如例 6-10 所示）。

**例 6-10　验证 R1 快速以太网接口 Fast Ethernet 0/1 上的链路本地地址**

```
R1# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::1
    2001:DB8:CAFE:1::1
FastEthernet0/1            [up/up]
    FE80::21B:CFF:FEC2:82D9
Serial0/0/0                [up/up]
    FE80::1
    2001:DB8:CAFE:A001::1
Serial0/0/1                [up/up]
    FE80::1
    2001:DB8:CAFE:A003::1
R1#
```

请注意，命令 **ipv6 enable** 并不禁止在接口上配置全局单播地址或唯一本地单播地址。以上案例只是为了说明不需要为接口配置这些单播地址。此外，可以采取手工方式配置接口的链路本地地址，此时会覆盖采取 EUI-64 生成的链路本地地址。

## 6.4 以 EUI-64 选项配置全局单播地址

绝大多数情况下，采取手工方式配置接口的全局单播接口都是最佳做法。不过，在很多场合下，使用 EUI-64 技术生成接口 ID 的配置方式也是非常有用的。例 6-11 以路由器 R1 的快速以太网接口 Fast Ethernet 0/1 为例加以说明。

例 6-11 中的命令 **show interface fastethernet 0/1** 显示了 MAC 地址 001b.0cc2.82d9，该 MAC 地址将用于全局单播地址的接口 ID。

**例 6-11 验证 R1 快速以太网接口 Fast Ethernet 0/1 上的 MAC 地址**

```
R1# show interface fastethernet 0/1
FastEthernet0/1 is up, line protocol is up
  Hardware is MV96340 Ethernet, address is 001b.0cc2.82d9 (bia 001b.0cc2.82d9)
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
<rest of output omitted for brevity)
```

使用 EUI-64 技术配置全局单播地址的接口级命令如下所示：

```
Router(config-if)# ipv6 address ipv6-prefix/prefix-length eui-64
```

如例 6-12 所示，R1 采取 EUI-64 格式配置了其快速以太网接口 Fast Ethernet 0/1 上的全局单播地址。

**例 6-12 使用 EUI-64 格式配置 R1 的快速以太网接口 Fast Ethernet 0/1**

```
R1(config)# interface fastethernet 0/1
R1(config-if)# ipv6 address 2001:0db8:cafe:1234::/64 ?
  anycast  Configure as an anycast
  eui-64   Use eui-64 interface identifier
  <cr>

R1(config-if)# ipv6 address 2001:0db8:cafe:1234::/64 eui-64
R1(config-if)# end

R1# show ipv6 interface fastethernet 0/1
FastEthernet0/1 is up, line protocol is up
```

```
  IPv6 is enabled, link-local address is FE80::21B:CFF:FEC2:82D9
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:CAFE:1234:21B:CFF:FEC2:82D9, subnet is 2001:DB8:CAFE:1234::/64 [EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FFC2:82D9
<output omitted for brevity>
R1#
```

例 6-12 中的命令 **show interface fastethernet 0/1** 显示的全局单播地址是 2001:0DB8:CAFE:1234:021B:0CFF:FEC2:82D9。该地址的创建方式如下。

- **前缀 2001:DB8:CAFE:1234::**：该前缀是由 **ipv6 address** 命令中的 *ipv6-prefix* 分配的。
- **使用 EUI-64 进程生成的接口 ID 021B:0CFF:FEC2:82D9**。
  - 接口的 MAC 地址：001b.0cc2.82d9。
  - 在 MAC 地址中间插入 FFFE。
  - 反转第 7 比特，将十六进制值由 0 更改为 2。

从例 6-13 中的 **show running-config** 命令可以看出，R1 的快速以太网接口 Fast Ethernet 0/1 采取 EUI-64 方式配置了其全局单播地址。

**例 6-13　利用 show running-config 命令检查 R1 的接口配置**

```
R1# show running-config
!
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:1::1/64
!
interface FastEthernet0/1
 no ip address
 duplex auto
 speed auto
 ipv6 address 2001:DB8:CAFE:1234::/64 eui-64
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A001::1/64
```

```
 clock rate 64000
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A003::1/64
```

从例 6-13 的输出结果可以看出，运行配置并不显示真实地址，如在其他接口上手工配置的地址，而且运行配置中显示的只有链路本地地址是手工配置的。例 6-13 中的 Fast Ethernet 0/1 的链路本地地址是使用 EUI-64 自动分配的。

## 6.5 删除 IPv6 地址

命令 **no ipv6 address** 的作用是删除接口的 IPv6 地址。如果该命令没有包含特定地址，那么该接口上的所有 IPv6 地址都会被删除，包括全局单播地址和链路本地地址。**no ipv6 address** 命令的格式如下：

```
Router(config-if)# no ipv6 address [ipv6-address/prefix-length]
```

**ipv6 enable** 命令有一个例外。如果之前使用 **ipv6 enable** 命令配置了接口，那么 **no ipv6 address** 命令只能删除除链路本地地址之外的所有 IPv6 地址。必须使用 **no ipv6 enable** 命令才能删除链路本地地址。

在例 6-14 中，路由器 R1 的快速以太网接口的所有 IPv6 地址都被删除了。首先，使用 **no ipv6 address** 命令删除全局单播地址（使用该命令时可以不指定任何 IPv6 前缀和前缀长度）。由于之前使用了 **ipv6 enbale** 命令，因而必须使用 **no ipv6 enable** 命令来删除链路本地地址。从 **show ipv6 fastethernet 0/1** 和 **show ipv6 interface brief** 的输出结果可以看出，该接口的所有 IPv6 地址都被删除了。

**例 6-14　删除 R1 的快速以太网接口 Fast Ethernet 0/1 的所有 IPv6 地址**

```
R1(config)# interface fastethernet 0/1
R1(config-if)# no ipv6 address 2001:DB8:CAFE:1234:21B:CFF:FEC2:82D9/64
R1(config-if)# no ipv6 enable
R1(config-if)# end
R1#
R1# show ipv6 interface fastethernet 0/1

R1#

R1# show ipv6 interface fastethernet 0/1
```

```
R1#show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::1
    2001:DB8:CAFE:1::1
FastEthernet0/1            [up/up]
    unassigned
Serial0/0/0                [up/up]
    FE80::1
    2001:DB8:CAFE:A001::1
Serial0/0/1                [up/up]
    FE80::1
    2001:DB8:CAFE:A003::1
R1#
```

## 6.6　启用 IPv6 包转发与 ND 路由器宣告

IPv6 路由器除了可以转发 IPv6 包，还可以利用 ND 路由器宣告消息来通告前缀、前缀长度、默认网关以及其他配置。为了启用路由器的 IPv6 单播包转发功能，需要使用全局配置命令 **ipv6 unicast-routing**：

```
Router(config)# ipv6 unicast-routing
```

如果禁用了命令 **ipv6 unicast-routing**，虽然仍能为路由器配置 IPv6 接口。但是禁用命令 **ipv6 unicast-routing** 会产生以下后果：

- 禁止转发 IPv6 单播包；
- 禁止配置静态 IPv6 路由或动态 IPv6 路由协议；
- 禁止发送邻居发现协议使用的 ICMPv6 路由器宣告消息。

如例 6-15 所示，路由器 R1 以及拓扑结构中的其他路由器都应用了命令 **ipv6 unicast-routing**，从 **debug ipv6 nd** 命令的输出结果可以看出，路由器 R1 的 Fast Ethernet 0/0 接口已经发出了 ND 路由器宣告消息。

**例 6-15　验证路由器 R1 上的路由器宣告消息**

```
R1(config)# ipv6 unicast-routing

R1# debug ipv6 nd
ICMP Neighbor Discovery events debugging is on
R1#
*Feb 11 18:03:23.439: ICMPv6-ND: Request to send RA for FE80::21B:CFF:FEC2:82D8
*Feb 11 18:03:23.439: ICMPv6-ND: Sending RA from FE80::21B:CFF:FEC2:82D8 to FF02::1
on FastEthernet0/0
```

```
*Feb 11 18:03:23.439: ICMPv6-ND:      MTU = 1500
*Feb 11 18:03:23.439: ICMPv6-ND:      prefix = 2001:DB8:CAFE:1::/64 onlink autoconfig
*Feb 11 18:03:23.439: ICMPv6-ND:             2592000/604800 (valid/preferred)
R1#
```

拓扑结构中的 PC-1A 和 PC-1B 都是 R1 的 LAN 上的主机，而且都被配置为自动获取其 IPv6 地址。路由器 R1 从接口 Fast Ethernet 0/0 发出包含了前缀、前缀长度和默认网关信息的 ND 路由器宣告消息。这两台主机使用的操作系统都是 Windows XP，该操作系统使用 EUI-64 格式创建其接口 ID。例 6-16 显示了 **ipconfig** 命令的输出结果。请注意，这两台主机的全局单播地址和链路本地地址的接口 ID 都是通过它们的以太网 MAC 地址和 EUI-64 技术生成的。

**例 6-16　PC-1A 和 PC-1B 的 IPv6 地址**

```
PC-1A> ipconfig

Windows XP IP Configuration

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2001:0db8:cafe:0001:0250:56ff:fe8a:0013
   Link-local IPv6 Address . . . . . : FE80::0250:56ff:fe8a:0013
   Default Gateway . . . . . . . . . : FE80:021b:0cff:0fec2:82d8

PC-1B> ipconfig

Windows XP IP Configuration

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2001:0db8:cafe:0001:0250:56ff:fe8a:001D
   Link-local IPv6 Address . . . . . : FE80::0250:56ff:fe8a:001D
   Default Gateway . . . . . . . . . : FE80:021b:0cff:0fec2:82d8
```

PC-1A 的 MAC 地址 00-50-56-8A-00-13，PC-1B 的 MAC 地址 00-50-56-8A-00-1D。

注：在以前的案例中，主机使用的都是 Windows Vista 或更新的操作系统，这些操作系统会为接口 ID 生成随机的 64 比特值。本案例的主机使用的是 Windows XP，目的是举例说明主机对 EUI-64 技术的使用情况。

## 6.7　邻居缓存表

IPv6 邻居缓存表或邻居发现缓存表类似于 IPv4 中的 ARP 缓存表。可以利用命令 **show ipv6 neighbors** 显示邻居缓存表。

在例 6-17 中，R1 ping 主机 PC-1A 的全局单播地址。与 IPv4 中的 ARP 相似，R1 发出一条邻居请求消息以请求获得PC-1A的二层MAC地址。PC-1A使用包含了其MAC地址的 ND 邻居宣告消息进行响应。R1 使用 PC-1A 的 MAC 地址封装其 ICMPv6 回显请求消息并发送给 PC-1A，PC-1A 发生 ICMPv6 回显应答消息。在路由器 R1 上应用命令 **show ipv6 neighbors**，即可在输出结果中看到 R1 上的邻居缓存表以及 PC-1A 的 IPv6 地址和相应的 MAC 地址。

**例 6-17　ping PC-1A 并显示邻居缓存表**

```
R1# ping 2001:db8:cafe:1:250:56ff:fe8a:13

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:1:250:56FF:FE8A:13, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/4 ms
R1# show ipv6 neighbors
IPv6 Address                              Age Link-layer Addr State Interface
2001:DB8:CAFE:1:250:56FF:FE8A:13            0 0050.568a.0013  REACH Fa0/0

R1#
```

在例 6-18 中，R1 ping 主机 PC-1B。目前 R1 的邻居缓存表中已经包含了 PC-1A 和 PC-1B 的 IPv6 地址和 MAC 地址。这两台主机的链路本地地址也都因为它们发送的邻居请求消息而位于邻居缓存表中了。

**例 6-18　ping PC-1B 并显示邻居缓存表**

```
R1# ping 2001:db8:cafe:1:250:56ff:fe8a:1d

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:1:250:56FF:FE8A:1D, timeout is 2
seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/1/4 ms
R1#
```

```
R1# show ipv6 neighbors
IPv6 Address                              Age Link-layer Addr State Interface
2001:DB8:CAFE:1:250:56FF:FE8A:13            0 0050.568a.0013  STALE Fa0/0
2001:DB8:CAFE:1:250:56FF:FE8A:1D            0 0050.568a.001d  REACH Fa0/0
FE80::250:56FF:FE8A:13                      0 0050.568a.0013  STALE Fa0/0
FE80::250:56FF:FE8A:1D                      0 0050.568a.001d  REACH Fa0/0

R1#
```

如第 5 章所述，邻居缓存表是通过邻居请求消息和邻居宣告消息动态填充的，这与 IPv4 中的 ARP 缓存表类似。静态表项也可以插入邻居缓存表中。例 6-19 解释了如何利用以下全局配置命令将静态表项插入 R1 的邻居缓存表中。

```
Router(config)# ipv6 neighbor ipv6-address interface-type interface-number
hardware-address
```

使用以下命令可以清除邻居缓存表：

```
Router# clear ipv6 neighbors
```

**例 6-19　向 R1 的邻居缓存表添加静态表项**

```
R1(config)# ipv6 neighbor 2001:db8:cafe:1::1234 fastethernet 0/0 0050.568a.5555
R1(config)# end

R1# show ipv6 neighbors
IPv6 Address                              Age Link-layer Addr State Interface
2001:DB8:CAFE:1:250:56FF:FE8A:13          208 0050.568a.0013  STALE Fa0/0
2001:DB8:CAFE:1::1234                       -  0050.568a.5555  REACH Fa0/0
2001:DB8:CAFE:1:250:56FF:FE8A:1D          206 0050.568a.001d  STALE Fa0/0
FE80::250:56FF:FE8A:13                    207 0050.568a.0013  STALE Fa0/0
FE80::250:56FF:FE8A:1D                    205 0050.568a.001d  STALE Fa0/0

R1#
```

## 6.8　调节邻居发现参数

第 5 章详细讨论了邻居发现协议以及邻居宣告（RA）消息。路由器可以周期性地发出 RA 消息，也可以将 RA 消息作为路由器请求（RS）消息的响应消息，用于为主机提供编址及其他配置信息，是 SLAAC 的重要组成部分。在默认情况下，Cisco 路由器会在以太网接口、快速以太网接口、吉比特以太网接口以及 10 吉比特以太网接口上发

送路由器宣告消息。

如例 6-20 所示，命令 **show ipv6 interface** 的输出结果显示了路由器宣告消息的大多数参数。

**例 6-20　默认的 ND 路由器宣告消息参数**

```
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::1
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:CAFE:1::1, subnet is 2001:DB8:CAFE:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ICMP unreachables are sent
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
! RA interval:
  ND router advertisements are sent every 200 seconds
! RA lifetime:
  ND router advertisements live for 1800 seconds
  ND advertised default router preference is Medium
! RA Managed-Config-Flag is 0. Hosts are to use stateless, not stateful
! configuration:
  Hosts use stateless autoconfig for addresses.
R1#
```

利用 **ipv6 nd** 命令可以配置路由器宣告消息以及其他邻居发现消息的参数。例 6-21 给出了邻居发现协议的一些配置选项，特别是在配置 ND 路由器宣告消息时用到的一些配置选项。

**例 6-21　ipv6 nd 命令的配置选项**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 nd ?
  advertisement-interval  Send an advertisement interval option in RA's
  dad                     Duplicate Address Detection
```

```
  managed-config-flag     Hosts should use DHCP for address config
  ns-interval             Set advertised NS retransmission interval
  other-config-flag       Hosts should use DHCP for non-address config
  prefix                  Configure IPv6 Routing Prefix Advertisement
  ra                      Router Advertisement control
  reachable-time          Set advertised reachability time
  router-preference       Set default router preference value

R1(config-if)# ipv6 nd ra ?
  interval  Set IPv6 Router Advertisement Interval
  lifetime  Set IPv6 Router Advertisement Lifetime
  suppress  Suppress IPv6 Router Advertisements

R1(config-if)#
```

IPv6 路由器不一定要参与路由器宣告消息的发送进程。可以使用以下接口级命令来抑制 RA 消息：

```
Router(config-if)# ipv6 nd ra suppress
```

注：命令 **ipv6 nd suppress-ra** 的作用只是抑制周期性的 RA 消息，请求式的 RA 消息（此时的 RA 消息是 RS 消息的响应消息）仍会发生。

注：从 Cisco IOS Release 12.4(2)T 开始，命令 **ipv6 nd suppress-ra** 就已经被 **ipv6 nd ra suppress** 取代了。

在例 6-20 中，命令 **show ipv6 interface** 解释了与路由器宣告消息相关的一些默认参数。其中，RA 间隔（RA interval）是连续两条路由器宣告消息之间的时间间隔（以秒为单位）。默认情况下，Cisco 路由器的 RA 间隔是 200 秒。可以使用 **ipv6 nd ra-interval** 命令调整该参数：

```
Router(config-if)# ipv6 nd ra interval {maximum-secs [minimum-secs]
| msec maximum-ms [minimum-ms]}
```

例 6-22 将 RA 间隔调整为 180 秒。

**例 6-22　配置路由器宣告消息的参数**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 nd ra interval 180
R1(config-if)# ipv6 nd ra lifetime 3600
R1(config-if)# ipv6 nd managed-config-flag
R1(config-if)# ipv6 nd other-config-flag
```

```
R1(config-if)# end
R1# show ipv6 interface fastethernet 0/0
*Feb 12 00:21:44.465: %SYS-5-CONFIG_I: Configured from console by console
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::1
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:CAFE:1::1, subnet is 2001:DB8:CAFE:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ICMP unreachables are sent
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 180 seconds    ! ipv6 nd ra interval 180
  ND router advertisements live for 3600 seconds         ! ipv6 nd ra lifetime 3600
  ND advertised default router preference is Medium
  Hosts use DHCP to obtain routable addresses.           ! ipv6 nd managed-config-flag
  Hosts use DHCP to obtain other configuration.          ! ipv6 nd other-config-flag
R1#
```

RA 生存期（RA lifetime）定义了主机将该路由器视为有效默认网关的持续时间。该参数并不会应用到路由器宣告消息中包含的其他配置参数。Cisco 路由器的默认 RA 生存期是 1800 秒（30 分钟）。RA 生存期为 0 表示该路由器不是默认网关，也不能将其加入主机的默认路由器列表中。可以使用命令 **ipv6 nd ra-lifetime** 来调整 RA 生存期。

```
Router(config-if)# ipv6 nd ra lifetime seconds
```

例 6-22 将 RA 生存期调整为 3600 秒（60 分钟）。

注：在某些文档和某些版本的 IOS 中，用于调整 RA 间隔和 RA 生存期参数的命令，在“**ra**”后面有一个连字符，即 **ipv6 nd ra-interval** 和 **ipv6 nd ra-lifetime**。

路由器宣告消息中的 M 标记和 O 标记都与 SLAAC 有关。M 标记（被管地址配置标记）的作用是告诉被配置为自动获取其配置信息的主机是否使用 SLAAC 或状态化配

置（DHCPv6）。默认情况下，M 标记为 0，也就是告诉主机使用 SLAAC。如果 M 标记被设置为 1，那么就是告诉主机使用状态化配置（DHCPv6）。如果希望将 M 标记设置为 1（DHCPv6），那么就可以使用命令 **ipv6 nd managed-config-flag**：

```
Router(config-if)# ipv6 managed-config-flag
```

例 6-22 将路由器宣告消息的 M 标记设置为 1（DHCPv6）。使用命令 **no ipv6 nd managed-config-flage** 可以将 M 标记设置为默认值 0（SLAAC）。

O 标记（其他配置标记）的作用是告诉主机是否可以从 DHCPv6 服务器获得额外配置信息，如与 DNS 相关的信息。O 标记默认值 0 表示不能从 DHCPv6 服务器获得额外配置信息。如果 O 标记被设置为 1，那么就是告诉主机可以从 DHCPv6 服务器获得额外配置信息。如果希望将 O 标记设置为 1，那么就可以使用命令 **ipv6 nd other-config-flag**：

```
Router(config-if)# ipv6 other-config-flag
```

例 6-22 将路由器宣告消息的 O 标记设置为 1。使用命令 **no ipv6 nd other-config-flag** 可以将 O 标记设置为默认值 0。M 标记和 O 标记的默认值都是 0，表示无法从 DHCPv6 服务器获取信息。

从 R1 的 **debug ipv6 nd** 命令输出结果中可以看出，M 标记和 O 标记都为置位了（如例 6-23 所示）。

**例 6-23　验证 M 标记和 O 标记**

```
R1# debug ipv6 nd
*Feb 12 00:22:59.205: ICMPv6-ND: Request to send RA for FE80::1
*Feb 12 00:22:59.205: ICMPv6-ND: Sending RA from FE80::1 to FF02::1 on
FastEthernet0/0
*Feb 12 00:22:59.205: ICMPv6-ND:      Managed address configuration
*Feb 12 00:22:59.205: ICMPv6-ND:      Other stateful configuration
*Feb 12 00:22:59.205: ICMPv6-ND:      MTU = 1500
*Feb 12 00:22:59.205: ICMPv6-ND:      prefix = 2001:DB8:CAFE:1::/64 onlink autoconfig
*Feb 12 00:22:59.205: ICMPv6-ND:       2592000/604800 (valid/preferred)
R1# undebug all
```

如例 6-24 所示，在每条命令中都使用了选项 **no**，从而将所有 ND 路由器宣告消息都复位了，并且通过命令 **show ipv6 interface** 加以验证。

**例 6-24　将路由器宣告消息的参数配置回默认值**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# no ipv6 nd ra interval 180
R1(config-if)# no ipv6 nd ra lifetime 3600
```

```
R1(config-if)# no ipv6 nd managed-config-flag
R1(config-if)# no ipv6 nd other-config-flag
R1(config-if)# end
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::1
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:CAFE:1::1, subnet is 2001:DB8:CAFE:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ICMP unreachables are sent
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds    ! no ipv6 nd ra interval 180
  ND router advertisements live for 1800 seconds         ! no ipv6 nd ra lifetime 3600
  ND advertised default router preference is Medium
  Hosts use stateless autoconfig for addresses.          ! no ipv6 nd managed-config-flag
                                                         ! no ipv6 nd other-config-flag
R1#
```

如果同一个以太网上存在多台路由器，那么所有路由器都会发出路由器宣告消息，除非使用命令 **ipv6 nd ra suppress** 加以抑制。为了显示来自链路上其他路由器的路由器宣告消息，可以使用以下命令：

```
Router# show ipv6 routers
```

## 6.9　最终配置

例 6-25 显示了路由器 R1、R2、R3 以及 ISP 的最终运行配置文件。请注意，这 4 台路由器上的每个接口配置中都有 **no ip address**，指的是该接口上没有配置 IPv4 地址，而不是 IPv6 地址。

例 6-25 路由器 R1、R2、R3 以及 ISP 的部分运行配置

```
R1# show running-config
!
hostname R1
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:1::1/64
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A001::1/64
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A003::1/64
!
```

```
R2# show running-config
!
hostname R2
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:2::1/64
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:A001::2/64
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:A002::1/64
```

```
R3# show running-config
```

```
!
hostname R3
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:3::1/64
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:A003::2/64
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:A002::2/64
!
interface Serial0/1/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:FEED:1::1/64
!
```

```
ISP# show running-config
!
hostname ISP
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::FEED link-local
 ipv6 address 2001:DB8:FACE:C0DE::1/64
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::FEED link-local
 ipv6 address 2001:DB8:FEED:1::2/64
```

例 6-26 通过 ping 邻居路由器的邻接接口来验证配置情况。但是在 ping 远程网络的接口时，ping 操作却失败了（如例 6-27 所示）。这是因为到现在为止，还没有配置静态路由或动态路由，从而无法到达这些远程网络。这一点将在第 7 章和第 8 章中加以解决。第 7 章将讨论静态 IPv6 路由的配置，第 8 章将揭示动态 IPv6 路由协议的配置。

### 例 6-26 验证串行链路的连接性

```
R1# ping 2001:db8:cafe:a001::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:A001::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
R1# ping 2001:db8:cafe:a003::2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:A003::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
R1#

R3# ping 2001:db8:cafe:a002::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:A002::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
R3# ping 2001:db8:feed:0001::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:FEED:1::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
R3#
```

### 例 6-27 无法与远程网络进行通信

```
R1# ping 2001:db8:cafe:3::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:3::1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
R1# ping 2001:db8:feed::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:FEED::2, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
R1#
```

## 6.10 IPv6 访问控制列表

虽然 IPv6 ACL（Access Control List，访问控制列表）的配置与 IPv4 ACL 相似，但两者之间也有一些显著区别。假定大家已经熟悉了 IPv4 ACL 的配置。IPv4 有两种基本的 ACL：标准 ACL 和扩展 ACL。总体来说，标准 ACL 只能基于源地址来允许或拒绝流量，而扩展 ACL 提供了更多的控制能力，可以根据源地址和目的地址，以及协议和服务（TCP/UDP 端口号）来允许和拒绝流量。IPv6 仅支持名称式（与编号式相对）ACL。

> 注：IPv6 也支持其他类型的 ACL，包括反射式（reflective）ACL、基于时间的（time-based）ACL 以及基于区的（zone-based）ACL。不过这些内容都已超出了本书写作范围，有关这些 IPv6 ACL 的详细信息，可参考 *Cisco IOS IPv6 Configuration Guide,Implementing Traffic Filters and Firewalls for IPv6 Security*: www.cisco.com/en/US/docs/ios/ipv6/ configuration/ guide/ip6- sec_trfltr_fw.html。

无论是标准 ACL，还是扩展 ACL，可用的 ACL 类型完全取决于路由器的 IOS 版本。*Cisco IOS IPv6 Configuration Guide,Implementing Traffic Filters and Firewalls for IPv6 Security* 中谈到：

“从 Cisco IOS Release 12.2(2)T 到 Cisco IOS Release 12.2(13)T，以及 Cisco IOS Release 12.0(22)S 及以后版本都只支持标准的 IPv6 ACL 功能。在 Cisco IOS Release 12.0(23)S 和 12.2(13)T 及以后版本，扩展了标准的 IPv6 ACL 功能，支持基于 IPv6 选项报头的流量过滤机制，而且作为可选项，还可以支持基于上层协议的类型信息来精细化地控制流量过滤操作（类似于 IPv4 中的扩展 ACL 功能）。”

本节将会用到扩展式 IPv6 ACL。

在 IPv4 中，每条访问控制列表的末尾都有一个隐式的 **deny any any** 语句。IPv6 ACL 不但也有相似的隐式语句 **deny ipv6 any any**，而且默认情况下还包含了另外两条语句：

- **permit icmp any any nd-na**
- **permit icmp any any nd-ns**

由于 IPv6 ND（Neighbor Discovery，邻居发现）进程需要用到 IPv6 网络层服务。因此在默认情况下，IPv6 ACL 需要隐式地允许接口发送和接收 IPv6 邻居发现包。也就是说，必须在链路上允许 ND 邻居请求消息和 ND 邻居宣告消息。虽然 IPv4 中的 ARP 消息（等同于 IPv6 邻居发现进程）并不通过 IPv4 进行发送，但 IPv6 的邻居发现消息却要用到 IPv6 服务，且必须隐式包含在 IPv6 ACL 中。

IPv4 ACL 与 IPv6 ACL 之间的另一个显著区别就是对编号访问列表的使用。IPv6 ACL 只能通过唯一的名字加以定义，不支持编号列表。IPv4 ACL 与 IPv6 ACL 无法共享同一个名字。

### 6.10.1 拒绝从 FACE:C0DE 到 CAFE 的访问

IPv6 ACL 的配置与 IPv4 ACL 非常相似。以图 6-2 所示拓扑结构为例，对网络 2001:db8:face:c0de::/48（包括了 R4 的 LAN）发出的流量进行过滤，拒绝这些流量进入网络 2001:db8:cafe::/48。为此，可以在路由器 R3 上配置扩展式 IPv6 ACL，以拒绝来自 ISP 路由器的流量进入网络。

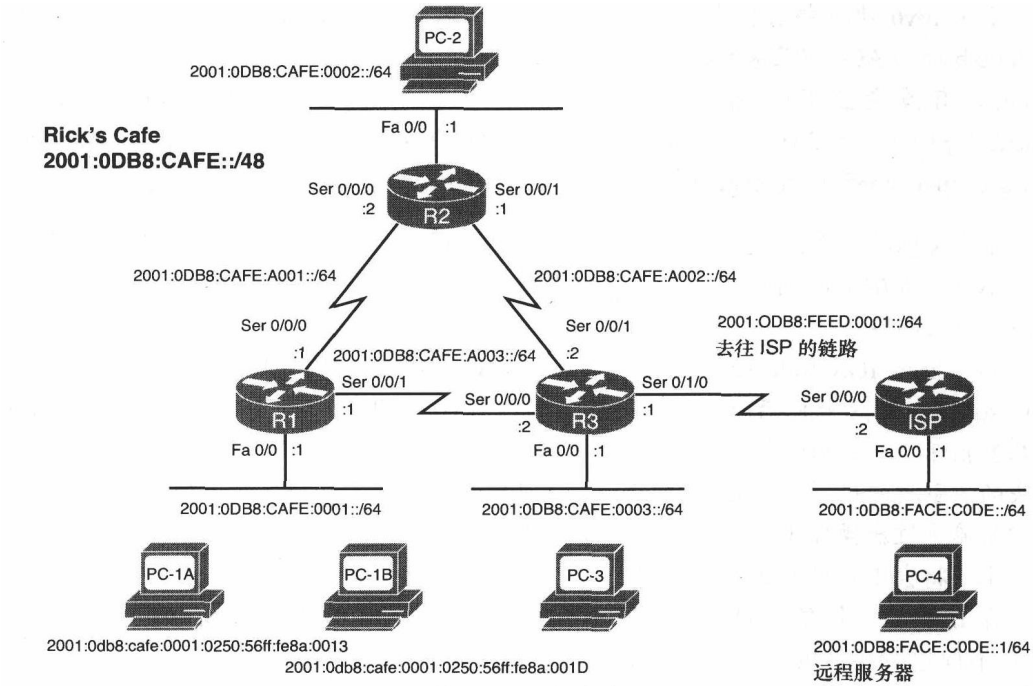



图 6-2 IPv6 拓扑结构

注：为了方便讨论 IPv6 ACL，这里假设例 6-2 中的拓扑结构具有完全的可达性。有关 IPv6 路由的问题将在第 7 章和第 8 章详细讨论。

首先在 R3 上使用命令 **ipv6 access-list**，该命令与 IPv4 的 **ip access-list** 相似，唯一的区别就是指定了 IPv6。该命令的语法如下：

```
Router(config)# ipv6 access-list access-list-name
```

在例 6-28 中，在 R3 上配置了一条访问列表名为 no-face-c0de 的 ACL。在 ACL 配置模式下，可以配置允许和拒绝语句。同样，这些命令与 IPv4 ACL 的区别就是指定了 IPv6。利用 ACL 命令 **deny 2001:db8:face:c0de::/48 any**，R3 将拒绝所有源 IPv6 地址包含前缀 2001:db8:face:c0de::/48 的数据包。下面的一条语句 **permit any any** 是允许其他所有的 IPv6 包。例 6-28 还给出了另一种扩展式 IPv6 ACL 的配置方式。

**例 6-28　在 R3 上配置扩展式 IPv6 ACL**

```
R3(config)# ipv6 access-list no-face-c0de
R3(config-ipv6-acl)# deny 2001:db8:feed::/48 any ?
  dest-option             Destination Option header (all types)
  dest-option-type        Destination Option header with type
  dscp                    Match packets with given dscp value
  flow-label              Flow label
  fragments               Check non-initial fragments
  log                     Log matches against this entry
  log-input               Log matches against this entry, including input
  mobility                Mobility header (all types)
  mobility-type           Mobility header with type
  routing                 Routing header (all types)
  routing-type            Routing header with type
  sequence                Sequence number for this entry
  time-range              Specify a time-range
  undetermined-transport  Transport cannot be determined or is missing
  <cr>

R3(config-ipv6-acl)# deny 2001:db8:face:c0de::/48 any
R3(config-ipv6-acl)# permit any any

OR

R3(config)# ipv6 access-list no-face-c0de deny 2001:db8:face:c0de::/48 any
R3(config)# ipv6 access-list no-face-c0de permit any any
```

注：与 IPv4 ACL 一样，IPv6 也可以根据 IPv6 报头和扩展报头中的信息进行过滤。该特性与平台相关，其内容已经超出了本书范围，本书仅关注于基本的 IPv6 ACL 配置。

与 IPv4 使用命令 **ip access-group** 将 ACL 应用于接口不同，IPv6 使用的是命令 **ipv6 traffic-filter**。命令 **ipv6 traffic-filter** 的作用是过滤被路由器转发的流量，而不过滤源自本路由器的流量。如果希望过滤接口的入站或出站链路，那么就可以在接口配置模式下

使用命令 **ipv6 traffic-filter**：

```
Router(config-if)# ipv6 traffic-filter access-list-name {in | out}
```

为了过滤 IPv6 前缀为 2001:db8:face:c0de::/48 的数据包，需要在 R3 面向 ISP 路由器的串行接口 serial 0/0/0 上配置命令 **ipv6 traffic-filter no-face-c0de**（如例 6-29 所示）。

**例 6-29　在接口上应用 ACL**

```
R3(config)# interface serial 0/1/0
R3(config-if)# ipv6 traffic-filter no-face-c0de in
R3(config-if)# end
R3#
```

例 6-30 利用命令 **show ipv6 access-list** 验证了该 ACL 配置。请注意与每个语句相关联的默认序列号。此外，例 6-30 还通过命令 **show running-config** 验证了名为 no-face-c0de 的 ACL 配置情况。

**例 6-30　验证 R3 的 ACL**

```
R3# show ipv6 access-list
IPv6 access list no-face-c0de
    deny ipv6 2001:DB8:FACE::/48 any sequence 10
    permit ipv6 any any sequence 20
R3#

R3# show running-config
<output omitted for brevity)
!
interface Serial0/1/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:FEED:1::1/64
 ipv6 traffic-filter no-face-c0de in
 clock rate 64000
!
ipv6 access-list no-face-c0de
 deny ipv6 2001:DB8:FACE::/48 any
 permit ipv6 any any
!
<output omitted for brevity)
```

可以在 ISP 路由器上利用 **ping** 命令测试 ACL 的配置情况，并且从 Fast Ethernet

0/0 接口（是网络 2001:db8:face:c0de:1::/64 的成员）发出 ICMPv6 回显请求消息。如例 6-31 所示，以包含 2001:db8:face:c0de::/48 低阶前缀为源 IPv6 地址发起的 ping 操作失败。

**例 6-31　测试名为 no-face-c0de 的 ACL**

```
ISP# ping 2001:db8:cafe:1::1 source fastethernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:1::1, timeout is 2 seconds:
Packet sent with a source address of 2001:DB8:FACE:C0DE::1
AAAAA
Success rate is 0 percent (0/5)
ISP#

ISP# ping 2001:db8:cafe:1::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/70/72 ms
ISP#
```

可以看出，R3 上的 IPv6 测试字符均为 A，表明这些数据包因管理原因而不可达，通常是被访问列表所阻塞。表 6-1 列出了 IPv6 ping 测试字符。

**表 6-1　IPv6 ping 测试字符**

| 字符 | 描述 |
|---|---|
| ! | 每个感叹号都表示收到一次应答 |
| . | 每个句号都表示网络服务器在等待应答时已超时 |
| ? | 未知错误 |
| @ | 因未知原因而不可达 |
| A | 管理性不可达，通常该输出结果表明是访问列表阻塞了流量 |
| B | 数据包超大 |
| H | 主机不可达 |
| N | 网络不可达（超出范围） |
| P | 端口不可达 |
| R | 参数问题 |
| S | 源地址与入站/出站策略相抵触 |

续表

| 字符 | 描述 |
| --- | --- |
| T | 超时 |
| U | 无去往主机的路由 |
| X | 拒绝路由到目的地 |

在讨论下一个案例之前，先删除 R3 上的名为 no-face-c0de 的 ACL，以免出现不可预料的情况。例 6-32 显示了删除 no-face-c0de ACL 的方法以及验证方法。

**例 32　删除名为 no-face-c0de 的 ACL**

```
R3(config)# no ipv6 access-list no-face-c0de
R3(config)# end
R3# show ipv6 access-list

R3#
R3(config)# interface serial 0/1/0
R3(config-if)# no ipv6 traffic-filter no-face-c0de in
R3(config-if)# end
R3#
```

### 6.10.2　允许本地 Telnet 访问

例 6-33 显示了仅允许本地 Telnet 访问路由器 R3 的访问列表配置情况。为了基于 IPv6 访问列表过滤来去路由器的入站和出站连接，可以使用线路配置命令 **ipv6 access-class**：

```
Router(config-line) ipv6 access-class ipv6-access-list-name {in | out}
```

例 6-33 配置了名为 **no-outside-vty** 的访问列表，语句 **permit 2001:db8:cafe::/48 any** 允许来自网络 2001:db8:cafe::/48 的数据包访问。该 ACL 的最后还有一条隐式语句 **deny any any**，用于拒绝源地址在网络 2001:db8:cafe::/48 之外的任何数据包。

**例 6-33　配置 ACL 以仅允许通过本地 Telnet 方式访问 R3**

```
R3(config)# ipv6 access-list no-outside-vty
R3(config-ipv6-acl)# permit 2001:db8:cafe::/48 any
R3(config-ipv6-acl)# end
R3#

R3(config)# line vty 0 4
R3(config-line)# password luigi
```

```
R3(config-line)# login
R3(config-line)# ipv6 access-class no-outside-vty in
R3(config-line)# end
R3#
```

例 6-33 给出了在 R3 上配置 Telnet 访问的示例。命令 **ipv6 accessclass no-outside-vty in** 使用 no-outside-vty ACL 来过滤 Telnet 连接。

例 6-34 证实了从网络 2001:db8:cafe::/48 之外 Telnet 访问 R3 被拒绝了。例中的 Telnet 连接试图使用 R3 的两个接口。

**例 6-34 验证无法从网络外部 Telnet 访问 R3**

```
ISP# telnet 2001:db8:feed:1::1
Trying 2001:DB8:FEED:1::1 ...
% Connection refused by remote host

ISP# telnet 2001:db8:cafe:a003::2
Trying 2001:DB8:CAFE:A003::2 ...
% Connection refused by remote host
```

例 6-35 给出了在网络 2001:db8:cafe::/48 内部使用 Telnet 的不同结果。由于 R1 的源 IPv6 地址在语句 **permit 2001:db8:cafe::/48 any** 的允许范围之内，因而 R3 接受该 Telnet 连接

**例 6-35 验证可以从网络内部 Telnet 访问 R3**

```
R1# telnet 2001:db8:cafe:a003::2
Trying 2001:DB8:CAFE:A003::2 ... Open


User Access Verification

Password:
R3>
```

## 6.11 本章小结

本章解释了如何为特定拓扑结构（下一章仍将用到该拓扑结构）配置地址。本章不但介绍了很多新命令，而且还回顾了很多在前面各章已经学习过的配置命令。

### 6.11.1 编址命令

- **配置全局单播地址**：配置全局单播地址或唯一本地单播地址的接口级命令如下所示：

```
Router(config-if)# ipv6 address ipv6-address/prefix-length
```

- **配置链路本地地址**：手工配置链路本地地址的接口级命令如下所示：

```
Router(config-if)# ipv6 address ipv6-address link-local
```

- **启用接口的 IPv6 功能**：为了在不指定全局单播地址或唯一本地地址的情况下启用接口的 IPv6 功能，可以使用以下命令：

```
Router(config-if)# ipv6 enable
```

- **以 EUI-64 选项配置全局单播地址**：使用 EUI-64 格式配置全局单播地址的命令如下所示：

```
Router(config-if)# ipv6 address ipv6-prefix/prefix-length eui-64
```

- **删除 IPv6 地址**：删除接口上所有 IPv6 地址（包括全局单播地址和链路本地地址，除非配置了命令 **ipv6 enable**）的命令如下所示：

```
Router(config-if)# no ipv6 address [ipv6-address/prefix-length]
```

### 6.11.2 转发 IPv6 单播包

为了让路由器能够转发 IPv6 单播包且发送 ND 路由器宣告消息，应使用以下命令：

```
Router(config)# ipv6 unicast-routing
```

### 6.11.3 邻居缓存表

- **邻居缓存表**：利用以下命令可以显示邻居缓存表（类似于 IPv4 的 ARP 缓存表）：

```
Router# show ipv6 neighbors
```

- **配置静态邻居缓存表表项**：利用以下命令可以向邻居缓存表添加静态表项：

```
Router(config)# ipv6 neighbor ipv6-address interface-type interface-number
hardware-address
```

- **清除邻居缓存表**：利用以下命令可以清除邻居缓存表：

```
Router# clear ipv6 neighbors
```

### 6.11.4 调节邻居发现参数

- **抑制路由器宣告消息**：利用以下命令可以抑制路由器宣告消息：

```
Router(config-if)# ipv6 nd ra suppress
```

- **修改 RA 间隔**：Cisco 路由器的默认 RA 间隔为 200 秒，利用以下命令可以修改 RA 间隔：

```
Router(config-if)# ipv6 nd ra interval {maximum-secs [minimum-secs] | msec
maximum-ms [minimum-ms]}
```

- **修改 RA 生存期**：RA 生存期是主机将该路由器视为有效默认网关的持续时间，利用以下命令可以修改 RA 生存期：

```
Router(config-if)# ipv6 nd ra lifetime seconds
```

- **设置 M 标记（被管地址配置标记）**：为了将 M 标记设置为 1，以告诉主机使用状态化 DHCPv6 而不是 SLAAC，可以使用以下命令

```
Router(config-if)# ipv6 managed-config-flag
```

- **设置 O 标记（其他配置标记）**：为了将 O 标记设置为 1，以告诉主机可以从 DHCPv6 服务器获得额外配置信息，可以使用以下命令：

```
Router(config-if)# ipv6 other-config-flag
```

- **显示从其他路由器收到的路由器宣告消息**：利用以下命令可以显示从以太网、令牌环或 FDDI 链路上的其他路由器收到的路由器宣告：

```
Router# show ipv6 routers
```

### 6.11.5 IPv6 ACL

从本章的案例研究中可以看出，IPv6 ACL 的配置与 IPv4 ACL 非常相似。与 IPv4 一样，IPv6 访问控制列表的末尾都包含一个隐式的 **deny any any** 语句。而且由于 IPv6 ND（Neighbor Discovery，邻居发现）进程需要用到 IPv6 网络层服务，因而每条 IPv6 ACL 的末尾还隐式包含另外两条语句：

- **permit icmp any any nd-na**
- **permit icmp any any nd-ns**

命令 **ipv6 access-list** 的作用是配置与 IPv6 相关的 ACL。IPv6 ACL 不能采取编号方式，只能配置为名称式访问列表。命令 **ipv6 traffic-filter** 的作用是将 IPv6 ACL 应用到接口上。如果需要将 ACL 应用于虚接口（如 vty 线路），那么就要使用线路配置命令 **ipv6 access-class**。

## 6.12 参考文献

Cisco IOS IPv6 Command Reference:www.cisco.com/en/US/docs/ios/ipv6/command/reference/ipv6_book.html

Cisco IOS IPv6 Configuration Guide, Release 12.4: www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/12_4/ipv6_12_4_book.html

Implementing Traffic Filters and Firewalls for IPv6 Security, www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-sec_trfltr_fw.html

# 第 7 章　IPv6 路由概述

前面的章节详细讨论了 IPv6 协议及其相关协议，如 ICMPv6 和邻居发现协议。如前所述，IPv6 并不仅仅意味着 128 比特地址空间，而且 IPv6 的配置并不是与 IPv4 完全不同。从第 6 章中可以看出，IPv6 接口地址的配置是非常直观的。

本章将讨论与 IPv6 路由相关的两个主题：

- IPv6 路由表；
- IPv6 静态路由。

在接口上配置 IPv6 地址以及配置 IPv6 静态路由与 IPv4 的配置非常相似。IPv6 路由表也与 IPv4 路由表非常相似。如果大家熟悉 IPv4 静态路由的配置，那么就会发现 IPv6 静态路由的配置没什么不同。因此请绝对放心，最具挑战性的学习就是前面章节讨论过的各种 IPv6 协议及其相关进程，而且本章将要讨论的 IPv6 相关配置也与 IPv4 非常相似。

本章仍以第 6 章使用的拓扑结构为例（如图 7-1 所示）。第 6 章已经为每台路由器都配置了全局单播地址和链路本地地址（不过图 7-1 并没有显示链路本地地址）。虽然可以 ping 邻接路由器的接口，但由于此时还没有配置任何静态路由或动态路由协议，因而无法访问任何远程网络。

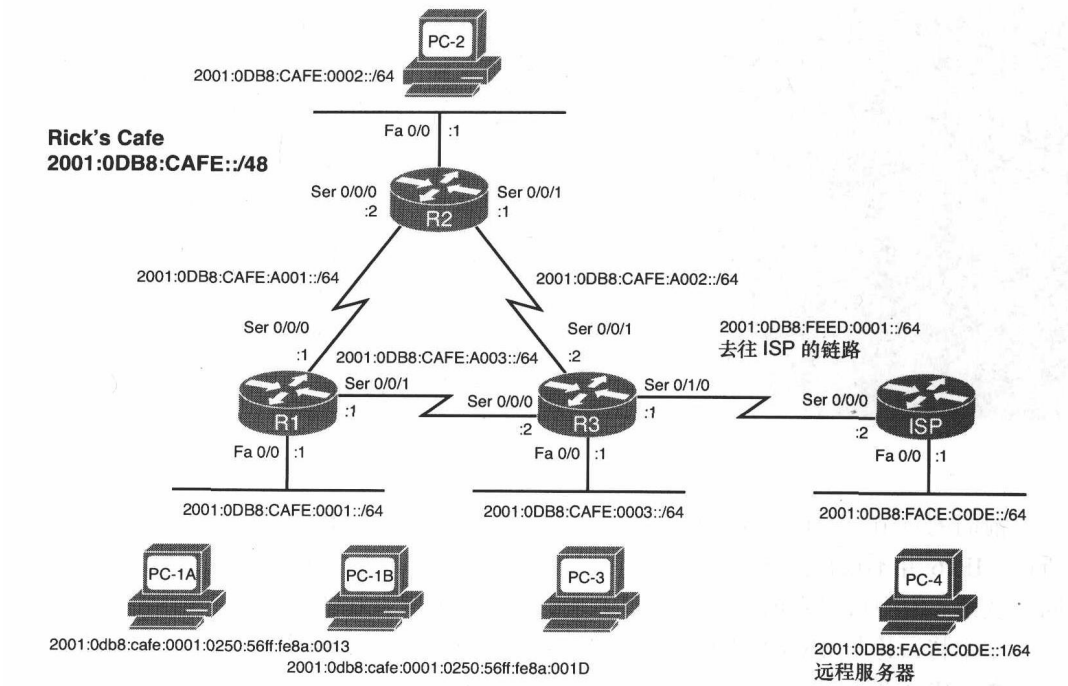



图 7-1 Rick's Café 网络拓扑结构

## 7.1 IPv6 路由表

首先讨论 IPv6 路由表。Cisco IOS 分别为 IPv4 和 IPv6 维护一张独立的路由表，可以使用命令 **show ipv6 route** 显示 IPv6 路由表：

```
Router# show ipv6 route
```

命令 **show ipv6 route** 有多个选项，本节将讨论其中的一些主要选项，完整的命令语法如下：

```
Router# show ipv6 route [ipv6-address | ipv6-prefix/prefix-length
[longer-prefixes] | [protocol] [updated [boot-up] [day month]
[time]] | interface interface-type interface-number | nsf | table
table-id | watch]
```

Cisco IOS IPv6 Command Reference 中详细描述了以下语法选项，这里列出以供参考。

- *ipv6-address*（可选）：显示指定 IPv6 地址的路由信息。必须按照 RFC 4291 的要求使用该选项，地址采取十六进制格式，所有的冒号之间使用 16 比

特值。

- *ipv6-prefix*（可选）：显示指定 IPv6 网络的路由信息。必须按照 RFC 4291 的要求使用该选项，地址采取十六进制格式，所有的冒号之间使用 16 比特值。
- */prefix-length*（可选）：IPv6 前缀的长度。十进制值表示该地址的前缀（地址的网络部分）是由多少个高阶连续比特组成的。十进制数值之前必须有一条斜线。
- **longer-prefixes**（可选）：显示具有更长前缀的路由项。
- *protocol*（可选）：通过关键字（如 **bgp**、**isis**、**eigrp**、**ospf** 或 **rip**）显示指定路由协议的路由，或者通过关键字（如 **connected**、**local**、**mobile** 或 **static**）显示指定类型的路由。
- **updated**（可选）：显示带时间戳的路由。
- **boot-up**（可选）：显示启动后的路由信息。
- *day month*（可选）：显示指定月份与日期后的路由。
- *time*（可选）：显示指定时间后的路由，时间应采取 *hh:mm* 格式。
- **interface** *interface-type*（可选）：接口类型。利用问号（?）在线帮助功能，可以获得所支持的接口类型信息。
- *interface-number*（可选）：接口号。利用问号（?）在线帮助功能，可以获得所支持接口类型的编号信息。
- **nsf**（可选）：显示处于不间断转发状态的路由。
- **table** *table-id*（可选）：显示指定表 ID 的 IPv6 RIB（Routing Information Base，路由信息库）表信息。表采取十六进制格式，表 ID 的取值范围是 0～0xFFFFFFFF。
- **watch**（可选）：显示路由监控器（route watcher）的信息。

IPv4 的 **show ip route** 命令显示的每条路由都带有时间戳，也就是路由最后被更新的时间（以小时:分钟:秒来表示）。但默认情况下，IPv6 路由表不显示时间戳，必须在命令中包含选项 **updated**，即 **show ipv6 route updated**，才能显示路由最后被更新的时间（以小时:分钟:秒，日，月，年来表示）。

> 注：由于大多数选项都超出了本书写作范围，因而不再详细讨论。如果希望进一步了解这些选项，请参考 Cisco IOS IPv6 Command Reference: www.cisco.com/en/US/docs/ios/ipv6/command/reference/ipv6_16.html#wp2669925。

下面分析路由器 R1 的路由表。没错，这里写的路由表的确是复数。在例 7-1 中，命令 **show ip route** 显示了 IPv4 路由表的内容。由于 R1 未配置任何 IPv4 接口，因而 IPv4 路由表为空。接下来使用命令 **show ipv6 route** 来显示 R1 的 IPv6 路由表。注意到此时的路由表中已经有了 IPv6 路由，下面就来仔细分析该路由表。

### 例 7-1 显示 R1 的 IPv6 路由表

```
R1# show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

R1# show ipv6 route
IPv6 Routing Table - 8 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
C   2001:DB8:CAFE:1::/64 [0/0]
     via ::, FastEthernet0/0
L   2001:DB8:CAFE:1::1/128 [0/0]
     via ::, FastEthernet0/0
C   2001:DB8:CAFE:A001::/64 [0/0]
     via ::, Serial0/0/0
L   2001:DB8:CAFE:A001::1/128 [0/0]
     via ::, Serial0/0/0
C   2001:DB8:CAFE:A003::/64 [0/0]
     via ::, Serial0/0/1
L   2001:DB8:CAFE:A003::1/128 [0/0]
     via ::, Serial0/0/1
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
R1#
```

紧跟在命令 **show ipv6 route** 之后的第一行输出结果中显示了路由表中的路由条数：

```
IPv6 Routing Table - 8 entries
```

8 条路由看起来似乎有些奇怪，因为 R1 只有 3 个接口，也没有配置任何静态路由

和动态路由，这个问题稍后再解释。在代码（Code）之后显示的是组成 R1 的 IPv6 路由表的 8 条路由。

### 7.1.1　代码：直连路由

代码字段 C 表示这是一个直连网络，这与相同代码的 IPv4 路由相似。当配置了全局单播地址或唯一本地单播地址的 IPv6 接口处于 up 状态时，包括接口的状态、线路协议、IPv6 前缀以及前缀长度在内的信息都被加入路由表，作为直连路由。如果接口只有一个链路本地地址，那么就不会显示在路由表中。稍后将讨论链路本地地址与路由表的相关内容。在例 7-1 中，IPv6 路由表中的第一条表项是 R1 的 Fast Ethernet 0/0 接口的直连网络。

```
C   2001:DB8:CAFE:1::/64 [0/0]
     via ::, FastEthernet0/0
```

该表项包含了 Fast Ethernet 0/0 接口的前缀和前缀长度 2001:DB8:CAFE:1::/64。[0/0] 表示该路由的管理距离（administrative distance）和度量（metric）。直连网络的管理距离和度量均为 0。该路由表项的第二行显示的是下一跳地址，对本案例来说是未指定地址。由于直连网络无下一跳地址，因而使用 IPv6 未指定地址。此外，第二行还包含了出接口 Fast Ethernet 0/0。

注：管理距离是路由信息源的可信度（trustworthiness）或优先权（preference）。如果路由器从多个路由源学到路由，那么就会选择管理距离较低的源。管理距离仅具有本地意义，不会通过路由更新进行宣告。

注：关于管理距离为 0 存在一定的误传，只有 IPv4 中的直连网络或者 IPv6 中的直连网络或本地接口，其管理距离才为 0。配置了出接口的静态路由，其管理距离不为 0。静态路由和动态路由的管理距离永远也不会为 0，即使配置了出接口的 IPv4 静态路由会在路由表中显示“directly connected（直连）”，但该静态路由的管理距离仍然是默认值 1。

R1 的路由表中存在三条直连路由，每个配置了可路由 IPv6 地址的接口都有一条直连路由。在命令 **show ipv6 route** 中使用 **connected** 选项，就可以对路由表进行过滤以仅显示直连网络（如例 7-2 所示）。

**例 7-2　显示 R1 的直连路由**

```
R1# show ipv6 route ?
     Hostname or X:X:X:X::X  IPv6 name or address
     X:X:X:X::X/<0-128>      IPv6 prefix
     bgp                     BGP routes
     connected               Connected routes
     eigrp                   EIGRP routes
```

```
  interface              interface specific routes
  isis                   IS-IS routes
  local                  Local routes
  ospf                   OSPFv3 routes
  rip                    RIPng routes
  static                 Static routes
  summary                Summary display
  |                      Output modifiers
  <cr>

R1# show ipv6 route connected
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
C   2001:DB8:CAFE:1::/64 [0/0]
     via ::, FastEthernet0/0
C   2001:DB8:CAFE:A001::/64 [0/0]
     via ::, Serial0/0/0
C   2001:DB8:CAFE:A003::/64 [0/0]
     via ::, Serial0/0/1
R1#
```

### 7.1.2 代码：本地路由

R1 的路由表中除了 3 条直连路由之外，还有 5 条代码为 L 的其他路由。代码 L 表示本地路由。大家可能第一感觉会将这些路由认为是链路本地网络，但实际上却并非如此。本地路由是/128 路由，本质上是路由器的 IPv6 单播地址的主机路由。本地路由能够让路由器更加高效地处理直接发送给路由器自身的数据包(而不是发送给该路由器直连子网的数据包)。

从例 7-1 中的 5 条本地路由可以看出，紧跟在每个直连网络之后的就是其本地网络，即主机路由或该接口的 IPv6 单播地址。例如，上一小节曾经讨论过 R1 的 Fast Ethernet 0/0 接口的直连网络，该接口以手工方式配置了一个 IPv6 全局单播地址 2001:DB8:CAFE:1::1，因而被显示为一条本地/128 主机路由：

```
L   2001:DB8:CAFE:1::1/128 [0/0]
     via ::, FastEthernet0/0
```

与直连网络类似，本地路由的管理距离和度量也均为 0。由于 R1 的 3 个接口都配置了可路由地址，因而 IPv6 路由表中的 5 条本地路由中的 3 条就是由此而产生的。

IPv6 接口上的链路本地地址情况如何呢？由于携带链路本地地址的数据包不会被路由到链路之外，因而 IPv6 路由表中不包含这些地址。IPv6 路由表显式地列出链路本地前缀 FE80::/10，并将其出接口设置为 Null0，这样就能保证路由器会丢弃所有去往链路本地地址的数据包（自己的链路本地地址除外）：

```
L   FE80::/10 [0/0]
     via ::, Null0
```

R1 的 IPv6 路由器中的最后一条本地路由是去往多播地址 FE80::/8 的 Null0 路由。

```
L   FF00::/8 [0/0]
     via ::, Null0
```

该路由用于路由多播包，如果多播包所去往的多播组未显式列在 IPv6 路由表中，那么路由器就会丢弃这些多播组。如果在命令 **show ipv6 route** 中使用选项 **local**，那么就可以对路由表进行过滤，以仅显示本地网络（如例 7-3 所示）。

**例 7-3　显示 R1 的本地路由**

```
R1# show ipv6 route local
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
L   2001:DB8:CAFE:1::1/128 [0/0]
     via ::, FastEthernet0/0
L   2001:DB8:CAFE:A001::1/128 [0/0]
     via ::, Serial0/0/0
L   2001:DB8:CAFE:A003::1/128 [0/0]
     via ::, Serial0/0/1
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
R1#
```

### 7.1.3　IPv6 与 IPv4 路由表对比

很多情况下，IPv6 路由表比 IPv4 路由表更易于理解。IPv4 路由表按照有类别方式组成（如例 7-4 所示），而 IPv6 则根本不关心有类别网络或 VLSM（Variable-Length Subnet Masking，可变长子网掩码）。IPv4 路由表是按照有类别网络地址组织的，需要在每条

有类别路由表项下列出所有子网。IPv6 路由表则显得更为直观。由于 IPv6 中不存在有类别和无类别编址，因而 IPv6 路由仅显示为单个网络。

**例 7-4 IPv4 路由表示例**

```
Router# show ip route
Codes: I - IGRP derived, R - RIP derived, O - OSPF derived,
       C - connected, S - static, E - EGP derived, B - BGP derived,
       * - candidate default route, IA - OSPF inter area route,
       i - IS-IS derived, ia - IS-IS, U - per-user static route,
       o - on-demand routing, M - mobile, P - periodic downloaded static route,
       D - EIGRP, EX - EIGRP external, E1 - OSPF external type 1 route,
       E2 - OSPF external type 2 route, N1 - OSPF NSSA external type 1 route,
       N2 - OSPF NSSA external type 2 route

    192.168.10.0/30 is subnetted, 3 subnets
C      192.168.10.0 is directly connected, Serial0/0/0
C      192.168.10.4 is directly connected, Serial0/0/1
O      192.168.10.8 [110/128] via 192.168.10.2, 14:27:57, Serial0/0/0
    172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
O      172.16.1.32/29 [110/65] via 192.168.10.6, 14:27:57, Serial0/0/1
C      172.16.1.16/28 is directly connected, FastEthernet0/0
    10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
O      10.10.10.0/24 [110/65] via 192.168.10.2, 14:27:57, Serial0/0/0
C      10.1.1.1/32 is directly connected, Loopback0
```

**注**：由 Rick Graziani 和 Allan Johnson 编著的 *Routing Protocols and Concepts* 提供了有关 IPv4 路由表的详细信息，包括路由表结构和查询进程。

虽然有关动态路由和汇总路由的内容将在第 8 章进行讨论，但为了完整起见，例 7-5 到例 7-7 列出了路由器 R2、R3 和 ISP 的所有路由表信息。

**例 7-5 显示 R2 的 IPv6 路由表**

```
R2# show ipv6 route
IPv6 Routing Table - 8 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
C   2001:DB8:CAFE:2::/64 [0/0]
     via ::, FastEthernet0/0
L   2001:DB8:CAFE:2::1/128 [0/0]
     via ::, FastEthernet0/0
```

```
C   2001:DB8:CAFE:A001::/64 [0/0]
     via ::, Serial0/0/0
L   2001:DB8:CAFE:A001::2/128 [0/0]
     via ::, Serial0/0/0
C   2001:DB8:CAFE:A002::/64 [0/0]
     via ::, Serial0/0/1
L   2001:DB8:CAFE:A002::1/128 [0/0]
     via ::, Serial0/0/1
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
R2#
```

## 例 7-6　显示 R3 的 IPv6 路由表

```
R3# show ipv6 route
IPv6 Routing Table - 10 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
C   2001:DB8:CAFE:3::/64 [0/0]
     via ::, FastEthernet0/0
L   2001:DB8:CAFE:3::1/128 [0/0]
     via ::, FastEthernet0/0
C   2001:DB8:CAFE:A002::/64 [0/0]
     via ::, Serial0/0/1
L   2001:DB8:CAFE:A002::2/128 [0/0]
     via ::, Serial0/0/1
C   2001:DB8:CAFE:A003::/64 [0/0]
     via ::, Serial0/0/0
L   2001:DB8:CAFE:A003::2/128 [0/0]
     via ::, Serial0/0/0
C   2001:DB8:FEED:1::/64 [0/0]
     via ::, Serial0/1/0
L   2001:DB8:FEED:1::1/128 [0/0]
     via ::, Serial0/1/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
R3#
```

**例 7-7 显示 ISP 的 IPv6 路由表**

```
ISP# show ipv6 route
IPv6 Routing Table - 6 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
C   2001:DB8:FACE:C0DE::/64 [0/0]
     via ::, FastEthernet0/0
L   2001:DB8:FACE:C0DE::1/128 [0/0]
     via ::, FastEthernet0/0
C   2001:DB8:FEED:1::/64 [0/0]
     via ::, Serial0/0/0
L   2001:DB8:FEED:1::2/128 [0/0]
     via ::, Serial0/0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
ISP#
```

## 7.2 配置 IPv6 静态路由

本节将解释如何为拓扑结构中所有路由器配置静态路由，以实现所有网络的可达性。如果要路由 IPv6 包，Cisco 路由器要求必须在全局配置模式下配置命令 **ipv6 unicast-routing**。Cisco IOS 在默认情况下是关闭该命令的，因而为了保证路由器能够路由 IPv6 包，就必须启用该命令并配置 IPv6 路由协议。该命令与 IPv4 的 **ip routing** 命令相似，但 **ip routing** 命令是默认启用的。虽然第 6 章已经打开了所有路由器的 IPv6 路由能力，但为了方便起见，仍然在例 7-8 中列出了这些命令。

**例 7-8 利用命令 ipv6 unicast-routing 打开路由器的 IPv6 路由能力**

```
R1(config)# ipv6 unicast-routing
R2(config)# ipv6 unicast-routing
R3(config)# ipv6 unicast-routing
ISP(config)# ipv6 unicast-routing
```

注：如果路由器希望发送邻居发现协议的路由器宣告消息，那么也必须应用命令 **ipv6 unicast-routing**。

IPv6 静态路由的配置与 IPv4 静态路由的配置非常相似。为了方便起见，这里假设大家都已经熟悉了 IPv4 静态路由的配置过程。本章将要使用的配置 IPv6 静态路由的基本命令语法如下：

```
Router(config)# ipv6 route ipv6-prefix/prefix-length {ipv6-address
| interface-type interface-number [administrative-distance]
```

配置 IPv6 静态路由的完整命令语法如下：

```
Router(config)# ipv6 route [vrf vrf-name] ipv6-prefix/prefix-length
{ipv6-address | interface-type interface-number [ipv6-address]}
[nexthop-vrf [vrf-name1 | default]] [administrative-distance]
[administrative-multicast-distance | unicast | multicast]
[next-hop-address] [tag tag] [name name]
```

根据 Cisco IOS IPv6 命令参考，**ipv6 route** 命令的语法描述如下。

- *ipv6-prefix*：静态路由的目的 IPv6 网络。配置静态主机路由时，也可以是主机名。
- */prefix-length*：IPv6 前缀的长度。十进制值表示前缀（地址的网络部分）是由多少个高阶连续比特组成的。十进制数值之前必须有一条斜线。
- **vrf**（可选）：指定所有 VRF（VPN[Virtual Private Network，虚拟专用网] routing/forwarding，VPN 路由转发）表或者与某 IPv4 地址或 IPv6 地址相对应的特定 VRF 表。
- *vrf-name*（可选）：与某 IPv4 地址或 IPv6 地址相对应的特定 VRF 表的名字。
- *ipv6-address*：用于到达指定网络的下一跳的 IPv6 地址。下一跳的 IPv6 地址不必是直连网络，可以通过递归查找来发现直连下一跳的 IPv6 地址。如果指定了接口类型和接口号，那么就可选指定数据包将要输出的下一跳的 IPv6 地址。请注意，如果将链路本地地址用作下一跳（链路本地下一跳必须是邻接路由器），那么就必须指定接口类型和接口号。必须按照 RFC 4291 的要求使用该选项，即使用十六进制形式的以冒号分隔的 16 比特值。
- *interface-type*：接口类型。利用问号（?）在线帮助功能，可以获得所支持的接口类型信息。通过 *interface-type* 可以指定静态路由直接输出到点到点接口（如串行接口或隧道接口）和广播接口（如以太网接口）。与点到点接口一起使用 *interface-type* 时，无需指定下一跳的 IPv6 地址。与广播接口一起使用 *interface-type* 时，必须指定下一跳的 IPv6 地址或者确保将指定前缀分配给本链路。下一跳地址的作用只是确定下一跳路由器的正确链路层（MAC）地址，

以避免在路由表中执行下一跳地址的递归查找。链路本地地址通常与出接口相关联。

- *interface-number*：接口号。利用问号（?）在线帮助功能，可以获得所支持的接口类型信息。
- **nexthop -vrf**（可选）：指示下一跳是 VRF。
- *vrf-name1*（可选）：下一跳 VRF 的名字。
- **default**（可选）：指示下一跳是默认 VRF。
- *administrative-distance*（可选）：管理距离。默认值为 1，因而静态路由的优先级要高于除直连路由之外的其他路由。
- *administrative-multicast-distance*（可选）：当该路由被选择为多播 RPF（Reverse Path Forwarding，反向路径转发）时使用的距离。
- **unicast**（可选）：指定不能用于多播 RPF 选择进程的路由。
- **multicast**（可选）：指定不能安装到单播 RIB（Routing Information Base，路由信息库）中的路由。
- *next-hop-address*（可选）：可用于到达指定网络的下一跳的地址。
- **tag** *tag*（可选）：通过路由映射来控制路由重分发时，该标签值可以被用作“匹配”值。
- **name** *route-name*（可选）：指定路由的名字。

注：命令 **show ipv6 route** 的大多数选项都超出了本书写作范围，因而不再详细讨论。如果希望进一步了解这些选项，请参考 Cisco IOS IPv6 Command Reference：www.cisco.com/en/US/docs/ios/ipv6/command/reference/ipv6_08.html# wp2233116。

下面首先为路由器 R1 配置 IPv6 静态路由。例 7-9 显示 R1 上的第一条静态路由。该静态路由去往 R2 与 R3 之间的串行网络 2001:db8:cafe:a002::/64。该例中的静态路由是以 R3 的下一跳地址进行配置的，即 R3 的 serial 0/0/0 全局单播地址 2001:db8: cafe:a003::2。

**例 7-9 以下一跳接口配置静态路由**

```
R1(config)# ipv6 route ?
  X:X:X:X::X/<0-128>  IPv6 prefix x:x::y/<z>

R1(config)# ipv6 route 2001:db8:cafe:a002::/64 ?
  Async               Async interface
  BVI                 Bridge-Group Virtual Interface
  CDMA-Ix             CDMA Ix interface
  CTunnel             CTunnel interface
  Dialer              Dialer interface
```

```
  FastEthernet          FastEthernet IEEE 802.3
  Lex                   Lex interface
  Loopback              Loopback interface
  MFR                   Multilink Frame Relay bundle interface
  Multilink             Multilink-group interface
  Null                  Null interface
  Port-channel          Ethernet Channel of interfaces
  Serial                Serial
  Tunnel                Tunnel interface
  Vif                   PGM Multicast Host interface
  Virtual-Dot11Radio    Virtual dot11 interface
  Virtual-PPP           Virtual PPP interface
  Virtual-Template      Virtual Template interface
  Virtual-TokenRing     Virtual TokenRing
  X:X:X:X::X            IPv6 address of next-hop
  XTagATM               Extended Tag ATM interface

R1(config)# ipv6 route 2001:db8:cafe:a002::/64 2001:db8:cafe:a003::2 ?
  <1-254>     Administrative distance
  multicast   Route only usable by multicast
  tag         Tag value
  unicast     Route only usable by unicast
  <cr>

R1(config)# ipv6 route 2001:db8:cafe:a002::/64 2001:db8:cafe:a003::2
```

注：在例 7-9 中，虽然也可以将 R3 的 serial 0/0/0 链路本地地址用作下一跳地址，但那样还需要增加 R1 的出接口。这是因为链路本地地址不在路由表中，因为无需对任何链路保持其唯一性。

例 7-10 配置了第二条去往相同网络 2001:db8:cafe:a002::/64 的静态路由。这次使用的是 R1 的出接口 serial 0/0/0，与前一条静态路由将数据包转发给 R3 不同，本静态路由是将去往 2001:db8:cafe:a002::/64 的数据包转发给 R2。

**例 7-10　以出接口配置静态路由**

```
R1(config)# ipv6 route 2001:db8:cafe:a002::/64 serial 0/0/0
```

例 7-11 所示为配置了两条静态路由之后的 R1 的路由表。根据不同的交换方式——进程交换（process switching）、快速交换（fast switching）以及 CEF（Cisco Express Forwading，Cisco 快速转发），R1 会对去往该目的网络（位于 R2 与 R3 之间）的数据包进行负载均衡。由于这两条静态路由的管理距离均为 1，因而均显示在路由表中。

**例 7-11 配置了两条静态路由之后的 R1 路由表**

```
R1# show ipv6 route
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
C   2001:DB8:CAFE:1::/64 [0/0]
     via ::, FastEthernet0/0
L   2001:DB8:CAFE:1::1/128 [0/0]
     via ::, FastEthernet0/0
C   2001:DB8:CAFE:A001::/64 [0/0]
     via ::, Serial0/0/0
L   2001:DB8:CAFE:A001::1/128 [0/0]
     via ::, Serial0/0/0
S    2001:DB8:CAFE:A002::/64 [1/0]
     via 2001:DB8:CAFE:A003::2
     via ::, Serial0/0/0
C   2001:DB8:CAFE:A003::/64 [0/0]
     via ::, Serial0/0/1
L   2001:DB8:CAFE:A003::1/128 [0/0]
     via ::, Serial0/0/1
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
R1#
```

注：如前所述，以出接口配置的静态路由的默认管理距离为 1，与以下一跳地址配置的静态路由相同。静态路由的管理距离不能为 0。

有关 CEF 的相关内容将在第 8 章进行讨论。所有的 Cisco IOS 版本和所有的路由器平台都支持进程交换，其转发决策是基于每个数据包做出的。对于进入路由器的每个数据包来说，路由进程必须检查目的 IPv6（或 IPv4）地址、在路由表中确定最佳路径、将数据包封装到二层帧中并通过特定出接口转发出去，这种方式要求 CPU 参与每个数据包的转发决策。通过在 R1 上使用命令 **debug ipv6 packet**，并且 ping R2 的地址 2001:db8:cafe:a002::1（如例 7-12 所示），可以看出 R1 采用的是进程交换方式处理 IPv6 数据包。由于采用的是进程交换方式，因而源接口在不同路径之间来回切换。从例 7-12 可以看出，数据包来回在接口 serial 0/0/0 与 serial 0/0/1 之间进行处理。对于高端 Cisco

平台来说，通常不使用进程交换。

**例 7-12　数据包处理的接口切换**

```
R1# debug ipv6 packet
IPv6 unicast packet debugging is on
R1# ping 2001:db8:cafe:a002::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:A002::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/40/60 ms
R1#
*Feb 18 02:37:28.610: IPv6: SAS picked source 2001:DB8:CAFE:A001::1 for
2001:DB8:CAFE:A002::1 (Serial0/0/0)
*Feb 18 02:37:28.610: IPV6: source 2001:DB8:CAFE:A001::1 (local)
*Feb 18 02:37:28.610:       dest 2001:DB8:CAFE:A002::1 (Serial0/0/0)
*Feb 18 02:37:28.610:       traffic class 0, flow 0x0, len 100+0, prot 58, hops
64, originating
<output omitted>
*Feb 18 02:37:28.642: IPv6: SAS picked source 2001:DB8:CAFE:A003::1 for
2001:DB8:CAFE:A002::1 (Serial0/0/1)
*Feb 18 02:37:28.642: IPv6: nexthop 2001:DB8:CAFE:A003::2,
*Feb 18 02:37:28.642: IPV6: source 2001:DB8:CAFE:A003::1 (local)
*Feb 18 02:37:28.642:       dest 2001:DB8:CAFE:A002::1 (Serial0/0/1)
*Feb 18 02:37:28.642:       traffic class 0, flow 0x0, len 100+0, prot 58, hops
64, originating
<output omitted>
```

虽然配置为快速交换的接口仍然使用 CPU，但只需要处理数据流中的第一个数据包，利用进程交换处理完第一个数据包之后，如何到达目的地的信息就存储在快速交换缓存中，后续去往同一个目的地的数据包都直接使用快速交换缓存中的下一跳信息，不再使用 CPU，因而能够更快地转发数据包。

> 注：快速交换正逐步被 CEF 所取代。

到现在为止，仅在 R1 上配置了两条去往同一个网络 2001:db8:cafe:a002::/64 的静态路由。例 7-13 给出了去往该拓扑结构中其余网络的静态路由配置情况。

例 7-13 中配置了两条静态路由（分别去往 R2 的 LAN 和 R3 的 LAN）和两条/48 汇总路由（分别去往网络 2001:db8:feed::/48 和 2001:db8:face::/48）。此外，例 7-13 中的 **show running-config | include ipv6 route** 命令的作用是过滤输出结果，仅显示以“ipv6 route”开头的行。利用命令 **show ipv6 route static**，还可以进一步验证静态路由的配置情况（如例 7-14 所示）。

### 例 7-13 R1 其余静态路由的配置与验证

```
R1(config)# ipv6 route 2001:db8:cafe:2::/64 serial 0/0/0
R1(config)# ipv6 route 2001:db8:cafe:3::/64 serial 0/0/1
R1(config)# ipv6 route 2001:db8:feed::/48 serial 0/0/1
R1(config)# ipv6 route 2001:db8:face::/48 serial 0/0/1
R1(config)# end
R1#
R1# show running-config | include ipv6 route
ipv6 route 2001:DB8:CAFE:2::/64 Serial0/0/0
ipv6 route 2001:DB8:CAFE:3::/64 Serial0/0/1
ipv6 route 2001:DB8:CAFE:A002::/64 Serial0/0/0
ipv6 route 2001:DB8:CAFE:A002::/64 2001:DB8:CAFE:A003::2 5
ipv6 route 2001:DB8:FACE::/48 Serial0/0/1
ipv6 route 2001:DB8:FEED::/48 Serial0/0/1
R1#
```

### 例 7-14 利用命令 show ipv6 route static 验证静态路由的配置情况

```
R1# show ipv6 route ?
  Hostname or X:X:X:X::X  IPv6 name or address
  X:X:X:X::X/<0-128>      IPv6 prefix
  bgp                     BGP routes
  connected               Connected routes
  eigrp                   EIGRP routes
  interface               interface specific routes
  isis                    IS-IS routes
  local                   Local routes
  ospf                    OSPFv3 routes
  rip                     RIPng routes
  static                  Static routes
  summary                 Summary display
  |                       Output modifiers
  <cr>

R1# show ipv6 route static
IPv6 Routing Table - 13 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
S   2001:DB8:CAFE:2::/64 [1/0]
```

```
     via ::, Serial0/0/0
S    2001:DB8:CAFE:3::/64 [1/0]
     via ::, Serial0/0/1
S    2001:DB8:CAFE:A002::/64 [1/0]
     via ::, Serial0/0/0
S    2001:DB8:FACE::/48 [1/0]
     via ::, Serial0/0/1
S    2001:DB8:FEED::/48 [1/0]
     via ::, Serial0/0/1
R1#
```

注：本示例拓扑结构中，路由器的静态路由配置只是为了解释命令 **ipv6 route** 的使用方法，并不一定是使用静态路由配置网络的最有效方法。

例 7-15 给出了路由器 R2 的静态路由配置及验证情况。请注意，命令 **show running-config | section ipv6 route** 中使用的是选项 **section**，而不再是 **include section** 的作用是过滤运行配置的输出结果，以仅显示特定部分的内容。

**例 7-15　R2 的静态路由配置与验证**

```
R2(config)# ipv6 route 2001:db8:cafe:0001::/64 serial 0/0/0
R2(config)# ipv6 route 2001:db8:cafe:a003::/64 serial 0/0/1
R2(config)# ipv6 route 2001:db8:cafe:0003::/64 serial 0/0/1
R2(config)# ipv6 route 2001:db8:feed::/48 serial 0/0/1
R2(config)# ipv6 route 2001:db8:face::/48 serial 0/0/1
R2(config)# end

R2# show running-config | section ipv6 route
ipv6 route 2001:DB8:CAFE:1::/64 Serial0/0/0
ipv6 route 2001:DB8:CAFE:3::/64 Serial0/0/1
ipv6 route 2001:DB8:CAFE:A003::/64 Serial0/0/1
ipv6 route 2001:DB8:FACE::/48 Serial0/0/1
ipv6 route 2001:DB8:FEED::/48 Serial0/0/1
R2#

R2# show ipv6 route static
IPv6 Routing Table - 13 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
S   2001:DB8:CAFE:1::/64 [1/0]
```

```
       via ::, Serial0/0/0
S    2001:DB8:CAFE:3::/64 [1/0]
       via ::, Serial0/0/1
S    2001:DB8:CAFE:A003::/64 [1/0]
       via ::, Serial0/0/1
S    2001:DB8:FACE::/48 [1/0]
       via ::, Serial0/0/1
S    2001:DB8:FEED::/48 [1/0]
       via ::, Serial0/0/1
R2#
```

例 7-16 和例 7-17 分别给出了路由器 R3 和 ISP 路由器的静态路由配置及验证情况。请注意，在例 7-16 路由器 R3 的配置中，使用以下命令配置了一条静态默认路由：

```
R3(config)# ipv6 route ::/0 ser 0/1/0
```

### 例 7-16 R3 的静态路由配置与验证

```
R3(config)# ipv6 route 2001:db8:cafe:a001::/64 serial 0/0/0
R3(config)# ipv6 route 2001:db8:cafe:0001::/64 serial 0/0/0
R3(config)# ipv6 route 2001:db8:cafe:0002::/64 serial 0/0/1
R3(config)# ipv6 route ::/0 ser 0/1/0
R3(config)# end

R3# show running-config | include ipv6 route
ipv6 route 2001:DB8:CAFE:1::/64 Serial0/0/0
ipv6 route 2001:DB8:CAFE:2::/64 Serial0/0/1
ipv6 route 2001:DB8:CAFE:A001::/64 Serial0/0/0
ipv6 route ::/0 Serial0/1/0

R3# show ipv6 route static
IPv6 Routing Table - 14 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
S    ::/0 [1/0]
     via ::, Serial0/1/0
S    2001:DB8:CAFE:1::/64 [1/0]
     via ::, Serial0/0/0
S    2001:DB8:CAFE:2::/64 [1/0]
     via ::, Serial0/0/1
```

```
S   2001:DB8:CAFE:A001::/64 [1/0]
     via ::, Serial0/0/0
R3#
```

**例 7-17　ISP 的静态路由配置与验证**

```
ISP(config)# ipv6 route 2001:db8:cafe::/48 serial 0/0/0
ISP(config)# end

ISP# show running-config | include ipv6 route
ipv6 route 2001:DB8:CAFE::/48 Serial0/0/0

ISP# show ipv6 route static
IPv6 Routing Table - 7 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
S   2001:DB8:CAFE::/48 [1/0]
     via ::, Serial0/0/0
ISP#
```

第 3 章曾经讲到，::/0 表示默认路由，与 IPv4 中的 0.0.0.0 相似。这一点很明显，因为 IPv6 中的::表示的就是全 0 地址，与 IPv4 的 0.0.0.0 非常相似。

### 7.2.1　修改管理距离

从例 7-11 的 R1 路由表可以看出，R1 为 R2 与 R3 之间的串行网络 2001:db8:cafe:a003::/64 配置了两条静态路由。其中一条静态路由使用了 R3 的全局单播下一跳地址，另一条静态路由使用出接口将数据包转发给 R2。通过下一跳接口可以修改静态路由的管理距离，命令如下（如例 7-18 所示）。

```
R1(config)# ipv6 route 2001:db8:cafe:a002::/64 2001:db8:cafe:a003::2 5
```

该命令最后的数值 5 将管理距离从默认值 1 更改为 5。由于路由表中还有一条指向相同网络且管理距离较低（为 1）的路由，因而管理距离为 5 的路由将不会出现在路由表中。利用命令 **show ipv6 route static** 可以验证这一点（如例 7-18 所示）。此时路由表中只有一条使用出接口且管理距离为 1 的静态路由。命令 **show running-config | include ipv6 route** 显示了第二条路由仍然位于运行配置中，但是仅当管理距离较小的当前路由

不可达的时候，才会将该路由安装到路由表中。

**例 7-18 修改静态路由的管理距离**

```
R1(config)# ipv6 route 2001:db8:cafe:a002::/64 2001:db8:cafe:a003::2 5
R1(config)# end

R1# show ipv6 route static
IPv6 Routing Table - 13 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
S   2001:DB8:CAFE:2::/64 [1/0]
     via ::, Serial0/0/0
S   2001:DB8:CAFE:3::/64 [1/0]
     via ::, Serial0/0/1
S   2001:DB8:CAFE:A002::/64 [1/0]
     via ::, Serial0/0/0
S   2001:DB8:FACE::/48 [1/0]
     via ::, Serial0/0/1
S   2001:DB8:FEED::/48 [1/0]
     via ::, Serial0/0/1
R1#
R1# show running-config | include ipv6 route
ipv6 route 2001:DB8:CAFE:2::/64 Serial0/0/0
ipv6 route 2001:DB8:CAFE:3::/64 Serial0/0/1
ipv6 route 2001:DB8:CAFE:A002::/64 Serial0/0/0
ipv6 route 2001:DB8:CAFE:A002::/64 2001:DB8:CAFE:A003::2 5
ipv6 route 2001:DB8:FACE::/48 Serial0/0/1
ipv6 route 2001:DB8:FEED::/48 Serial0/0/1
R1#
```

例 7-19 通过 **show ipv6 static** 和 **show ipv6 static detail** 命令进一步验证了上述配置。在这两条命令中，去往 2001:db8:cafe:a002::/64 且管理距离为 5 的静态路由出现在输出结果中，但是没有标记*，因而表明该路由没有被安装到 RIB 或路由表中。请注意，命令 **show ipv6 static detail** 的输出结果中还包含了用于所有只使用下一跳接口的静态路由的已解析的出接口。

### 例 7-19　使用命令 show ipv6 static

```
R1# show ipv6 static
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:CAFE:2::/64 via interface Serial0/0/0, distance 1
* 2001:DB8:CAFE:3::/64 via interface Serial0/0/1, distance 1
* 2001:DB8:CAFE:A002::/64 via interface Serial0/0/0, distance 1
  2001:DB8:CAFE:A002::/64 via nexthop 2001:DB8:CAFE:A003::2, distance 5
* 2001:DB8:FACE::/48 via interface Serial0/0/1, distance 1
* 2001:DB8:FEED::/48 via interface Serial0/0/1, distance 1
R1#

R1# show ipv6 static detail
IPv6 Static routes
Code: * - installed in RIB
* 2001:DB8:CAFE:2::/64 via interface Serial0/0/0, distance 1
* 2001:DB8:CAFE:3::/64 via interface Serial0/0/1, distance 1
* 2001:DB8:CAFE:A002::/64 via interface Serial0/0/0, distance 1
  2001:DB8:CAFE:A002::/64 via nexthop 2001:DB8:CAFE:A003::2, distance 5
     Resolves to 1 paths (max depth 1)
     via Serial0/0/1
* 2001:DB8:FACE::/48 via interface Serial0/0/1, distance 1
* 2001:DB8:FEED::/48 via interface Serial0/0/1, distance 1
R1#
```

接下来从 R1 向 2001:db8:cafe:a002::1 发起 ping 操作。此时路由表中只有一条路由。通过对比例 7-20 和例 7-12 的输出结果可以看出，这些数据包仅使用 R2 路由表中的当前路由进行转发。

### 例 7-20　验证去往 2001:db8:cafe:a002::/64 的单一路由

```
R1# debug ipv6 packet
IPv6 unicast packet debugging is on
R1# ping 2001:db8:cafe:a002::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:A002::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
R1#
*Feb 18 02:42:44.494: IPv6: SAS picked source 2001:DB8:CAFE:A001::1 for
2001:DB8:CAFE:A002::1 (Serial0/0/0)
*Feb 18 02:42:44.494: IPV6: source 2001:DB8:CAFE:A001::1 (local)
*Feb 18 02:42:44.494:       dest 2001:DB8:CAFE:A002::1 (Serial0/0/0)
```

```
*Feb 18 02:42:44.494:        traffic class 0, flow 0x0, len 100+0, prot 58, hops
64, originating
<output omitted>
*Feb 18 02:42:44.522: IPv6: SAS picked source 2001:DB8:CAFE:A001::1 for
2001:DB8:CAFE:A002::1 (Serial0/0/0)
*Feb 18 02:42:44.526: IPV6: source 2001:DB8:CAFE:A001::1 (local)
*Feb 18 02:42:44.526:        dest 2001:DB8:CAFE:A002::1 (Serial0/0/0)
*Feb 18 02:42:44.526:        traffic class 0, flow 0x0, len 100+0, prot 58, hops
64, originating
<output omitted>
```

### 7.2.2 最终配置与验证

在 R1 和 ISP 路由器上利用 **ping** 命令可以验证本章示例拓扑结构的可达性（如例 7-21 所示）。当然，在每台路由器上向每个网络都发起一系列 **ping** 命令，能够更好地验证整个拓扑结构的可达性。

**例 7-21 验证可达性**

```
R1# ping 2001:db8:feed:1::2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:FEED:1::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
R1# ping 2001:db8:face:c0de::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:FACE:C0DE::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/56 ms
R1#

ISP# ping 2001:db8:cafe:1::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/70/72 ms
ISP#
```

通过检查每台路由器的路由表可以验证网络的连接性。例 7-22 显示了路由器 R1 的最终路由表信息。命令 **show ipv6 route summary** 显示了 R1 路由表中的许多前缀（如例 7-22 所示）。

## 例 7-22　通过检查 R1 的路由表来验证静态路由

```
R1# show ipv6 route
IPv6 Routing Table - 13 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
C   2001:DB8:CAFE:1::/64 [0/0]
     via ::, FastEthernet0/0
L   2001:DB8:CAFE:1::1/128 [0/0]
     via ::, FastEthernet0/0
S   2001:DB8:CAFE:2::/64 [1/0]
     via ::, Serial0/0/0
S   2001:DB8:CAFE:3::/64 [1/0]
     via ::, Serial0/0/1
C   2001:DB8:CAFE:A001::/64 [0/0]
     via ::, Serial0/0/0
L   2001:DB8:CAFE:A001::1/128 [0/0]
     via ::, Serial0/0/0
S   2001:DB8:CAFE:A002::/64 [1/0]
     via ::, Serial0/0/0
C   2001:DB8:CAFE:A003::/64 [0/0]
     via ::, Serial0/0/1
L   2001:DB8:CAFE:A003::1/128 [0/0]
     via ::, Serial0/0/1
S   2001:DB8:FACE::/48 [1/0]
     via ::, Serial0/0/1
S   2001:DB8:FEED::/48 [1/0]
     via ::, Serial0/0/1
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
R1#

R1# show ipv6 route summary
IPv6 Routing Table Summary - 13 entries
  5 local, 3 connected, 5 static, 0 RIP, 0 BGP, 0 IS-IS, 0 OSPF, 0 EIGRP
  Number of prefixes:
    /8: 1, /10: 1, /48: 2, /64: 6, /128: 3
R1#
```

## 7.3 IPv6 CEF

上一节讨论了进程交换和快速交换，也谈到了第三种交换方式 CEF。虽然有关 CEF 的详细信息已经超出了本书范围，但是为了大家更好地理解 CEF，本节还是要简单介绍一下 CEF。

CEF 是用于 IPv4 Cisco 路由器和多层交换机的交换技术，也可以用于 IPv6（称为 CEFv6）。CEF 使用的交换方法优化了路由查找操作，能够实现非常快速的包交换。

> 注：由于 CEFv6 的操作行为与 IPv4 CEF 完全一样，因而也将 CEFv6 称为 CEF。

CEF 使用两种表来实现快速转发操作。

- **FIB（Forwarding Information Base，转发信息库）**：FIB 类似于 IP 路由表，它是一种有序表，为路由表中每个 IPv6 前缀/前缀长度（IPv4 的每个 IPv4 网络/子网）首先列出最明细的路由。FIB 是 IP 路由表中所包含的转发信息的镜像，当网络中的路由或拓扑结构发生变化后，首先会更新 IP 路由表，然后才会将这些变化反应到 FIB 中。FIB 基于 IP 路由表中的信息来维护下一跳地址信息。
- **邻接表（Adjacency Table）**：FIB 包含了三层网络和下一跳信息。为了流水化处理包转发操作，将 FIB 表中每个下一跳表项相对应的二层信息存储在邻接表中。邻接表是通过 IPv6 邻居缓存表（或 IPv4 的 ARP 缓存表）构建而成的，包含了三层与二层地址的映射信息。邻接表中包含了下一跳节点（如路由器和主机）的 MAC 地址。只要路由器更新了其 IPv6 邻居缓存表（或 IPv4 的 ARP 缓存表），邻接表也会进行相应的更新。

有关 CEF 操作的具体处理过程已经超出了本书范围，大家可以通过 Cisco Press 出版的 *Implementing Cisco Switched Networks (SWITCH)* 获取详细信息。

Cisco IOS Release 12.2(13)T 及以后版本和 12.2(9)S 及以后版本都支持 CEFv6。默认情况下，CEFv6 功能是关闭的。在使用 IPv6 CEF 之前，必须先启用 IPv4 CEF。大多数平台在默认情况下都是启用 IPv4 CEF 的。如果没有启用，那么就需要通过全局配置命令 **ip cef** 来打开 IPv4 CEF 特性。启用了 IPv4 CEF 之后，就可以在全局配置模式下通过命令 **ipv6 cef** 来启用 IPv6 CEF：

```
Router(config) ip cef
Router(config) ipv6 cef
```

## 7.4 本章小结

本章讨论了以下两个与 IPv6 路由相关的主题：

- IPv6 路由表；
- IPv6 静态路由。

Cisco IOS 为 IPv4 和 IPv6 分别维护一张独立的路由表。可以使用命令 **show ipv6 route** 显示 IPv6 路由表。代码字段 C 表示这是一个直连网络。当配置了全局单播地址或唯一本地单播地址的 IPv6 接口处于 up 状态时，前缀以及前缀长度等信息都会被加入路由表中。链路本地地址不在路由表中。接口的单播地址也会被添加到 IPv6 路由表中，作为本地/128 主机路由，其代码字段值为 L。

必须在全局配置模式下通过命令 **ipv6 unicast-routing** 打开 Cisco 路由器路由 IPv6 数据包的能力，该特性在默认情况下是关闭的。IPv6 静态路由的配置与 IPv4 静态路由的配置非常相似。配置 IPv6 静态路由的基本命令语法如下：

```
Router(config)# ipv6 route ipv6-prefix/prefix-length {ipv6-address | interface-
type interface-number [administrative-distance]
```

将::/0 作为 *ipv6-prefix* / *prefix-length* 即可以配置IPv6静态默认路由。利用命令 **show ipv6 route static 和 show running-config | include ipv6 route** 可以验证静态路由的配置情况。

CEF 是设计用于 IPv4 Cisco 路由器和多层交换机的交换技术，也可以用于 IPv6（称为 CEFv6），CEF 使用的交换方法优化了路由查找操作，能够实现非常快速的包交换。在使用以下命令启用 IPv6 CEF 之前，必须先启用 IPv4 CEF：

```
Router(config) ipv6 cef
```

## 7.5　参考文献

Cisco IOS IPv6 Command Reference, www.cisco.com/en/US/docs/ios/ipv6/command/reference/ipv6_16.html

Cisco IOS IPv6 Configuration Guide, www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/12_4/ipv6_12_4_book.html

Implementing Static Routes for IPv6, www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-stat_routes.html

Implementing Traffic Filters and Firewalls for IPv6 Security, www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-sec_trfltr_fw.html

# 第 8 章　IPv6 IGP 路由协议

在第 6 章，我们已经为 Rick's Café 网络的拓扑结构进行了编址。第 7 章也使用了相同的拓扑结构，并利用静态路由实现了整个拓扑结构的可达性。静态路由对于进出末梢（stub）网络的路由选择来说是非常适用的，也被用作默认路由。但是对于包含多台路由器以及存在冗余链路的网络来说，如果只配置静态路由，那么非常不好了。使用静态路由的一些缺陷如下：

- 配置和维护工作比较费时；
- 配置容易出错，特别是在大型网络中；
- 无法动态适应网络的变化；
- 扩展性差。

动态路由协议可以解决静态路由的上述缺陷。动态路由协议的扩展性非常好，易于管理，而且能够确定去往远程网络的最佳路径，能够自动适应网络拓扑结构的变化。本章将讨论以下三种 IPv6 IGP（Interior Gateway Protocol，内部网关协议）路由协议：

- IPv6 RIPng（Routing Information Protocol next generation，下一代路由信息协议）；
- IPv6 EIGRP（Enhanced Interior Gateway Routing Protocol，增强型内部网关路由协议）；
- OSPFv3（Open Shortest Path First version 3，开放最短路径优先版本 3）。

注：IS-IS（Intermediate System-to-Intermediate System，中间系统到中间系统）是另一种可同时用于 IPv4 和 IPv6 的 IGP 路由协议。Cisco IOS 同时支持 IPv4 和 IPv6 的 IS-IS，但有关 IS-IS 的详细内容已经超出了本书范围，大家可以参考 *Cisco IOS IPv6 Configuration Guide, Implementing IS-IS for IPv6*: www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-is-is.html。

注：与 EGP（Exterior Gateway Protocol，外部网关协议）用于自治系统之间的路由选择相对，IGP 路由协议用于自治系统内部的路由选择。BGP（Border Gateway Protocol，边界网关协议）是目前 Internet 上在自治系统之间使用的 EGP。当前用于 IPv4 的 BGP 版本是 BGP-4，定义在 RFC 4271 “A Border Gateway Protocol 4 (BGP-4)” 中。

增强型的 BGP 版本 BGP+或多协议 BGP 扩展了 BGP-4 规范，以包含其他地址簇，如用于 IPv6 的 BGP。多协议 BGP 定义在 RFC 4760 “Multiprotocol Extensions for BGP-4” 和 RFC 2545 “Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing” 中。有关 BGP 的详细信息不在本书写作范围之内。

RIPng、IPv6 EIGRP 和 OSPFv3 与其对应的 IPv4 路由协议使用独立的进程，这些路由协议采取“夜间行船（ships in the night）”运行模式，拥有独立的路由表、独立的数据库以及独立的进程。IS-IS 单个进程可以同时支持两种编址方案，未来的 OSPFv3 单一进程也将同时支持两种编址方案。

IPv6 路由协议与其对应的 IPv4 路由协议之间的差异不大，IPv6 路由协议的处理及操作基本上都与 IPv4 相同。例如，IPv6 EIGRP 仍然使用 DUAL（Diffusing Update Algorithm，扩散更新算法）。OSPFv3 也仍然是一种多区域链路状态路由协议，即便是 IPv6 路由协议的配置，也与对应的 IPv4 路由协议配置极为相似。

注：正如本书前言所述，本书假定读者都已经掌握了 IPv4 以及 IPv4 路由协议，大家应该对掌握并理解 IPv4 的常用路由协议，如 RIPv2、EIGRP 和 OSPFv2。如果需要了解这些路由协议的详细信息，可以参考 Cisco Press 出版的 *Routing Protocols and Concepts* 或 *Routing TCP/IP*。本书主要讨论 IPv6 路由协议的基本概念和基本配置，相当于 CCNA 的理解水平。如果希望了解更多的内容，可以参考 Cisco Press 出版的 *Implementing Cisco IP Routing (ROUTE)*和 *Cisco Self Study: Implementing Ciseo IPv6 Net works*。

本章在讨论 RIPng、IPv6 EIGRP 和 OSPFv3 时，都使用曾经在第 6 章和第 7 章用到过的拓扑结构，并将讨论以下内容：

- IPv6 路由协议与其对应的 IPv4 路由协议之间的异同点；
- 配置 IPv6 路由协议以及分发默认路由的基本命令；
- 用于部署、确认网络收敛以及测试可达性的相关命令。

## 8.1 IPv6 RIPng

RIP 是目前仍在使用的最古老的距离矢量路由协议。虽然 RIP 缺少很多高级路由协议的高级功能，但是其简单性以及长期使用的现状很好地证明了其长期存在的必要性。很多人可能会认为 RIP 是一个“即将过时”的路由协议，但无论如何，RIP 仍然有自己

的 IPv6 形式，即 RIPng。

Charles Hedrick 于 1988 年在 RFC 1058“Routing Information Protocol”中制定了最早的 RIP 规范。此后 RIP 经过了多次更新，包括 1994 年的 RIPv2 和 1997 年的 RIPng。RIPng（下一代 RIP 或用于 IPv6 的 RIP）与 IPv4 中的 RIPv2 拥有相同的功能和相同的优势。RIPng 定义在 RFC 2080“RIPng for IPv6”中，RIPv2 定义在 RFC 2453“RIP Version 2”中。本书假定大家已经掌握了 RIPv2，而且相关内容也不在本书写作范围之内。

### 8.1.1　IPv6 RIPng 与 RIP2 对比

RIPng 是 IPv4 RIPv2 的 IPv6 对应协议。RIPng 的操作与配置都与 RIPv2 非常相似，但是也有一些非常重要的区别。下面是两者之间的主要特性对比。

- **宣告的路由：**RIPv2 宣告 IPv4 路由，而 RIPng 宣告路由以构造 IPv6 路由表。
- **距离矢量：**与链路状态路由协议中的每台路由器都有网络的拓扑映射相对，RIPv2 和 RIPng 都是使用 Bellman-Ford 算法的距离矢量协议。在链路状态环境中，路由器不但有其直连邻居的视图，而且还有整个路由域中其他所有路由器和网络的视图信息。
- **度量：**RIPv2 和 RIPng 都使用值为 1～16（表示跳数）的度量。由于最大跳数为 15，因而跳数达到 16（无穷大）时就表示不可达。
- **IPv6 路由表度量：**对于 RIPv2（和 RIPv1 来说），在路由更新中收到的度量就是接收路由器在其路由器中用于该网络的度量。例如，如果路由器 X 向路由器 Y 发送一条关于其直连网络的路由更新，路由器 X 宣告该路由的跳数为 1。那么下一跳路由器 Y 收到该 RIPv2 更新后，就将该路由及其度量 1 加入自己的 IPv4 路由表中，而不递增度量值。在 RIPv2 中，IPv4 路由表中显示的度量或跳数是以下一跳路由器为第一跳，到达远程网络所需的跳数。

在 Cisco 实现的 RIPng 中，虽然会发送相同的度量，但收到路由更新的路由器会递增该度量，这是与 IPv4 RIP 的重大区别。仍然以前面的例子为例，路由器 X 向路由器 Y 发送一条关于其直连网络的路由更新，路由器 X 宣告该路由的跳数为 1，那么接收该更新的邻接路由器 Y 在将该路由加入自己的 IPv4 路由表之前，会将该路由的度量加 1，也就是度量为 2。因此，在 Cisco 实现中，最终的 RIPng 跳数（度量）为：输入了命令 **show ipv6 route** 的路由器将自己作为第一跳，因而会将正常情况下 RIPv2 路由表中见到的跳数加 1。详细原因将在下一节解释。

- **基于 UDP 协议：**RIPv2 和 RIPng 消息都被封装在 UDP 报文段中。RIPv2 使用的端口号是 520，而 RIPng 使用的端口号是 521。
- **路由更新：**RIPv2 和 RIPng 都以 30 秒为周期发送路由更新，而且这两种路由协议也都支持触发更新。
- **定时器：**RIPv2 和 RIPng 都使用抑制定时器（holddown timer）和其他定时器

来防止路由环路。

- **水平分割**：RIPv2 和 RIPng 都使用水平分割（split horizon）或水平分割加毒性反转（poison reverse）机制来防止路由环路。
- **自动汇总**：RIPv2 能够按照有类别边界进行自动汇总。由于 IPv6 中不存在有类别编址的概念，因而 RIPng 不支持该功能，IPv6 地址被认为是无类别地址。
- **源地址和目的地址**：RIPv2 向 IPv4 多播地址 224.0.0.9（RIPv2 的多播组地址）发送消息，并且这些消息使用出接口的源 IPv4 地址。RIPng 向 IPv6 多播地址 FF02::9（链路本地范围内的全部 RIP 路由器多播地址）发送消息，并使用出接口的链路本地地址作为这些消息的源地址。
- **认证**：RIPv2 使用明文认证或 MD5（Message Digest 5，消息摘要 5）认证机制。RIPng 没有自己的认证机制，所有的认证和/或加密操作都由 IPv6 定义的标准 IPSec 功能来完成。

### 8.1.2 在 Cisco 路由器上配置 RIPng

下面将以第 6 章曾经用过的拓扑结构为例，为 2001:0DB8:CAFE::/48 网络启用 RIPng（如图 8-1 所示）。为此需要配置路由器 R1、R2 和 R3，使用 RIPng 共享路由信息。由于该网络属于末梢网络，因而 R3 配置了一条经由 ISP 路由器的默认路由。利用 RIPng 在 RIPng 域中分发 R3 的默认路由。ISP 路由器配置了一条去往 2001:0DB8:CAFE::/48 的静态路由。

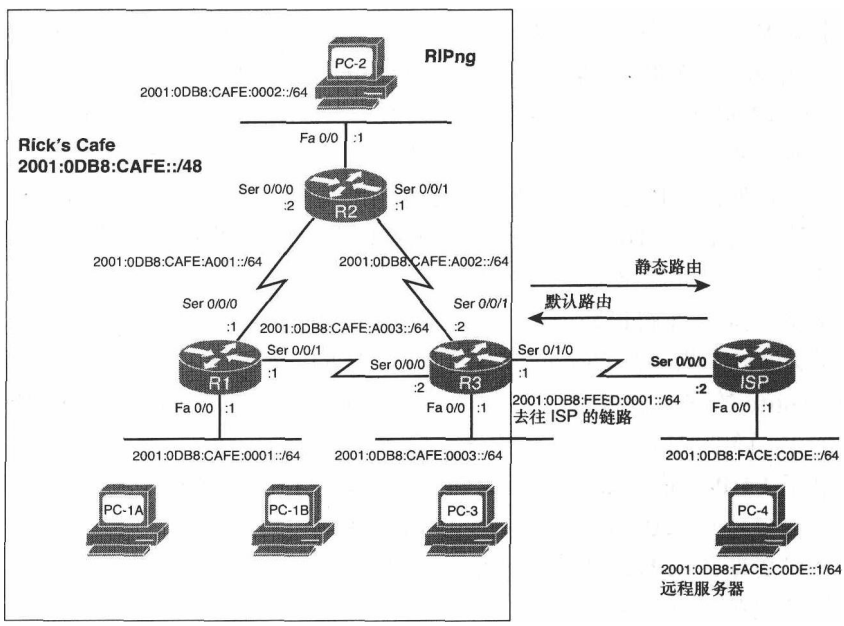



图 8-1 Rick's Café 网络的 RIPng 拓扑结构

RIPng 以及其他 IPv6 路由协议都使用路由器的链路本地地址作为其消息的源地址。首先检查第 6 章曾经为这 4 台路由器的接口所做的配置（如例 8-1 所示），每台路由器的每个接口上都配置了相同的链路本地地址（第 6 章采取手工配置方式以易于识别每台路由器）。请注意，链路本地地址只要在本链路具有唯一性即可。

**例 8-1　接口的全局单播地址和链路本地地址**

```
R1# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::1
    2001:DB8:CAFE:1::1
Serial0/0/0                [up/up]
    FE80::1
    2001:DB8:CAFE:A001::1
Serial0/0/1                [up/up]
    FE80::1
    2001:DB8:CAFE:A003::1
R1#
```

```
R2# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::2
    2001:DB8:CAFE:2::1
Serial0/0/0                [up/up]
    FE80::2
    2001:DB8:CAFE:A001::2
Serial0/0/1                [up/up]
    FE80::2
    2001:DB8:CAFE:A002::1
R2#
```

```
R3# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::3
    2001:DB8:CAFE:3::1
Serial0/0/0                [up/up]
    FE80::3
    2001:DB8:CAFE:A003::2
Serial0/0/1                [up/up]
    FE80::3
    2001:DB8:CAFE:A002::2
Serial0/1/0                [up/up]
    FE80::3
    2001:DB8:FEED:1::1
```

```
R3#
```

```
ISP# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::FEED
    2001:DB8:FACE:C0DE::1
Serial0/0/0                [up/up]
    FE80::FEED
    2001:DB8:FEED:1::2
ISP#
```

为了配置 RIPng 路由进程，首先要在全局配置模式下使用命令 **ipv6 router rip**。命令 **ipv6 router rip** 与命令 **router rip** 很相似，区别在于该命令是专用于 IPv6 的命令。该命令的语法格式如下：

```
Router(config)# ipv6 router rip name
```

该命令可以为 IPv6 启用 RIPng 路由进程。其中，*name* 用于标识特定的路由进程。

例 8-2 在路由器 R1 上试图通过命令 **ipv6 router rip RIPNG-R1** 来启用 RIPng，但是却出现以下错误信息：

```
% IPv6 routing not enabled
```

这是因为此时还未启用 IPv6 路由，因而需要首先在 R1 上应用全局配置命令 **ipv6 unicast-routing**，然后再输入命令 **ipv6 router rip RIPNG-R1**。

**例 8-2　启用 RIPng**

```
R1(config)# ipv6 router rip RIPNG-R1
% IPv6 routing not enabled
R1(config)#
R1(config)# ipv6 unicast-routing
R1(config)# ipv6 router rip RIPNG-R1
R1(config-rtr)#
```

名字用来唯一地表示本地路由器上的 RIPng 进程。名字区分大小写且仅具有本地意义。虽然 RIPng 路由域中所有路由器的 RIPng 进程名不必相同，但为了操作方便，通常还是将路由域中所有路由器的 RIPng 进程名设置为相同，以便因大量不同的名字而让 NOC（Network Operations Center，网络操作中心）人员产生混淆。

例 8-3 显示了调整 RIPng 进程的 IPv6 路由器子命令。

### 例 8-3　IPv6 路由器子命令

```
R1(config)# ipv6 router rip RIPNG-R1
R1(config-rtr)# ?
  default           Set a command to its defaults
  distance          Administrative distance
  distribute-list   Filter networks in routing updates
  exit              Exit from IPv6 routing protocol configuration mode
  maximum-paths     Forward packets over multiple paths
  no                Negate a command or set its defaults
  poison-reverse    Poison reverse updates
  port              Port and multicast address
  redistribute      Redistribute IPv6 prefixes from another routing protocol
  shutdown          Shutdown protocol
  split-horizon     Split horizon updates
  timers            Adjust routing timers

R1(config-rtr)#
```

如果要在接口上配置 RIPng 路由进程，那么就要在每个接口上使用接口命令 **ipv6 rip** *name* **enable** 以启用 RIPng。其中，*name* 用于标识特定的路由进程。例 8-4 给出了路由器 R1 的 RIPng 接口配置命令。请注意，接口命令中使用的名字必须与全局配置模式下启用 RIPng 进程所使用的名字相同，而且名字区分大小写，必须完全匹配才行。

### 例 8-4　在 R1 上启用 RIPng

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 rip RIPNG-R1 enable
R1(config-if)# exit
R1(config)# interface serial 0/0/0
R1(config-if)# ipv6 rip RIPNG-R1 enable
R1(config-if)# exit
R1(config)# interface serial 0/0/1
R1(config-if)# ipv6 rip RIPNG-R1 enable
R1(config-if)# end
R1#
```

**注**：对于 RIPng 和其他 IPv6 路由协议来说，Cisco IOS 并不像配置 IPv4 路由协议那样在路由器配置模式下使用 **network** 命令，而是在接口上直接启用 IPv6 路由协议。

例 8-5 显示的命令是用于完成拓扑结构中路由器 R2 和 R3 的 RIPng 配置。由于 R3 的 serial 0/1/0 接口被用作去往 ISP 路由器的默认路由，因而该接口未包含在 RIPng 进程中。虽然在第 7 章的时候已经配置了命令 **ipv6 unicast-routing** 以启用 IPv6 路由能力，

但这里仍然首先配置了该命令。请注意，虽然每台路由器都有一个唯一的名字，但是在全局配置模式下用来启用RIPng的名字也被用来在接口上启用RIPng。为了解释这一点，路由器R1、R2和R3都配置了不同的名字。不过在实际部署中，同一路由域中的所有路由器通常都应该使用相同的名字。

**例8-5 在路由器R2和R3上启用RIPng**

```
R2(config)# ipv6 unicast-routing
R2(config)# ipv6 router rip RIPNG-R2
R2(config-rtr)# exit
R2(config)# interface fastethernet 0/0
R2(config-if)# ipv6 rip RIPNG-R2 enable
R2(config-if)# exit
R2(config)# interface serial 0/0/0
R2(config-if)# ipv6 rip RIPNG-R2 enable
R2(config-if)# exit
R2(config)# interface serial 0/0/1
R2(config-if)# ipv6 rip RIPNG-R2 enable
R2(config-if)#
```

```
R3(config)# ipv6 unicast-routing
R3(config)# ipv6 router rip RIPNG-R3
R3(config-rtr)# exit
R3(config)# interface fastethernet 0/0
R3(config-if)# ipv6 rip RIPNG-R3 enable
R3(config-if)# exit
R3(config)# interface serial 0/0/0
R3(config-if)# ipv6 rip RIPNG-R3 enable
R3(config-if)# exit
R3(config)# interface serial 0/0/1
R3(config-if)# ipv6 rip RIPNG-R3 enable
R3(config-if)#
```

在为网络2001:db8:cafe::/48中的3台路由器启用了RIPng后，接下来就应该在R3与ISP路由器之间配置静态路由了。首先在路由器R3上配置默认路由（如例8-6所示）。

为了将IPv6默认路由分发到RIPng域中，需要使用接口命令**ipv6 rip default-information**。如例8-6所示，R3的两个接口上都配置了该命令，目的是将该默认路由宣告给邻接的RIPng路由器。命令**ipv6 rip default-information**的完整语法格式如下：

**例 8-6　在 R3 上配置默认路由并通过 RIPng 进行分发**

```
! This is the default route
R3(config)# ipv6 route ::/0 serial 0/1/0
R3(config)# interface serial 0/0/0
R3(config-if)# ipv6 rip RIPNG-R3 default-information originate
R3(config-if)# exit
R3(config)# interface serial 0/0/1
R3(config-if)# ipv6 rip RIPNG-R3 default-information originate
R3(config-if)#
```

```
Router(config-if)# ipv6 rip name default-information {only | originate} [metric
metric-value]
```

该命令语法中的术语如下。

- *name*：标识 IPv6 RIP 路由进程的名字。
- **only**：仅宣告 IPv6 默认路由（::/0），抑制所有其他路由的宣告。
- **originate**：宣告 IPv6 默认路由（::/0），其他路由的宣告不受影响。
- **metric** *metric-value*（可选）：为默认路由指定度量值，取值范围是 1～15。

注：RIPng 中的接口命令 **ipv6 rip default-information** 与 RIPv2 中的全局配置命令 **default-information originate** 相似，区别在于 RIPng 命令是在接口上配置的，而 RIPv2 命令是在全局配置模式下配置的。

例 8-7 给出了在 ISP 路由器上配置去往网络 2001:0DB8:CAFE::/48 的静态路由。

**例 8-7　在 ISP 路由器上配置去往 2001:0DB8:CAFE::/48 的静态路由**

```
ISP(config)# ipv6 route 2001:db8:cafe::/48 serial 0/0/0
```

### 8.1.3　验证 RIPng

本节将使用与 RIPv2 中相似的命令来验证上述配置情况。如例 8-8 所示，命令 **show ipv6 protocols** 用来显示所有处于激活状态的 IPv6 路由协议进程的参数及其当前状态。默认情况下，只要启用了命令 **ipv6 unicast-routing**，IPv6 直连路由和静态路由就处于激活状态。R1 上的输出结果还显示了其 3 个接口上的 RIPng 进程 RIPNG-R1 都处于激活状态。

例 8-8 还通过命令 **show ipv6 interface fastethernet 0/0** 验证了 R1 的 Fast Ethernet 0/0 接口目前是 RIPng 多播组 FF02::9 的成员。RIPng 将 FF02::9 用作 RIP 更新消息的目的地址。

例 8-8 命令 show ipv6 protocols

```
R1# show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "static"
IPv6 Routing Protocol is "rip RIPNG-R1"
  Interfaces:
    Serial0/0/1
    Serial0/0/0
    FastEthernet0/0
  Redistribution:
    None
R1#
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::1
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:CAFE:1::1, subnet is 2001:DB8:CAFE:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::9
    FF02::1:FF00:1
  MTU is 1500 bytes
<output omitted>
```

命令 **show ipv6 route rip** 可以显示 R1 路由表中的 RIPng 路由（如例 8-9 所示）。请注意，在 R3 上配置的默认路由（::/0）已经通过 RIPng 分发给 R1 了。接下来详细分析以下路由中的信息：

```
R   2001:DB8:CAFE:2::/64 [120/2]
  via FE80::2, Serial0/0/0
```

该路由使用了以下术语。

- **R**：表示生成该路由的路由协议是 RIPng。
- **2001:DB8:CAFE:2::/64**：是远程网络的前缀和前缀长度。
- **[120/2]**：方括号中的第一个数值是信息源的管理距离，此处为 120，这与 RIPv1 和 RIPv2 相同。第二个数值是该路由的度量。请注意，这里的跳数就是 RIPv1 和 RIPv2 中的跳数。但是与 IPv4 RIP 不同的是，在 Cisco 的 RIPng 实现中，路由器从邻居收到路由更新后会将度量加 1。事实上，路由器是将自己纳入到达目的网络的跳数之内了。

- **via FE80::2**：下一跳路由器的链路本地地址。
- **Serial0/0/0**：下一跳路由器的出接口。

**例 8-9　R1 的 IPv6 路由表**

```
R1# show ipv6 route rip
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
R   ::/0 [120/2]
     via FE80::3, Serial0/0/1
R   2001:DB8:CAFE:2::/64 [120/2]
     via FE80::2, Serial0/0/0
R   2001:DB8:CAFE:3::/64 [120/2]
     via FE80::3, Serial0/0/1
R   2001:DB8:CAFE:A002::/64 [120/2]
     via FE80::2, Serial0/0/0
     via FE80::3, Serial0/0/1
R1#
```

例 8-10 使用 **ping** 命令来验证网络的可达性。可以看出，对 R2、R3 和 ISP 路由器的快速以太网接口发起的 ping 操作均成功。

**例 8-10　使用 ping 命令来验证可达性**

```
R1# ping 2001:db8:cafe:2::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:2::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
R1# ping 2001:db8:cafe:3::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:3::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
R1# ping 2001:db8:face:c0de::1

Type escape sequence to abort.
```

```
Sending 5, 100-byte ICMP Echos to 2001:DB8:FACE:C0DE::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/60 ms
R1#
```

例 8-11 中的 **show ipv6 rip** 命令显示了当前 RIPng 进程的相关信息。除了增加了进程名之外，UDP 端口号、多播组、进程 ID（pid）以及大多数信息都与 RIPv2 相似（RIPv2 使用 UDP 端口号 520，而 RIPng 使用 UDP 端口号 521）。

**例 8-11　在 R1 上使用 show ipv6 rip 命令**

```
R1# show ipv6 rip
RIP process "RIPNG-R1", port 521, multicast-group FF02::9, pid 179
     Administrative distance is 120. Maximum paths is 16
     Updates every 30 seconds, expire after 180
     Holddown lasts 0 seconds, garbage collect after 120
     Split horizon is on; poison reverse is off
     Default routes are not generated
     Periodic updates 12, trigger updates 6
  Interfaces:
    Serial0/0/1
    Serial0/0/0
    FastEthernet0/0
  Redistribution:
    None
R1#
```

在 Cisco IOS 的 RIPng 软件实现中，每个 IPv6 RIP 进程都维护一个被称为 RIB（路由信息库）的本地路由表。RIPng 会试图将其本地数据库中每条未过期的路由都插入主 IPv6 RIB（也被称为 IPv6 路由表）。如果路由器学到了多条去往同一网络的路由，那么 RIPng 仅在本地 RIPng RIB 中存储开销最小的路由。如果从管理距离比 RIPng 更优的路由源学到了同一条路由，那么该 RIP 路由虽然不会被加入到 IPv6 路由表中，但是仍然位于 RIPng RIB 中。命令 **show ipv6 rip database** 的作用就是显示 RIPng RIB（如例 8-12 所示）。

**例 8-12　R1 的 RIPng 路由信息库**

```
R1# show ipv6 rip database
RIP process "RIPNG-R1", local RIB
 2001:DB8:CAFE:2::/64, metric 2, installed
     Serial0/0/0/FE80::2, expires in 173 secs
 2001:DB8:CAFE:3::/64, metric 2, installed
```

```
     Serial0/0/1/FE80::3, expires in 176 secs
 2001:DB8:CAFE:A001::/64, metric 2
     Serial0/0/0/FE80::2, expires in 173 secs
 2001:DB8:CAFE:A002::/64, metric 2, installed
     Serial0/0/1/FE80::3, expires in 176 secs
     Serial0/0/0/FE80::2, expires in 173 secs
 2001:DB8:CAFE:A003::/64, metric 2
     Serial0/0/1/FE80::3, expires in 176 secs
 ::/0, metric 2, installed
     Serial0/0/1/FE80::3, expires in 176 secs
R1#
```

命令 **show ipv6 rip next-hops** 可以显示下一跳地址以及使用该下一跳地址的 RIPng 路由数（不论该路由是否在 IPv6 路由表中）。显示的信息源于 RIPng RIB，而非 IPv6 路由表。例 8-13 给出了在 R1 上应用命令 **show ipv6 rip next-hops** 的输出结果。

**例 8-13　在 R1 上应用命令 show ipv6 rip next-hops**

```
R1# show ipv6 rip next-hops
 RIP process "RIPNG-R1", Next Hops
  FE80::2/Serial0/0/0 [3 paths]
  FE80::3/Serial0/0/1 [4 paths]
R1#
```

例 8-14 中的命令 **debug ipv6 rip** 显示了路由器 R1 所发送和接收到的 RIPng 路由消息。请注意，路由更新的源地址是接口的链路本地地址，目的地址是 FF02::9，即链路本地范围内的全部 RIP 路由器（all-RIP-routers）多播地址。输出结果中还包括了路由的度量和前缀/前缀长度。与 RIPv2 不同，接收端 RIPng 路由器会对度量递增 1，事实上是将自己作为第一跳路由器。

**例 8-14　在 R1 上应用命令 debug ipv6 rip**

```
R1# debug ipv6 rip
RIP Routing Protocol debugging is on
R1#
*Feb 20 18:01:18.511: RIPng: response received from FE80::3 on Serial0/0/1 for
RIPNG-R1
*Feb 20 18:01:18.515:        src=FE80::3 (Serial0/0/1)
*Feb 20 18:01:18.515:        dst=FF02::9
*Feb 20 18:01:18.515:        sport=521, dport=521, length=112
*Feb 20 18:01:18.515:        command=2, version=1, mbz=0, #rte=5
*Feb 20 18:01:18.515:        tag=0, metric=1, prefix=2001:DB8:CAFE:3::/64
*Feb 20 18:01:18.515:        tag=0, metric=1, prefix=2001:DB8:CAFE:A003::/64
```

```
*Feb 20 18:01:18.515:          tag=0, metric=1, prefix=2001:DB8:CAFE:A002::/64
*Feb 20 18:01:18.515:          tag=0, metric=2, prefix=2001:DB8:CAFE:2::/64
*Feb 20 18:01:18.515:          tag=0, metric=1, prefix=::/0
R1#
*Feb 20 18:01:25.071: RIPng: Sending multicast update on Serial0/0/1 for RIPNG-R1
*Feb 20 18:01:25.071:          src=FE80::1
*Feb 20 18:01:25.071:          dst=FF02::9 (Serial0/0/1)
*Feb 20 18:01:25.071:          sport=521, dport=521, length=92
*Feb 20 18:01:25.071:          command=2, version=1, mbz=0, #rte=4
*Feb 20 18:01:25.071:          tag=0, metric=1, prefix=2001:DB8:CAFE:1::/64
*Feb 20 18:01:25.071:          tag=0, metric=1, prefix=2001:DB8:CAFE:A001::/64
*Feb 20 18:01:25.071:          tag=0, metric=1, prefix=2001:DB8:CAFE:A003::/64
*Feb 20 18:01:25.071:          tag=0, metric=2, prefix=2001:DB8:CAFE:2::/64
```

例 8-15 显示了从这 4 台路由器的最终运行配置中抽取的与 RIPng 相关的内容。

**例 8-15 运行配置中与 RIPng 相关的内容**

```
R1# show running-config
!
ipv6 unicast-routing
!
ipv6 router rip RIPNG-R1
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:1::1/64
 ipv6 rip RIPNG-R1 enable
!
interface Serial0/0/0
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A001::1/64
 ipv6 rip RIPNG-R1 enable
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A003::1/64
 ipv6 rip RIPNG-R1 enable
!
R1#

R2# show running-config
```

```
!
ipv6 router rip RIPNG-R2
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:2::1/64
 ipv6 rip RIPNG-R2 enable
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:A001::2/64
 ipv6 rip RIPNG-R2 enable
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:A002::1/64
 ipv6 rip RIPNG-R2 enable
!
R2#
```

---

```
R3# show running-config
!
ipv6 unicast-routing
!
ipv6 router rip RIPNG-R3
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:3::1/64
 ipv6 rip RIPNG-R3 enable
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:A003::2/64
 ipv6 rip RIPNG-R3 enable
 ipv6 rip RIPNG-R3 default-information originate
!
interface Serial0/0/1
```

```
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:A002::2/64
 ipv6 rip RIPNG-R3 enable
 ipv6 rip RIPNG-R3 default-information originate
!
interface Serial0/1/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:FEED:1::1/64
!
ipv6 route ::/0 Serial0/1/0
!
R3#

ISP# show running-config
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::FEED link-local
 ipv6 address 2001:DB8:FACE:C0DE::1/64
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::FEED link-local
 ipv6 address 2001:DB8:FEED:1::2/64
!
ipv6 route 2001:DB8:CAFE::/48 Serial0/0/0
!
ISP#
```

在讨论 IPv6 EIGRP 之前，例 8-16 给出了删除 RIPng 进程的命令。全局配置命令 **no ipv6 router rip** *name* 可以删除 RIPng 进程，包括该进程使用的所有接口配置命令。

**例 8-16　删除 RIPng 进程**

```
R1(config)# no ipv6 router rip RIPNG-R1
R2(config)# no ipv6 router rip RIPNG-R2
R3(config)# no ipv6 router rip RIPNG-R3
```

## 8.2　IPv6 EIGRP

EIGRP（Enhanced Interior Gateway Routing Protocol，增强型内部网关路由协议）是一种距离矢量无类别路由协议，最早于 1992 年在 Cisco IOS9.21 在发布。顾名思义，EIGRP 是 Cisco IGRP（Interior Gateway Routing Protocol，内部网关路由协议）的增强型协议。这两种协议都是 Cisco 专有协议，只能运行在 Cisco 路由器之上。Cisco 开发 EIGRP 的主要目的是创建一种 IGRP 的无类别版本。EIGRP 包含了很多其他距离矢量路由协议（如 RIP，包括 RIPv1 和 RIPv2）没有的功能特性。

**注**：有时也用术语混合路由协议（hybrid routing protocol）来定义 EIGRP，但是，该术语带有很大的误导性，因为 EIGRP 并不是距离矢量路由协议与链路状态路由协议的混合体。EIGRP 只是距离矢量路由协议，因而 Cisco 不再使用该术语来称呼 EIGRP 了。

EIGRP 的独特之处在于其 RTP（Reliable Transport Protocol，可靠传输协议）。RTP 可以为 RIGRP 包提供可靠或不可靠传输机制。此外，EIGRP 还可以与启用了 EIGRP 的直连路由器建立邻居关系，邻居关系被用于跟踪这些邻居的状态。RTP 和跟踪邻居的邻接关系为 EIGRP 的 DUAL（Diffusing Update Algorithm，扩散更新算法）提供了工作条件。

作为驱动 EIGRP 的计算引擎，DUAL 是该路由协议的核心，可以保证整个路由域中的路径与备份路径都是无环路的。DUAL 会选择一条主路由（称为后继路由[successor]）安装到路由表中，同时还维护这一个无环路备用路由列表（称为可行后继路由[feasible successor]）。后继路由和可行后继路由都被维护在 EIGRP 的拓扑表中。

IPv6 EIGRP 是与 IPv4 EIGRP 对应的用来路由 IPv6 前缀的路由协议。IPv4 EIGRP 运行在 IPv4 网络层之上，负责 IPv4 EIGRP 对等体之间的通信，并且只宣告 IPv4 路由。IPv6 EIGRP 的功能与 IPv4 EIGRP 相同，只是运行于 IPv6 网络层之上，负责 IPv6 EIGRP 对等体之间的通信，并且仅宣告 IPv6 路由。本书假定大家已经掌握了 EIGRP，而且相关内容也不在本书写作范围之内。

### 8.2.1　IPv4 EIGRP 与 IPv6 EIGRP 对比

Cisco IOS Release 12.4(6)T 及以后版本都支持 IPv6 EIGRP。虽然 IPv4 EIGRP 和 IPv6 EIGRP 都是独立配置和度量管理的，但是两者在配置上非常相似。两种协议都支持很多相同的进程和操作功能。

IPv4 EIGRP 与 IPv6 EIGRP 主要功能特性的对比情况如下。

- **宣告的路由**：IPv6 EIGRP 宣告 IPv4 路由，而 IPv6 EIGRP 宣告路由以构造 IPv6 路由表。

- **距离矢量**：IPv4 EIGRP 与 IPv6 EIGRP 都是距离矢量路由协议。
- **收敛技术**：IPv4 EIGRP 与 IPv6 EIGRP 都使用 DUAL 算法，使用相同的收敛技术和进程，包括后继路由、可行后继路由、可行距离（feasible distance）和报告距离（reported distance）。
- **度量**：IPv4 EIGRP 与 IPv6 EIGRP 都使用带宽、时延、可靠性和负荷来作为它们的复合度量，而且默认情况下只使用带宽和时延。
- **传输协议**：IPv4 EIGRP 与 IPv6 EIGRP 都使用 RTP 将 EIGRP 包可靠、有序地传送给所有邻居。
- **部分 SPF**：IPv4 EIGRP 与 IPv6 EIGRP 在目的地发生变化后都只发送增量更新，而不是周期性地发送路由表的全部内容。
- **邻居发现机制**：IPv4 EIGRP 与 IPv6 EIGRP 都使用简单的 Hello 机制来了解邻居路由器并建立邻接关系。IPv6 EIGRP 使用链路本地地址来实现邻居发现和建立邻居邻接关系。
- **自动汇总**：IPv4 EIGRP 能够按照有类别边界进行自动汇总。由于 IPv6 EIGRP 中不存在有类别编址的概念，因而该功能不适用于 IPv6 EIGRP。
- **源地址和目的地址**：IPv4 EIGRP 向多播地址 224.0.0.10（EIGRP 多播组地址）发送消息，这些消息使用出接口的源 IPv4 地址；IPv6 EIGRP 向 IPv6 多播地址 FF02::10（链路本地范围内的全部 EIGRP 路由器[all-EIGRP-router]多播地址），并使用出接口的链路本地地址作为这些消息的源地址。
- **认证**：IPv4 EIGRP 可以使用明文认证或 MD5（Message Digest 5，消息摘要 5）认证机制，IPv6 EIGRP 也使用 MD5 认证机制。
- **路由器 ID**：IPv4 EIGRP 与 IPv6 EIGRP 都使用 32 比特的路由器 ID。32 比特的路由器 ID 采取点分十进制表示形式，通常就是 IPv4 地址。如果路由器没有配置 IPv4 地址，那么就需要通过 EIGRP 的 **router-id** 命令，使用 IPv4 地址来配置路由器 ID。两种协议都使用相同的进程来确定 32 比特路由器 ID。

### 8.2.2 在 Cisco 路由器上配置 IPv6 EIGRP

下面仍然以第 6 章曾经用过的拓扑结构为例，为 2001:0DB8:CAFE::/48 网络启用 EIGRP（如图 8-2 所示）。

为此需要配置路由器 R1、R2 和 R3，在 IPv6 EIGRP 路由域中共享路由信息。仍然为 R3 配置一条经由 ISP 路由器的默认路由，并配置 R3 将该默认路由分发给其他 EIGRP 路由器。同时，仍然为 ISP 路由器配置一条去往 2001:0DB8:CAFE::/48 的静态路由。这两条静态路由的情况如例 8-17 所示。

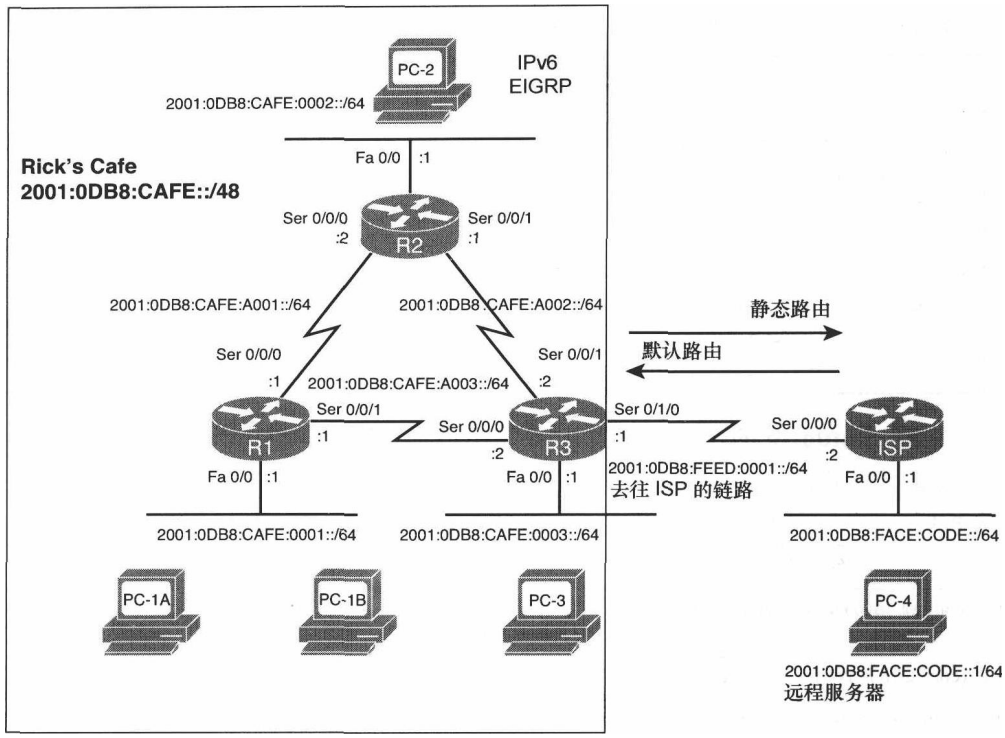



图 8-2　Rick's Café 网络的 IPv6 EIGRP 拓扑结构

### 例 8-17　R3 和 ISP 路由器上的静态路由

```
R3(config)# ipv6 route ::/0 serial 0/1/0

ISP(config)# ipv6 route 2001:db8:cafe::/48 serial 0/0/0
```

例 8-18 解释了路由器 R1 的 IPv6 EIGRP 配置。首先通过命令 **ipv6 unicast-routing** 启用 IPv6 路由功能，然后再配置 EIGRP，在全局配置模式下使用命令 **ipv6 router eigrp 100** 创建一个 IPv6 EIGRP 路由进程（使用 as-number[AS 号] 100）。该命令类似于 IPv4 EIGRP 中的 **eigrp** *as-number* 命令。与 IPv4 EIGRP 一样，IPv6 EIGRP 路由域中所有路由器上的 as-number 必须相同。继续看例 8-18，通过命令 **router-id** 来配置 EIGRP 的路由器 ID。路由器 ID 是以点分十进制形式表示的 32 比特 IPv4 地址。

该命令对本拓扑结构中的所有 EIGRP 路由器来说都是必需的，因为这些路由器都没有配置 IPv4 地址。例 8-18 中最后的路由器配置命令是 **no shutdown**。

例 8-18 在 R1 上启用 IPv6 EIGRP

```
R1(config)# ipv6 unicast-routing

R1(config)# ipv6 router eigrp 100
R1(config-rtr)# router-id ?
  A.B.C.D  EIGRP Router-ID in IP address format

R1(config-rtr)# router-id 10.1.1.1
R1(config-rtr)# no shutdown
R1(config-rtr)# end
R1#
```

命令 **no shutdown** 的作用是激活 IPv6 路由进程，默认是 **shutdown**。表 8-1 给出了这些命令的语法格式。

表 8-1 创建 IPv6 EIGRP 路由进程的命令

| 命令 | 描述 |
|---|---|
| Router(config)# **ipv6 router eigrp** *as-number* | 启用 IPv6 EIGRP 路由进程，as-number 用于标识特定的路由进程，EIGRP 路由域中所有路由器的 as-number 都必须相同 |
| Router(config-rtr)# **router-id** *router-id* | 路由器为 IPv6 EIGRP 路由进程使用 32 比特 IPv4 地址来选择路由器 ID |
| Router(config-rtr)# **no shutdown** | 该命令是开始运行 EIGRP 进程必可不少的命令 |

例 8-19 通过命令 **ipv6 eigrp** *as-number* 为 R1 的接口启用了 IPv6 EIGRP 路由进程。接口命令中的 as-number 必须与在路由器上启用 IPv6 EIGRP 路由进程的 as-number 相同。为特定接口启用 IPv6 EIGRP 的命令语法如下：

```
Router(config-if)# ipv6 eigrp as-number
```

命令中的 as-number 的作用是标识特定的路由进程，必须与用来创建路由进程的命令 **ipv6 router eigrp** *as-number* 中的 as-number 相同。

例 8-19 给出了在 R1 的接口上启用 IPv6 EIGRP 路由进程的示例。

例 8-19 在 R1 的接口上启用 IPv6 EIGRP 路由进程

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 eigrp 100
R1(config-if)# exit
R1(config)# interface serial 0/0/0
R1(config-if)# ipv6 eigrp 100
R1(config-if)# exit
R1(config)# interface serial 0/0/1
R1(config-if)# ipv6 eigrp 100
R1(config-if)# end
R1#
```

如果未正确配置 EIGRP 路由进程，那么使用命令 **show ipv6 eigrp** 之后，就会收到相应的 IOS 通告消息（如例 8-20 和例 8-21 所示）。如果 EIGRP 路由进程被关闭了，那么就会出现例 8-20 所示的 IOS 通告消息“% EIGRP 100 is in SHUTDOWN”。如果配置了 EIGRP 进程，但是由于无有效的 IPv4 地址，使得无法自动配置路由器 ID，就会出现例 8-21 所示的 IOS 通告消息“% No router ID for EIGRP 100”。

**例 8-20　IPv6 EIGRP 处于关闭状态**

```
R1# show ipv6 eigrp neighbors
IPv6-EIGRP neighbors for process 100
% EIGRP 100 is in SHUTDOWN
R1#
```

**例 8-21　IPv6 EIGRP 无有效的路由器 ID**

```
R1# show ipv6 eigrp neighbors
IPv6-EIGRP neighbors for process 100
% No router ID for EIGRP 100
R1#
```

路由器 R2 和 R3 都使用了与 R1 相同的 as-number 配置其 IPv6 EIGRP（如例 8-22 所示）。

**例 8-22　在 R2 和 R3 上启用 IPv6 EIGRP**

```
R2(config)# ipv6 unicast-routing
R2(config)# ipv6 router eigrp 100
R2(config-rtr)# router-id 10.2.2.2
R2(config-rtr)# no shutdown
R2(config-rtr)# exit
R2(config)#
R2(config)# interface fastethernet 0/0
R2(config-if)# ipv6 eigrp 100
R2(config-if)# exit
R2(config)# interface serial 0/0/0
R2(config-if)# ipv6 eigrp 100
R2(config-if)# exit
R2(config)# interface serial 0/0/1
R2(config-if)# ipv6 eigrp 100
R2(config-if)# end
R2#
```

```
R3(config)# ipv6 unicast-routing
```

```
R3(config)# ipv6 router eigrp 100
R3(config-rtr)# router-id 10.3.3.3
R3(config-rtr)# no shutdown
R3(config-rtr)# exit
R3(config)#
R3(config)# interface fastethernet 0/0
R3(config-if)# ipv6 eigrp 100
R3(config-if)# exit
R3(config)# interface serial 0/0/0
R3(config-if)# ipv6 eigrp 100
R3(config-if)# exit
R3(config)# interface serial 0/0/1
R3(config-if)# ipv6 eigrp 100
R3(config-if)# end
R3#
```

如前所述，IPv4 EIGRP 与 IPv6 EIGRP 默认都使用带宽和延时作为其复合度量，因而修改接口的带宽和时延参数会给 IPv4 EIGRP 与 IPv6 EIGRP 路由进程带来影响。

例 8-17 显示了之前在 R3 上配置的一条以 ISP 路由器为下一跳的默认路由。在例 8-23 中，该路由通过接口命令 **ipv6 summary-address eigrp** 作为汇总路由被分发给了路由器 R1 和 R2。该命令的语法格式如下：

```
Router(config-if)# ipv6 summary-address eigrp as-number ipv6-address
[admin-distance]
```

该命令的作用是为指定接口配置汇总聚合地址。默认情况下，IPv6 EIGRP 汇总路由的管理距离为 5。

**例 8-23 将默认路由作为汇总路由进行传播**

```
R3(config)# interface serial 0/0/0
R3(config-if)# ipv6 summary-address eigrp 100  ::/0
R3(config-if)# exit
R3(config)# interface serial 0/0/1
R3(config-if)# ipv6 summary-address eigrp 100  ::/0
R3(config-if)# end
R3#
```

由于R3拥有两个EIGRP邻居，因而需要在R3的两个接口进行配置，并为IPv6 EIGRP汇总路由应用默认的管理距离 5。命令 **ipv6 summary-address eigrp** 用于配置接口级地址汇总，管理距离 5 宣告的汇总地址无需安装到汇总了该路由的路由器的路由表中。

注：汇总路由以 null0 为下一跳被安装到路由表和 EIGRP 拓扑表中，这样做的目的是正确丢弃那些无明细路由的数据包，而不至于在网络中一直路由直至超出跳数限制。

下面就来验证将默认路由传播到 R1 和 R2 的情况。

### 8.2.3　验证 IPv6 EIGRP

例 8-24 通过命令 **show ipv6 protocols** 显示了当前所有处于激活状态的 IPv6 路由进程（包括 IPv6 EIGRP）的参数及当前状态。以高亮方式显示的输出结果是在例 8-23 中以汇总路由方式分发给 R1 和 R2 的默认路由（::/0）。汇总路由（是默认路由）的管理距离为 5，因而不在 R3 的路由表中，而之前采用默认管理距离 1 配置的静态默认路由则在 R3 的路由表中（如例 8-25 所示）。请注意，IPv6 EIGRP 路由的管理距离为 90，并且所有路由都源自该接口的链路本地地址。

**例 8-24　在 R3 上使用命令 show ipv6 protocols**

```
R3# show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "static"
IPv6 Routing Protocol is "eigrp 100"
  EIGRP metric weight K1=1, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Interfaces:
    FastEthernet0/0
    Serial0/0/0
    Serial0/0/1
  Redistribution:
    None
! Summary-address command configured on
! Serial 0/0/0 and Serial 0/0/1 in Example 8-23:
! ipv6 summary-address eigrp 100 ::/0
  Address Summarization:
    ::/0 for Serial0/0/0, Serial0/0/1
      Summarizing with metric 28160
  Maximum path: 16
  Distance: internal 90 external 170

R3#
R3# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::3
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:CAFE:3::1, subnet is 2001:DB8:CAFE:3::/64
```

```
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::A
    FF02::1:FF00:3
  MTU is 1500 bytes
<output omitted>
```

如例 8-24 所示，命令 **show ipv6 interface fastethernet 0/0** 验证了 R3 的 Fast Ethernet 0/0 接口目前是 IPv6 EIGRP 多播组 FF02::A 的成员。IPv6 EIGRP 将 FF02::A 用作 EIGRP 更新消息的目的地址。

**例 8-25　R3 的路由表**

```
R3# show ipv6 route
IPv6 Routing Table - 14 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
S   ::/0 [1/0]
     via ::, Serial0/1/0
D   2001:DB8:CAFE:1::/64 [90/20514560]
     via FE80::1, Serial0/0/0
D   2001:DB8:CAFE:2::/64 [90/20514560]
     via FE80::1, Serial0/0/0
     via FE80::2, Serial0/0/1
D   2001:DB8:CAFE:A001::/64 [90/21024000]
     via FE80::1, Serial0/0/0
     via FE80::2, Serial0/0/1
<output omitted for brevity?
R3#
```

例 8-26 中的命令 **show ipv6 route eigrp** 显示了 R1 路由表中的 IPv6 EIGRP 路由，包括由 R3 分发的默认路由。内部 EIGRP 路由的管理距离均为 90（与 IPv4 EIGRP 相同）。

例 8-27 通过命令 **show ipv6 eigrp neighbors** 验证了 R1 的邻居邻接关系。该命令与 IPv4 对应命令的唯一区别就是该命令使用 IPv6 链路本地地址来标识 EIGRP 对等体。IPv6 EIGRP 消息均以路由器的链路本地地址为源地址，这就是建议大家采取手工方式配置路由器的链路本地地址的原因，因为这样能够更容易地识别出源或目的路由器。从例 8-27 的输出结果中的 Address 例可以很容易地看出，FE80::3 是 R3 的链路本地地址，

而 FE80::2 是 R2 的链路本地地址。

**例 8-26　R1 的 IPv6 EIGRP 路由表表项**

```
R1# show ipv6 route eigrp
IPv6 Routing Table - 11 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
! Default route
D   ::/0 [90/2172416]
     via FE80::3, Serial0/0/1
D   2001:DB8:CAFE:2::/64 [90/2172416]
     via FE80::2, Serial0/0/0
D   2001:DB8:CAFE:A002::/64 [90/2681856]
     via FE80::2, Serial0/0/0
R1#
```

**例 8-27　验证 IPv6 EIGRP 中的邻居邻接关系**

```
R1# show ipv6 eigrp neighbors
IPv6-EIGRP neighbors for process 100
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
1   Link-local address:     Se0/0/1           14 00:02:34   49   294  0  8
    FE80::3
0   Link-local address:     Se0/0/0           13 00:13:08   28   200  0  9
    FE80::2
R1#
```

例 8-28 使用命令 **show ipv6 eigrp topology** 显示了 IPv6 EIGRP 的拓扑表，R3 的全部路由都处于被动状态（passive state），并且某些路由还包含了后继路由和可行后继路由。除了将链路本地地址作为下一跳地址之外，其他信息都与 IPv4 EIGRP 相同。拓扑表中的第一条表项是在例 8-23 中通过接口命令 **ipv6 summary-address eigrp** 创建的汇总路由。

注：如果希望了解更多有关 DUAL 或者命令 **show ipv6 topology** 输出结果的信息，可以参考 Cisco Press 出版的 *Routing Protocols and Concepts* 或 *Implementing Cisco IP Routing (ROUTE)*。

例 8-28 验证 IPv6 EIGRP 拓扑表

```
R3# show ipv6 eigrp topology
IPv6-EIGRP Topology Table for AS(100)/ID(10.3.3.3)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P ::/0, 1 successors, FD is 28160
        via Summary (28160/0), Null0
P 2001:DB8:CAFE:A003::/64, 1 successors, FD is 20512000
        via Connected, Serial0/0/0
        via FE80::2 (21536000/2681856), Serial0/0/1
        via FE80::1 (21024000/2169856), Serial0/0/0
P 2001:DB8:CAFE:3::/64, 1 successors, FD is 28160
        via Connected, FastEthernet0/0
P 2001:DB8:CAFE:A002::/64, 1 successors, FD is 20512000
        via Connected, Serial0/0/1
        via FE80::2 (21024000/2169856), Serial0/0/1
        via FE80::1 (21536000/2681856), Serial0/0/0
P 2001:DB8:CAFE:2::/64, 1 successors, FD is 20514560
        via FE80::2 (20514560/28160), Serial0/0/1
        via FE80::1 (21026560/2172416), Serial0/0/0
P 2001:DB8:CAFE:A001::/64, 2 successors, FD is 21024000
        via FE80::2 (21024000/2169856), Serial0/0/1
        via FE80::1 (21024000/2169856), Serial0/0/0
P 2001:DB8:CAFE:1::/64, 1 successors, FD is 20514560
        via FE80::1 (20514560/28160), Serial0/0/0
        via FE80::2 (21026560/2172416), Serial0/0/1
R3#
```

例 8-29 使用 **ping** 命令来验证网络的可达性。可以看出，对 R2、R3 和 ISP 路由器的快速以太网接口发起的 ping 操作均成功。

例 8-30 中的命令 **show ipv6 eigrp traffic** 显示了发送和接收到的 EIGRP 包的数量。IPv6 EIGRP 中的 EIGRP 包种类与 IPv4 EIGRP 相同：

- Hello；
- 更新（Update）；
- 查询（Query）；
- 应答（Reply）；
- 确认（Acknowledgment）。

### 例 8-29　使用 ping 命令来验证可达性

```
R1# ping 2001:db8:cafe:1::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/0 ms
R1# ping 2001:db8:cafe:2::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:2::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
R1# ping 2001:db8:face:c0de::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:FACE:C0DE::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/60 ms
R1#
```

### 例 8-30　验证 IPv6 EIGRP 消息

```
R3# show ipv6 eigrp traffic
IPv6-EIGRP Traffic Statistics for AS 100
  Hellos sent/received: 2689/1789
  Updates sent/received: 50/37
  Queries sent/received: 1/3
  Replies sent/received: 3/1
  Acks sent/received: 23/35
  Input queue high water mark 2, 0 drops
  SIA-Queries sent/received: 0/0
  SIA-Replies sent/received: 0/0
  Hello Process ID: 190
  PDM Process ID: 179

R3#
```

命令 **show ipv6 eigrp interfaces** 可以列出已经配置了 IPv6 EIGRP 的接口（如例 8-31 所示）。

### 例 8-31 验证 IPv6 EIGRP 接口

```
R3# show ipv6 eigrp interfaces
IPv6-EIGRP interfaces for process 100

                    Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface        Peers  Un/Reliable  SRTT  Un/Reliable  Flow Timer  Routes
Fa0/0              0        0/0       0       0/10          0         0
Se0/0/0            1        0/0       25      7/190         290       0
Se0/0/1            1        0/0       44      7/190         382       0
R3#
```

例 8-32 显示了从这 4 台路由器的最终运行配置中抽取的与 IPv6 EIGRP 相关的内容。

### 例 8-32 运行配置中与 IPv6 EIGRP 相关的内容

```
R1# show running-config
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:1::1/64
 ipv6 eigrp 100
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A001::1/64
 ipv6 eigrp 100
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A003::1/64
 ipv6 eigrp 100
!
ipv6 router eigrp 100
 router-id 10.1.1.1
 no shutdown
!
R1#
```

```
R2# show running-config
```

```
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:2::1/64
 ipv6 eigrp 100
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:A001::2/64
 ipv6 eigrp 100
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:A002::1/64
 ipv6 eigrp 100
!
ipv6 router eigrp 100
 router-id 10.2.2.2
 no shutdown
!
R2#
```

```
R3# show running-config
!
hostname R3
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:3::1/64
 ipv6 eigrp 100
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:A003::2/64
 ipv6 eigrp 100
 ipv6 summary-address eigrp 100 ::/0 5
```

```
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:A002::2/64
 ipv6 eigrp 100
 ipv6 summary-address eigrp 100 ::/0 5
!
interface Serial0/1/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:FEED:1::1/64
!
ipv6 route ::/0 Serial0/1/0
!
ipv6 router eigrp 100
 router-id 10.3.3.3
 no shutdown
!
R3#
```

```
ISP# show running-config
!
hostname ISP
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::FEED link-local
 ipv6 address 2001:DB8:FACE:C0DE::1/64
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::FEED link-local
 ipv6 address 2001:DB8:FEED:1::2/64
!
ipv6 route 2001:DB8:CAFE::/48 Serial0/0/0
!
!
ISP#
```

在讨论OSPFv3之前，例8-33给出了删除IPv6 EIGRP进程的命令。全局配置命令**no ipv6 router eigrp** *as-name* 可以删除IPv6 EIGRP进程，包括该进程使用的所有接口配置命令。

例 8-33　删除 IPv6 EIGRP 进程

```
R1(config)# no ipv6 router eigrp 100
R2(config)# no ipv6 router eigrp 100
R3(config)# no ipv6 router eigrp 100
```

## 8.3　OSPFv3

OSPF（Open Shortest Path First，开放最短路径优先）是一种链路状态路由协议，主要用来替代距离矢量路由协议 RIP。虽然 RIP 能够满足早期网络互连的路由选择需要，但是 RIP 将跳数作为选择最优路由的唯一度量已越来越无法满足大型网络对健壮路由解决方案的需要。OSPF 是一种无类别路由协议，使用区域的概念来实现其优良的扩展性。

1989 年，RFC 1131 “The OSPF Specification” 定义了 OSPFv1 规范。1991 年，John Moy 又在 RFC 1247 “OSPF Version 2” 中定义了 OSPFv2，OSPFv2 对 OSPFv1 做了很多重大改进。1998 年又在 RFC 2328 “OSPF Version 2” 中对 OSPFv2 规范做了更新，目前该 RFC 是 OSPFv2 的最新规范。RFC 2328 将 OSPF 度量定义为可以取任意值的开销（cost），Cisco IOS 将带宽作为 OSPF 的开销度量。

1999 年，在由 John Moy、Rob Coltun 和 Dennis Ferguson 合作编写的 RFC 2740“OSPF for IPv6”中发布了用于 IPv6 的 OSPFv3，后来又在 RFC 5340“OSPF for IPv6”中对 OSPFv3 做了更新。OSPFv3 不但是一种可部署于 IPv6 网络的新路由协议，而且还对该协议做了重大重写。本书假定大家已经掌握了 OSPFv2，而且相关内容也不在本书写作范围之内。

### 8.3.1　OSPFv2 与 OSPFv3 对比

与 RIPng 和 IPv6 EIGRP 类似，OSPFv3 的进程与操作也都与 OSPFv2 相似。下面列出了 OSPFv2 与 OSPFv3 在主要特性上的对比情况。

- **宣告的路由**：OSPFv2 宣告 IPv4 路由，而 OSPFv3 宣告 IPv6 路由。

> 注：Cisco IOS 也维护了 OSPFv3 地址族，同时支持 IPv4 和 IPv6 单播流量。RFC 5838 “Support of Address Families in OSPFv3” 定义了在 OSPFv3 中同时支持 IPv4 和 IPv6 的相关规范，但有关 OSPFv3 地址族的功能特性已经超出了本书范围，大家可以参考 *Cisco IOS IPv6 Configuration Guide*，*Implementing OSPF for IPv6*：www.cisco.com/en/US/docs/ios/ipv6/configuration/ guide/ip6- ospf_ps6922_TSD_ Products _Configuration_Guide_Chapter.html。

- **链路状态**：OSPFv2 与 OSPFv3 都是链路状态路由协议。
- **度量**：OSPFv2 与 OSPFv3 的 RFC 中都将度量定义为将数据包从接口发送出去

的开销。在 Cisco 的实现中，度量是出站接口到目的网络的累积带宽（$10^8$/以 bit/s 为单位的带宽）。OSPFv2 与 OSPFv3 中的参考带宽 $10^8$ 都可以在路由器配置模式下，通过命令 **auto-cost reference-bandwidth** *ref-bw* 来加以修改，而且参数 ref-bw（参考带宽）都是以 Mbit/s 为单位。由于是在路由器配置模式下使用命令，因而仅影响所配置的 OSPF 度量。例如，如果是为 OSPFv3 输入的该命令，那么就不会影响 OSPFv2 的路由度量。

- **OSPF 包类型**：OSPFv3 使用了与 OSPFv2 相同的 5 种基本的数据包：
  - Hello；
  - DBD（Database Description，数据库描述）；
  - LSR（Link-state Request，链路状态请求）；
  - LSU（Link-state Update，链路状态更新）；
  - LSAck（Link-state Acknowledgment，链路状态确认）。

注：如果熟悉 OSPFv2 中的链路状态更新和各种 LSA（Link-state Advertisement，链路状态宣告），那么大家就可能会对 OSPFv3 新增的两类 LSA 感兴趣：链路 LSA（Link-LSA，8 类 LSA）和区域内前缀 LSA（Intra-Area-Prefix-LSA，9 类 LSA）。而且某些已有的 LSA 也被重新命名，但有关 LSA 类型的相关内容已经超出了本书写作范围，请参考 RFC 5340 “OSPF for IPv6”。

- **LSA 泛洪机制**：链路状态数据库的老化机制以及通过提前老化进程（premature aging process）从路由域中清除 LSA 等机制都保持不变。运行 OSPFv2 或 OSPFv3 的 Cisco 路由器每隔 30 秒都要就周期性地泛洪其链路状态，并在 60 秒之后清除 LSA。
- **路由器 ID**：OSPFv2 与 OSPFv3 都采取 32 比特数值以点分十进制形式来表示路由器 ID，通常是 IPv4 地址。如果没有为路由器配置 IPv4 地址，那么就需要使用命令 **router-id** 来配置路由器 ID，而且这两种协议确定 32 比特路由器 ID 的进程都相同。
- **DR/BDR 选举进程**：DR/BDR（Designated Router/Backup Designated Router，指派路由器/备份指派路由器）选举进程在 OSPFv3 中保持不变。
- **邻居发现机制**：邻居状态机（包括邻居状态和事件列表）都保持不变。OSPFv2 与 OSPFv3 都使用 Hello 机制来了解邻居路由器并构造邻接关系。但是，与 OSPFv2 相比，OSPFv3 在构造邻居邻接关系的时候没有“相同子网”的需求，这是因为 OSPFv3 利用链路本地地址（而不是全局单播地址）来构造邻居邻接关系。
- **源地址和目的地址**：OSPFv3 用到的两个标准多播地址如下所示。
  - FF02::5：表示链路本地范围内的全部 SPF 路由器，等同于 OSPFv2 中的 224.0.0.5。
  - FF02::6：表示链路本地范围内的全部 DR，等同于 OSPFv2 中的 224.0.0.6。

  OSPFv2 消息以出接口的 IPv4 地址为源地址，而 OSPFv3 消息以出接口的链路

本地地址为源地址。

- **认证**：OSPFv2 可以使用明文认证或 MD5（Message Digest 5，消息摘要 5）认证机制，OSPFv3 使用 IPSec 提供的认证和/或加密功能。
- **NBMA 拓扑**：NBMA（NonBroadcast MultiAccess，非广播多路接入）和点到多点拓扑（如帧中继）上的 OSPFv3 操作与 OSPFv2 相同。OSPFv3 同时支持 RFC 遵从模式和 Cisco 模式。
- **区域（area）**：OSPFv3 中的多区域概念与 OSPFv2 相同，目的是为 OSPF 域提供最少的链路状态泛洪和更好的稳定性。此外，还通过末梢区域（stub area）、完全末梢区域（totally stubby area）和 NSSA（Not-So-Stubby Area，非完全末梢区域）来最小化区域内路由器的链路状态数据库和路由表大小。

注：有关多区域 OSPF 的内容已经超出了本书范围，如果希望了解多区域 OSPF 的更多信息，可以参考 Cisco Press 出版的 *Implementing Cisco IP Routing (ROUTE)*和 *Cisco Self Study:Implementing Cisco IPv6 Networks*。

### 8.3.2　在 Cisco 路由器上配置 OSPFv3

与 RIPng 和 IPv6 EIGRP 类似，下面也将使用与第 6 章示例拓扑结构相似的拓扑结构来解释 OSPFv3 的配置。示例拓扑结构如图 8-3 所示。

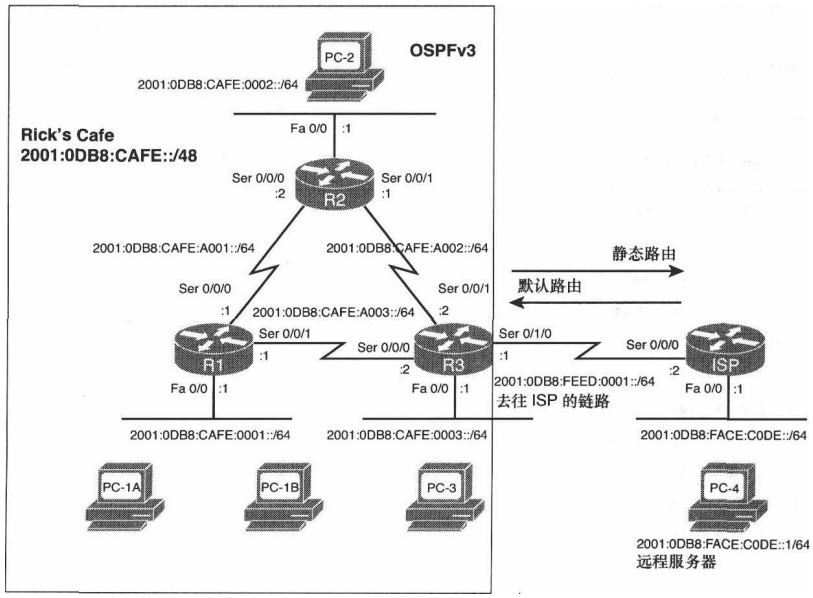



图 8-3　Rick's Café 网络的 OSPFv3 拓扑结构

同样，需要配置路由器 R1、R2 和 R3 以共享路由信息，此时在 IGP 路由域中使用的是 OSPFv3。仍然为 R3 配置一条经由 ISP 路由器的默认路由，并配置 R3，通过 OSPFv3

将该默认路由分发给其他 OSPFv3 路由器。同时仍然为 ISP 路由器配置一条去往 2001:0DB8:CAFE::/48 的静态路由。这两条静态路由的情况如例 8-34 所示。

**例 8-34 R3 和 ISP 路由器上的静态路由**

```
R3(config)# ipv6 route ::/0 serial 0/1/0

ISP(config)# ipv6 route 2001:db8:cafe::/48 serial 0/0/0
```

命令 **ipv6 unicast-routing** 的作用是在 R1 上启用 IPv6 路由功能（如例 8-35 所示）。接下来在路由器 R1 上使用全局配置命令 **ipv6 router ospf 1** 配置 OSPFv3 进程。该命令与 OSPFv2 中的 **router ospf** *process-id* 相似。与 OSPFv2 一样，OSPFv3 中的 process-id（进程号）也只具有本地意义，无需与 OSPF 域中的其他路由器相同。在路由器配置模式下，命令 **router-id** 的作用是配置 OSPF 路由器 ID。由于这些路由器的接口上都没有配置 IPv4 地址，因而必须在拓扑结构中的所有路由器上都配置该 OSPF 命令 **router-id**。因此，路由器无法动态选择路由器 ID，必须采取手工方式配置路由器 ID。

这些命令的语法格式如表 8-2 所示。

**表 8-2 创建 OSPFv3 路由进程的命令**

| 命令 | 描述 |
|---|---|
| Router(config)# **ipv6 router ospf** *process-id* | 启用 IPv6 的 OSPF 路由进程，process-id 用于标识特定的路由进程，OSPF 路由域中所有路由器的 process-id 都不必相同 |
| Router(config-rtr)# **router-id** *router-id* | 路由器为 OSPFv3 使用 32 比特以点分十进制形式表示的 IPv4 地址来选择路由器 ID |

**例 8-35 在 R1 上启用 OSPFv3**

```
R1(config)# ipv6 unicast-routing

R1(config)# ipv6 router ospf 1
R1(config-rtr)# router-id 10.1.1.1
R1(config-rtr)# exit

R1(config)# interface fastethernet 0/0
R1(config-if)# ipv6 ospf 1 area 0
R1(config-if)# exit
R1(config)# interface serial 0/0/0
R1(config-if)# ipv6 ospf 1 area 0
R1(config-if)# exit
R1(config)# interface serial 0/0/1
R1(config-if)# ipv6 ospf 1 area 0
R1(config-if)# end
R1#
```

继续看例 8-35 中的配置。通过命令 **ipv6 ospf** *process-id* **area** *area-id* 为 R1 的接口启用 OSPFv3，其中的 process-id 必须与命令 **ipv6 router ospf** *process-id* 中使用的 process-id 相同。在指定接口上启用 OSPFv3 路由进程的命令语法如下：

Router(config-if)# **ipv6 ospf** *process-id* **area** *area-id*

其中的 *process-id* 用于标识特定的路由进程，必须与用于创建路由进程的命令 **ipv6 router ospf** *process-id* 中的 process-id 相同。参数 *area-id*（区域号）是与该 OSPFv3 接口相关联的区域，虽然可以为该区域配置任意值，这里选择区域号 0 的原因是 area 0 是骨干区域。所有其他区域都要连接到骨干区域之上，这样就能方便未来根据需要迁移到多个区域（多区域 OSPF）。

请注意例 8-36 中路由器 R2 和 R3 的 OSPFv3 配置情况。虽然可以使用不同的 process-id，不过为了保持一致性，这里使用的 process-id 仍然为 1。

**例 8-36　R2 和 R3 的 OSPFv3 配置**

```
R2(config)# ipv6 unicast-routing
R2(config)# ipv6 router ospf 1
R2(config-rtr)# router-id 10.2.2.2
R2(config-rtr)# exit
R2(config)# interface fastethernet 0/0
R2(config-if)# ipv6 ospf 1 area 0
R2(config-if)# exit
R2(config)# interface serial 0/0/0
R2(config-if)# ipv6 ospf 1 area 0
R2(config-if)# exit
R2(config)# interface serial 0/0/1
R2(config-if)# ipv6 ospf 1 area 0
R2(config-if)# end
R2#
```

```
R3(config)# ipv6 unicast-routing
R3(config)# ipv6 router ospf 1
R3(config-rtr)# router-id 10.3.3.3
R3(config-rtr)# exit
R3(config)# interface fastethernet 0/0
R3(config-if)# ipv6 ospf 1 area 0
R3(config-if)# exit
R3(config)# interface serial 0/0/0
R3(config-if)# ipv6 ospf 1 area 0
R3(config-if)# exit
R3(config)# interface serial 0/0/1
R3(config-if)# ipv6 ospf 1 area 0
R3(config-if)# end
R3#
```

为了完成 OSPFv3 的配置，需要将 R3 上的默认路由分发给 OSPF 路由域中的其他路由器，因此在 R3 上配置了路由器配置命令 **default-information originate**（如例 8-37 所示）。

命令 **default-information originate** 的完整语法格式如下：

```
Router(config-rtr)# default-information originate [always | metric metric-value |
metric-type type-value | route-map map-name]
```

该命令会生成一条默认外部路由并进入 OSPFv3 路由域，其参数如下。

- **always**（可选）：无论软件是否有默认路由，都要宣告该默认路由。
- **metric** *metric-value*（可选）：度量用于生成默认路由，如果省略了该度量值或者没有利用设置默认度量的路由器配置命令来指定度量值，那么默认度量值就为 10。默认度量值范围是 0～16 777 214。
- **metric-type** *type-value*（可选）：将与默认路由相关联的外部链路类型宣告到 IPv6 OSPF 路由域中，类型值为：
  - 1：1 类外部路由；
  - 2：2 类外部路由。

**例 8-37　在 R3 上分发默认路由**

```
R3(config)# ipv6 router ospf 1
R3(config-rtr)# ?
  area                   OSPF area parameters
  auto-cost              Calculate OSPF interface cost according to bandwidth
  default                Set a command to its defaults
  default-information    Distribution of default information
  default-metric         Set metric of redistributed routes
  discard-route          Enable or disable discard-route installation
  distance               Administrative distance
  distribute-list        Filter networks in routing updates
  exit                   Exit from IPv6 routing protocol configuration mode
  ignore                 Do not complain about specific event
  interface-id           Source of the interface ID
  log-adjacency-changes  Log changes in adjacency state
  maximum-paths          Forward packets over multiple paths
  no                     Negate a command or set its defaults
  passive-interface      Suppress routing updates on an interface
  process-min-time       Percentage of quantum to be used before releasing CPU
  redistribute           Redistribute IPv6 prefixes from another routing
                         protocol
  router-id              router-id for this OSPF process
  shutdown               Shutdown protocol
  summary-prefix         Configure IPv6 summary prefix
  timers                 Adjust routing timers

R3(config-rtr)# default-information originate
R3(config-rtr)# end
R3#
```

- 默认值为 2 类外部路由：2 类路由的开销总是到达该路由的外部开销，而不是内部开销（OSPF 域内开销）。1 类开销是到达该路由的外部开销加内部开销。
- **route-map** *map-name*（可选）：在满足路由映射的情况下，路由进程会生成该默认路由。

### 8.3.3　验证 OSPFv3

为了验证 OSPFv3 域是否已完全收敛，首先检查 R1 的路由表。通过命令 **show ipv6 route** 可以查看 OSPF 域中的远程网络以及 R3 所分发的默认路由（如例 8-38 所示）。OSPFv3 的管理距离为 110，与 OSPFv2 相同。

在 R1 上使用 **ping** 命令可以看出，所有远程网络均可达（如例 8-39 所示）。

**例 8-38　R1 的路由表**

```
R1# show ipv6 route ospf
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
OE2   ::/0 [110/1], tag 1
      via FE80::3, Serial0/0/1
O     2001:DB8:CAFE:2::/64 [110/65]
      via FE80::2, Serial0/0/0
O     2001:DB8:CAFE:3::/64 [110/65]
      via FE80::3, Serial0/0/1
O     2001:DB8:CAFE:A002::/64 [110/128]
      via FE80::2, Serial0/0/0
R1#
```

**例 8-39　利用 ping 命令验证可达性**

```
R1# ping 2001:db8:cafe:1::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:1::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/4 ms
R1# ping 2001:db8:cafe:2::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:2::1, timeout is 2 seconds:
```

```
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/28 ms
R1# ping 2001:db8:face:c0de::1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:FACE:C0DE::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/56/60 ms
R1#
```

例 8-40 给出了命令 **show ipv6 protocols** 的输出结果。与本章中其他 IPv6 路由协议相似，该命令会显示路由器上运行的直连路由、静态路由以及 IPv6 OSPF 路由进程。

**例 8-40　在 R1 上运行命令 show ipv6 protocols**

```
R1# show ipv6 protocols
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "static"
IPv6 Routing Protocol is "ospf 1"
  Interfaces (Area 0):
    Serial0/0/1
    Serial0/0/0
    FastEthernet0/0
  Redistribution:
    None
R1#
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::1
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:CAFE:1::1, subnet is 2001:DB8:CAFE:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::5
    FF02::6
    FF02::1:FF00:1
  MTU is 1500 bytes
<output omitted>
```

如例 8-40 所示，命令 **show ipv6 interface fastethernet 0/0** 验证了 R1 的 Fast Ethernet 0/0 接口当前是 OSPFv3 多播组 FF02::5（全部 SPF 路由器多播地址）的成员。R1 将会处理任何目的

地址为 FF02::5 的 OSPFv3 消息（如 OSPFv3 Hello 消息）。同时，R1 的 Fast Ethernet 0/0 接口还是该网段的指派路由器，需要侦听目的地址为 FF02::6（全部 DR 路由器）的 OSPFv3 消息。

例 8-41 通过命令 **show ipv6 ospf neighbor** 列出了 OSPFv3 的邻居邻接关系。注意到两个邻居的路由器 ID 都是它们用于 OSPFv3 的 32 比特路由器 ID。

R1 上运行命令 **show ipv6 ospf interface** 的输出结果与 OSPFv2 相似（如例 8-42 所示），唯一的区别在于 OSPFv3 消息的源地址为接口的链路本地地址（对本案例来说，就是 R1 的 FE80::1），而 OSPFv2 消息的源地址时接口的 IPv4 地址。

**例 8-41　在 R1 上运行命令 show ipv6 ospf neighbor 的输出结果**

```
R1# show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
10.3.3.3          1   FULL/  -        00:00:32    6               Serial0/0/1
10.2.2.2          1   FULL/  -        00:00:31    6               Serial0/0/0
R1#
```

**例 8-42　在 R1 上运行命令 show ipv6 ospf interface 的输出结果**

```
R1# show ipv6 ospf interface serial 0/0/0
Serial0/0/0 is up, line protocol is up
  Link Local Address FE80::1, Interface ID 6
  Area 0, Process ID 1, Instance ID 0, Router ID 10.1.1.1
  Network Type POINT_TO_POINT, Cost: 64
  Transmit Delay is 1 sec, State POINT_TO_POINT,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:01
  Index 1/2/2, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 3
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 10.2.2.2
  Suppress hello for 0 neighbor(s)
R1#
```

例 8-43 显示了从这 4 台路由器的最终运行配置中抽取的与 OSPFv3 相关的内容。

### 例 8-43 运行配置中与 OSPFv3 相关的内容

```
R1# show running-config
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:1::1/64
 ipv6 ospf 1 area 0
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A001::1/64
 ipv6 ospf 1 area 0
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:A003::1/64
 ipv6 ospf 1 area 0
!
ipv6 router ospf 1
 router-id 10.1.1.1
 log-adjacency-changes
!
R1#
```

```
R2# show running-config
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:2::1/64
 ipv6 ospf 1 area 0
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:A001::2/64
 ipv6 ospf 1 area 0
```

```
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::2 link-local
 ipv6 address 2001:DB8:CAFE:A002::1/64
 ipv6 ospf 1 area 0
!
ipv6 router ospf 1
 router-id 10.2.2.2
 log-adjacency-changes
!
R2#
```

---

```
R3# show running-config
!
hostname R3
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:3::1/64
 ipv6 ospf 1 area 0
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:A003::2/64
 ipv6 ospf 1 area 0
!
interface Serial0/0/1
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:CAFE:A002::2/64
 ipv6 ospf 1 area 0
!
interface Serial0/1/0
 no ip address
 ipv6 address FE80::3 link-local
 ipv6 address 2001:DB8:FEED:1::1/64
!
ipv6 route ::/0 Serial0/1/0
!
ipv6 router ospf 1
 router-id 10.3.3.3
```

```
 log-adjacency-changes
 default-information originate
!
R3#
```

```
ISP# show running-config
!
hostname ISP
!
ipv6 unicast-routing
!
interface FastEthernet0/0
 no ip address
 ipv6 address FE80::FEED link-local
 ipv6 address 2001:DB8:FACE:C0DE::1/64
!
interface Serial0/0/0
 no ip address
 ipv6 address FE80::FEED link-local
 ipv6 address 2001:DB8:FEED:1::2/64
!
ipv6 route 2001:DB8:CAFE::/48 Serial0/0/0
!
!
ISP#
```

如果要删除 OSPFv3，可以使用全局配置命令 **no ipv6 router ospf** *process-id*。该命令将删除 OSPFv3 进程，包括所有 OSPFv3 接口配置命令。

## 8.4 本章小结

IPv6 路由协议与对应的 IPv4 路由需协议之间并无太多差异，本章讨论了以下三种路由协议：

- IPv6 RIPng；
- IPv6 EIGRP；
- OSPFv3。

对每种路由协议来说，都讨论了以下内容：

- IPv6 路由协议与其对应的 IPv4 路由协议之间的异同点；
- 配置 IPv6 路由协议以及分发默认路由的基本命令；
- 用于部署、确认网络收敛以及测试可达性的相关命令。

在配置 IPv6 路由协议之前，必须首先在路由器上启用 IPv6 路由功能，相应的命令为：

```
Router(config)# ipv6 unicast-routing
```

### 8.4.1　RIPng

RIPng 是 IPv4 RIPv2 的 IPv6 对等协议。两者功能非常相似，最大的区别在于 Cisco 的 RIPng 实现中，路由器会将自己加入到路由的度量或跳数中。在路由器上输入命令 **show ipv6 router** 会发现路由器将自己作为第一跳，因而会将正常情况下 RIPv2 路由表中见到的跳数加 1。

#### 1. 配置命令

以下是启用和配置 RIPng 的命令。

- Router(config)# **ipv6 router rip** *name*：该命令可以为 IPv6 启用 RIPng 路由进程。其中，*name* 用于标识特定的路由进程；
- Router(config-if)# **ipv6 rip** *name* **enable** 以启用 RIPng。其中，*name* 用于标识特定的路由进程。
- Router(config-if)# **ipv6 rip** *name* **default-information**：将默认路由分发到 RIPng 路由域中。

#### 2. 验证命令

以下是验证 RIPng 的命令。

- Router# **show ipv6 protocols**：显示所有处于激活状态的 IPv6 路由进程的参数和当前状态。
- Router# **show ipv6 route rip**：显示 IPv6 路由表中的 RIPng 路由。
- Router# **show ipv6 rip**：显示当前 RIPng 进程的信息。
- Router# **show ipv6 rip database**：显示 RIPng RIB。
- Router# **show ipv6 rip next-hops**：显示下一跳地址以及使用该下一跳地址的 RIPng 路由数，而不论这些路由是否位于 IPv6 路由表中。
- Router# **debug ipv6 rip**：显示 RIPng 路由器发送和接收的 RIPng 路由消息。

### 8.4.2　IPv6 EIGRP

IPv6 EIGRP 是 IPv4 EIGRP 的对等协议，用于路由 IPv6 前缀。两者都将 DUAL 用作计算引擎，以保证无环路路径。

#### 1. 配置命令

以下是启用和配置 IPv6 EIGRP 的命令。

- Router(config)# **ipv6 router eigrp** *as-number*：启用 IPv6 EIGRP 路由进程。as-number 用于标识特定的路由进程。EIGRP 路由域中所有路由器的 as-number 都必须相同。
- Router(config-rtr)# **router-id** *router-id*：路由器为 IPv6 EIGRP 路由进程使用 32 比特 IPv4 地址来选择路由器 ID。
- Router(config-if)# **ipv6 eigrp** *as-number*：在接口上启用 IPv6 EIGRP 路由进程。

as-number（自治系统号）的作用是标识特定的路由进程，必须与用来创建路由进程的命令 **ipv6 router eigrp** *as-number* 中的 as-number 相同。

- Router(config-if)# **ipv6 summary-address eigrp** *as-number ipv6-address* [ *admin-distance* ]：为指定接口配置汇总聚合地址。默认情况下，IPv6 EIGRP 汇总路由的管理距离为 5。可用来将默认路由分发给 EIGRP 邻居。

**2. 验证命令**

以下是验证 IPv6 EIGRP 的命令。

- Router# **show ipv6 protocols**：显示所有处于激活状态的 IPv6 路由进程的参数和当前状态。
- Router# **show ipv6 route eigrp**：显示 IPv6 路由表中的 EIGRP 路由。
- Router# **show ipv6 eigrp neighbors**：显示 IPv6 EIGRP 邻居的邻接关系。
- Router# **show ipv6 eigrp topology**：显示 IPv6 EIGRP 的拓扑表。
- Router# **show ipv6 eigrp traffic**：显示发送和收到的 IPv6 EIGRP 包数量。
- Router# **show ipv6 eigrp interfaces**：显示启用了 IPv6 EIGRP 的接口。

### 8.4.3 OSPFv3

OSPFv3 是 IPv4 OSPFv2 的对等协议，虽然两者的进程、数据包类型等方面很相似，但是 OSPFv3 对协议内核做了重大重写。

**1. 配置命令**

以下是启用和配置 OSPFv3 的命令。

- Router(config)# **ipv6 router ospf** *process-id*：启用 IPv6 的 OSPF 路由进程。process-id 用于标识特定的路由进程。OSPF 路由域中所有路由器的 process-id 都不必相同。
- Router(config-rtr)# **router-id** *router-id*：路由器为 OSPFv3 使用 32 比特以点分十进制形式表示的 IPv4 地址来选择路由器 ID。
- Router(config-if)# **ipv6 ospf** *process-id* **area** *area-id*：在接口上启用 OSPFv3 路由进程，其中的参数 *process-id* 用于标识特定的路由进程，必须与用于创建路由进程的命令 **ipv6 router ospf** *process-id* 中的 process-id 相同。参数 *area-id* 是与该 OSPFv3 接口相关联的区域。
- Router(config-rtr)# **default-information originate**：生成一条默认外部路由并进入 OSPFv3 路由域。

**2. 验证命令**

以下是验证 OSPFv3 的命令。

- Router# **show ipv6 protocols**：显示所有处于激活状态的 IPv6 路由进程的参数

和当前状态。

- Router# **show ipv6 route ospf**：显示 IPv6 路由表中的 OSPFv3 路由。
- Router# **show ipv6 eigrp neighbors**：显示 OSPFv3 邻居的邻接关系。
- Router# **show ipv6 ospf interfaces**：显示启用了 OSPFv3 的接口上的 OSPF 信息。

## 8.5 参考文献

RIPng：

RFC 1058, Routing Information Protocol , C. Hedrick, Rutgers University, IETF, www.ietf.org/rfc/rfc1058 , June 1998

RFC 2453, RIP Version 2 , G. Malkin, Bay Networks, IETF, www.ietf.org/rfc/rfc2453 , November 1998

RFC 2080, RIPng for IPv6 , G. Malkin, Xylogics, www.ietf.org/rfc/rfc2080 , January 1997

Cisco IOS IPv6 Configuration Guide, Implementing RIP for IPv6, www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-rip_ps6922_TSD_Products_Configuration_Guide_Chapter.html

IPv6 EIGRP：

Cisco IOS IPv6 Configuration Guide, Implementing EIGRP for IPv6, www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-eigrp.html

OSPFv3：

RFC 2328, OSPF Version 2 , J. Moy, Ascend Communications, www.ietf.org/rfc/rfc2328 , April 1998

RFC 5340, OSPF for IPv6 , R. Coltun, Acoustra Productions, www.ietf.org/rfc/rfc5340 , July 2008

RFC 5838, Support of Address Families in OSPFv3 , A. Lindem, Ericsson, http://tools.ietf.org/html/rfc5838 , April 2010

"OSPFv3 Support for Address Families," www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6554/ps6599/ps6629/whitepaper_c11-668030.html

Cisco IOS IPv6 Configuration Guide, Implementing OSPF for IPv6, www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-ospf_ps6922_TSD_Products_Configuration_Guide_Chapter.html

# 第 9 章　DHCPv6

IPv6 对大部分上层协议的影响都非常小，需要做出变动的主要原因是该协议或应用程序需要在其提供的服务中使用扩展后的 128 比特 IPv6 地址。其中变动较明显的一个上层协议就是用于 IPv6 的 DHCP，即 DHCPv6（Dynamic Host Configuration Protocol for IPv6，用于 IPv6 的动态主机配置协议）。除了在 DHCP 消息中使用了更大的 IPv6 地址之外，协议方面也做了一些改动。

IPv6 的动态地址分配比 IPv4 更复杂，而且提供了更多的配置选项。与 IPv4 相似，DHCPv6 提供了状态化 DHCP 服务。但是正如第 4 章中所述，IPv6 还可以提供无状态地址自动配置（即 SLAAC）。DHCP 服务器可以提供也可以不提供 SLAAC 服务。无状态地址自动配置可能仅包含路由器及其路由器宣告（RA）消息或 RA 消息以及从 DHCP 服务器获得的其他配置参数。

理解 DHCPv6 进程以及可用的动态地址分配选项是非常重要的。

本章将讨论以下内容：

- DHCPv6 服务；
- DHCPv6 通信；
- 其他上层协议：DNS、TCP 和 UDP。

## 9.1　DHCPv6 服务

前面已经对 IPv6 地址分配的无状态本质做了很多讨论。设备可以在没有 DHCPv6 服务器的情况下，获取其 IPv6 地址和其他配置信息。利用 ICMPv6 路由器宣告消息以及设备创建其接口 ID 的能力，SLAAC 可以提供无 DHCP 服务的动态编址能力。对某些应用场合来说，这种无状态地址自动配置是 IPv6 网络必须提供的动态地址分配能力。

但是对另外一些应用场景来说，状态化 DHCP 服务却能更好地满足其网络运行需要以及地址分配的管理需要。

例如，某个企业可能需要某种特定的 IP 编址方案，不希望主机使用 MAC 地址来创建其接口 ID，或者不希望 SLAAC 在提供基本的地址/网关信息之外还提供其他额外的配置信息。有些人认为状态化 DHCPv6 服务没有必要，但也有很多人不同意这个观点。过去的许多年里，双方就此展开了大量争论。

目前 DHCPv6 与 SLAAC 都是 IPv6 中的现实存在。DHCPv6 定义在 RFC 3315 "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)" 中。过去的很多年里，大家就该规范做了很多研究工作。2001 年，Cisco 公司的 Steve Deering 在 IETF 51"Proceedings of the 51st Internet Engineering Task Force" 中申明，DHCPv6 规范是所有 Internet 草案中修订版本号最高的规范。

注：SLAAC 定义在 RFC 4862 "IPv6 Stateless Address Autoconfiguration" 中，详细内容请见第 5 章。

DHCPv6 包括下面两种形式。

- **状态化 DHCPv6 服务**：RFC 3315"Dynamic Host Configuration Protocol for IPv6 (DHCPv6)"。
- **无状态 DHCPv6 服务**：RFC 3736"Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6"。

注：随着服务提供商逐步部署 IPv6，某些 ISP 已开始向公众提供 IPv6 接入。对于广泛部署的 "实时在线" 介质（如 DSL 和 Cable 互联网接入）来说，需要一种有效的向客户站点授权地址前缀的机制。RFC 3769 "Requirements for IPv6 Prefix Delegation" 讨论了该授权机制，目的是将向客户站点的连网设备通知其被分配使用的地址前缀的过程自动化。有关 DHCPv6 前缀授权的详细信息请参考 RFC 3769。

状态化 DHCPv6 可以提供与 IPv4 DHCP（DHCPv4）相同的自动配置服务，主机可以直接从 DHCPv6 服务器获取全部的编址信息以及其他配置信息。而无状态 DHCPv6 的差异之处在于主机从路由器宣告消息中获取其编址信息，并从 DHCPv6 服务器获取其他配置信息。

虽然 DHCPv6 与 DHCPv4 的功能很相似，但是 DHCPv6 也几乎彻底改写了 DHCPv4 协议，而且这两种协议也是相互独立的。如果在使用 DHCP 的网络上启用了双栈功能，那么就需要启用两个独立的 DHCP 服务，分别用于 IPv4 和 IPv6。

虽然 DHCPv6 中的进程和消息与 DHCPv4 相比没有太大的区别，但是对 DHCPv4 来说，客户端被配置为使用 DHCP 时，只是简单地在 Windows 中选择 "自动获取 IP 地址" 选项或者在 Mac 系统中选择 "使用 DHCP"。而对于 IPv6 来说，虽然客户端的配置一样，但是客户端并不知道配置信息将来自于路由器宣告消息（SLAAC），还是来自

于 DHCPv6 服务器（状态化 DHCPv6），或者是来自于 SLAAC 和 DHCPv6 两者。

> 注：在网络中为主机部署动态 IPv6 编址技术之前，最好先确认主机操作系统的特性，目前已经发现并不是所有的操作系统都能识别路由器宣告消息中的 M 和 O 比特（标记），因而可能会得到非期望的结果。

IPv6 客户端如何接收其配置信息取决于路由器所发送的路由器宣告消息，而不取决于客户端本身。有关路由器宣告消息的配置选项将在本章稍后进行讨论。

本章将首先讨论 DHCPv6 的基础知识，介绍一些新却很相似的术语、消息类型以及运行方式，然后讨论路由器宣告消息是如何通知客户端究竟是使用 SLAAC、状态化 DHCPv6 还是无状态 DHCPv6 的。理解了 DHCPv6 协议及其运行方式之后，将讨论如何配置路由器的路由器宣告消息，以告知链路上的主机如何动态获取其编址信息以及其他配置参数，包括无状态 DHCPv6。

> 注：本章重点讨论 DHCPv6 的基本概念以及路由器作为 DHCPv6 服务的角色功能，而绝不是 DHCPv6 的权威应用指南。有关 Cisco 路由器的 DHCPv6 实现细节，请参考：
>
> - RFC 3315 " Dynamic Host Configuration Protocol for IPv6 (DHCPv6)";
> - Cisco IOS IPv6 Configuration Guide, Implementing DHCP for IPv6 : www.cisco.com/en/US/docs/ios/ipv6/configura tion /guide/ip6- dhcp.html。

### 9.1.1　DHCPv6 术语、多播地址和消息类型

DHCPv6 来源于 DHCPv4，两者拥有很多相似点。从大面上来说，DHCPv6 提供了与 DHCPv4 相似的服务，但也有一些差别。以下三个 DHCPv6 术语与 DHCPv4 中对应的术语完全一样。

- **DHCP 客户端：**DHCP 客户端向 DHCP 服务器发送请求以获取其配置参数。
- **DHCP 服务器：**DHCP 服务器被配置为响应来自 DHCP 客户端的 DHCP 请求，这些 DHCP 客户端可能与服务器位于同一条链路，也可能位于不同链路。
- **DHCP 中继代理：**DHCP 客户端经常与 DHCP 服务器位于不同网络或不同链路上，中继代理是一种中间设备（通常是路由器），负责接收客户端的请求并转发给一个或多个位于其他网络上的 DHCP 服务器。DHCP 中继代理的操作对客户端来说是完全透明的。

而接下来的三个术语（DUID、IA 和 IAID）是 DHCPv6 中的专用术语，不过目前正在制定相应的机制将这些术语转化到 DHCPv4 中。虽然有关这些术语的讨论并不很重要，但是理解这些术语对于配置 DHCPv6 提供的某些选项来说还是很有用的。

- **DUID（DHCP Unique Identifier，DHCP 唯一标识符）**：每个 DHCP 参与方（包括服务器和客户端）都有一个唯一标识该设备的 DUID。DUID 能够标识每个客户端和服务器。DUID 包括一个 2 字节类型代码，其后是字节数可变的用于组成实际标识符的字段。不包含类型代码的话，DUID 最长可以达到 128 字节，RFC 3315 定义了三种 DUID：
- 链路层地址加时间；
- 厂商分配的唯一 ID（基于 IANA 维护的私有企业号[Private Enterprise Number]）；
- 链路层地址。

**注**：DUID 的选择与设备无关。DUID 按照“不透明值（opaque values）”进行处理，除了唯一地表示 DHCPv6 客户端或服务器之外没有任何意义。这三种 DUID 之间的区别以及 DUID 的选择进程不在本书写作范围之内，大家可参考 RFC 3315 以获取更多信息。

- **IA（Identity Association，身份关联）**：IA 是分配给客户端的地址集，每个客户端至少为每个使用 DHCPv6 服务的接口分配一个 IA。每个 IA 都有一个由客户端分配的 IAID（Identity Association Identifier，身份管理标识符），用于唯一地标识 IA。
- **IAID（Identity Association Identifier，身份管理标识符）**：每个 IA 都有一个由客户端分配的 IAID，而且该客户端的所有 IAID 都必须是唯一的。

DHCPv6 客户端和服务器使用以下多播地址。

- **全部 DHCP 中继代理和服务器（All_DHCP_Relay_Agents_and_Servers）多播地址（FF02::1:2）**：所有的 DHCPv6 服务器和中继代理都是该链路本地范围多播组的成员，客户端利用该地址与链路上的 DHCPv6 服务器和中继代理进行通信。
- **全部 DHCP 服务器（All_DHCP_Servers）多播地址（FF05::1:3）**：所有的 DHCPv6 服务器都是该站点本地范围多播组的成员，中继代理利用该多播地址向站点内的所有 DHCPv6 服务器发送消息，或者在不知道 DHCPv6 服务器单播地址的情况下使用该多播地址。

DHCPv6 使用以下 UDP 端口。

- **UDP 端口 546**：DHCPv6 服务器和中继代理利用 UDP 目的端口 546 向客户端发送消息，客户端在 UDP 端口 546 上侦听 DHCPv6 消息。
- **UDP 端口 547**：DHCPv6 客户端通过 UDP 目的端口 547 向服务器和中继代理发送消息，服务器和中继代理在 UDP 端口 547 上侦听 DHCPv6 消息。

DHCPv6 为客户端-服务器之间的通信定义了不同类型的消息，有关 DHCPv6 消息的完整列表定义在 RFC 3315 并列在表 9-1 中。下一节将讨论 DHCPv6 通信时解释究竟使用了多少种消息。如果希望获得更详细的信息，请参考本章前面列出的相关资源。

DHCPv6 客户端与服务器之间通信用到的 4 种主要 DHCPv6 消息如下所示。

- **SOLICIT (1)**：DHCPv6 客户端利用 SOLICIT（请求）消息定位服务器。

- **ADVERTISE (2):** DHCPv6 服务器发送 ADVERTISE（宣告）消息作为客户端 SOLICIT 消息的响应消息，表明其可以提供 DHCPv6 服务。
- **REQUEST (3):** DHCPv6 客户端发送 REQUEST（要求）消息以便从特定的 DHCPv6 服务器请求配置参数（包括 IP 地址）。
- **REPLY (7):** 作为客户端 REQUEST 消息的响应消息，DHCPv6 服务器发送的 REPLY（应答）消息包含了已分配地址和配置参数。如果使用了快速分配选项（Rapid Commit Option），那么 REPLY 消息也可以作为 SOLICIT 消息的响应消息。有关快速分配选项的详细信息将在本节后面讨论。此外，REPLY 消息还可以作为 INFORMATION-REQUEST（信息要求）、CONFIRM（确认）、RELEASE（释放）或 DECLINE（拒绝）消息的响应消息。

**表 9-1　　RFC 3315 描述的 DHCPv6 消息类型列表**

| DHCPv6 消息类型 | 描述 |
|---|---|
| SOLICIT (1) | 客户端发送 SOLICIT（请求）消息以定位服务器 |
| ADVERTISE (2) | 服务器发送 ADVERTISE（宣告）消息以表明其 DHCP 服务可用，以响应从客户端接收到的 SOLICIT（请求）消息 |
| REQUEST (3) | 客户端发送 REQUEST（要求）消息以便从特定服务器请求包括 IP 地址在内的配置参数 |
| CONFIRM (4) | 客户端向所有可用的 DHCP 服务器发送 CONFIRM（确认）消息，以确定所分配的地址是否适合该客户端所连接的链路 |
| RENEW (5) | 客户端向最初为客户端提供客户端地址和配置参数的服务器发送 RENEW（更新）消息，以扩展分配给该客户端的地址的生存时间，并更新其他配置参数 |
| REBIND (6) | 客户端向所有可用服务器发送 REBIND（重新绑定）消息，以扩展分配给该客户端的地址的生存时间，并更新其他配置参数；该消息是在客户端没有收到 RENEW 消息的响应消息之后发送的 |
| REPLY (7) | 服务器从客户端收到 SOLICIT、REQUEST、RENEW 或 REBIND 消息后，会发送包含已分配地址和相关配置参数的 REPLY（应答）消息作为响应消息；服务器发送包含配置信息的 REPLY 消息作为 INFORMATION-REQUEST 消息的响应消息；服务器发送 REPLY 消息来响应 CONFIRM 消息，以确认或否认已分配给客户端的地址对于客户端所连接的链路是合适的；此外，服务器还会通过发送 REPLY 消息来确认已收到 RELEASE 或 DECLINE 消息 |
| RELEASE (8) | 客户端向为其分配地址的服务器发送 RELEASE（释放）消息，表明客户端不再使用一个或多个已分配地址 |
| DECLINE (9) | 客户端向服务器发送 DECLINE（拒绝）消息，表明客户端已确定服务器所分配的一个或多个地址已在客户端所连接链路上被占用 |
| RECONFIGURE (10) | 服务器向客户端发送 RECONFIGURE（重新配置）消息，以告诉客户端该服务器有新的或更新后的配置参数，客户端与服务器发起 RENEW/REPLY 或 INFORMATION-REQUEST/REPLY 事物以接收更新后的信息 |
| INFORMATION-REQUEST (11) | 客户端向服务器发送 INFORMATION-REQUEST（信息要求）消息，以请求不带有客户端 IP 地址分配的配置参数 |
| RELAY-FORW (12) | 中继代理直接或通过其他中继代理向服务器发送 RELAY-FORW（中继转发）消息来转发消息，收到的消息（客户端消息或来自其他中继代理的 RELAY-FORW 消息）都被封装到 RELAY-FORW 消息的一个选项中 |

续表

| DHCPv6 消息类型 | 描述 |
| --- | --- |
| RELAY-REPL (13) | 服务器向中继代理发送 RELAY-REPL（中继应答）消息，该消息中包含了中继代理发送给客户端的消息，RELAY-REPL 消息可以通过其他中继代理传送给目标中继代理，服务器将客户端消息作为一个选项封装在 RELAY-REPL 消息中，由中继代理提取并发送给客户端 |

### 9.1.2 DHCPv6 通信

图 9-1 解释了 DHCPv6 通信过程（包括客户端、服务器和路由器）的各个步骤。请注意，DHCPv6 与 DHCPv4 之间的区别在于路由器决定如何动态分配分配地址。被配置为自动获取其编址及其他配置信息的客户端可以采用以下 3 种方式：

- SLAAC（Stateless Address Autoconfiguration，无状态地址自动分配）；
- 状态化 DHCPv6；
- 无状态 DHCPv6。

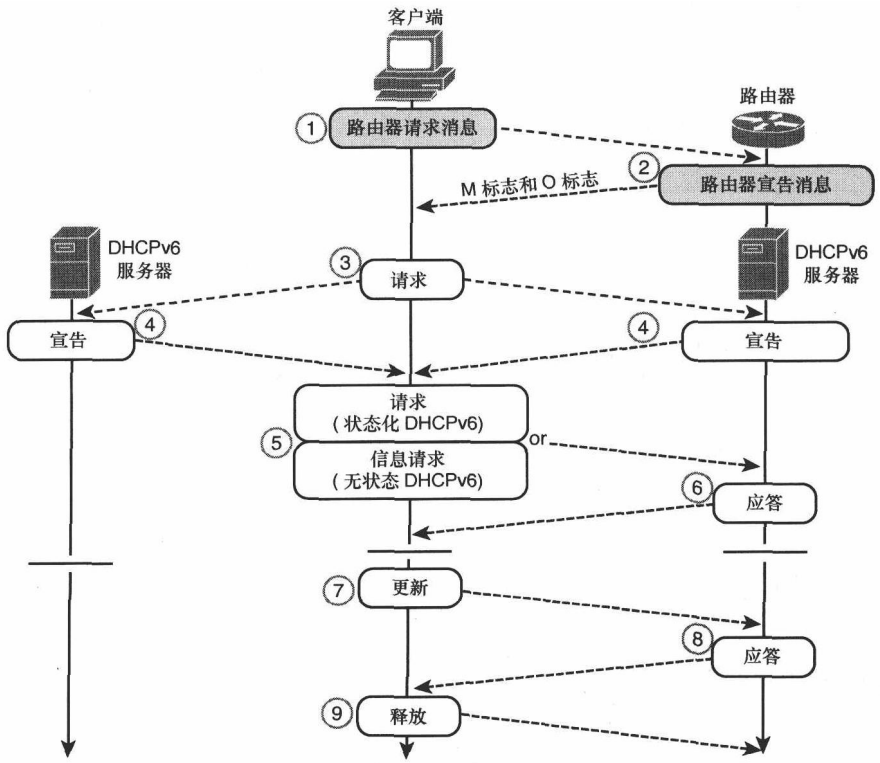



图 9-1 DHCPv6

主机无需决定应该使用哪种方法来动态获取其编址信息。如果主机依赖于路由器宣告（RA）消息，那么路由器就会通过其 RA 消息中包含的信息（特别是 M 标记和 O 标记）来控制该决策。取决于不同的操作系统，主机可以忽略路由器的 RA 消息，并使用状态化 DHCPv6 来获取其编址信息。如前所述，最好在网络部署之前先验证主机操作系统的处理行为。

**注：**除了 M 标记和 O 标记之外，路由器宣告消息还包含一个 A（Autonomous Address Configuration，自主地址分配）标记。A 标记表示是否可以将 RA 中的前缀用于 SLAAC。默认情况下，A 标记为 1，表示 RA 中的前缀将用于 SLAAC。A 标记为地址配置提供了额外选项，但相关内容已超出了本书写作范围，大家可参考 http://blogs.cisco. com/borderless/ipv6-automatic-addressing 和 RFC 4861 “Neighbor Discovery for IP version 6 (IPv6)”。

**注：**有关 ICMPv6 路由器请求（RS）消息和路由器宣告（RA）消息的细节请参见第 5 章内容。

以图 9-1 为例，DHCPv6 通信过程如下。

**路由器请求消息和路由器宣告消息**

**第 1 步：**如果客户端未收到路由器宣告消息，那么就会向全部路由器多播组 FF 02::2 发送路由器请求消息。当主机被配置为动态获取其前置、前缀长度、默认网关及其他配置信息时，就会发出路由器请求消息。

**第 2 步：**路由器周期性地发送路由器宣告消息或者在收到主机的路由器请求消息后作为响应而发出路由器宣告消息。RA 中包含了两种用于决定主机将通过何种方式收到其编址信息及其他配置信息的标记：M 标记和 O 标记。Cisco IOS 提供了以下三种选项：

- **SLAAC：**Cisco 路由器在默认情况下将 M 标记和 O 标记均设置为 0，表示不能从 DHCPv6 服务器获取配置信息。需要动态地址分配的主机会从 RA 消息中获得全部信息。
- **状态化 DHCPv6：**M 标记为 1 时，告知主机将使用状态化 DHCPv6 服务。必须从 DHCPv6 服务器直接获取其全部编址和配置信息。需要重点注意的是，即使没有路由器，用于地址配置的 DHCPv6 服务也依然可能可用。
- **无状态 DHCPv6：**M 标记为 0 且 O 标记为 1 时，告知主机应该从 RA 消息获取前缀、前缀长度、默认网关及其他信息（如 MTU）。但是还可以从 DHCPv6 服务器获取一些额外配置信息，如 DNS 服务器的地址。

**注：**RFC 6101 “IPv6 Router Advertisement Options for DNS Configuration” 标准化了 RA 消息中的一个选项，为主机提供一个递归式 DNS 服务器列表及其相关联的生存时间。同样，在部署该选项之前最好先验证主机操作系统的处理行为。不是所有的操作系统都支持 RFC 6101。如果 M 标记和 O 标记均为 0，那么就使用 SLAAC，DHCPv6 进程将终止。

如果 M 标记和 O 标记均为 1，那么 DHCPv6 进程将继续，表示正在使用无状态或状态化 DHCPv6。第 5 步解释了客户端如何向服务器发送相应的请求（取决于正在使用的是状态化 DHCPv6 还是无状态 DHCPv6）。

**DHCPv6 消息**

路由器 RA 中的 M 标记和/或 O 标记被设置为 1。

**第 3 步：** 主机（目前是 DHCPv6 客户端）向目的地址为全部 DHCP 中继代理和服务器（All_DHCP_Relay_Agents_and_Servers）多播地址的 FF02::1:2 发送 SOLICIT 消息以定位 DHCPv6 服务器。某些情况下，可以配置客户端使用单播地址来到达指定的服务器。

**第 4 步：** 一台或多台服务器用 ADVERTISE 消息进行响应，以告知客户端 DHCPv6 服务可用。如果客户端收到了多条 ADVERTISE 消息，那么就需要使用 RFC 3315 定义的决策进程来选择合适的服务器。需要在客户端和服务器上均配置这些选项。

注：有关客户端从多台服务器收到多条 ADVERTISE 消息后，DHCPv6 服务器选择标准的问题不在本书写作范围，大家可以参考 RFC 3315 section 17.1.3 “Receipt of Advertise Messages”。

**第 5 步：** 客户端向服务器发送 REQUEST 或 INFORMATION-REQUEST 消息（取决于是否正在使用状态化或无状态 DHCPv6）。客户端发送的消息类型取决于路由器 RA 消息中的 M 标记，存在以下两种可能性：

- 如果客户端正在使用状态化 DHCPv6（路由器 RA 消息中的 M 标记为 1），那么客户端就将发送 REQUEST 消息，以获取 IPv6 地址以及其他配置参数。
- 如果客户端正在使用无状态 DHCPv6（路由器 RA 消息中的 M 标记为 0 且 O 标记为 1），那么客户端就将发送 INFORMATION-REQUEST 消息，仅向服务器请求不包含编址信息的配置参数。

**第 6 步：** 服务器收到客户端发送的 REQUEST 消息后，会发送包含已分配地址及其他配置参数的 REPLY 消息。如果是响应 INFORMATION-REQUEST 消息，那么服务器就会发送仅包含配置参数的 REPLY 消息。

**更新和释放**

**第 7 步：** DHCPv6 租约到期时，客户端会向最初为该客户端提供地址及配置信息的服务器发送 RENEW 消息，以扩展租约的生存时间。

**第 8 步：** 服务器向客户端发送 REPLY 消息，以确认正在更新地址和配置参数的租约。

**第 9 步：** 客户端不再需要一个或多个已分配地址时，就会发送 RELEASE 消息以告知 DHCPv6 服务器。

注：服务器还会通过发送 REPLY 消息来确认已收到 RELEASE 或 DECLINE 消息。

注：如果客户端确定某个已分配地址已被占用（通常利用 DAD 机制），那么就会向服务器发送 DECLINE 消息。此时主机会暂停使用该地址，服务器将该地址标记为被未知主机占用，并为客户端分配其他地址。

**配置无状态 DHCPv6**

无状态 DHCPv6 的配置非常简单直观。除了路由器宣告消息的配置之外，DHCPv6 服务的基本配置与 IPv4 中的 DHCP 服务配置完全相似。配置无状态 DHCPv6 服务的三个基本步骤如下。

**第 1 步**：配置 DHCPv6 服务器池的名字及配置参数。

**第 2 步**：在接口上启用 DHCPv6 服务器池。

**第 3 步**：修改 RA 消息的参数，让主机知道可以从 DHCPv6 服务器获取其他配置信息。

表 9-2 列出了创建 DHCPv6 池的命令，并描述了一些可用的配置参数。

注：DHCPv6 的配置涉及很多内容，这里仅列出了一小部分，有关 DHCPv6 配置选项和配置命令的全部内容请参考 Cisco IOS IPv6 Configuration Guide, Implementing DHCP for IPv6: www.cisco.com/en/ US/docs/ios/ipv6/configuration/guide/ip6-dhcp.html。

表 9-2　　DHCPv6 池配置命令

| 命令 | 描述 |
| --- | --- |
| Router(config)# **ipv6 dhcp pool** *poolname* | 创建 DHCPv6 池并进入 DHCPv6 池配置模式 |
| Router(config-dhcp)# **dns-server** *ipv6-address* | 指定可用于 DHCPv6 客户端的 IPv6 DNS 服务器 |
| Router(config-dhcp)# **domain** *domain* | 为 DHCPv6 客户端配置域名 |
| Router(config-dhcp)# **prefix-delegation** *ipv6-prefix /prefix-length client-duid* [ **iaid** *iaid* ] [ **lifetime** ] | 指定一个手工配置的数字前缀（被授权给指定客户端的 IAID） |

如果要在接口上启用 DHCPv6 服务，那么就要在接口配置模式下使用命令 **ipv6 dhcp server**。表 9-3 列出了将 DHCPv6 池与接口进行关联的配置命令。其中，命令 **ipv6 dhcp server** 中的快速分配选项将在下一节进行讨论。

表 9-3　　DHCPv6 池配置命令

| 命令 | 描述 |
| --- | --- |
| Router(config)# **interface** *type number* | 指定接口类型和接口号并进入路由器的接口配置模式。 |
| Router(config-dhcp)# **ipv6 dhcp server** *poolname* [ **rapid-commit** ] | 在接口上启用 DHCPv6 服务，*pool-name*（可选）是用户为本地前缀池定义的名字，池名可以是有意义的字符串（如 Engineering），也可以是整数（如 0）。<br>**rapid-commit**（可选）允许为前缀授权采取双消息交换方法 |

表 9-4 列出了用于配置路由器的 RA 消息，以便让主机知道可以从 DHCPv6 服务器获得其他配置信息的相关命令。

**表 9-4　　将 RA 消息中的 O 标记设置为 1**

| 命令 | 描述 |
|---|---|
| Router(config)# **interface** *type number* | 指定接口类型和接口号并进入路由器的接口配置模式 |
| Router(config-dhcp)# **ipv6 nd other-config-flag** | 将路由器宣告消息中的 O 标记设置为 1，表示可以从 DHCPv6 服务器获得其他配置信息，该命令的选项 **no** 可以将 O 标记设置为默认值 0 |

利用 Rick's Café 网络的拓扑结构，将 R1 的 LAN 配置为使用无状态 DHCPv6。LAN 上的主机将使用 SLAAC 得到其地址信息（包括前缀、前缀长度和默认网关）。然而，并不是主机所需的全部信息都来源于路由器的 RA 消息，主机只是从 DHCPv6 服务器获取其他配置数据，如 DNS 服务器地址。为了实现该要求，需要：

- 将 R1 配置为其 LAN 上的无状态 DHCPv6 服务器并提供 DNS 服务器地址；
- 修改路由器 RA 消息中的 O 标记，以便让主机知道可以从 DHCPv6 服务器获取其他配置信息。

图 9-2 显示了 R1 与 PC-1C 之间的无状态 DHCPv6 处理过程。第 1 步和第 2 步显示 PC-1C 从 RA 消息中获得其前缀、前缀长度和默认网关等信息。RA 消息中的 O 标记被设置为 1，表明 PC-1C 必须从 DHCPv6 服务器获取其他额外的配置信息。第 3～8 步则显示了 PC-1C 与无状态 DHCPv6 服务器（路由器 R1）之间的 DHCPv6 处理过程。

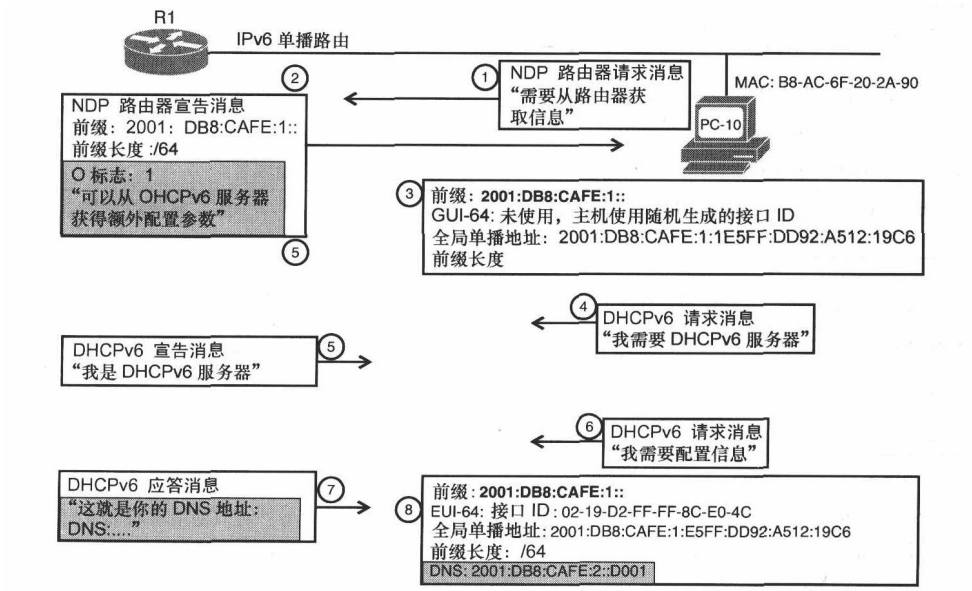



图 9-2　无状态 DHCPv6

将路由器 R1 配置为无状态 DHCPv6 服务器的相关命令如例 9-1 所示。

**例 9-1　在 R1 上配置无状态 DHCPv6**

```
! The first two commands are the DHCP configuration pool commands
R1(config)# ipv6 dhcp pool cafe-1-pool
R1(config-dhcp)# dns-server 2001:db8:cafe:2::d001
R1(config-dhcp)# exit

R1(config)# interface fa 0/0
! The next command enables DHCPv6 service on the interface
R1(config-if)# ipv6 dhcp server cafe-1-pool
! The next command sets the Router Advertisement O flag (Other Configuration flag)
! to 1
R1(config-if)# ipv6 nd other-config-flag
R1(config-if)# end
R1#
```

命令 **ipv6 dhcp pool cafe-1-pool** 的作用是创建 DHCPv6 池 cafe-1-pool，并让路由器进入 DHCPv6 池配置模式。命令 **dns-server 2001:db8:cafe:2::d001** 的作用是为客户端指定 DNS 服务器的地址。

例 9-1 解释了如何在面向客户端的接口（Fast Ethernet 0/0）上启用 DHCPv6 服务，并利用命令 **ipv6 dhcp server cafe-1-pool** 将其与 cafe-1-pool 关联起来。

最后的配置工作就是修改 RA 消息。默认情况下 O 标记为 0。将 O 标记设置为 1 将意味着告知主机可以从 DHCPv6 服务器获得额外的配置信息。利用命令 **ipv6 nd other-config-flag** 即可将 O 标记设置为 1。

利用命令 **show ipv6 interface fastethernet 0/0** 可以验证已经修改了 R1 的 O 标记（如例 9-2 所示）。请注意，该接口目前已经是全部 DHCP 中继代理和服务器多播组（All_DHCP_Relay_Agents_and_Servers）FF02::1:2 的成员了。从输出结果的最后两行可以看出，主机虽然还在使用 SLAAC 来获取编址信息（M 标记为 0），但主机已经可以从 DHCPv6 服务器获取其他配置参数了（O 标记为 1）。

**例 9-2　验证 RA 消息中的标记**

```
R1# show ipv6 interface fastethernet 0/0
FastEthernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::1
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:CAFE:1::1, subnet is 2001:DB8:CAFE:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
```

```
    FF02::5
    FF02::6
    FF02::1:2     ! All_DHCP_Relay_Agents_and_Servers multicast group
    FF02::1:FF00:1
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ICMP unreachables are sent
  ND DAD is enabled, number of DAD attempts: 1
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  ND advertised default router preference is Medium
  Hosts use stateless autoconfig for addresses.    ! RA's M Flag set to 0
  Hosts use DHCP to obtain other configuration.    ! RA's O Flag set to 1

R1#
```

在 PC-1C 上使用 **ipconfig /all** 命令可以看出，客户端已经收到了其全部编址和配置信息（如例 9-3 所示）。其中，前缀、前缀长度和默认网关都是从 R1 的路由器宣告消息中收到的（由于 PC-1C 是 Windows 主机，因而其接口 ID 是随机生成的），而 IPv6 DNS 地址是从 DHCPv6 服务器（对本案例来说也是 R1）获取的。

**例 9-3　PC-1C**

```
PC-1C> ipconfig /all
Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Intel(R) 82567LM-3 Gigabit Network Connection
   Physical Address. . . . . . . . . : B8-AC-6F-20-2A-90
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv6 Address. . . . . . . . . . . : 2001:db8:cafe:1:e5ff:dd92:a512:19c6(Preferred)
   Link-local IPv6 Address . . . . . : fe80::e5ff:dd92:a512:19c6
   Default Gateway . . . . . . . . . : fe80::1
   DHCPv6 IAID . . . . . . . . . . . : 250629538
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-15-EF-49-66-B8-AC-6F-20-2A-90
   DNS Servers . . . . . . . . . . . : 2001:db8:cafe:2::d001
```

利用命令 **show ipv6 dhcp** 和 **show ipv6 dhcp interface** 可以检查路由器上的

DHCPv6 服务(如例 9-4 所示)。命令 **show ipv6 dhcp** 可以显示路由器的 DUID。DHCPv6 服务器和客户端都有一个 DUID，DHCP 利用 DUID 来唯一地标识设备。命令 **show ipv6 dhcp interface** 可以显示接口上的 DHCP 信息，包括与之相关联的 DHCP 池以及是否正在使用快速分配选项。

**例 9-4 在 R1 上验证 DHCP 服务**

```
R1# show ipv6 dhcp
This device's DHCPv6 unique identifier(DUID): 00030001001B0CC282D8
R1#

R1# show ipv6 dhcp interface fastethernet 0/0
FastEthernet0/0 is in server mode
  Using pool: cafe-1-pool
  Preference value: 0
  Hint from client: ignored
  Rapid-Commit: disabled
R1#
```

虽然通常都是采取手工方式为路由器接口配置 IPv6 地址，但有时也可能会遇到路由器需要利用 SLAAC 自动获取其 IPv6 地址的情况。通过命令 **ipv6 address autoconfig interface** 即可执行 IPv6 SLAAC，利用 RA 消息发现链路上的前缀，然后再将基于 EUI-64 的地址分配给接口。命令 **ipv6 address autoconfig interface** 的语法格式如表 9-5 所示。

**表 9-5 为路由器接口配置 SLAAC**

| 命令 | 描述 |
|---|---|
| Router(config)# **interface** *type number* | 指定接口类型和接口号并进入路由器的接口配置模式 |
| Router(config-dhcp)# **ipv6 address autoconfig** | 在路由器接口上启用基于 SLAAC 机制的 IPv6 地址自动配置功能，并在接口上启用 IPv6 处理功能 |

### 9.1.3 快速分配选项

DHCPv6 客户端与服务器之间交换的 DHCPv6 消息可以从 4 条减少至 2 条，实现方式是在 DHCPv6 客户端发送给服务器的 SOLICIT 消息中包含快速分配选项（Rapid Commit Option），其作用是告知服务器，客户端希望将交换的消息数从 4 条减少至 2 条（如图 9-3 所示）。如果服务器支持快速分配选项，那么就会以包含了快速分配选项的 REPLY 消息进行响应，并分配包含在 REPLY 消息中的已分配地址。因而客户端与服务器之间只交换了 2 条消息（SOLICIT 和 REPLY 消息），而不是之前讨论的 4 条消息（SOLICIT、ADVERTISE、REQUEST 和 REPLY 消息）。

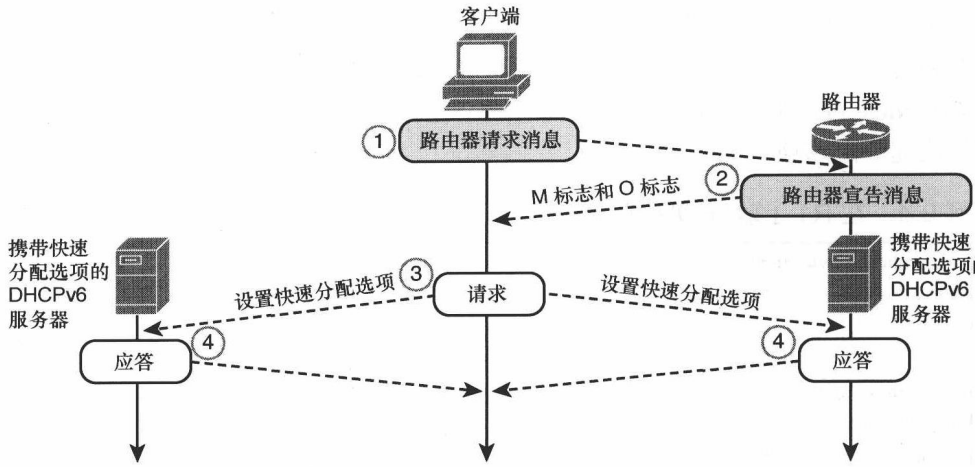



图 9-3 使用快速分配选项的 DHCPv6

使用快速分配选项时，服务器将不再期望从收到 REPLY 消息的客户端接收任何信息，因而如果有多台服务器都对包含了快速分配选项的 SOLICIT 消息做出响应，那么只有一台服务器分配的地址最终被客户端所使用。因此，只为一台服务器配置快速分配选项可以最小化该未使用地址的问题。如果客户端发送了带快速分配选项的 SOLICIT 消息，那么就会优选携带快速分配选项的服务器发送的 REPLY 消息。如果客户端未收到任何携带快速分配选项的 REPLY 消息，那么就会接受服务器的 ADVERTISE 消息并执行常规的四消息交换进程。

**配置快速分配选项**

在 **ipv6 server dhcp interface** 命令中包含 **rapid-commit** 参数，即可为路由器配置快速分配选项。例 9-5 修改了 R1 之前的 DHCPv6 配置，支持快速分配双消息交换机制。通过命令 **show ipv6 dhcp interface** 可以证实快速分配选项已被启用。

**例 9-5 配置并验证快速分配选项**

```
R1(config)# interface fastethernet 0/0
! Including the Rapid Commit Option:
R1(config-if)# ipv6 dhcp server cafe-1-pool rapid-commit
R1(config-if)# end

R1# show ipv6 dhcp interface fastethernet 0/0
FastEthernet0/0 is in server mode
  Using pool: cafe-1-pool
  Preference value: 0
  Hint from client: ignored
  Rapid-Commit: enabled
R1#
```

例 9-6 显示了 R1 运行配置中的 DHCPv6 和其他相关命令。

**例 9-6 R1 的运行配置**

```
R1# show running-config
!
ipv6 unicast-routing
!
ipv6 dhcp pool cafe-1-pool
 dns-server 2001:DB8:CAFE:2::D001
!
interface FastEthernet0/0
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:1::1/64
 ipv6 nd other-config-flag
 ipv6 dhcp server cafe-1-pool
 ipv6 dhcp server cafe-1-pool rapid-commit
 ipv6 ospf 1 area 0
!
<output omitted for brevity>
```

### 9.1.4 中继代理通信

有时 DHCP 服务器与请求编址和其他配置参数的客户端位于不同的网络之中，此时就要配置路由器或中继代理为不同网络中的客户端和服务器转发 DHCP 消息。如果大家熟悉 DHCPv4 和 Cisco 路由器，那么可能也就熟悉 DHCPv4 中使用的 **ip helper** 命令。不过，路由器或中继代理在转发 DHCPv6 消息时与 DHCPv4 还是有一些差别的。

图 9-4 解释了使用中继代理的 DHCPv6 进程。从客户端的角度来看，一切都没有变化。虽然图中只显示了一台路由器，但实际上，客户端与服务器间的路径上可以存在多个中继代理，相应的处理过程如下。

**第 1～3 步**：这些步骤与前面讨论的常规 DHCPv6 通信过程一样，客户端向全部 DHCP 中继代理和服务器（All_DHCP_Relay_Agents_and_Server）多播地址 FF02::1:2 发送 SOLICIT 消息。顾名思义，该消息会发送给所有的 DHCPv6 服务器和中继代理。

**第 4 步**：中继代理创建 RELAY-FORW 消息（包含从客户端收到的原始 SOLICIT 消息），并使用站点本地范围内的全部 DHCP 服务器（All_DHCP_Server）多播地址 FF05::1:3 将该消息转发给服务器。也可以配置中继代理使用单播地址将该消息发送给特定 DHCPv6 服务器。

**第 5 步**：服务器向中继代理返回一条 RELAY-REPL 消息，该消息中包含（封装）由目标中继代理发送给客户端的 ADVERTISE 消息（如果使用了快速分配选项，那么 RELAY-REPL 消息中将包含 REPLY 消息以及所需的编址和/或其他配置参数）。

**第 6 步**：面向客户端的中继代理（路径上可能存在多个中继代理）收到 RELAY-REPL 消息。解封装该消息后，中继代理就将 ADVERTISE 消息转发给客户端。

**第 7 步**：客户端向服务器发送 REQUEST 或 INFORMATION-REQUEST 消息（取决于是否正在使用状态化或无状态 DHCPv6），这与前面讨论的进程完全相同。

**第 8 步**：中继代理创建另一条 RELAY-FORW 消息，此时包含（封装）了来自客户端的 REQUEST 或 INFORMATION-REQUEST 消息，并将该消息转发给服务器。

**第 9 步**：服务器向中继代理响应以包含了 REPLY 消息的 RELAY-REPL 消息。

**第 10 步**：中继代理收到 RELAY-REPL 消息，面向客户端的中继代理解封装该消息并将其中的 REPLY 消息转发给客户端。

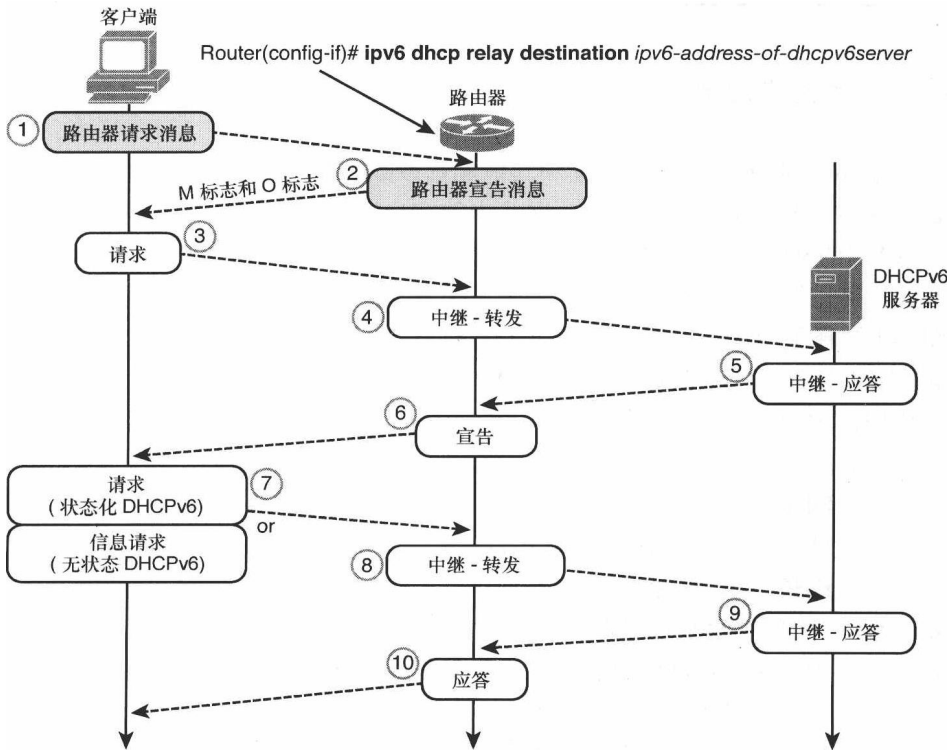



图 9-4 DHCPv6 中继代理通信

**配置中继代理**

为了将路由器配置为 DHCPv6 中继代理，需要在面向客户端的接口上配置命令 **ipv6 dhcp relay destination**。在接口上启用了中继代理服务之后，该接口上就会收到 DHCPv6 消息并转发给所有已配置的中继目的地。入站 DHCPv6 消息可以来自于该接口上的客户端，也可以由其他中继代理转发而来，表 9-6 显示了将路由器配置为 DHCPv6 中继代理的相关命令语法。

**表 9-6　　DHCPv6 中继代理命令**

| 命令 | 描述 |
| --- | --- |
| Router(config)# **interface** *type number* | 指定接口类型和接口号并进入路由器的接口配置模式 |
| Router(config-dhcp)# **ipv6 dhcp relay destination** *ipv6-address* [ *interface-type interface-number* ] | 指定客户端数据包将要被转发至的目的地址，并在接口上启用 DHCPv6 中继代理。<br>参数 *ipv6-address* 定义了中继目的地址，中继目的地址包括以下两种类型：<br>■ 链路范围内的单播和多播 IPv6 地址，用户必须为该类地址指定一个出接口；<br>■ 全局或站点范围内的单播或多播 IPv6 地址 |

如果命令 **ipv6 dhcp relay destination** 中未包含出接口，那么就由 IPv6 路由表来决定出接口。

注：DHCPv6 的很多内容都超出了本书写作范围，包括提供安全性考虑（DHCPv6 欺骗）、状态化 DHCPv6 服务以及配置其他客户端/服务器选项。与 IPv6 中的很多内容一样，DHCPv6 似乎也一直处于不断发展变化之中。本章在很多地方以及小结中列出了很多 RFC 以及与 DHCPv6 相关的其他资源，这些将有助于大家更深入地学习 DHCPv6 的配置以及协议本身。

## 9.2　其他上层协议

由于底层的 IPv6 提供了更大的地址空间，因而很多上层协议可能只需要做一些很小的变动即可，很多都不需要重新配置 IPv6 设备。例如，传输层协议 TCP 和 UDP 都只为 IPv6 做了非常少的改动。由于 TCP 和 UDP 都将 IP 地址作为校验和的一部分，因而需要根据 IPv6 对检验和的计算方式进行修改。此外，还有一些上层协议也要做相应的改动，如 DNS。

### 9.2.1　DNS

用于 IPv4 和 IPv6 的 DNS 负责处理域名与地址之间的映射。与 IPv4 相似，IPv6 也支持 IPv6 地址到 DNS 名的反向映射。在很多情况下，由于 IPv6 拥有更大的地址空间，

因而 IPv6 对 DNS 的需求更为强烈。

名字服务器或 DNS 服务器需要跟踪与域名相关联的信息，包括主机名到地址的映射数据库。每个名字都可以映射到一个或多个 IPv4 地址、IPv6 地址或这两种地址。与 IPv4 一样，每个主机名在 DNS 服务器中都有两条 DNS 记录：资源记录（或地址记录）和反向映射指针记录。

在 IPv4 中，DNS 的 A 资源记录负责将域名映射为 IPv4 地址。与此相似，IPv6 中的 AAAA（“4A”）资源记录也是负责将域名映射为 IPv6 地址。有关 AAAA 记录的详细信息定义在 RFC 3596 “DNS Extensions to Support IP Version 4” 中。4A（AAAA）表示 IPv6 地址尺寸是 IPv4 地址的 4 倍。AAAA 记录的结构与 A 记录极其相似，只是更大而已。IPv4 或 IPv6 都可以被用于传输这两类记录。

表 9-7 中的这两类记录都使用 FQDN（Fully Qualified Domain Name，完全合格域名/全称域名）。FDDN 可以指定 DNS 层次化结构中的确切位置，如 www.test.org 或 mail.example.com。

表 9-7　　用于 IPv4 和 IPv6 的 DNS 资源记录

| 协议 | 资源记录 | DNS 映射 |
|---|---|---|
| IPv4 | A 记录 | www.test.org A 209.165.200.225 |
| IPv6 | AAAA 记录 | www.test.org AAAA 2001:DB8:CAFE:1234:0:0:0:A1 |

对反向 DNS 查找来说，IPv6 地址使用特殊域名 ip6.arpa。等同于 IPv4 中的 PTR（pointer record，指针记录），IPv6 的指针记录也负责将 IPv6 地址映射为主机名。IPv6 指针记录看起来就是一个域名以及一串由句点（.）逆序分隔的半字节（十六进制值）。以 www.test.org 为例，相应的 IPv4 和 IPv6 指针记录如表 9-8 所示。

表 9-8　　IPv4he IPv6 的 DNS 指针记录

| 协议 | 指针记录格式 |
|---|---|
| IPv4 | 225.200.165.209 www.test.org |
| IPv6 | 0.0.1.a.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.4.3.2.1.e.f.a.c.0.8.b.d.1.0.0.2 www.test.org |

BIND（Berkeley Internet Name Domain，伯克利互联网域名）是目前互联网上应用最为广泛的 DNS 软件。BIND 是一个可以为互联网部署 DNS 协议的开源软件。BIND 自版本 9 开始支持 IPv6 的 DNS。

DNS 解析器是 DNS 通信过程的客户侧，负责向服务器发起域名到 IP 地址的解析请求。支持 IPv6 AAAA 记录的 DNS 服务器并不需要经 IPv6 网络进行查询，而是可以利用 IPv4 对查询请求做出应答。

与 IPv4 一样，可以为 IPv6 地址配置静态主机名。**ipv6 host** 命令与 IPv4 中的 **ip host** 命令相似，只是指定了 IPv6。命令 **ipv6 host** 可以在主机名缓存中定义一条静态的主机

名到地址的映射，其命令语法如下

```
Router(config)# ipv6 host name [port] ipv6-address1 [ipv6-address2...ipv6-
address4]
```

命令中的参数信息如下。

- *name*：IPv6 主机名，第一个字符可以是字母或数字。如果使用数字，那么所能执行的操作将会受到限制。
- **port**（可选）：用于相关联的 IPv6 地址的默认 Telnet 端口号。
- *ipv6-address1*：相关联的 IPv6 地址，该参数必须采用 TFC 2373 规定的表达形式，即十六进制形式的以冒号分隔的 16 比特值。
- *ipv6-address2...ipv6-address4*（可选）：其他相关联的 IPv6 地址。最多可以将 4 个地址绑定到一个主机名上。

例 9-7 在路由器 R1 上配置了命令 **ipv6 host**，以映射路由器 R2 和 R3 的 IPv6 接口地址。对 IPv6 来说，将主机名映射到地址比 IPv4 更为有利，这是因为 IPv6 地址较长，因而通过 **ipv6 host** 命令使用 DNS 服务器或静态映射就显得更有吸引力。与 IPv4 相比，简单的 ping 或 Telnet 一个 IPv6 地址就是一件非常繁琐的工作，而且很容易敲错。从例 9-7 可以看出，使用主机名更加容易些。当然，必须在企业内设计一套合理的命名方案，以便于记住这些名字。

**例 9-7 R1 上的静态主机名到 IPv6 地址的映射**

```
R1(config)# ipv6 host R2-s0 2001:db8:cafe:a001::2
R1(config)# ipv6 host R3-any 2001:db8:cafe:a003::2 2001:db8:cafe:a002::2
R1(config)# end

R1# ping r2-s0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:CAFE:A001::2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32 ms
R1#

R1# telnet r3-any
Trying R3-any (2001:DB8:CAFE:A003::2)... Open

User Access Verification

Password:
R3>
```

利用命令 **show hosts** 可以验证上述配置（如例 9-8 所示）。命令 **show hosts** 可以显示默认域名、名字查询服务方式、名字服务器主机列表以及缓存的主机名和地址列表。

**例 9-8 命令 show hosts**

```
R1# show hosts
Name lookup view: Global
Default domain is not set
Name/address lookup uses static mappings

Codes: UN - unknown, EX - expired, OK - OK, ?? - revalidate
       temp - temporary, perm - permanent
       NA - Not Applicable None - Not defined

Host                      Port  Flags      Age Type   Address(es)
R2-s0                     None  (perm, OK)  0  IPv6   2001:DB8:CAFE:A001::2
R3-any                    None  (perm, OK)  0  IPv6   2001:DB8:CAFE:A003::2
2001:DB8:CAFE:A002::2
R1#
```

如果只有少量的主机名到 IPv6 地址的映射，那么使用命令 **show hosts** 还比较方便。但是当映射数量非常多时，就应该考虑使用 DNS 服务器的服务了。Cisco IOS 中的 DNS 解析器可以接受 IPv4 地址和/或 IPv6 地址作为名字服务器。命令 **ip name-server** 与 IPv4 中的命令完全相同，作用是指定一台或多台可用于名字和地址解析的 DNS 服务器。该命令的语法格式如下：

```
Router(config)# ip name-server server-address1 [server-address2...server-address6]
```

命令中的参数信息如下。

- *server-address1*：名字服务器的 IPv4 或 IPv6 地址。
- server-address2...server-address6（可选）：其他名字服务器的 IP 地址（最多六台名字服务器）。

注：如果配置了多台 DNS 服务器，那么路由器通常会使用配置中列出的第一台服务器。不过根据网络活动程度，路由器也可以查询配置中列出的多台 DNS 服务器。

例 9-9 在路由器 R1 上配置了命令 **ip name-server**。请注意，该命令同时指定了一个 IPv6 地址和一个 IPv4 地址。

**例 9-9 为名字服务器指定地址**

```
R1(config)# ip name-server 2001:db8:cafe:2::77 192.168.2.77
R1(config)#
```

**DNS 查询与响应**

IPv6 中的域名解析与 IPv4 中的域名解析几乎没有什么区别。例如，假设使用 IPv6

客户端的某个用户在其浏览器中输入 www.facebook.com。

Web 浏览器识别出域名请求，并调用客户端的本地解析器，将域名和地址关联在一起。在查询其 DNS 服务器之前，客户端的解析器将首选检查其域名缓存和本地主机表。如果都找不到该域名，那么解析器就会向客户端上静态或动态配置的本地 DNS 服务器发送请求。

接下来由本地 DNS 服务器负责为客户端解析该域名。此时需要在本地 DNS 服务器和根域名服务器、TLD（Top Level Domain，顶级域）域名服务器以及权威域名服务器之间执行多次递归查询。本地 DNS 服务器收到所请求域名的 IPv6 地址后，就会将该信息返送给发起该域名查询请求的客户端。这样客户端就知道了相应的 IPv6（被用做数据包的目的 IP 地址）。

以 www.facebook.com 为例，例 9-10 解释了由客户端发送给 DNS 服务器的 DNS 查询过程。DNS 知道答案后，就将 IPv6 地址（即 www.facebook.com 的 IPv6 地址 2a03:2880:2110:3f01:face:b00c::）作为响应发送给客户端。DNS 服务器对此域名解析的响应情况如例 9-11 所示。

**例 9-10 对 www.facebook.com 的 DNS 查询**

```
Domain Name System (query)
    [Response In: 7]
    Transaction ID: 0x01e1
    Flags: 0x0100 (Standard query)
        0... .... .... .... = Response: Message is a query
        .000 0... .... .... = Opcode: Standard query (0)
        .... ..0. .... .... = Truncated: Message is not truncated
        .... ...1 .... .... = Recursion desired: Do query recursively
        .... .... .0.. .... = Z: reserved (0)
        .... .... ...0 .... = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    Queries
        www.facebook.com: type AAAA, class IN
            Name: www.facebook.com
            Type: AAAA (IPv6 address)
            Class: IN (0x0001)
```

通过命令 **nslookup**（名字服务器查询）查询 DNS 服务器，可以验证名字到地址的映射情况（如例 9-12 所示）。

有关 IPv6 DNS 的更多信息，请参考 RFC 3596“DNS Extensions to Support IP Version 6”：http://tools.ietf.org/html/rfc3596。

### 例 9-11 DNS 服务器对 www.facebook.com 的响应

```
Domain Name System (response)
    Transaction ID: 0x01e1
    Flags: 0x8180 (Standard query response, No error)
        1... .... .... .... = Response: Message is a response
        .000 0... .... .... = Opcode: Standard query (0)
        .... .0.. .... .... = Authoritative: Server is not an authority for domain
        .... ..0. .... .... = Truncated: Message is not truncated
        .... ...1 .... .... = Recursion desired: Do query recursively
        .... .... 1... .... = Recursion available: Server can do recursive queries
        .... .... .0.. .... = Z: reserved (0)
        .... .... ..0. .... = Answer authenticated: Answer/authority portion was not
authenticated by the server
        .... .... ...0 .... = Non-authenticated data: Unacceptable
        .... .... .... 0000 = Reply code: No error (0)
    Questions: 1
    Answer RRs: 1
    Authority RRs: 5
    Additional RRs: 5
    Queries
        www.facebook.com: type AAAA, class IN
            Name: www.facebook.com
            Type: AAAA (IPv6 address)
            Class: IN (0x0001)
    Answers
        www.facebook.com: type AAAA, class IN, addr 2a03:2880:2110:3f01:face:b00c::
            Name: www.facebook.com
            Type: AAAA (IPv6 address)
            Class: IN (0x0001)
            Time to live: 47 minutes, 40 seconds
            Data length: 16
            Addr: 2a03:2880:2110:3f01:face:b00c::
```

### 例 9-12 利用 nslookup 命令解析域名

```
PC> nslookup
> set type=AAAA
> www.facebook.com
Non-authoritative answer:
www.facebook.com  AAAA IPv6 address = 2a03:2880:2110:3f01:face:b00c::

PC> nslookup
```

### 9.2.2 TCP 和 UDP

不同协议在不同层面上都使用了校验和。校验和对数据块进行定长计算，用于检测可能因传输原因导致的意外差错。任何将 IP 地址纳入校验和计算的传输层协议或其他上层协议都必须进行修改，以便能在 IPv6 网络中正常使用。这是包含更大的 128 比特 IPv6 地址的需要。TCP 和 UDP 的报头中都包含一个 16 比特校验和字段（如图 9-5 所示）。

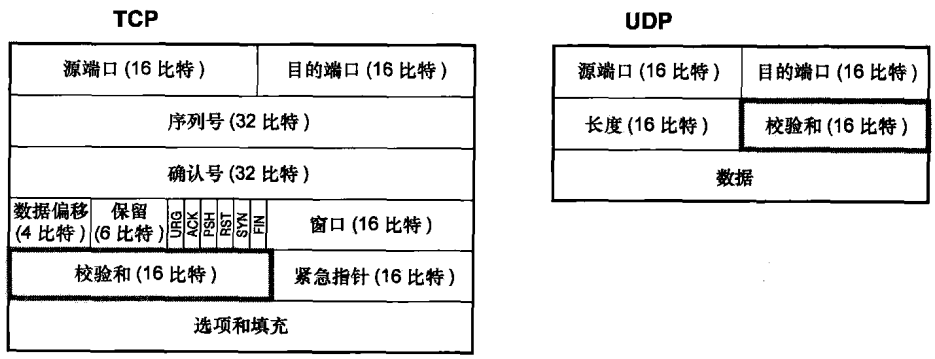



图 9-5 TCP 和 UDP

TCP 和 UDP 都会为校验和计算生成一个伪报头。伪报头中包含了用于计算校验和的网络层和传输层字段。当 TCP 或 UDP 通过 IPv6 网络进行传输时，校验和将包括：

- 源 IPv6 地址；
- 目的 IPv6 地址；
- 上层净荷长度（TCP/UDP 报头和数据）；
- IPv6 下一报头值（包括所有扩展报头）。

因而需要修改 TCP 和 UDP 协议，以便能够通过 IPv6 网络进行传输。由于它们的校验和都使用 IPv6 地址，因而必须重写协议内核以适应更长地址尺寸的需求。虽然地址尺寸更长，但 IPv6 中使用的校验和计算方法却与 IPv4 完全相同。

由于 IPv4 报头中有一个校验和字段，因而 UDP 通过 IPv4 进行传输时校验和是可选项。但 IPv6 为了提高处理速度，从报头中删除了校验和字段。而 TCP 和 UDP 除了传输层协议之外，还都有校验和功能。考虑到 IPv6 不包含校验和，因而通过 IPv6 进行传输时，校验和不再是可选项，而是强制项。

由于 IP 是一种尽力而为型传送协议，因而需要由传输层协议来确保数据的完整性。有关上层协议校验和的内容定义在 RFC 2460 “Internet Protocol, Version 6 (IPv6) Specification” 中。

## 9.3 本章小结

本章主要讨论了 DHCPv6，解释了 DHCPv6 与 DHCPv4 之间的异同点。DHCPv6

有下面两种形式。

- **状态化 DHCPv6**：主机的编址和配置信息都是从 DHCPv6 服务器获取的。
- **无状态 DHCPv6**：主机通过路由器宣告（RA）消息获取其编址信息，并从 DHCPv6 服务器获取其他配置信息。

被配置为自动获取其编址和其他配置信息的客户端可以采用以下方式：

- SLAAC（Stateless Address Autoconfiguration，无状态地址自动分配）；
- 状态化 DHCPv6；
- 无状态 DHCPv6。

主机采用哪种方式取决于路由器 RA 消息中所包含的信息，特别是 M 标记和 O 标记。

- **SLAAC**：Cisco 路由器在默认情况下将 M 标记和 O 标记均设置为 0，表示不能从 DHCPv6 服务器获取配置信息。需要动态地址分配的主机会从 RA 消息中获得全部信息；
- **状态化 DHCPv6**：M 标记为 1 时，告知主机将使用状态化 DHCPv6 服务，必须从 DHCPv6 服务器直接获取其全部编址和配置信息。
- **无状态 DHCPv6**：M 标记为 0 且 O 标记为 1 时，告知主机应该从 RA 消息获取前缀、前缀长度、默认网关及其他信息（如 MTU）。但是还可以从 DHCPv6 服务器获取一些额外配置信息，如 DNS 服务器的地址。

配置无状态 DHCPv6 服务的三个基本步骤如下。

**第 1 步**：配置 DHCPv6 服务器池的名字及配置参数。

**第 2 步**：在接口上启用 DHCPv6 服务器池。

**第 3 步**：修改 RA 消息的参数，让主机知道可以从 DHCPv6 服务器获取其他配置信息。

配置 DHCPv6 服务器池名字和配置参数的命令如下。

- Router(config)# **ipv6 dhcp pool** *poolname*：创建 DHCPv6 池并进入 DHCPv6 池配置模式。
- Router(config-dhcp)# **dns-server** *ipv6-address*：指定可用于 DHCPv6 客户端的 IPv6 DNS 服务器。
- Router(config-dhcp)# **domain** *domain*：为 DHCPv6 客户端配置域名。

在接口上启用 DHCPv6 服务器池的命令如下。

- Router(config)# **interface** *type number* 指定接口类型和接口号并进入路由器的接口配置模式。
- Router(config-dhcp)# **ipv6 dhcp server** *poolname* [ **rapid-commit** ]在接口上启用 DHCPv6 服务。**rapid-commit**（可选）允许为前缀授权采取双消息交换方法。

默认情况下，路由器宣告消息中的 O 标记为 0，将 O 标记设置为 1，表示主机可以从 DHCPv6 服务器获得其他配置信息。用于修改路由器宣告消息中 O 标记的命令为：

```
Router(config-if)# ipv6 nd other-config-flag
```

与 DHCPv4 一样，DHCPv6 路由器也可以充当中继代理，负责为处于不同网络的客户端和服务器转发 DHCPv6 消息，用于指定客户端数据包将要被转发至的目的地址，并在接口上启用 DHCPv6 中继代理：

```
Router(config-if)# ipv6 dhcp relay destination ipv6-address [interface-type
interface-number]
```

与 IPv4 中的 DNS 负责将域名解析为 IPv4 地址一样，IPv6 中的 DNS 也负责将域名解析为 IPv6 地址。Cisco 为 IPv4 和 IPv6 使用相似的配置命令来静态地映射地址。IPv4 使用的命令是 **ip host**，IPv6 使用的命令是 **ipv6 host**，作用是在主机名缓存中定义一条静态的主机名到地址的映射：

```
Router(config)# ipv6 host name [port] ipv6-address1 [ipv6-address2...ipv6-
address4]
```

为了指定一台或多台可用于名字和地址解析的 DNS 服务器，IPv6 使用了与 IPv4 相同的命令：

```
Router(config)# ip name-server server-address1 [server-address2...server-address6]
```

## 9.4 参考文献

RFC：

RFC 2460, Internet Protocol, Version 6 (IPv6) Specification , S. Deering, Cisco Systems, www.ietf.org/rfc/rfc2460 , December 1988

RFC 3315, Dynamic Host Configuration Protocol for IPv6 (DHCPv6) , R. Droms, Cisco Systems, www.ietf.org/rfc/rfc3315 , July 2003

RFC 3596, DNS Extensions to Support IP Version 6 , Thompson, Cisco Systems, www.ietf.org/rfc/rfc3596 , October 2003

RFC 3736, Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6 , R. Droms, Cisco Systems, www.ietf.org/rfc/rfc3736 , April 2004

RFC 4862, IPv6 Stateless Address Autoconfiguration , S. Thompson, Cisco Systems, www.ietf.org/rfc/rfc4862 , September 2007

网站：

Cisco IOS IPv6 Configuration Guide, Implementing DHCP for IPv6, www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-dhcp.html

# 第10章 双栈与隧道

接下来的两章将讨论 IPv4 与 IPv6 的共存和融合策略。虽然短期内 IPv4 不可能消亡，但正如第 1 章所述，IPv4 向 IPv6 的迁移是不可阻挡的潮流，至少在可预见的未来，IPv4 和 IPv6 将长期共存。与之前软硬件开发者所处理的“千年虫”问题不同，从 IPv4 向 IPv6 过渡并没有明确的截止日期或转换时间，过渡过程可能会持续数年时间。IETF 制定了多种协议、工具和机制，来帮助网络管理员将网络迁移到 IPv6，这些过渡技术可以分为三类：

- **双栈（Dual-stack）**：双栈技术允许 IPv4 和 IPv6 共存于同一个网络。双栈设备可以是同时配置了 IPv4 协议栈和 IPv6 协议栈的主机、服务器或路由器。
- **隧道（Tunneling）**：隧道技术是一种在 IPv4-only 网络中传输 IPv6 包的方法，其实现方式是将 IPv6 包封装到 IPv4 包中。
- **转换（Translation）**：NAT64（Network Address Translation 64，从 IPv6 到 IPv4 的网络地址转换）利用与 IPv4 NAT 相似的技术，可以实现纯 IPv6 设备与纯 IPv4 设备之间的通信。有关 NAT64 的详细内容将在第 11 章进行讨论。

注：虽然第 11 章还会讨论 NAT-PT（Network Address Translation-Protocol Translation，网络地址转换-协议转换）技术，但是由于 NAT-PT 与 DNS 耦合度太高以及转换时的诸多限制，使得 IETF 决定废弃该技术，具体原因请见 RFC 4996 “Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status”。

## 10.1 双栈

双栈设备完全支持 IPv4 和 IPv6，可以是同时支持这两种协议的主机、打印机、服务器、路由器或任何设备。对于 IPv4 来说，包括 IPv4 地址、ARP（Address Resolution

Protocol，地址解析协议）和 IPv4 ICMP（Internet Control Message Protocol，互联网控制报文协议）。IPv4 路由器支持 IPv4 静态路由和 IPv4 路由协议（如 EIGRP 和 OSPFv2）。对于 IPv6 来说，支持的含义不仅仅是拥有更长地址的网络报头，还包括 IPv6 全局单播地址、链路本地地址以及 ICMPv6 操作（包括 SLAAC、DAD）等。IPv6 路由器需要使用静态路由或 IPv6 路由协议（如 IPv6 EIGRP 和 OSPFv3）来路由 IPv6 包。此外，IPv6 路由器还要向外发送 ICMPv6 路由器宣告消息，并且还能提供隧道或转换服务。

如本书始终申明的那样，IPv6 并不仅仅只是拥有 128 比特的源地址和目的地址。支持 IPv6 也就意味着要实现新协议并处理 SLAAC 和 DAD 等机制。近年来大多数主机操作系统（如 Windows、Mac OS 和 Linux）都支持 IPv6。Cisco 自 2000 年发布的 Cisco IOS Release 12.2(2)T 就开始支持 IPv6 了。

运行 IPv6 的双栈设备并不是新鲜事物。在 IPv4 成为主要网络协议之前，许多组织机构的网络设备上除了运行 IPv4 之外，还同时运行着 IPX、AppleTalk、DECnet 或 SNA。

双栈设备与 IPv4 设备进行通信时，其行为特性就像纯 IPv4 设备。与 IPv6 设备进行通信时，其行为特性又像纯 IPv6 设备，图 10-1 给出了一种纯 IPv4 应用，该应用承载在 TCP 或 UDP 报文段中（由传输层端口号来辨识）。然后报文段又被封装到 IPv4 包中（相应的数据或净荷通过 IPv4 协议字段值来标识），为了通过链路传送该 IPv4 包，又将 IPv4 包封装到以太网帧中，并由以太网类型字段值 0x800 来标识。

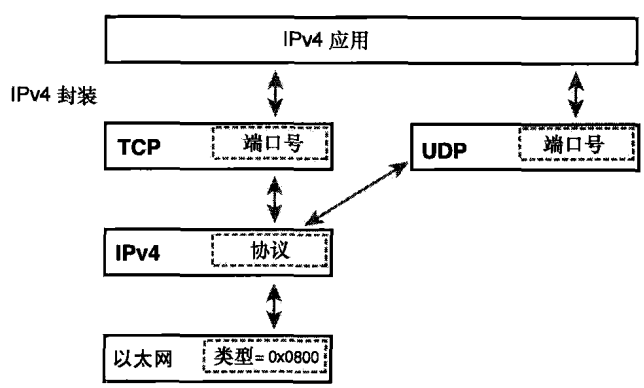



图 10-1　使用 IPv4 协议栈的 IPv4 应用

对于双栈设备上同时支持 IPv4 和 IPv6 协议的应用来说，其处理过程几乎与此完全一样（如图 10-2 所示）。TCP 和 UDP 的端口号可以标识具体应用。当选择 IPv6 协议栈时，IPv6 基本报头中的下一报头字段可以识别传输协议。IPv6 包被封装在以太网帧中，以太网类型字段值为 0x86DD。请注意，由于以太网是二层网络，因而在承载 IPv6 包时，虽然有不同的类型字段，但是以太网交换机在处理以太网帧时没有任何区别。如果典型的二层接入交换机不支持 IPv6，那么仅仅表明无法使用 IPv6 来管理该交换机。这些交换机仍然能够转发这些以太网帧，而不管其中的净荷是 IPv4 包还是 IPv6 包。

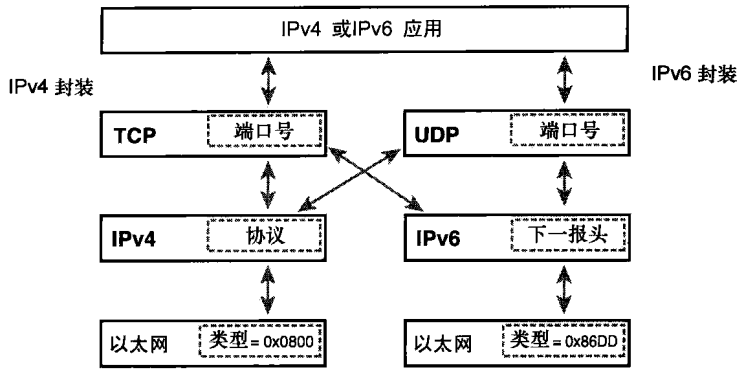



图 10-2　同时使用 IPv4 和 IPv6 协议栈的应用

虽然应用同时支持 IPv4 和 IPv6，但是设备并不会随机选择将要使用的协议。图 10-3 给出了一个双栈主机示例，如果应用同时支持 IPv4 和 IPv6 协议栈，那么就会向 DNS 服务器请求其 FQDN（Fully Qualified Domain Name，完全合格域名/全称域名）的所有可用地址，DNS 服务器也会以所有可用地址（包括 IPv4 地址和 IPv6 地址）进行响应。如果 DNS 服务器同时发送了 IPv4 地址和 IPv6 地址，那么操作系统会选用其中的一种地址。大多数情况下，应用程序默认选择的是 IPv6，不过具体选择取决于操作系统。然后，设备就利用选定的 IP 协议栈连接源端。

在图 10-3 所示的第 1 步中，双栈主机 A 对 www.example.com 的 4A（AAAA）记录进行查询；第 2 步中，DNS 服务器返回一条包含了 www.example.com 的 A 记录和 4A 记录的 DNS 查询响应消息，然后主机使用 4A 记录开始与 www.example.com 服务器进行通信。

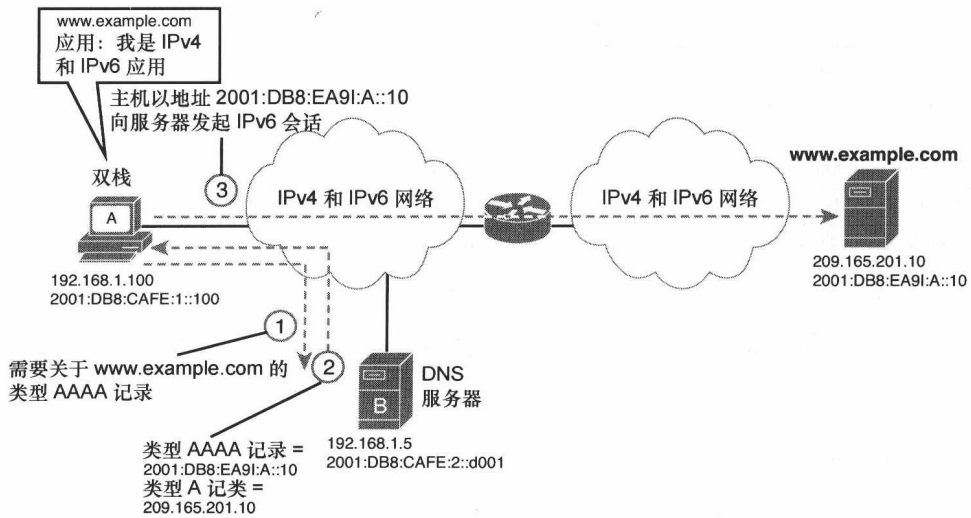



图 10-3　同时感知 IPv4 和 IPv6 的应用

注：如果大家对 IPv6 API（Application Programming Interface，应用编程接口）以及 DNS 的操作细节感兴趣，可参考 RFC 3493“Basic Socket Interface Extensions for IPv6”和 RFC 4472 “Operational Considerations and Issues with IPv6 DNS”。通常由底层操作系统来确定传输方式。

### 10.1.1 以 URL 语法表示的 IPv6 地址格式

当使用 IP 地址而不是域名时，应用就使用与特定 IP 地址相关联的协议栈。如果用户输入 IPv4 地址，那么应用就使用 IPv4 地址。如果用户输入的是 IPv6 地址，那么应用就选择 IPv6 地址。使用 Web 浏览器的时候，绝大多数时候输入的都不是地址，而是 URL（Uniform Resource Locator，统一资源定位符），如 http://www.example.com。虽然也可以使用 IP 地址，但是使用域名的最大好处就是可以避免记忆复杂的 IP 地址。不过在某些场合下（如调试或通过浏览器接口配置设备），需要在浏览器内输入 IP 地址而不是 URL，如可以输入 http://192.168.1.1 来连接并配置 Linksys 家用路由器。

那么，如何使用 IPv6 地址来完成上述工作呢？由于 IPv6 地址中包含了令人讨厌的冒号（:），因而不能直接与 URL 相兼容，浏览器会将 IPv6 地址中的冒号解析为端口号。例如，假设要在端口 5000 上访问 Linksys，那么在浏览器中输入的 URL 就是 http://192.168.1.1:5000。RFC 3986 “Uniform Resource Identifier (URI): Generic Syntax”定义了 URL 中 IPv6 地址的格式，其出发点是简化将 IPv6 地址剪切并粘贴到浏览器地址栏中的操作，将 IPv6 地址放置在括号中，后面是其他 URL 语法。表 10-1 给出了 RFC 3986 定义的使用 URL 语法的 IPv6 地址表述示例。

表 10-1 以 URL 语法表示的 IPv6 地址格式

| IPv6 地址 | URL |
|---|---|
| FEDC:BA98:7654:3210:FEDC:BA98:7654:3210 | http://[FEDC:BA98:7654:3210:FEDC:BA98:7654:3210]:80/index.html |
| 1080:0:0:0:8:800:200C:4171 | http://[1080:0:0:0:8:800:200C:417A]/index.html |
| 3ffe:2a00:100:7031::1 | http://[3ffe:2a00:100:7031::1] |
| 1080::8:800:200C:417A | http://[1080::8:800:200C:417A]/foo |
| ::192.9.5.5 | http://[::192.9.5.5]/ipng |
| ::FFFF:129.144.52.38 | http://[::FFFF:129.144.52.38]:80/index.html |
| 2010:836B:4179::836B:4179 | http://[2010:836B:4179::836B:4179] |

### 10.1.2 配置双栈网络

配置双栈网络意味着为设备同时配置 IPv4 和 IPv6。这里所说的设备指的是主机、

路由器、多层交换机、打印机以及各种拥有 IP 地址的设备。至于 IP 地址，它们是两个完全独立的网络，仅仅共享相同的“空间”而已，就像住在同一套公寓中没有真正见过或注意过对方，有点儿像“夜间行船”的室友一般。

在第 7 章中，为 Rick's Café 网络配置了 IPv6 和 OSPFv3。本节将讨论如何再在这些路由器上配置 IPv4 和 OSPFv2，从而在该网络上开启双栈功能。如此一来，双栈主机就可以通过 Rick's Café 网络传送 IPv4 包和 IPv6 包了。图 10-4 显示了 IPv4 和 IPv6 寻址方案。图中的路由器已经配置了 IPv6 和 OSPFv3。在第 6 章中为这些路由器接口配置了 IPv6 地址。在第 8 章中为这些路由器配置了 OSPFv3。只要检查路由器的运行配置即可显示之前的 IPv6 配置情况。

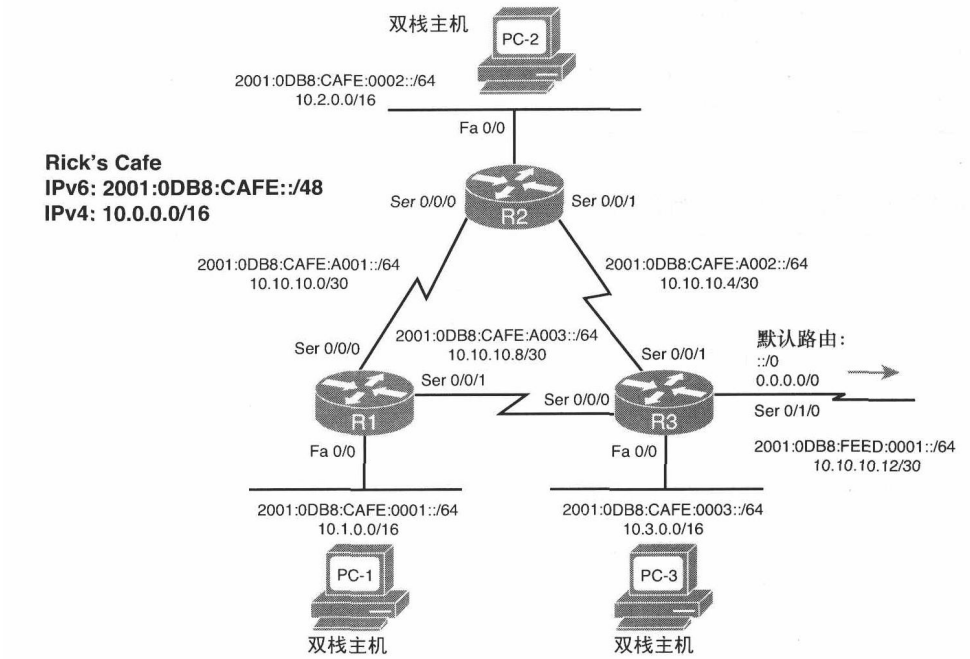



图 10-4　Rick's Café 网络的 OSPFv3 拓扑结构

首先配置路由器 R1，为其快速以太网接口和两个串行接口配置 IPv4 地址。配置完接口之后，接下来的任务就是配置 OSPFv2。例 10-1 给出了为 R1 接口配置 IPv4 地址并在这 3 个接口上启用 OSPFv2 的配置示例。这些命令都是 IPv4 中使用的命令，就像 R1 上根本不存在 IPv6 一样。当然实际上是存在的。

例 10-2 查看了路由器 R1 的当前运行配置。可以看出同时包含了 IPv4 和 IPv6 配置命令。这两种协议的配置命令都共存于同一台路由器上，每个接口都有一个 IPv4 地址和一个可路由的 IPv6 地址。IPv6 接口还有一个先前采取静态方式配置以更好识别的链路本地地址。

### 例 10-1 在 R1 上配置 IPv4 地址和 OSPFv2

```
R1(config)# interface FastEthernet0/0
R1(config-if)# ip address 10.1.0.1 255.255.0.0
R1(config-if)# exit
R1(config)# interface Serial0/0/0
R1(config-if)# ip address 10.10.10.1 255.255.255.252
R1(config-if)# exit
R1(config)# interface Serial0/0/1
R1(config-if)# ip address 10.10.10.9 255.255.255.252
R1(config-if)# exit

R1(config)# router ospf 2
R1(config-rtr)# network 10.1.0.0 0.0.255.255 area 0
R1(config-rtr)# network 10.10.10.0 0.0.0.3 area 0
R1(config-rtr)# network 10.10.10.8 0.0.0.3 area 0
```

### 例 10-2 节选的 R1 运行配置输出结果

```
interface FastEthernet0/0
 ip address 10.1.0.1 255.255.0.0          ! IPv4 address
 ipv6 address FE80::1 link-local          ! IPv6 address
 ipv6 address 2001:DB8:CAFE:1::1/64       ! IPv6 address
 ipv6 ospf 1 area 0                       ! IPv6 - OSPFv3 command
!
interface Serial0/0/0
 ip address 10.10.10.1 255.255.255.252    ! IPv4 address
 ipv6 address FE80::1 link-local          ! IPv6 address
 ipv6 address 2001:DB8:CAFE:A001::1/64    ! IPv6 address
 ipv6 ospf 1 area 0                       ! IPv6 - OSPFv3 command
!
interface Serial0/0/1
 ip address 10.10.10.9 255.255.255.252    ! IPv4 address
 ipv6 address FE80::1 link-local          ! IPv6 address
 ipv6 address 2001:DB8:CAFE:A003::1/64    ! IPv6 address
 ipv6 ospf 1 area 0                       ! IPv6 - OSPFv3 command
!
ipv6 router ospf 1                        ! IPv6 - OSPFv3 commands
 router-id 10.1.1.1
 log-adjacency-changes
!
router ospf 2                             ! IPv4 - OSPFv2 commands
 network 10.1.0.0 0.0.255.255 area 0
 network 10.10.10.0 0.0.0.3 area 0
```

```
network 10.10.10.8 0.0.0.3 area 0
log-adjacency-changes
```

继续分析 R1 的运行配置。与 OSPFv2 不同，OSPFv3 并不使用 **network** 命令在接口上启用 OSPF，而是通过命令 **ipv6 ospf** 直接在接口上启用 OSPF。

路由器 R1 是双栈路由器，就像戴了“两顶帽子”一样：一顶是 IPv4，另一顶是 IPv6。

- **接口**：每个接口都配置了适当的 IPv4 和 IPv6 地址。
  - **IPv4**：IPv4 地址是通过命令 **ip address** 配置的。
  - **IPv6**：全局单播 IPv6 地址和链路本地 IPv6 地址都是通过命令 **ipv6 address** 配置的。
- **路由表**：路由器维护了两种路由表，分别用于 IPv4 和 IPv6。
  - **IPv4**：命令 **show ip route** 可以显示 IPv4 路由表。
  - **IPv6**：命令 **show ipv6 route** 可以显示 IPv6 路由表。
- **路由协议**：R1 为 IPv4 和 IPv6 配置的动态路由协议都是 OSPF。R1 为每种协议都维护了独立的路由数据库、路由表和路由进程。
  - **IPv4**：R1 上用于 IPv4 的路由协议是 OSPFv2。
  - **IPv6**：R1 上用于 IPv6 的路由协议是 OSPFv3。
- **IP 进程**：IPv4 和 IPv6 使用不同的进程完成地址解析、发送消息以及各种基本网络操作。
  - **IPv4**：IPv4 路由器参与 ARP、ICMPv4 以及其他 IPv4 协议。
  - **IPv6**：IPv6 路由器参与 ICMPv6 邻居发现和重复地址检测等各种进程和协议。

接下来为路由器 R2 配置 IPv4 和 OSPFv2。不过与前面的配置略有不同，从例 10-3 可以看出，R2 的 OSPFv2 配置与 R1 不同。R1 使用的是 OSPF **network** 命令，而 R2 是直接在接口上通过接口命令 **ip ospf area** 在接口上启用 OSPF。这一点与在接口上使用接口命令 **ipv6 ospf area** 配置 OSPFv3 相似。用于 OSPFv2 的接口命令 **ip ospf area** 集成在 Cisco IOS Release 12.3(11)T 中。

**例 10-3　在 R2 上配置 IPv4 地址和 OSPFv2**

```
R2(config)# interface fastethernet 0/0
R2(config-if)# ip address 10.2.0.1 255.255.255.0
! Replaces the OSPFv2 network command:
R2(config-if)# ip ospf 2 area 0
R2(config-if)# exit
R2(config)# interface serial 0/0/0
R2(config-if)# ip address 10.10.10.2 255.255.255.252
! Replaces the OSPFv2 network command:
R2(config-if)# ip ospf 2 area 0
R2(config-if)# exit
R2(config)# interface serial 0/0/1
```

```
R2(config-if)# ip address 10.10.10.5 255.255.255.252
! Replaces the OSPFv2 network command:
R2(config-if)# ip ospf 2 area 0
R2(config-if)#
```

从例 10-4 的 R2 运行配置可以看到用于 IPv4 和 IPv6 协议的所有接口和 OSPF 命令。这两种协议的配置方式非常相似。在很多情况下，两者的配置命令几乎完全相同，区别仅仅在于 IPv4 是 **ip**，而 IPv6 是 **ipv6**。当然，每种协议都有各自不同的命令选项，但毋庸置疑的是，只要大家掌握了 IPv4 的配置和验证命令，那么学习相应的 IPv6 配置和验证命令就会非常简单。

**例 10-4　节选的 R2 运行配置输出结果**

```
interface FastEthernet0/0
 ip address 10.2.0.1 255.255.0.0          ! IPv4 address
 ip ospf 2 area 0                         ! IPv4 - OSPFv2 command
 ipv6 address FE80::2 link-local          ! IPv6 address
 ipv6 address 2001:DB8:CAFE:2::1/64       ! IPv6 address
 ipv6 ospf 1 area 0                       ! IPv6 - OSPFv3 command
!
interface Serial0/0/0
 ip address 10.10.10.2 255.255.255.252    ! IPv4 address
 ip ospf 2 area 0                         ! IPv4 - OSPFv2 command
 ipv6 address FE80::2 link-local          ! IPv6 address
 ipv6 address 2001:DB8:CAFE:A001::2/64    ! IPv6 address
 ipv6 ospf 1 area 0                       ! IPv6 - OSPFv3 command
!
interface Serial0/0/1
 ip address 10.10.10.5 255.255.255.252    ! IPv4 address
 ip ospf 2 area 0                         ! IPv4 - OSPFv2 command
 ipv6 address FE80::2 link-local          ! IPv6 address
 ipv6 address 2001:DB8:CAFE:A002::1/64    ! IPv6 address
 ipv6 ospf 1 area 0                       ! IPv6 - OSPFv3 command
!
router ospf 2                             ! IPv4 - OSPFv2 commands
 log-adjacency-changes
!
ipv6 router ospf 1                        ! IPv6 - OSPFv3 commands
 router-id 10.2.2.2
 log-adjacency-changes
```

例 10-5 给出了路由器 R3 完成 IPv4 配置后的运行配置情况（未包含 R3 的配置命令）。

## 例 10-5　节选的 R3 运行配置输出结果

```
interface FastEthernet0/0
 ip address 10.3.0.1 255.255.0.0          ! IPv4 address
 ip ospf 2 area 0                         ! IPv4 - OSPFv2 command
 ipv6 address FE80::3 link-local          ! IPv6 address
 ipv6 address 2001:DB8:CAFE:3::1/64       ! IPv6 address
 ipv6 ospf 1 area 0                       ! IPv6 - OSPFv3 command
!
interface Serial0/0/0
 ip address 10.10.10.10 255.255.255.252   ! IPv4 address
 ip ospf 2 area 0                         ! IPv4 - OSPFv2 command
 ipv6 address FE80::3 link-local          ! IPv6 address
 ipv6 address 2001:DB8:CAFE:A003::2/64    ! IPv6 address
 ipv6 ospf 1 area 0                       ! IPv6 - OSPFv3 command
!
interface Serial0/0/1
 ip address 10.10.10.6 255.255.255.252    ! IPv4 address
 ip ospf 2 area 0                         ! IPv4 - OSPFv2 command
 ipv6 address FE80::3 link-local          ! IPv6 address
 ipv6 address 2001:DB8:CAFE:A002::2/64    ! IPv6 address
 ipv6 ospf 1 area 0                       ! IPv6 - OSPFv3 command
!
interface Serial0/1/0
 ip address 10.10.10.13 255.255.255.252   ! IPv4 address
 ipv6 address FE80::3 link-local          ! IPv6 address
 ipv6 address 2001:DB8:FEED:1::1/64       ! IPv6 address
!
!
ip route 0.0.0.0 0.0.0.0 Serial0/1/0 ! IPv4 static default route
!
ipv6 route ::/0 Serial0/1/0          ! IPv6 static default route
!
router ospf 2
 log-adjacency-changes
 default-information originate       ! Propagate IPv4 default route
                                     ! into OSPFv2 domain
!
ipv6 router ospf 1
 router-id 10.3.3.3
 log-adjacency-changes
 default-information originate      ! Propagate IPv6 default route
                                    ! into OSPFv3 domain
```

与 R2 相似，R3 使用的也是接口命令 **ip ospf area**，而不是 OSPF 的 **network** 命令。并使用下述命令配置了一条与 IPv6 静态路由的出接口相同的 IPv4 静态路由：

```
R3(config)# ip route 0.0.0.0 0.0.0.0 serial 0/1/0
```

为了在 OSPFv2 中分发该默认路由，同样需要在路由器配置模式下使用命令 **default-information originate**：

```
R3(config)# router ospf 2
R3(config-rtr)# default-information originate
```

例 10-6 显示了每种协议的路由表。通过命令 **show ip route** 显示了所有直连路由、静态路由以及学自 IPv4 路由协议的动态路由。并在 IPv6 网络中使用命令 **show ipv6 route** 来显示相同信息。第 8 章讨论了 OSPFv3 的配置和验证命令，并检查了 IPv6 路由表。请注意，IPv6 路由没有 IPv4 路由中的时间戳。

### 例 10-6 验证 IPv4 和 IPv6 路由表

```
R1# show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 10.10.10.10 to network 0.0.0.0

     10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
C       10.10.10.8/30 is directly connected, Serial0/0/1
O       10.2.0.0/16 [110/65] via 10.10.10.2, 01:20:14, Serial0/0/0
O       10.3.0.0/16 [110/65] via 10.10.10.10, 01:20:14, Serial0/0/1
C       10.10.10.0/30 is directly connected, Serial0/0/0
C       10.1.0.0/16 is directly connected, FastEthernet0/0
O       10.10.10.4/30 [110/128] via 10.10.10.2, 01:20:14, Serial0/0/0
O*E2 0.0.0.0/0 [110/1] via 10.10.10.10, 01:15:51, Serial0/0/1
R1#

R1# show ipv6 route
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
```

```
       U - Per-user Static route
       I1 - ISIS L1, I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
       D - EIGRP, EX - EIGRP external
OE2  ::/0 [110/1], tag 1
     via FE80::3, Serial0/0/1
C    2001:DB8:CAFE:1::/64 [0/0]
     via ::, FastEthernet0/0
L    2001:DB8:CAFE:1::1/128 [0/0]
     via ::, FastEthernet0/0
O    2001:DB8:CAFE:2::/64 [110/65]
     via FE80::2, Serial0/0/0
O    2001:DB8:CAFE:3::/64 [110/65]
     via FE80::3, Serial0/0/1
C    2001:DB8:CAFE:A001::/64 [0/0]
     via ::, Serial0/0/0
L    2001:DB8:CAFE:A001::1/128 [0/0]
     via ::, Serial0/0/0
O    2001:DB8:CAFE:A002::/64 [110/128]
     via FE80::2, Serial0/0/0
C    2001:DB8:CAFE:A003::/64 [0/0]
     via ::, Serial0/0/1
L    2001:DB8:CAFE:A003::1/128 [0/0]
     via ::, Serial0/0/1
L    FE80::/10 [0/0]
     via ::, Null0
L    FF00::/8 [0/0]
     via ::, Null0
R1#
```

## 10.2　隧道

另一种 IPv4 到 IPv6 的过渡机制是隧道技术。与其他过渡技术相似，隧道技术也应该被认为是全面部署纯 IPv6 之前的一种临时解决方案。隧道技术的本质就是将 IP 包封装到其他包中，因而隧道既可以是将 IPv4 包封装在另一个 IPv4 包中，也可以是将任一种网络层协议封装到另一种网络层协议中。将 IPv6 集成到现有 IPv4 网络的一大挑战就是如何通过纯 IPv4 网络传输 IPv6 包。一种实现方式就是使用隧道，也就是 IPv6 中的叠加（overlay）隧道。叠加隧道的作用是将 IPv6 包封装到 IPv4 包中，并通过 IPv4 网络进行传输。

图 10-5 给出了一个纯 IPv6 网络或者是被纯 IPv4 网络隔离的“IPv6 孤岛”示例。

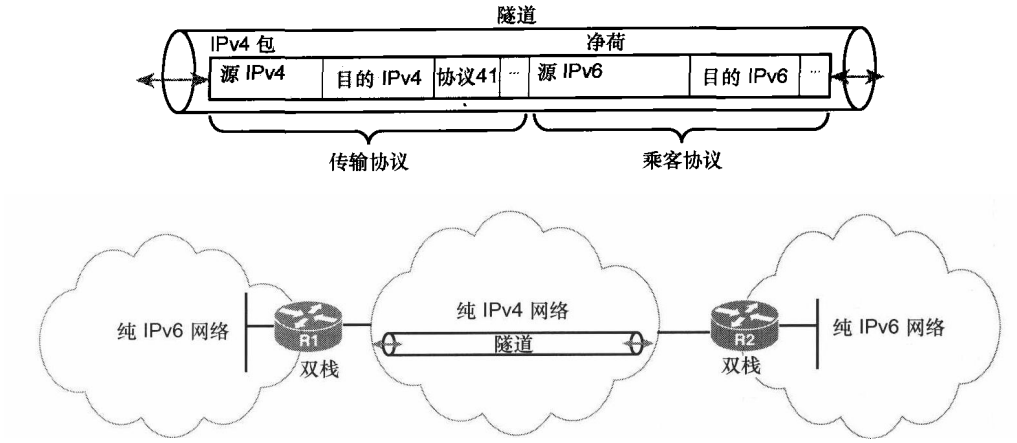



图 10-5 利用隧道连接 IPv6 孤岛

利用隧道技术，孤立的 IPv6 网络之间可以通过 IPv4 网络发送 IPv6 包。隧道包含两类协议：传输协议（transport protocol）和乘客协议（passenger protocol），图 10-5 解释了 IPv6 叠加隧道中的这两类协议。

- **传输协议**：IPv4 是传输协议，隧道是在 IPv4 中创建的。IPv4 报头中的协议字段值 41 表示所封装的数据部分是一个 IPv6 包。
- **乘客协议**：IPv6 是乘客协议，IPv6 被封装在隧道中并通过隧道进行传送。

隧道包含两类设备，分别是隧道端点和管理隧道的管理协议，隧道的这三个组件分别如下所示。

- **隧道入口点（Tunnel entry point）**：隧道入口点是将 IPv6 包（乘客协议）封装到 IPv4 包（传输协议）的位置。隧道入口点必须是双栈设备，负责接收 IPv6 包并将其封装到 IPv4 包中，以便通过 IPv4 网络进行传输。
- **隧道出口点（Tunnel exit point）**：隧道出口点是另一台双栈设备，负责接收 IPv4 包并将解封装出来的 IPv6 通过 IPv6 网络进行传输。
- **隧道管理（Tunnel management）**：隧道管理是由隧道入口点和隧道出口点完成的，负责处理隧道封装/解封装、编址、MTU、分段和差错处理。

根据隧道端点（入口点和出口点）的不同，隧道可以包括各种路由器与主机的组合。图 10-6 解释了以下三种可能存在的应用场景。

- **主机到主机（Host-to-host）**：孤立的双栈主机被纯 IPv4 网络所隔离，主机（充当隧道端点）负责创建在两台主机之间传输 IPv6 包的隧道。
- **主机到路由器（Host-to-router）**：隧道端点是由纯 IPv4 网络互连的主机和路由器。隧道一端是双栈主机，另一端是双栈路由器。路由器通过 IPv6 网络发送和接收纯 IPv6 包，与主机进行通信时，会将 IPv6 包封装到 IPv4 包中并通过隧道进行传输，主机则接收并处理 IPv6 包。

- **路由器到路由器（Router-to-router）**：隧道端点都是双栈路由器，双栈路由器之间可以创建隧道来互连 IPv6 主机孤岛。

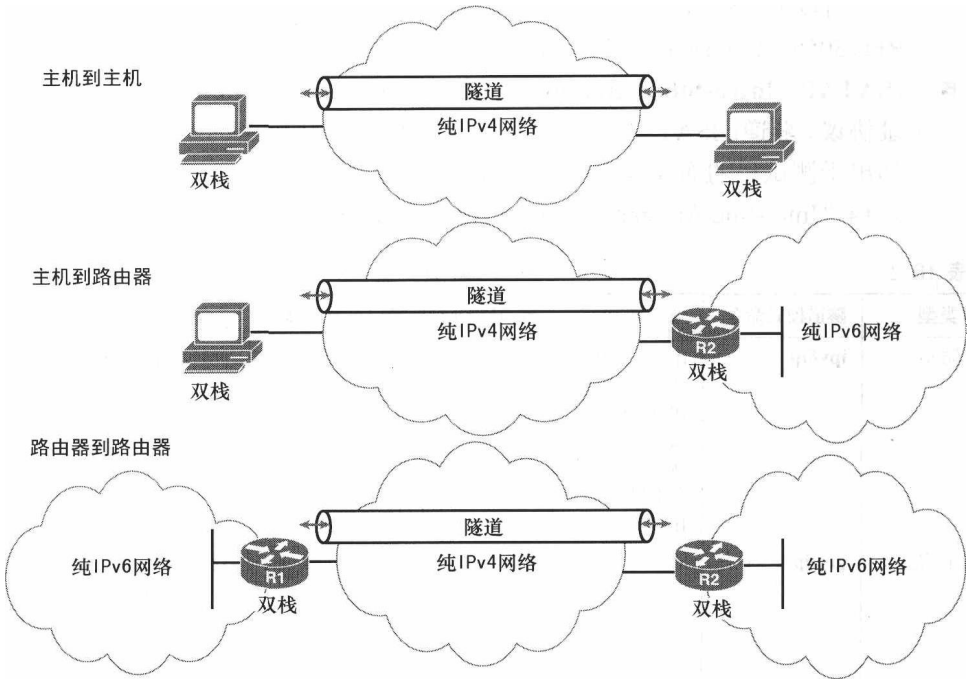



图 10-6　IPv6 隧道端点

为了在双栈节点之间建立隧道，IETF 定义了多种协议和技术。而隧道选项数量之多足以让人迷惑不解，因而为了更好地理解不同隧道之间的异同点，本章将详细描述这些隧道的细节信息。从本质上来看，所有的隧道做的都是同一件事：在两个隧道端点之间传输 IPv4 包内的 IPv6 包。此时需要做出的选择就是：

- 需要配置多少条隧道？如果只有几条隧道，那么手工隧道就很好，否则就需要选择动态隧道。
- 有哪些特殊需求（如通过隧道传输非 IP 包）使得某种隧道要优于其他隧道？

表 10-2 汇总了以下 IPv6 隧道类型。

- **手工隧道**：手工隧道等同于点到点链路，主要用于需要在两台边界路由器（或一个端系统和一台边界路由器）之间构建常规安全通信能力的稳定连接，或者用于连接远程 IPv6 网络。
- **与 IPv4 兼容的隧道**：与 IPv4 兼容的隧道无法适应大型网络的扩展性要求，Cisco 不推荐使用这类隧道，本章也不讨论该类隧道。
- **6to4 隧道**：6to4 隧道或自动 6to4 隧道的关键差区别就在于这类隧道是点到多点隧道，而不是点到点隧道。隧道的目的 IPv4 地址决定于数据包的目的 IPv6 地址，

6to4 隧道需要 IPv6 前缀或网络地址与 IPv4 隧道地址之间的关系，IPv6 地址是从 IPv4 隧道地址以反向工程方式得到的，地址格式为 2002: tunnel-IPv4-address ::/48，因而可以创建一条拥有多个目的端的隧道——点到多点隧道。6to4 隧道定义在 RFC 3056“Connection of IPv6 Domains via IPv4 Clouds”中。

- **ISATAP（Intra-Site Automatic Tunnel Addressing Protocol，站内自动隧道寻址协议）隧道**：ISATAP 用于在不具备纯 IPv6 基础设施的站点内传输 IPv6 包，如出于测试目的而仅部署了少量 IPv6 主机的应用场合。ISATAP 定义在 RFC 5214“Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)”中。

表 10-2 IPv6 隧道类型

| 隧道类型 | 隧道模式命令 | 隧道源端 | 隧道目的端 | 接口前缀或地址 | 备注 |
|---|---|---|---|---|---|
| 手工隧道 | **ipv6ip** | IPv4 地址或者是配置了 IPv4 地址的接口，可以是物理接口或环回接口的 IPv4 地址 | IPv4 地址 | IPv6 地址 | 只能承载 IPv6 包 |
| GRE 隧道 | **gre ip** | | IPv4 地址 | IPv6 地址 | 可以承载 IPv6、CLNS（Connectionless Network Service，无连接网络服务）以及其他多种数据包。<br>注：与 IPv4 兼容的隧道使用::/96 前缀，Cisco 不推荐使用该类隧道 |
| 与 IPv4 兼容的隧道 | **ipv6ip auto-tunnel** | | 不需要。这些都是点到多点隧道 | 不需要。接口地址格式为:: *tunnelsource/96* | |
| 6to4 隧道 | **ipv6ip 6to4** | | 利用 IPv6 目的地址逐包计算 IPv4 目的地址 | IPv6 地址 | |
| ISATAP 隧道 | **ipv6ip isatap** | | — | 采用改进型 EUI-64 格式的 IPv6 前缀，IPv6 地址是通过 IPv6 前缀和隧道源 IPv4 地址生成的 | 是可以在站点内连接系统的点到多点隧道<br>可以为站点内的个别双栈主机提供通信能力，站点可以使用任何 IPv6 单播地址 |

注：本章重点介绍上述隧道类型中最常见的三类通过 IPv4 网络传输 IPv6 包的隧道：手工隧道、6to4 隧道和 ISATAP 隧道。如果对其他隧道技术（包括表 10-2 未列出的隧道技术）感兴趣，建议参考 Cisco Press 出版的由 Regis Desmeules 编写的 *Cisco Self-Study:*

*Implementing Cisco IPv6 Networks* 和由 Diane Teare 编写的 *Implementing Cisco IP Routing (ROUTE)*。另一个比较好的参考内容是 *Implementing Tunneling for IPv6*: www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ ip6-tunnel.html。

### 10.2.1　手工隧道

手工隧道是双向点到点隧道，并且其 IPv4 端点地址由配置文件来决定，通常这是一种最简单的隧道实现方式。手工隧道的优点在于配置很直观；缺点在于扩展性较差。手工隧道是点到点隧道，如果应用环境需要在多台路由器之间建立多条隧道，那么就需要在每两台端点路由器之间都建立一条独立的隧道，从而导致端点的全网状连接问题，即所需的隧道数量为（n=端点数）：

隧道数量=n(n-1)/2

虽然在路由器数量有限的情况下，手工隧道能够很好地满足隧道互连的需求，但是随着连接数的增加，隧道数量将快速增加。图 10-7 给出了在两个 IPv6 孤岛（Rick's Café 的网络 2001:0DB8:CAFE::/48 和 Ace's Surfboards 的网络 2001:0DB8:ACE::/48）之间创建手工隧道的拓扑结构。这两个 IPv6 孤岛经纯 IPv4 网络进行互连，路由器 R1 和 R2 都是双栈路由器，充当经 IPv4 网络隧道化 IPv6 包的边缘路由器或边界路由器。同样，IPv4 报头中的协议字段值 41 表示被封装的数据部分是一个 IPv6 包。

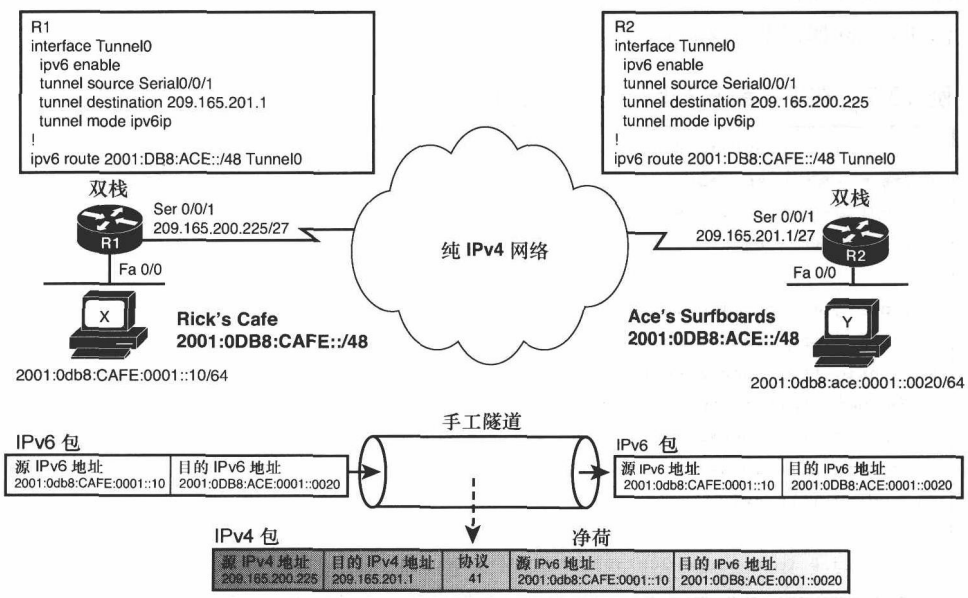



图 10-7　手工隧道

虽然隧道是点到点虚拟线路，但是两台边界路由器之间的纯 IPv4 网络却不必是点到点网络。也就是说，路由器 R1 和 R2 上的 IPv4 接口不必位于同一个子网中，就像任何普通的 IP 网一样，这两台路由器之间只要具备可达性即可。即 R1 必须能够到达 R2 串行接口所在网络 209.165.201.0/27，R2 也必须能够到达 R1 串行接口所在网络 209.165.200.224/24。

配置手工隧道的命令如表 10-3 所示。

**表 10-3　　手工隧道配置命令**

| 命令 | 描述 |
|---|---|
| Router(config)# **interface tunnel** *tunnel-number* | 指定隧道接口和隧道号，并进入路由器的接口配置模式 |
| Router(config-if)# **ipv6-address** *ipv6-prefix/prefix-length* [ **eui-64** ] | 指定分配给接口的IPv6地址并在接口上启用IPv6处理能力。如果不指定 IPv6 地址，那么也可以使用命令 ipv6 enable 来创建链路本地地址并在接口上启用 IPv6 |
| Router(config-if)# **tunnel source** { *ip-address* \| *interface-type interface number* } | 为隧道接口指定源 IPv4 地址或源接口类型和接口号，源 IPv4 地址必须可以从隧道对端可达<br>如果指定了接口，那么该接口必须配置 IPv4 地址，地址可以是物理地址，也可以是环回地址，但必须可以从隧道对端可达 |
| Router(config-if)# **tunnel destination** *ip-address* | 为隧道接口指定目的 IP4 地址或主机名 |
| Router(config-if)# **tunnel mode ipv6ip** | 指定手工 IPv6 隧道<br>注：**tunnel mode ipv6ip** 将 IPv6 指定为手工隧道的乘客协议，并将 IPv4 指定为封装和传输协议 |
| Router(config)# **ipv6 route** *ipv6-prefix/prefix-length* **tunnel** *tunnel-number* | 利用命令 **interface tunnel** 中指定的隧道号为 IPv6 前缀配置一条静态路由 |

例 10-7 解释了如何利用这些命令在 R1 和 R2 上创建手工隧道。

**例 10-7　在 R1 和 R2 上配置手工隧道**

```
R1(config)# interface Serial0/0/1
! The next command establishes the tunnel source for the tunnel from R1 to R2
R1(config-if)# ip address 209.165.200.225 255.255.255.224
R1(config-if)# exit

R1(config)# interface Tunnel 0
R1(config-if)# ipv6 enable
R1(config-if)# tunnel source Serial0/0/1
! The next command specifies the tunnel destination
R1(config-if)# tunnel destination 209.165.201.1
R1(config-if)# tunnel mode ipv6ip
R1(config-if)# exit

R1(config)# ipv6 route 2001:DB8:ACE::/48 Tunnel 0
```

```
R2(config)# interface Serial0/0/1
```

```
! The next command is the tunnel destination
R2(config-if)# ip address 209.165.201.1 255.255.255.224
R2(config-if)# exit

R2(config)# interface Tunnel 0
R2(config-if)# ipv6 enable
R2(config-if)# tunnel source Serial0/0/1
R2(config-if)# tunnel destination 209.165.200.225
R2(config-if)# tunnel mode ipv6ip
R2(config-if)# exit

R1(config)# ipv6 route 2001:DB8:CAFE::/48 Tunnel 0
```

下面来分析 R1 的配置。

```
R1(config)# interface Serial0/0/1
R1(config-if)# ip address 209.165.200.225 255.255.255.224
```

该命令为 R1 的串行接口配置 IPv4 地址。

```
R1(config)# interface Tunnel 0
```

该命令负责指定隧道接口/隧道号并进入接口配置模式。

```
R1(config-if)# ipv6 enable
```

该命令在隧道接口上配置 IPv6 地址并在接口上启用 IPv6 处理功能，但并不意味着可达性，因而可以采用多种配置选项。既可以用 **ipv6 address** 命令配置地址，在路由器上指定一个拥有 IPv6 地址的环回接口，也可以用 **ipv6 unnumbered** 命令配置地址，使用路由器某个物理接口的 IPv6 地址。同样，此处的 IPv6 地址本身并不重要，目的是为隧道接口启用 IPv6 处理功能。

```
R1(config-if)# tunnel source Serial0/0/1
```

命令 **tunnel source** 的作用是指定隧道的 IPv4 地址。该地址必须可以从隧道对端可达。这里既可以指定 IPv4 地址，也可以使用物理接口的 IPv4 地址（本配置示例使用的就是 Serial0/0/1）。无论采用何种方式来确定隧道的 IPv4 地址，都要求隧道对端必须能够到达该 IPv4 地址。该地址是传输协议的 IPv4 源地址。

```
R1(config-if)# tunnel destination 209.165.201.1
```

命令 **tunnel destination** 的作用是指定隧道对端的 IPv4 地址。对本例来说，该地址必须对 R1 可达，该地址是传输协议的 IPv4 目的地址。

```
R1(config-if)# tunnel mode ipv6ip
```

该命令将 IPv6 指定为该收工 IPv6 隧道的乘客协议（“ipv6ip”中的“ipv6”），同时

将 IPv4 指定为传输协议（“ipv6ip”中的“ip”）。

```
R1(config)# ipv6 route 2001:DB8:ACE::/48 Tunnel 0
```

必须告诉 R1 的 IPv6 路由进程如何到达 R2 上的网络 2001:0DB8:ACE::/48，为此可以使用将隧道作为出接口的 IPv6 静态路由。

从例 10-7 可以看出，R2 的配置与 R1 非常相似。例 10-8 验证了 R1 的隧道配置情况，包括隧道源端、隧道目的端和隧道类型等。

**例 10-8　验证隧道配置**

```
R1# show interface tunnel 0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 209.165.200.225 (Serial0/0/1), destination 209.165.201.1
  Tunnel protocol/transport IPv6/IP
<Rest of output omitted for brevity>
```

例 10-9 通过命令 **show ip interface brief**（用于 IPv4）和 **show ipv6 interface**（用于 IPv6）来验证 Tunnel 0 的状态。可以看出这两种协议的状态都是“up”，但接口 Tunnel 0 只有一个 IPv6 地址。Tunnel 0 的地址是链路本地地址 FE80::D1A5:C8E1，是通过应用于隧道接口的命令 **ipv6 enable** 自动创建的。该隧道的真实源 IPv4 地址是 R1 的接口 serial 0/0/1。

**例 10-9　验证 R1 上的接口 Tunnel 0**

```
R1# show ipv6 interface brief
FastEthernet0/0            [up/up]
    FE80::1
    2001:DB8:CAFE:1::1
Serial0/0/1                [up/up]
Tunnel0                    [up/up]
    FE80::D1A5:C8E1
R1#

R1# show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            unassigned      YES manual up                    up
Serial0/0/1                209.165.200.225 YES manual up                    up
Tunnel0                    unassigned      YES unset  up                    up
R1#
```

例 10-10 通过 ping 路由器 R2 上的 IPv6 地址验证了 R1 能够到达 Ace's Surfboards 的网络 2001:0DB8:ACE::/48R1。首先利用命令 **debug ip packet detail** 查看传输协议的 IPv4 源地址和目的地址，然后发起命令 **ping 2001:db8:ace:1::1 source fastethernet 0/0**，将 R1 的接口 Fast Ethernet 0/0 用作该 **ping** 命令的源 IPv6 地址。从调试输出结果可以看出，IPv4 源地址是 209.165.200.225（隧道源地址），IPv4 目的地址是 209.165.201.1（隧道目的地址）。

**例 10-10　检查传输协议的报头**

```
R1# debug ip packet detail
IP packet debugging is on (detailed)
R1# ping 2001:db8:ace:1::1 source fastethernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:ACE:1::1, timeout is 2 seconds:
Packet sent with a source address of 2001:DB8:CAFE:1::1
!!!!!

02:08:40: IP: s=209.165.200.225 (Tunnel0), d=209.165.201.1
(Serial0/0/1), len 120, sending, proto=41
02:08:40: IP: s=209.165.201.1 (Serial0/0/1), d=209.165.200.225
(Serial0/0/1), len 120, rcvd 3, proto=41
```

利用同样的 **ping** 命令还可以验证乘客协议的 IPv6 源地址和目的地址。也就是在发起 **ping** 命令之前先应用命令 **debug ip packet**（如例 10-11 所示）。可以看出，IPv6 源地址是 2001:db8:cafe::1，也就是 **ping** 命令中指定的 R1 的接口 Fast Ethernet 0/0 的 IPv6 地址。而调试输出结果中的 IPv6 目的地址则与 **ping** 命令中指定的地址相匹配（即 2001:db8:ace:1::1）。

**例 10-11　检查乘客协议的报头**

```
R1# debug ipv6 packet
IPv6 unicast packet debugging is on

R1# ping 2001:db8:ace:1::1 source fastethernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2001:DB8:ACE:1::1, timeout is 2 seconds:
Packet sent with a source address of 2001:DB8:CAFE:1::1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/67/68 ms
R1#
```

```
02:24:13: IPv6: SAS picked source 2001:DB8:CAFE:1::1 for 2001:DB8:ACE:1::1
(FastEthernet0/0)
02:24:13: IPV6: source 2001:DB8:CAFE:1::1 (local)
02:24:13:       dest 2001:DB8:ACE:1::1 (Tunnel0)
02:24:13:       traffic class 0, flow 0x0, len 100+0, prot 58, hops
64, originating
```

上面已经讨论了隧道的配置及验证操作，大家可能对实际的路由过程还有疑惑，毕竟现在是要通过 IPv4 网络来路由 IPv6 包。如例 10-11 所示，当发起 **ping** 命令时，所创建的 IPv6 包的源地址和目的地址如下所示。

- IPv6 源地址：2001:db8:cafe::1；
- IPv6 目的地址：2001:db8:ace:1::1。

例 10-12 解释了 R1 通过 IPv4 网络转发 IPv6 包的路由过程，并在后面的备注中做了进一步描述。小括号中的备注号与后面的描述顺序保持一致。

**例 10-12　路由过程**

```
R1# show ipv6 route
<Rest of output omitted for brevity>

S   2001:DB8:ACE::/48 [1/0]    ! (1)
     via ::, Tunnel0

R1# show ip route
<Rest of output omitted for brevity>

     209.165.200.0/27 is subnetted, 1 subnets
C       209.165.200.224 is directly connected, Serial0/0/1    ! (3)
     209.165.201.0/27 is subnetted, 1 subnets
S       209.165.201.0 is directly connected, Serial0/0/1   ! (4)
R1#


R1(config)# interface Tunnel 0
R1(config-if)# ipv6 enable
R1(config-if)# tunnel source Serial0/0/1   ! (3)
R1(config-if)# tunnel destination 209.165.201.1   ! (4)
R1(config-if)# tunnel mode ipv6ip          ! (2)
```

**乘客协议：**

1．查找 R1 的 IPv6 路由表，以找到与该数据包的 IPv6 目的地址 2001:db8:ace:1::1 最匹配的路由，决定使用前面创建的静态路由，出接口为 Tunnel 0。

2．Tunnel 0 的隧道模式表明该 IPv6 包将会被封装到 IPv4 包中，手工隧道模式表示 IPv4 源地址和目的地址（传输协议）都是在隧道接口命令中指定的。

**传输协议：**

3．命令 **tunnel source** 指向接口 serial 0/0/1 的 IPv4 地址 209.165.200.225。该地址是 IPv4 传输协议使用的 IPv4 源地址。源地址必须处于“up”状态，因而必须位于 R1 的路由表中。

4．命令 **tunnel destination** 指定隧道对端的 IPv4 地址 209.165.201.1。该地址是传输协议使用的 IPv4 目的地址。该地址必须对 R1 可达，也就是说该地址可以位于其路由表中，隧道目的地址209.165.201.1与IPv4路由表中的静态路由相匹配，出接口为serial 0/0/1。

至此，携带了 IPv6 净荷的 IPv4 包就可以从接口 serial 0/0/1 转发出去了。R2 收到该数据包后，会解封装出 IPv6 包，并利用其 IPv6 路由表将该 IPv6 包转发到目的地。

### 10.2.2　6to4 隧道

手工隧道虽然配置简单，但如前所说，难以满足大量隧道的扩展性需求。它需要为每对路由器都配置一条独立的隧道，只要新加入一台路由器，都要在现有的路由器上各配置一条新隧道。该缺点与在网络中使用静态路由实现网络的可达性相似。对某些企业来说，管理少数静态配置的手工隧道路由是可接受的，而其他企业则希望寻找更具扩展性的隧道解决方案。

IETF 定义了一种被称为 6to4 的隧道机制，可以通过一条已配置隧道自动连接多个 IPv6 网络。6to4 隧道定义在 RFC 3056“Connection of IPv6 Domains via IPv4 Clouds”中。6to4 隧道属于点到多点连接，单个 6to4 隧道可以连接任意数量的 IPv6 网络，也就是说单个 6to4 隧道可以有任意数量的隧道目的端（如图 10-8 所示）。路由器 R1 可以配置一个能够到达路由器 R2 到 R6 所在 IPv6 网络的 6to4 隧道。

6to4 隧道（或称为自动 6to4 隧道）与手工隧道的区别主要是，手工隧道必须静态配置隧道对端（隧道 IPv4 目的地址），而对于 6to4 隧道来说，隧道 IPv4 目的地址是从数据包的 IPv6 地址自动衍生而来的。也就是这两类地址之间必然存在某种联系，即 IPv6 地址是可达的 IPv4 地址加上一个被保留用作该隧道的特殊前缀（定义在 RFC 2056 中）。IANA 将前缀 2002::/16 永久分配为使用 6to4 隧道自动访问 IPv6 网络。

> 注：6to4 隧道的一个局限性是无法将流量发生到域外。隧道两端的路由器都必须被配置为 6to4 隧道路由器，否则就会出现流量黑洞。此外，6to4 隧道只支持静态路由，不支持动态路由。

因此，使用 32 比特 IPv4 地址和 2002::/16 可以得到一个 IPv6 地址 2002:*ipv4-address*::/48。下面来分析一下具体的实现过程。

图 10-9 解释了从 IPv4 地址反向得到 IPv6 地址的三个简单步骤。

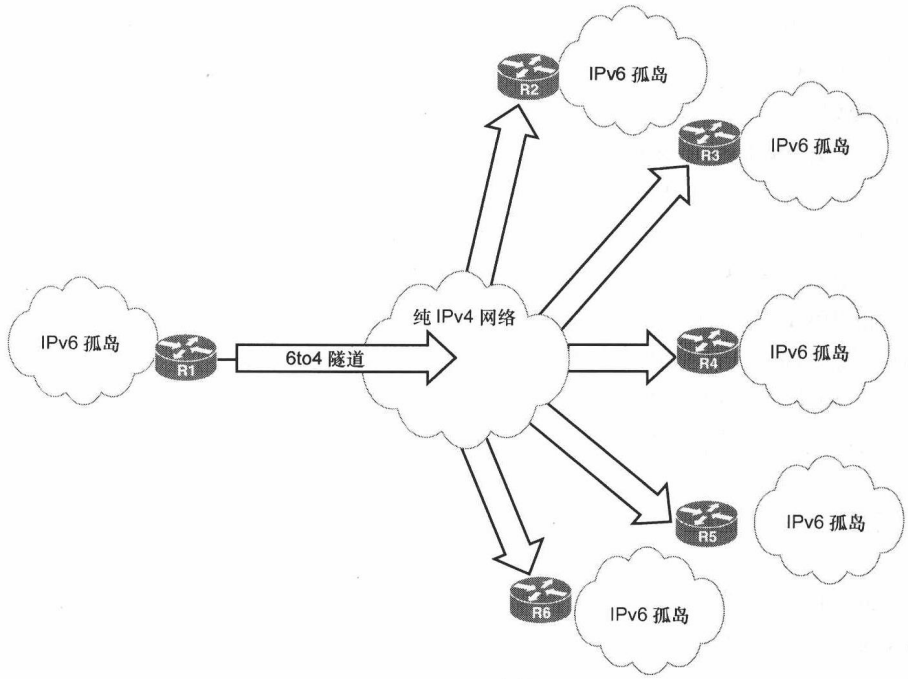
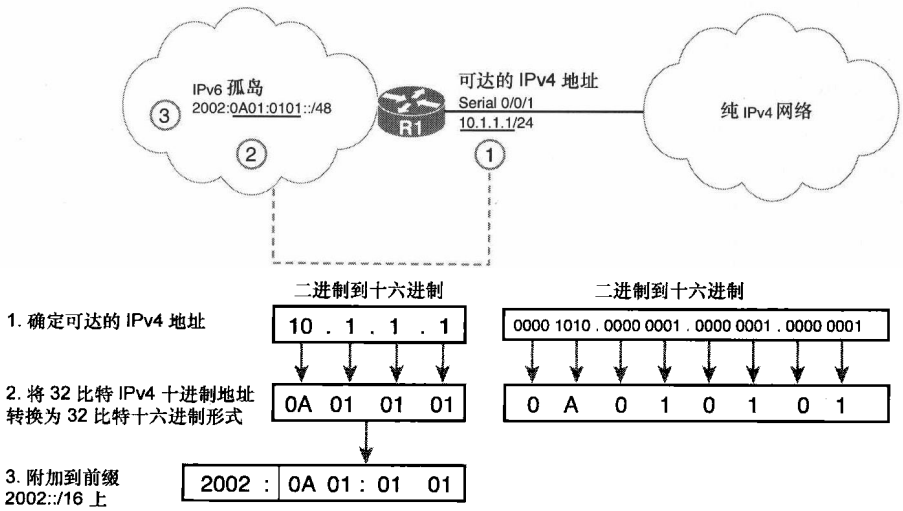



图 10-8 6to4 隧道




R1 的 LAN 的 IPv6 地址为 2002:0A01:0101::/48

图 10-9 6to4 从 IPv4 地址反向构造 IPv6 地址的步骤

**第 1 步**：确定希望用来构造 IPv6 地址的可达的 IPv4 地址。本案例使用可达的 IPv4 地址是 10.1.1.1/24（R1 的接口 serial 0/0/1），该地址对隧道对端的路由器来说是可达的。

注：本案例选用该地址的目的是简化地址转换过程。但是，如果 R1 是一台面向互联网的路由器，那么该地址就必须是一个公有 IPv4 地址，而不能是本案例使用的 RFC 1918 保留地址。

注：该 IPv4 地址对于所有的隧道对端路由器来说都是可达的，本章将在后面的案例中使用环回地址而不是物理接口地址来验证这一点。

**第 2 步**：将以十进制表示的 IPv4 地址 10.1.1.1 转化为十六进制形式，即 32 比特十六进制值 0A01:0101。

**第 3 步**：将该 32 比特十六进制值附加到 IANA 保留给 6to4 使用的前缀 2002::/16 上，得到 IPv6 前缀或网络地址 2002:0A01:0101::/48。因而 R1 的 IPv6 网络上的主机将拥有使用该前缀的 IPv6 地址，该前缀可以在 R1 的 IPv6 域内进行子网划分和路由。

将该方法扩展到图中的其他路由器。图 10-10 给出了其他网络使用各自面向提供商网络的接口的可路由 IPv4 地址确定得到（反向工程）的 IPv6 地址。虽然地址 10.0.0.0/8 不是公有地址，但本案例使用该地址的目的是便于解释如何将十进制 IPv4 地址转化为用于 IPv6 地址的 32 比特十六进制值。从图 10-10 可以看出，所有的 IPv6 地址都是在前缀 2002::/16 后附加了一个被转化之后的可路由接口的 32 比特 IPv4 地址。

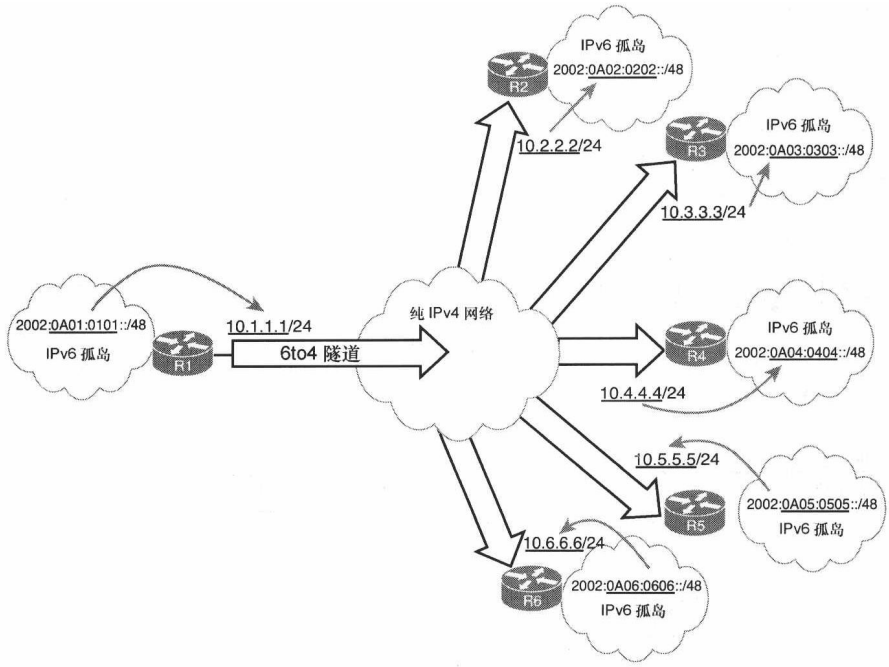



图 10-10　6to4 从 IPv4 地址反向构造 IPv6 地址示例

配置 6to4 隧道的命令如表 10-4 所示。可以看出这些命令与手工隧道的配置命令非常相似。例 10-13 以高亮方式突出了两者的配置区别以及 6to4 隧道的工作方式。

**表 10-4　　6to4 隧道配置命令**

| 命令 | 描述 |
|---|---|
| Router(config)# **interface tunnel** *tunnel-number* | 指定隧道接口和隧道号，并进入路由器的接口配置模式 |
| Router(config-if)# **ipv6-address** *ipv6-prefix/prefix-length* [ **eui-64** ] | 指定分配给接口的 IPv6 地址并在接口上启用 IPv6 处理能力。如果不指定 IPv6 地址，那么也可以使用命令 **ipv6 enable** 来创建链路本地地址并在接口上启用 IPv6<br>如果指定了 IPv6 地址，那么跟在前缀 2002::/16 之后的 32 比特必须与分配给隧道源端的 IPv4 地址相同 |
| Router(config-if)# **tunnel source** { *ip-address* \| *inter face-type interface number* } | 为隧道接口指定源 IPv4 地址或源接口类型和接口号，源 IPv4 地址必须可以从隧道对端可达<br>如果指定了接口，那么该接口必须配置 IPv4 地址，地址可以是物理地址，也可以是环回地址，但必须可以从隧道对端可达 |
| Router(config-if)# **tunnel mode ipv6ip 6to4** | 指定一个使用 6to4 地址的 IPv6 隧道，IPv4 目的地址将通过 6to4 技术得到 |
| Router(config)# **ipv6 route** *ipv6-prefix/ prefix-length* **tunnel** *tunnel-number* | 为 IPv6 6to4 前缀 2002::/16 配置一条去往指定隧道接口的静态路由<br>**ipv6 route** 命令中指定的隧道号必须与 **interface tunnel** 命令中的隧道号相同 |

学习这些命令使用方法的最好方式就是实际配置一条 6to4 隧道并分析其具体工作方式。下面就以图 10-11 所示的应用场景为例，首先在路由器 R1 和 R2 上配置 6to4 隧道，其他路由器的配置则完全相似。

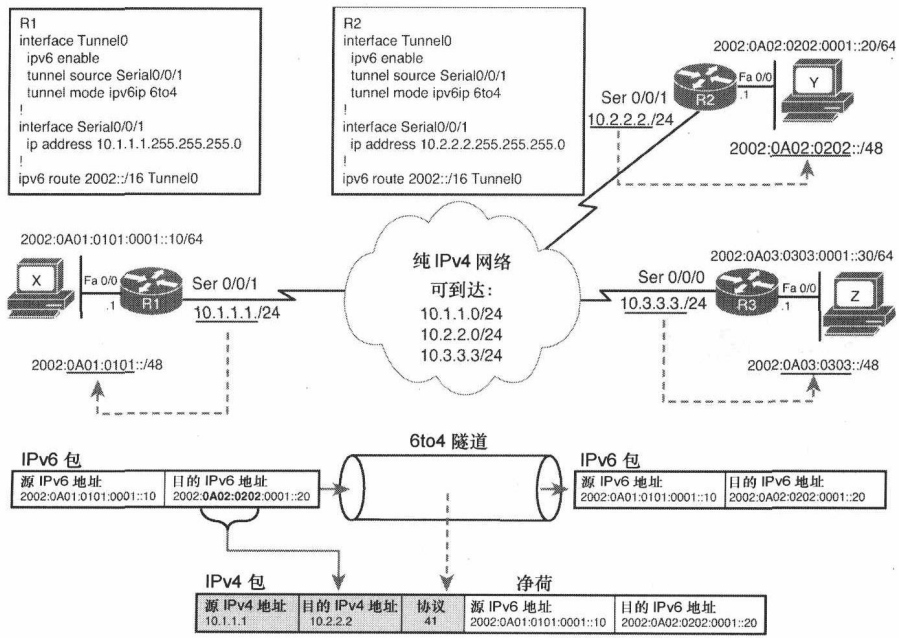



图 10-11　使用串行接口的 6to4 隧道

例 10-13 给出了路由器 R1 和 R2 的配置示例。大多数配置命令都与手工隧道的配

置相似。请注意，R1 和 R2 的接口 Fast Ethernet 0/0 的 IPv6 地址都是由各自串行接口的 IPv4 地址加上前缀 2002::/16 得到的。请记住，举这样例子的目的与前面讨论手工隧道时的例子完全一样。如图 10-5 所述，将 IPv6 包（乘客协议）封装到 IPv4 包（传输协议）中，如此就可以通过 IPv4 网络在 IPv6 孤岛网络之间传送 IPv6 包。

**例 10-13　在 R1 和 R2 上配置 6to4 隧道**

```
R1(config)# interface Serial0/0/1
! IPv4 address matches 32 bits of FastEthernet0/0 IPv6 address
R1(config-if)# ip address 10.1.1.1 255.255.255.0
R1(config-if)# exit

R1(config)# interface FastEthernet0/0
! 32 bits of IPv6 address matches Serial0/0/1 IPv4 address
R1(config-if)# ipv6 address 2002:A01:101:1::1/64
R1(config-if)# exit

R1(config)# interface Tunnel 0
R1(config-if)# ipv6 enable
R1(config-if)# tunnel source Serial0/0/1
! Use 6to4 technique to determine tunnel destination IPv4 address
R1(config-if)# tunnel mode ipv6ip 6to4
R1(config-if)# exit

R1(config)# ipv6 route 2002::/16 Tunnel 0
```

```
R2(config)# interface Serial0/0/1
! IPv4 address matches 32 bits of FastEthernet0/0 IPv6 address
R2(config-if)# ip address 10.2.2.2 255.255.255.0
R2(config-if)# exit

R2(config)# interface FastEthernet0/0
! 32 bits of IPv6 address matches Serial0/0/1 IPv4 address
R2(config-if)# ipv6 address 2002:A02:202:1::1/64
R2(config-if)# exit

R2(config)# interface Tunnel 0
R2(config-if)# ipv6 enable
R2(config-if)# tunnel source Serial0/0/1
! Use 6to4 technique to determine tunnel destination IPv4 address
R2(config-if)# tunnel mode ipv6ip 6to4
R2(config-if)# exit

R2(config)# ipv6 route 2002::/16 Tunnel 0
```

从例 10-13 所示配置来看，隧道接口命令虽然与手工隧道相似，但是也有两点区别：一个是命令 **tunnel mode ipv6ip 6to4**，另一个是没有命令 **tunnel destination**。命令 **tunnel mode ipv6ip 6to4** 的作用是告诉路由器使用 IPv6 目的地址（乘客协议）来确定隧道目的 IPv4 地址（传输协议）。由于隧道目的地址是自动从 IPv6 包得到的，因而不需要通过 **tunnel destination** 命令配置一个预设的目的地址，这就是 6to4 的魔力之所在！由于隧道目的端是由 IPv6 包的目的地址自动确定的，因而只要 IPv6 网络的地址格式是 2002:*ipv4-address*::/48，那么就可以通过 6to4 隧道可达。

最后的部分是配置静态路由，即 **ipv6 route 2002::/16 Tunnel 0**。所有前缀为 2002::/16 的 IPv6 包都被传送到 Tunnel 0 接口进行转发。如前所述，利用命令 **tunnel mode ipv6ip 6to4** 可以在将 IPv6 封装到 IPv4 包中时确定真实的 IPv4 地址。

如例 10-14 所示，从 R1 向 R2 和 R3 的 IPv6 网络发起的 ping 操作均成功。

**例 10-14　利用 ping 命令验证连接性**

```
R1# ping 2002:0a02:0202:1::1 source fastethernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:A02:202:1::1, timeout is 2 seconds:
Packet sent with a source address of 2002:A01:101:1::1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/72/92 ms

R1# ping 2002:0a03:0303:1::1 source fastethernet 0/0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2002:A03:303:1::1, timeout is 2 seconds:
Packet sent with a source address of 2002:A01:101:1::1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/66/68 ms
R1#
```

首先来看第一条 **ping** 命令。请注意 R1 是如何确定隧道对端的 IPv4 目的地址的。在 R1 上发起的该 **ping** 命令为：

```
R1# ping 2002:0a02:0202:1::1 source fastethernet 0/0
```

IPv6 路由表中与目的地址 2002:0a02:0202:1::1 最匹配的路由表项是已配置的静态路由：

```
R1(config)# ipv6 route 2002::/16 Tunnel 0
```

出接口是 Tunnel 0，从例 10-13 的 R1 配置可以看出该隧道转发 IPv6 包的方式。IPv6

包(乘客协议)被封装到 IPv4 包(传输协议)中。与手工隧道相似,也配置了 **tunnel source** 命令。对本例来说就是接口 serial 0/0/1 的 IPv4 地址 10.1.1.1。

与手工隧道不同的是，6to4 隧道没有配置 **tunnel destination** 命令，直接通过命令 **tunnel mode ipv6ip 6to4** 告诉路由器使用 IPv6 包的目的地址来确定隧道对端的 IPv4 地址。对于本例来说，将前缀 2002::/16 后面跟随的 32 比特值 0a02:0202 用作传输协议的 IPv4 地址，及 10.2.2.2（如图 10-11 底部所示)。

因此，只要 IPv6 地址符合 2002: *ipv4-address* ::/48 格式要求，任何 IPv6 孤岛都能通过单个 6to4 隧道进行通信。

**6to4 隧道及环回接口**

除了使用物理接口的 IPv4 地址之外，还可以使用环回地址。使用环回地址的好处是，只要路由器处于正常运行状态,环回地址就可用。与物理接口的 IPv4 地址一样,环回接口的 IPv4 对隧道对端的路由器来说也必须可达。图 10-12 所示的拓扑结构与前面的案例相同，但此时使用的是环回接口（而不是面向提供商网络的物理接口）来确定网络的 IPv6 地址，环回地址被用作隧道源 IPv4 地址。例 10-15 给出了 R1 和 R2 将环回地址用作隧道源地址的配置示例。

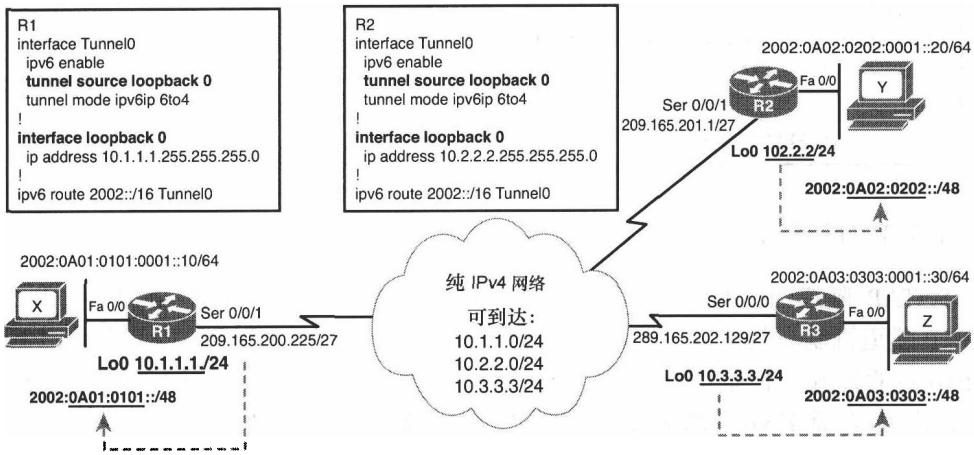



图 10-12　使用环回接口的 6to4 隧道

**例 10-15　使用环回接口在 R1 和 R2 上配置 6to4 隧道**

```
R1(config)# interface Serial0/0/1
R1(config-if)# ip address 209.165.200.225 255.255.255.224
R1(config-if)# exit
R1(config)# interface loopback0
R1(config-if)# ip address 10.1.1.1 255.255.255.0
R1(config)# interface FastEthernet0/0
R1(config-if)# ipv6 address 2002:A01:101:1::1/64
R1(config-if)# exit
```

```
R1(config)# interface Tunnel 0
R1(config-if)# ipv6 enable
R1(config-if)# tunnel source loopback0
R1(config-if)# tunnel mode ipv6ip 6to4
R1(config-if)# exit
```

```
R1(config)# ipv6 route 2002::/16 Tunnel 0
R2(config)# interface Serial0/0/1
R2(config-if)# ip address 209.165.201.1 255.255.255.224
R2(config-if)# exit
R2(config)# interface loopback0
R2(config-if)# ip address 10.2.2.2 255.255.255.0
R2(config)# interface FastEthernet0/0
R2(config-if)# ipv6 address 2002:A02:202:1::1/64
R2(config-if)# exit

R2(config)# interface Tunnel 0
R2(config-if)# ipv6 enable
R2(config-if)# tunnel source loopback0
R2(config-if)# tunnel mode ipv6ip 6to4
R2(config-if)# exit

R2(config)# ipv6 route 2002::/16 Tunnel 0
```

### 10.2.3 ISATAP

ISATAP 隧道的目的是在不具备纯 IPv6 基础设施的站点内部传输 IPv6 包。虽然 ISATAP 隧道机制与其他自动隧道机制（如 IPv6 6to4）相似，但 ISATAP 仅适用于站点内 IPv6 包的传输。不适用于站点间 IPv6 包的传输，ISATAP 使用明确定义的 IPv6 地址格式，由/64 单播 IPv6 前缀和 64 比特接口 ID（最后 32 比特是 IPv4 地址）组成。

与其他隧道技术相似，ISATAP 隧道可以存在于任何两台双栈设备之间：主机到路由器、路由器到路由器或主机到主机。不过，ISATAP 的主要功能是为双栈主机提供通过纯 IPv4 网络访问 IPv6 网络的能力。ISATAP 是一种 IPv6 过渡技术，允许 IPv4 网络中的主机与纯 IPv6 网络进行通信，这也是本章的主要关注点。

注：ISATAP 是一种访问 IPv6 资源的快速、临时性解决方案。如假设某栋写字楼内有 10 名开发人员需要访问 IPv6 资源，那么完全可以不必为整栋写字楼都部署 IPv6，而是可以利用 ISATAP 隧道技术为这些开发人员提供临时性的 IPv6 接入解决方案。

ISATAP 定义在 RFC 5214 “Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)”中。图 10-13 给出了双栈主机连接到双栈 ISATAP 路由器的示例拓扑结构。

通过图 10-13 提供的大量信息，大家可以对 ISATAP 有更深入的认识。

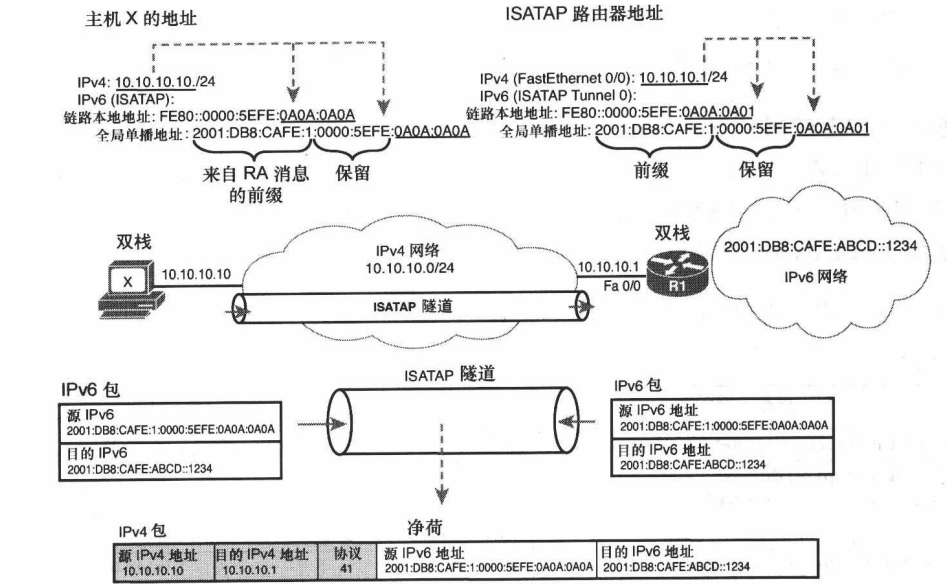



图 10-13　ISATAP 隧道

路由器 R1 是一台 ISATAP 路由器，隧道对端是一台双栈设备——主机 X。ISATAP 路由器利用 ISATAP 隧道机制为主机 X 提供经 IPv4 网络访问 IPv6 网络的能力。R1（ISATAP 路由器）利用路由器宣告消息为主机穿越 ISATAP 隧道提供相应的配置信息，这与纯 IPv6 网络中主机与路由器之间的 SLAAC 进程完全一样。下面将详细分析 ISATAP 的地址格式（如图 10-14 所示）。

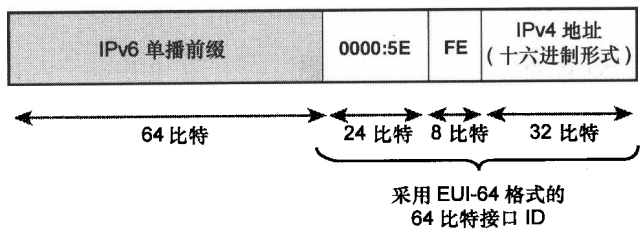



图 10-14　ISATAP 地址格式

ISATAP 地址包括以下内容。

- **前缀**：IPv6 单播前缀（64 比特）：可以是任何有效的单播前缀，包括全局可路由前缀、唯一本地前缀、链路本地前缀，甚至是 6to4 前缀。该前缀应该被选为网络编址方案的一部分。
- 接口 ID（EUI-64 格式）。
- **00-00-5E（24 比特）**：是 IANA 保留的以太网 OUI（Organizationally Unique Identifier，组织唯一标识符）。包括 ISATAP 在内的多种协议都要用到该 OUI。

该 OUI 用于 IP 地址与 IEEE 802 MAC 地址之间的动态映射。

- **FE（8 比特）**：表明该地址包含一个内嵌的 IPv4 地址。
- **IPv4 地址（32 比特）**：最后 4 字节是以十六进制表示的 IPv4 地址。

注：ISATAP 使用的 EUI-64 格式与第 4 章中讨论的 EUI-64 格式不同。那里的 EUI-64 是将 FFFE 插入以太网 MAC 地址的中间并反转 U/L 比特，从而得到 64 比特接口 ID。而 IETF 在 RFC 5342 “IANA & IETF Use of IEEE 802 Parameters” 中则将 EUI-64 格式定义为所有从 IANA 分配的 OUI 构建而成的地址。

表 10-5 列出了在路由器上配置 ISATAP 隧道的相关命令。

**表 10-5　　ISATAP 隧道配置命令**

| 命令 | 描述 |
|---|---|
| Router(config)# **interface tunnel** *tunnel-number* | 指定隧道接口和隧道号，并进入路由器的接口配置模式 |
| Router(config-if)# **ipv6-address** *ipv6-prefix/prefix-length* [ **eui-64** ] | 指定分配给接口的 IPv6 地址并在接口上启用 IPv6 处理能力<br>任何 IPv6 地址都可以，如果使用 eui-64 选项，那么将利用 ISATAP EUI-64 格式创建一个 IPv6 地址 |
| Router(config-if)# **no ipv6 nd suppress-ra** | 隧道接口在默认情况下是被禁止发送 IPv6 路由器宣告消息的，该命令的作用是启用 IPv6 路由器宣告消息以允许客户端自动配置，该命令也可以写成 **no ipv6 nd ra suppress** |
| Router(config-if)# **tunnel source** { *ip-address* \| *interface-type interface number* } | 为隧道接口指定源 IPv4 地址或源接口类型和接口号，源 IPv4 地址必须可以从隧道对端可达<br>如果指定了接口，那么该接口必须配置 IPv4 地址，地址可以是物理地址，也可以是环回地址，但必须可以从隧道对端可达 |
| Router(config-if)# **tunnel mode ipv6ip isatap** | 指定一个使用 ISATAP 地址的 IPv6 隧道 |

利用图 10-13 的示例拓扑结构，表 10-16 给出了将路由器 R1 配置为 ISATAP 路由器的配置示例。

**例 10-16　将 R1 配置为 ISATAP 路由器**

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 10.10.10.1 255.255.255.0
R1(config-if)# exit

R1(config)# interface tunnel 0
R1(config-if)# ipv6 address 2001:db8:cafe:1::/64 eui-64
R1(config-if)# no ipv6 nd suppress-ra
R1(config-if)# tunnel source fastethernet 0/0
R1(config-if)# tunnel mode ipv6ip isatap
R1(config-if)# exit

R1(config)# interface loopback 1
R1(config-if)# ipv6 address 2001:db8:cafe:abcd::1234/64
R1(config-if)#
```

下面是例 10-16 中的配置：

```
R1(config)# interface fastethernet 0/0
R1(config-if)# ip address 10.10.10.1 255.255.255.0
```

该地址是被主机 X 以及其他位于 10.10.10.0/24 LAN 上的客户端用作默认网关的 IPv4 地址，通常来自于 DHCP 服务器。

```
R1(config)# interface tunnel 0
```

这是用来经 IPv4 网络转发 IPv6 包的隧道接口。

```
R1(config-if)# ipv6 address 2001:db8:cafe:1::/64 eui-64
```

该命令的作用是在接口上启用 IPv6 处理功能并为隧道分配 IPv6 地址，任何 IPv6 地址都可以。由于本例使用了 **eui-64** 选项，因而使用 ISATAP EUI-64 格式创建了一个 IPv6 地址。

```
R1(config-if)# no ipv6 nd suppress-ra
```

默认情况下，Cisco IOS 在隧道接口上禁用 ICMPv6 的路由器宣告（RA）消息。该命令的作用是允许通过该隧道发送 RA 消息。主机可以利用这些 RA 消息自动配置其 IPv6 地址并收到 IPv6 默认网关地址。

```
R1(config-if)# tunnel source fastethernet 0/0
```

该命令的作用是将接口 Fast Ethernet 0/0 指定为隧道的源接口。该接口必须分配了 IPv4 地址，之前为该接口配置的 IPv4 地址为 10.10.10.1/24。

```
R1(config-if)# tunnel mode ipv6ip isatap
```

通过该命令，R1 的 Tunnel 0 被定义为 ISATAP 隧道，至此就可以通过 IPv4 网络传输 IPv6 包，并将 ISATAP 地址用作隧道的 IPv6 地址。

```
R1(config)# interface loopback 1
R1(config-if)# ipv6 address 2001:db8:cafe:abcd::1234/64
```

该命令是在 R1 上创建一个虚拟地址，以解释主机是如何到达远程 IPv6 网络的。该命令并不是专用于 ISATAP 配置的命令。

例 10-17 给出了多条用于验证 R1 地址配置的命令。命令 **show ip interface brief** 显示出接口 Fast Ethernet 0/0 静态配置的 IPv4 地址是 10.10.10.1。

例 10-17 中的命令 **show ipv6 interface brief** 显示了 R1 的 Tunnel 0 接口的 IPv6 地址。该 IPv6 地址采用了 ISATAP 格式，曾经在例 10-16 中通过命令 **ipv6 address 2001:db8:cafe:1::/64 eui-64** 配置了 Tunnel 0 的全部单播地址。该命令将前缀指定为 2001:db8:cafe:1。选项 **eui-64** 表示使用 ISATAP 的 EUI-64 格式来生成 64 接口 ID。前 32 比特是 ISATAP 保留的 OUI，然后是 FE 和 0000:5EFE，最后的低阶 32 比特是转化为十

六进制值 0A0A:0A01 的 IPv4 地址。

**例 10-17 验证 R1 的地址**

```
R1# show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            10.10.10.1      YES manual up                    up
Tunnel0                    unassigned      YES unset  up                    up
R1# show ipv6 interface brief
FastEthernet0/0            [up/up]
Tunnel0                    [up/up]
    FE80::5EFE:A0A:A01
    2001:DB8:CAFE:1:0:5EFE:A0A:A01
R1#

R1# show ipv6 interface tunnel 0
Tunnel0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::5EFE:A0A:A01
  No Virtual link-local address(es):
  Global unicast address(es):
    2001:DB8:CAFE:1:0:5EFE:A0A:A01, subnet is 2001:DB8:CAFE:1::/64 [EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF0A:A01
  MTU is 1480 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ICMP unreachables are sent
  ND DAD is not supported
  ND reachable time is 30000 milliseconds
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements live for 1800 seconds
  ND advertised default router preference is Medium
  Hosts use stateless autoconfig for addresses.
R1#
```

Tunnel 0 的链路本地地址也经历了同样的处理过程。链路本地地址使用保留的前缀 FE80::/10，加上相同的为全局单播地址创建的 ISATAP EUI-64 接口 ID：0000:5EFE:0A0A:0A01。命令 **show ipv6 interface tunnel 0** 显示了 Tunnel 0 的全局单播地址和链路本地地址（如例 10-17 所示）。

例 10-18 通过命令 **show interface tunnel 0** 验证了 Tunnel 0 接口的隧道类型是 ISATAP。

**例 10-18　验证 R1 的隧道协议**

```
R1# show interface tunnel 0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 10.10.10.1 (FastEthernet0/0), destination UNKNOWN
  Tunnel protocol/transport IPv6 ISATAP
<Rest of output omitted for brevity>
```

分析了 R1 的配置之后，接下来再讨论主机 X 是如何利用 ISATAP 隧道接收其 IPv6 配置信息的。与 6to4 隧道技术相似，ISATAP 也从 IPv4 地址自动创建其 IPv6 地址，不过 ISATAP 还使用了用于 SLAAC 的路由器宣告消息服务。

图 10-15 给出了主机 X 获取其 IPv6 全局单播地址、地址前缀和 IPv6 默认网关的过程。

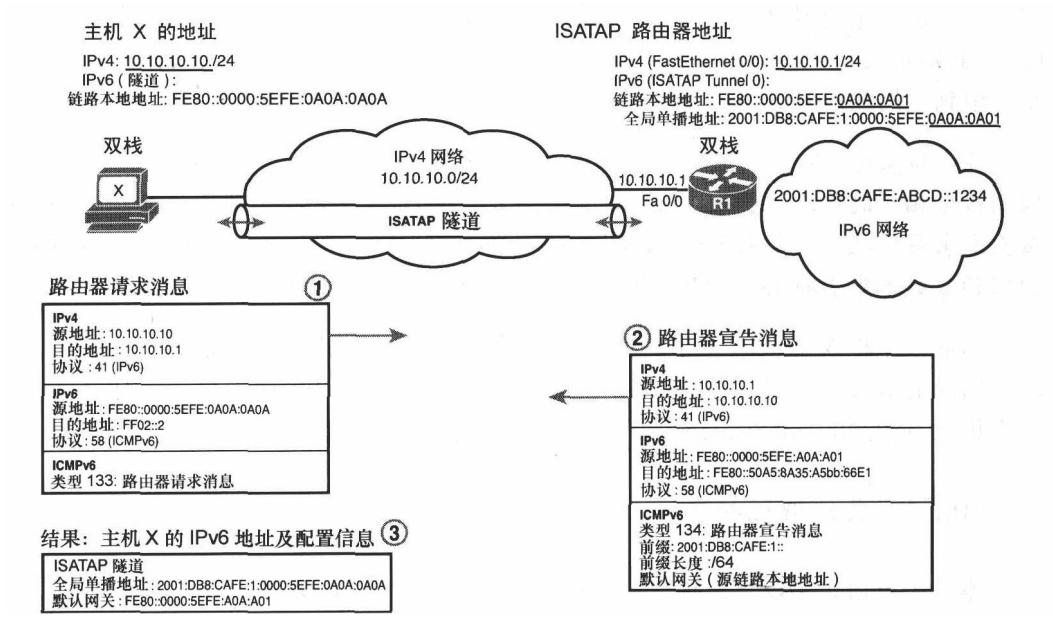



图 10-15　ISATAP 隧道上的路由器请求和路由器宣告消息

主机 X 配置（静态或动态）了 IPv4 地址 10.10.10.10/24 并启用了 ISATAP。包括 Windows XP、Windows Vista、Windows 7 和 Linux 在内的大多数常见操作系统都支持 ISATAP。主机 X 的 ISATAP 配置中包含了隧道对端的 IPv4 地址，对本例来说就是路由器 R1 的 IPv4 地址 10.10.10.1。主机 X 的接口利用 ISATAP 格式为其自动创建了一个 IPv6

链路本地地址，步骤如下。

**第 1 步**：主机 X 使用该链路本地地址经 ISATAP 隧道向全部路由器多播地址 FF02::2 发送一条路由器请求（RS）消息，RS 消息被封装在 IPv4 包中，并通过 ISATAP 隧道发送给路由器 R1。

**第 2 步**：R1 以包含了 IPv6 前缀、前缀长度和默认网关地址的路由器宣告（RA）消息进行响应，通过隧道将 ICMPv6 RA 消息发送给主机 X 之前还要将该消息封装到 IPv4 包中。

**第 3 步**：主机 X 已经有了创建可路由 IPv6 地址的相关信息并于 IPv6 默认网关进行通信。来自 R1 的路由器宣告消息包含了前缀和前缀长度 2001:DB8:CAFE:1::/64。利用 ISATAP 格式，主机 X 创建了 64 比特接口 ID，即 0000:5EFE 加上十六进制形式的 IPv4 地址 0A0A:0A0A，然后再加上前缀 2001:DB8:CAFE:1::/64 即可得到主机 X 的全局单播地址 2001:DB8:CAFE:1:0000:5EFE:0A0A:0A0A/64。

主机 X 的默认网关地址是 R1 接口 Fast Ethernet 0/0 的链路本地地址。该地址是 R1 用来向主机 X 发送路由器宣告消息的链路本地地址（如图 10-15 所示）。现在主机 X 的 ISATAP 隧道接口上就有了 IPv6 全局单播地址和 IPv6 默认网关地址。这样一来，主机 X 希望到达的任何 IPv6 网络都可以通过该隧道传输给路由器 R1。

注：如果要在客户端操作系统上启用 ISATAP，最好检查操作系统文档或者通过互联网检索相应的信息。对于 Windows 来说，需要在 DOS 命令提示符下使用 **netsh** 命令。截至本书写作之时，苹果的 MAC OS 还不支持 ISATAP，只是在 Mac OS 10.6.4 及以后的准预览版本中支持 ISATAP，包括 10.7.x(Lion)，但目前仍处于开发阶段。

至此，主机 X 已经获得了经过 IPv4 网络访问 IPv6 设备的全部信息。主机 X 有了其必需的 IPv6 编址信息，包括 IPv6 默认网关地址。图 10-13 的底部显示了 IPv6 数据包从主机 X 去往 2001:db8:cafe:abcd::1234 的封装过程。

### 10.2.4 其他隧道技术

本章主要讨论了以下三种通过 IPv4 承载 IPv6 包的隧道类型：

- 手工隧道；
- 6to4 隧道；
- ISATAP 隧道。

表 10-2 概况描述了一些常见的叠加隧道机制，包括前面已经讨论过的三种隧道。大家很容易迷失在这么多隧道和转换技术之中，因此全面掌握本章所讨论的这几种隧道机制将有助于学习可能需要在网络中用到的其他隧道技术，如 GRE、6RD 和 Teredo。

- GRE（Generic Routing Encapsulation，通用路由封装）隧道：类似于手工隧道，GRE 隧道也是一种点到点隧道，可以为两个端点之间提供安全通信能力，但是其承载的协议类型并不限于 IP。也就是说，GRE 允许包括 IP 协议在内的其他协议成为乘客协议，如 IS-IS。GRE 隧道的配置也与手工隧道相似，GRE 定义在 RFC 2784“Generic Routing Encapsulation (GRE)”中。GRE 在隧道化过程中增加了额外的报头。GRE 是 Cisco IOS **tunnel mode** 命令的默认隧道类型。

当 IPv4 报头的协议字段值为 47 时，表明其数据部分是 GRE 数据包。

- 6RD（IPv6 Rapid Deployment，IPv6 快速部署）隧道：6RD 隧道是 6to4 的扩展，6RD 允许 ISP 或服务提供商通过其 IPv4 网络为客户提供 IPv6 服务。6RD 与 6to4 隧道之间的主要区别如下：
- 6RD 不需要地址使用前缀 2002::/16，因而前缀可以来自于服务提供商自己的地址块；
- 不是所有的 32 比特 IPv4 目的地址都需要承载在 IPv6 净荷报头中，IPv4 目的地址来源于净荷报头中的部分比特以及路由器中配置的信息。

6RD 定义在 RFC 5569“IPv6 Rapid Deployment on IPv4 Infrastructures (6rd)”中。

- Teredo 隧道（RFC 4380）：Teredo 也被称为 shipworm（蛀船虫）。当某台双栈设备位于 IPv4 NAT 之后时，可以利用 Teredo 隧道在两台双栈设备之间发送 IPv6 包。Teredo 使用 UDP 端口号 3544 与 Teredo 服务器（充当 Teredo 客户端与 Teredo 中继的调度机）进行通信。虽然 Teredo 是由 Microsoft 提出的，但是 Linux 也有一个被称为 Miredo 的版本。Teredo 定义在 RFC 4380“Teredo:Tunneling IPv6 over UDP through Network Address Translations (NATs)”中。

有关 GRE、6RD 以及 Teredo 的内容都超出了本书写作范围，详细信息可参考：

- 由 Regis Desmeules 编写的 *Cisco Self-Study: Implementing Cisco IPv6 Networks*
- Implementing Tunneling for IPv6: www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-tunnel.html。

## 10.3　本章小结

本章详细讨论了从 IPv4 向 IPv6 迁移过程中使用的双栈技术和三种常见的隧道技术。双栈设备完全支持 IPv4 和 IPv6。理解 IPv6 协议及其细微差别是一件望而生畏的事情，而双栈设备则允许大家在继续保持稳定的 IPv4 基础设施的同时逐渐熟悉 IPv6。

本章还讨论了隧道或叠加隧道，这是实现 IPv4 设备与 IPv6 设备共存的另一种方式。隧道机制通过将 IPv6 包封装在 IPv4 包中，来实现 IPv6 设备之间经纯 IPv4 网络进行通信。在纯 IPv6 全面部署之前，隧道技术被认为是一种临时性的解决方案。

本章讨论的第一类隧道是手工隧道。手工隧道是点到点隧道，需要在路由器上静态配置 IPv4 源地址和目的地址。虽然手工隧道配置简单，但是需要在每两个端点之间都配置一条独立的隧道，因而手工隧道无法满足大型网络的扩展性需求。

本章讨论的第二类隧道是 6to4 隧道。6to4 隧道是点到多点隧道，隧道对端的 IPv4 地址可以从 IPv6 包的目的地址自动生成。IPv6 网络地址是由可达的 IPv4 地址加上一个特殊的保留给 6to4 隧道的前缀 2002::/16，32 比特的 IPv4 地址被用作 IPv6 地址的低阶比特，从而得到网络前缀为 2002: *ipv4-address* ::/48。利用该编址格式，单个 6to4 隧道可以到达任意数量的 2002: *ipv4-address* ::/48 网络。与手工隧道不同，不需要为 6to4 隧道配置目的 IPv4 地址。

本章讨论的最后一类隧道是 ISATAP 隧道，主要用于站点内部的隧道化。ISATAP 的一个主要特点就是允许主机通过 SLAAC 机制，利用 ISATAP 路由器发送的路由器宣告（RA）消息来获取其 IPv6 地址配置信息。IPv6 前缀源自 RA 消息包含的前缀信息，地址的 64 比特接口 ID 是 0000:5EFE 加上十六进制形式的 32 比特 IPv4 地址。与 6to4 隧道相似，也不需要为 ISATAP 隧道配置目的 IPv4 地址。

## 10.4 参考文献

RFC：

RFC 2784, Generic Routing Encapsulation (GRE) , D. Farinacci, Procet Networks, www.ietf.org/rfc/rfc2784 , March 2000

RFC 3056, Connection of IPv6 Domains via IPv4 Clouds (6to4) , B. Carpenter, www.ietf.org/rfc/rfc3056 , February 2001

RFC 3493, Basic Socket Interface Extensions for IPv6 , R. Gilligan, Intransa, Inc., www.ietf.org/rfc/rfc3493 , February 2003

RFC 3986, Uniform Resource Identifier (URI): Generic Syntax , T. Berners-Lee, W3C/MIT, www.ietf.org/rfc/rfc3986 , January 2005

RFC 4380, Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs) , C. Huitema, Microsoft, www.ietf.org/rfc/rfc4380 , February 2006

RFC 4966, Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status , C. Aoun, Energize Urnet, www.ietf.org/rfc/rfc4966 , July 2007

RFC 4472, Operational Considerations and Issues with IPv6 DNS , A. Duran, Comcast, www.ietf.org/rfc/rfc4472 , April 2006

RFC 5214, Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) , F. Templin, Boeing Phantom Works, www.ietf.org/rfc/rfc5214.txt , March 2008
RFC 5342, IANA & IETF Use of IEEE 802 Parameters , D. Eastlake, Eastlake Enterprises, www.ietf.org/rfc/rfc5342.txt , September 2009
RFC 5569, IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) , R. Despres, RD-IPtech, www.ietf.org/rfc/rfc5569 , January 2010

网站：

Implementing Tunneling for IPv6, www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-tunnel.html

# 第11章 NAT64

如前所述，在可预见的未来，IPv4 和 IPv6 将长期共存。企业何时和如何迁移到 IPv6 取决于各自的特殊需求。许多服务提供商和大型企业网都是 IPv6 的早期部署者，他们在几年前就开始规划、培训并制定相应的迁移策略。由于 IPv4 地址空间的匮乏，许多企业（特别是在亚洲和欧洲）都在积极部署 IPv6。RIPE NCC 通过网站 www.ipv6actnow.org/statistic 提供了 IPv6 部署策略建议。Google 也在互联网上收集了大量有关 IPv6 部署的相关统计信息，并通过 www.google.com/intl/en/ipv6/statistics 不断更新。

理想情况下，IPv6 应该尽可能运行在纯 IPv6 网络环境中。IPv6 设备通过 IPv6 网络进行相互通信。但是，从 IPv4 向 IPv6 迁移的过程是逐渐进行的，需要很长的时间。与此同时，IETF 制定了多种过渡技术来满足各种 IPv4 向 IPv6 迁移的应用场景，包括双栈、隧道和转换技术，上一章已经讨论了其中的双栈和隧道技术。

本章将讨论以下两类转换技术：

- NAT64（Network Address Translation IPv6 to IPv4，从 IPv6 到 IPv4 的网络地址转换）；
- NAT-PT（Network Address Translation-Protocol Translation，网络地址转换-协议转换）。

NAT（Network Address Translation，网络地址转换）是 IPv4 中一种常见的地址转换技术，通常在私有地址（RFC 1918）与公有 IPv4 地址空间之间进行转换。很明显，NAT64 是一种在纯 IPv6 网络与纯 IPv4 网络之间提供接入能力的转换技术。AFT（Address Family Translation，地址族转换）或者简单的转换机制为纯 IPv6 与纯 IPv4 主机和网络提供了通信能力。AFT 负责在这两者网络层协议之间执行 IP 报头和地址转换功能。与其他过渡技术一样，转换技术也不是一种长期策略，其最终目标也是构建纯 IPv6 网络。与隧道技术相比，转换技术具有以下两个重要优点：

- 转换技术提供了一种渐进式、无缝式的 IPv6 迁移机制；
- 内容提供商可以向 IPv6 互联网用户透明地提供服务。

NAT64 是 NAT-PT 的替代技术，具体情况定义在 RFC 6144“Framework for IPv4/IPv6 Translation”中。Cisco 建议不再使用 NAT-PT，而是支持其替代技术 NAT64。为了参考和连续性起见，本章也在后面讨论了 NAT-PT 技术，截至本书写作之时，NAT-PT 仍是部分 Cisco 课程的内容之一，虽然该技术最终必然过时。目前，与 NAT64 相比，Cisco 平台对 NAT-PT 的支持程度更为广泛，不过这种状况很快就会发生变化。

由于 NAT-PT 与 DNS 之间的紧耦合关系以及转换时的种种限制，使得 IETF 决定正式废止该技术，其原因详见 RFC 4966“Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status”，并将 NAT64 列为 NAT-PT 的替代技术。

## 11.1 NAT64

NAT64 是 IPv4 到 IPv6 的过渡机制，也是一种 IPv4 与 IPv6 共存机制，与 DNS64 一起，NAT64 的主要目的是允许纯 IPv6 客户端向纯 IPv4 服务器发起通信过程（如图 11-1 所示）。利用静态或手工绑定机制，NAT64 也允许纯 IPv4 客户端向纯 IPv6 服务器发起通信过程。本章将逐一讨论这两种应用场景。

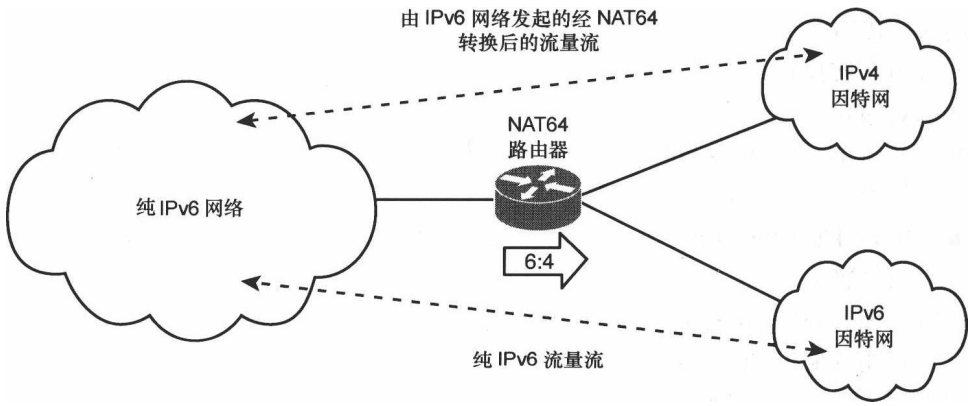



图 11-1 纯 IPv6 网络访问 IPv4 和 IPv6 因特网

NAT64 定义在 RFC 6146“Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers.”中。与 IPv4 使用的状态化 NAT 相似，状态化 NAT64 在执行协议转换的同时会创建或修改 IPv6 与 IPv4 之间的绑定关系。NAT64 通过以下操

作来完成 IPv6 与 IPv4 之间的转换。

- 使用 RFC 6145“IP/ICMP Translation Algorithm”定义的算法来转换两种协议的 IP 报头。
- 使用 RFC 6052“IPv6 Addressing of IPv4/IPv6 Translators”定义的算法来转换这两种协议的 IP 地址。

注：还有一种无状态 NAT64，与状态化 NAT64 不同，无状态 NAT64 在执行协议转换的同时，不需要维护任何形式的绑定关系或会话状态。本章仅讨论状态化 NAT64。

NAT64 包括以下 3 类组件。

- **NAT64 前缀**：为了通过纯 IPv6 网络传输数据包而与转化后的 IPv4 地址一起使用的任何/32、/40、/48、/56、/64 或/96 前缀，NAT64 前缀可以是 NSP（Network-Specific Prefix，特定网络前缀）或 WKP（Well-Known Prefix，周知前缀）。NSP 是由组织结构自行分配的前缀，通常是来自于组织机构自有 IPv6 前缀的子网。用于 NAT64 的 WKP 是 64:ff9b::/96。如果没有指定或配置 NSP，那么 NAT64 就将 WKP 附加到转化后的 IPv4 地址上，NAT64 前缀也被称为 Pref64::/n。
- **DNS64 服务器**：DNS64 服务器不但为 IPv6 AAAA 记录完成普通的 DNS 服务器功能，而且还要在 AAAA 记录不可用时试图定位 IPv4 A 记录。定位了 A 记录之后，DNS64 会利用 NAT64 前缀将 IPv4 A 记录转换为 IPv6 AAAA 记录，使得纯 IPv6 主机认为其可以通过 IPv6 与纯 IPv4 服务器进行通信。
- **NAT64 路由器**：NAT64 路由器将 NAT64 前缀宣告到纯 IPv6 网络中，并在纯 IPv6 网络与纯 IPv4 网络之间执行转换操作。

### 11.1.1 从纯 IPv6 客户端向纯 IPv4 服务器发送流量

在图 11-2 给出的应用场景中，位于纯 IPv6 网络 2001:DB8:CAFE::/48 中的客户端利用 NAT64 机制与纯 IPv4 服务器进行通信。这是 NAT64 最常见的应用场景，相应的通信过程如下。

**第 1 步**：主机 A 是一台纯 IPv6 主机，希望与服务器 www.example.com 进行通信，因而向 DNS64 服务器发起 DNS 查询（AAAA: www.example.com）。DNS64 是该进程的关键部件，DNS64 服务器是同时支持 IPv4 和 IPv6 的双栈 DNS 服务器，其作用是让 IPv6 客户端认为可以通过 IPv6 地址到达 IPv4 服务器。

图 11-3 更加详细地解释了 DNS64 操作中的第 1 步到第 7 步，这些步骤与图 11-2 中的步骤完全相同，利用这两张图可以更好地理解 DNS64 处理过程。

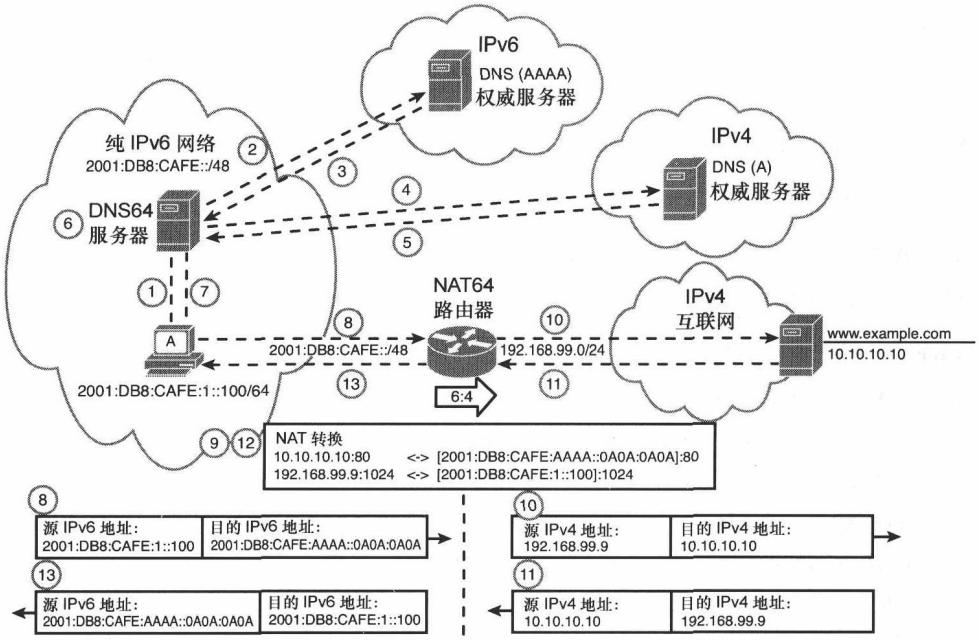



图 11-2 NAT64 路由器将 IPv6 流量转换为 IPv4 并负责返回流量

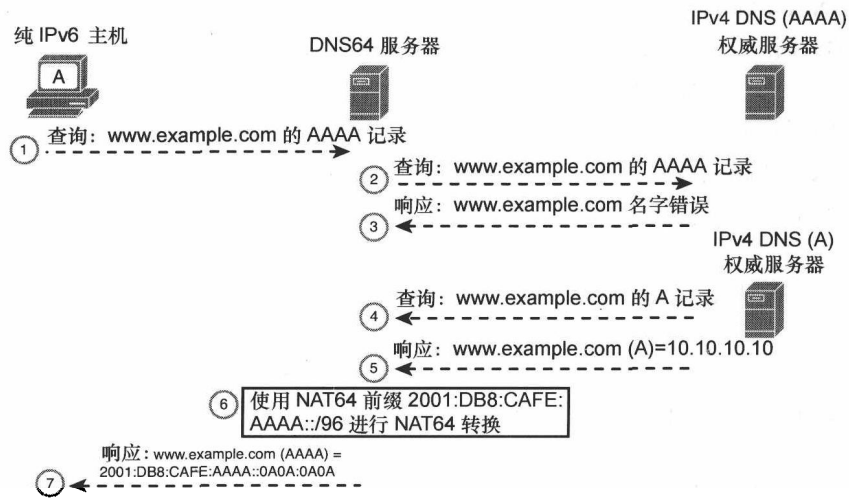



图 11-3 DNS64 操作

主机 A 向 DNS64 服务器发起 DNS 查询（AAAA: www.example.com）。对于主机 A 来说，这就是一条普通的查询 IPv6 服务器的 DNS AAAA 查询。

**第 2 步**：DNS64 服务器收到主机发送的 DNS AAAA 查询请求。为了解析该域名，DNS64 服务器会向 DNS AAAA 权威服务器发送关于 www.example.com 的查询请求。

**第 3 步**：IPv6 DNS AAAA 权威服务器返回响应，表明没有关于 www.example. com

的 AAAA 记录。

**第 4 步**：DNS64 服务器收到该 AAAA 查询的空白答案（名字错误）响应后，会向 IPv4 DNS A 权威服务器发送 A 查询（A: www.example.com）。

**第 5 步**：IPv4 DNS A 权威服务器有关于 www.example.com 的 A 记录，因而向 DNS 64 服务器返回一条包含了该服务器 IPv4 地址的响应消息（A: www.example.com 10.10.10.10）。

**第 6 步**：DNS64 服务器从 IPv4 DNS A 权威服务器收到 IPv4 地址后，将该 A 记录合成为 AAAA 记录，即使用 NAT64 前缀 2001:DB8:CAFE:1234::/96 并附加到转化为十六进制值 0A0A:0A0A 的 IPv4 地址上。

图 11-4 解释了 DNS64 将 IPv4 A 记录合成为 IPv6 AAAA 记录的过程。DNS64 服务器收到 www.example.com 的 IPv4 A 记录后，将 IPv4 地址转化为十六进制 0A0A:0A0A，然后将前缀 2001:DB8:CAFE:AAAA::/96 放到 0A0A:0A0A 之前，就可以得到 DNS64 合成后的关于 www.example.com 的 AAAA 资源记录 2001:DB8:CAFE:AAAA::0A0A:0A0A。该地址会被主机 A 用作前往 www.example.com 服务器的目的 IPv6 地址。

注：DNS 服务器不会自动执行 DNS64 服务，DNS 服务器必须支持 DNS64 并且被明确配置为执行 DNS64 操作才行。

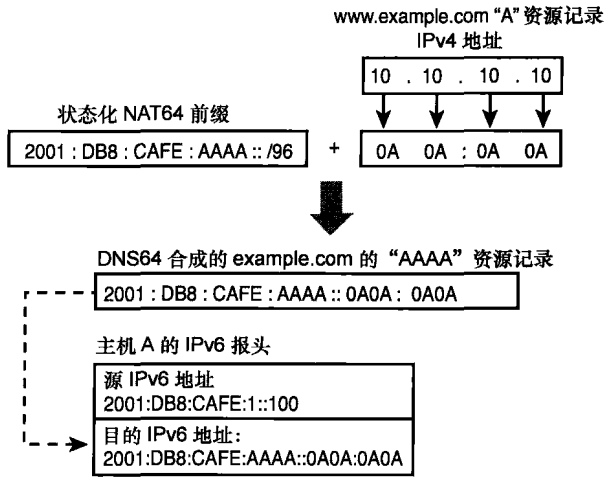



图 11-4 DNS64 将 A 记录合成为 AAAA 记录

注：DNS64 前缀必须是 NSP 或 WKP。

**第 7 步**：DNS64 服务器向主机 A 发送携带了 www.example.com 的 IPv6 地址 2001:DB8:CAFE:AAAA::0A0A:0A0A 的 AAAA 记录响应。

**第 8 步**：合成的 AAAA 记录对主机 A 来说完全透明。对主机 A 来说，看起来就像

可以通过 IPv6 网络和互联网到达 www.example.com 一样，此时主机 A 已经有了将 IPv6 包传输到 www.example.com 的编址信息。

- IPv6 目的地址：2001:DB8:CAFE:AAAA::0A0A:0A0A；
- IPv6 源地址：2001:DB8:CAFE:1::100。

**第 9 步：** NAT64 路由器在其启用了 NAT64 功能的接口上收到主机 A 发来的 IPv6 包。NAT64 路由器配置了状态化 NAT64 前缀 2001:DB8:CAFE:AAAA::/96。NAT64 路由器会试图转发该 IPv6 包。如果该 IPv6 包的目的地址的前 96 比特与已配置的状态化 NAT64 前缀不匹配，那么就会使用常规的 IPv6 路由对该 IPv6 包进行无协议翻译的转发。如果与状态化 NAT64 前缀匹配，那么就会对该 IPv6 包执行以下协议转换操作。

- 将 IPv6 报头转换为 IPv4 报头。
- 将 IPv6 目的地址转换为 IPv4 地址，首选移除该 IPv6 目的地址中的状态化 NAT64 前缀 2001:DB8:CAFE:AAAA::/96，然后再将剩余的 32 比特 0A0A:0A0A 转化为点分十进制形式的 IPv4 地址 10.10.10.10。
- 利用已配置的 IPv4 地址池将 IPv6 源地址转化为 IPv4 地址，根据 NAT64 的配置情况，可以是 1:1 的地址转换，也可以使用重叠型的 IPv4 地址，这一点与 IPv4 NAT 相似。对于本例来说，主机 A 的源 IPv6 地址被转换为 IPv4 地址 192.168.99.9。
- 为源地址和目的地址创建状态化 NAT64 IP 地址转换状态（如图 11-2 NAT64 路由器下面的 NAT 转换表所示）。在对数据包执行协议转换的第一时间就要创建这些状态。路由器通过维护这些状态来为数据流中的后续数据包提供服务。当流量结束或状态维护定时器到期后，该状态就会终止。

**注：** 本案例使用了私有地址空间，但是在实际网络环境中，为了确保可达性，必须使用公有可达的 IPv4 地址。因此，如果只有有限的公有可达的 IPv4 地址池，那么该方案可能不是最佳解决方案。

**第 10 步：** 完成 NAT64 转换后，利用常规的 IPv4 路由查找进程即可转发转换后的 IPv4 包。对本例来说，使用 IPv4 目的地址 10.10.10.10 来转发该 IPv4 包。

**第 11 步：** 地址为 10.10.10.10 的 www.example.com 服务器做出应答，最终 NAT64 路由器收到该应答。

**第 12 步：** NAT64 路由器在其启用了 NAT64 功能的接口上收到来自 www.exam ple.com 服务器的 IPv4 包后，会检查该 IPv4 包以确定是否存在该 IPv4 目的地址的 NAT64 转换状态。如果不存在这样的转换状态，那么就丢弃该数据包。如果存在该 IPv4 目的地址的转换状态，那么 NAT64 路由器就执行以下操作。

- 将 IPv4 报头转换为 IPv6 报头。

- 利用已有的 NAT64 转换状态将 IPv4 源地址转换为 IPv6 源地址。对于本例来说，源地址从 IPv4 地址 10.10.10.10 转换为 IPv6 地址 2001:DB8:CAFE:AA AA::0A0A:0A0A，目的地址则从 IPv4 地址 192.168.99.9 转换为 IPv6 地址 2001:DB8:CAFE:1::100。

**第 13 步**：完成以上协议转换后，即可利用常规的 IPv6 路由查找进程转发该 IPv6 包。

DNS64 服务器与 NAT64 路由器结合，为需要与 IPv4 服务器进行通信的纯 IPv6 客户端创建了一个透明的传输环境。DNS64 服务器让 IPv6 主机误以为 IPv4 目的端是一个 IPv6 地址。DNS64 服务器既可以充当典型的 IPv6 DNS 服务器，也可以充当使用 NAT64 前缀的 DNS64 转换器。

### 11.1.2 配置

表 1-1 列出了上述应用场景中将路由器配置为执行 NAT64 功能的基本命令。NAT64 的配置存在多种选项和应用场景，有关状态化 NAT64 的详细配置信息请参考 Stateful Network Address Translation 64：www.cisco.com/en/US/docs/ios-xml/ios/ipaddr_nat/ configuration/xe-3s/iadnatstateful-nat64.pdf。有关无状态 NAT64 的详细配置信息请参考 Stateless Network Address Translation 64：www.cisco.com/en/US/docs/ios-xml/ios/ipaddr_nat/configuration/xe-3s/iadnat-stateless-nat64.pdf。

表 11-1 NAT64 配置命令

| 命令 | 描述 |
|---|---|
| Router(config)# **interface** *type number* | 指定接口类型和接口号并进入路由器的接口配置模式，该接口是面向纯 IPv6 网络的接口，需要配置 IPv6 地址 |
| Router(config-if)# **ipv6 address** *ipv6-address/prefix-length* | 指定分配给该接口的 IPv6 地址和前缀长度 |
| Router(config-if)# **nat64 enable** | 在接口上启用 NAT64 翻译功能 |
| Router(config)# **interface** *type number* | 指定接口类型和接口号并进入路由器的接口配置模式，该接口是面向纯 IPv4 网络的接口，需要配置 IPv4 地址 |
| Router(config-if)# **ip address** *ipv4-address subnet-mask* | 指定分配给该接口的 IPv4 地址和子网掩码 |
| Router(config-if)# **nat64 enable** | 在接口上启用 NAT64 转换功能 |
| Router(config)# **nat64 prefix stateful** *ipv6-prefix/prefix-length* | 为状态化 NAT64 定义前缀和前缀长度：<br>- *ipv6-prefix*：包含在路由器宣告消息中的 IPv6 网络地址，必须采用 RFC 2363 规定的表达形式，即十六进制形式的以冒号分隔的 16 比特值；<br>- */prefix-length*：IPv6 前缀的长度，十进制值表示地址的多少个高阶连续比特是前缀（地址的网络部分），十进制值之前必须有一条斜线 |
| Router(config)# **nat64 v4 pool** *pool-name start-address-range end-address-range* | 启用 NAT64 IPv4 配置：<br>- **pool**：配置 IPv4 地址池；<br>- *pool-name*：IPv4 地址池的名字；<br>- *start-address-range*：地址池空间的起始地址；<br>- *end-address-range*：地址池空间的结束地址 |

续表

| 命令 | 描述 |
| --- | --- |
| Router(config)# **nat64 v6v4 list** *list-name* **pool** *pool-name* [ **overload** ] | 将 IPv6 源地址转换为 IPv4 源地址，并将 IPv4 目的地址转换为 IPv6 目的地址：<br>- **list**：将 IPv4 地址池与过滤机制相关联，以决定何时执行 IPv6 地址映射；<br>- *list-name*：IPv6 访问列表的名字；<br>- **pool**：指定用于动态地址映射的 NAT64 池；<br>- *pool-name*：NAT64 池的名字；<br>- **overload**：（可选）启用 NAT64 超量地址转换 |
| Router(config)# **ipv6 access-list** *access-list-name* | 定义 IPv6 ACL 并进入 IPv6 访问列表配置模式<br>*access-list-name*：指定 IPv6 ACL 的名字 |
| Router(config-ipv6-acl)# **ipv6 permit** *ipv6-address/ipv6-prefix-length* | 指定将要转换的 IPv6 地址和前缀长度 |

例 11-1 给出了 NAT64 路由器的配置并按照配置顺序列出了相应的配置命令，但没有显示与路由和重分发等配置命令。

### 例 11-1 NAT64 配置示例

```
NAT64-Router(config)# interface GigabitEthernet 0/0/1
NAT64-Router(config-if)# description Connected to IPv6 Network
NAT64-Router(config-if)# ipv6 address 2001:DB8:CAFE:1::1/64
NAT64-Router(config-if)# nat64 enable
NAT64-Router(config-if)# exit
NAT64-Router(config)# interface GigabitEthernet 0/0/2
NAT64-Router(config-if)# description Connected to IPv4 Network
NAT64-Router(config-if)# ip address 192.168.99.1 255.255.255.0
NAT64-Router(config-if)# nat64 enable
NAT64-Router(config-if)# exit

NAT64-Router(config)# nat64 prefix stateful 2001:DB8:CAFE:AAAA::/96
NAT64-Router(config)# nat64 v4 pool pool1 192.168.99.9 192.168.99.10
NAT64-Router(config)# nat64 v6v4 list mylist pool1 overload
NAT64-Router(config)# ipv6 access-list mylist

NAT64-Router(config-ipv6-acl)# permit ipv6 2001:DB8:CAFE::/48 any
```

```
NAT64-Router(config)# interface GigabitEthernet 0/0/1
```

该接口是连接纯 IPv6 网络的接口。

```
NAT64-Router(config-if)# ipv6 address 2001:DB8:CAFE:1::1/64
```

该命令为接口配置 IPv6 全局单播可路由地址。

```
NAT64-Router(config-if)# nat64 enable
```

该命令在 IPv6 接口上启用无状态 NAT64 协议翻译功能。

```
NAT64-Router(config)# interface GigabitEthernet 0/0/2
```

该接口是连接 IPv4-only 网络的接口。

```
NAT64-Router(config-if)# ip address 192.168.99.1 255.255.255.0
```

该命令为接口配置 IPv4 地址。

```
NAT64-Router(config-if)# nat64 enable
```

该命令在 IPv4 接口上启用无状态 NAT64 协议转换功能。

```
NAT64-Router(config)# nat64 prefix stateful 2001:DB8:CAFE:AAAA::/96
```

该命令启用 NAT64 的 IPv6-to-IPv4 地址映射，将使用 NAT64 前缀 2001:DB8:CAFE:AAAA::/96 并附加一个 IPv4 地址。

```
NAT64-Router(config)# nat64 v4 pool pool1 192.168.99.9 192.168.99.10
```

该命令定义状态化 NAT64 IPv4 地址池，这些地址都是用于 NAT64 转换的 IPv4 地址。

```
NAT64-Router(config)# nat64 v6v4 list mylist pool1 overload
```

该命令将 IPv4 源地址动态转换为 IPv6 源地址，并将 IPv6 目的地址转换为 IPv4 目的地址。参数 **overload** 的作用是启用 NAT64 超量地址转换功能（类似于 IPv4 的超量 NAT）。

```
NAT64-Router(config)# ipv6 access-list mylist
```

该命令定义了一个 IPv6 访问列表并进入 IPv6 访问列表配置模式。

```
NAT64-Router(config-ipv6-acl)# permit ipv6 2001:DB8:CAFE::/48 any
```

该命令指定将要转换的 IPv6 地址和前缀长度。

### 11.1.3 从纯 IPv4 客户端向纯 IPv6 服务器发送流量

在图 11-5 给出的应用场景中，位于纯 IPv4 网络的客户端利用 NAT64 与纯 IPv6 服务器进行通信（该操作过程的第 1 步到第 9 步将在后面详细描述），目的是为 IPv4 客户端透明地提供 IPv6 服务。对本应用场景来说，DNS64 服务器不是必需的，在 NAT64 路由器上配置了 IPv6 地址与 IPv4 地址之间的静态映射。

注：该应用场景在可预见的未来都是不可能出现的，绝大多数具备 IPv6 功能的服务器一般也都具备 IPv4 功能，最可能的情况就是 IPv6 服务器在相当长的时期内都运行双栈，虽然最终都将是纯 IPv6 服务器，但是目前离这一天还很遥远。

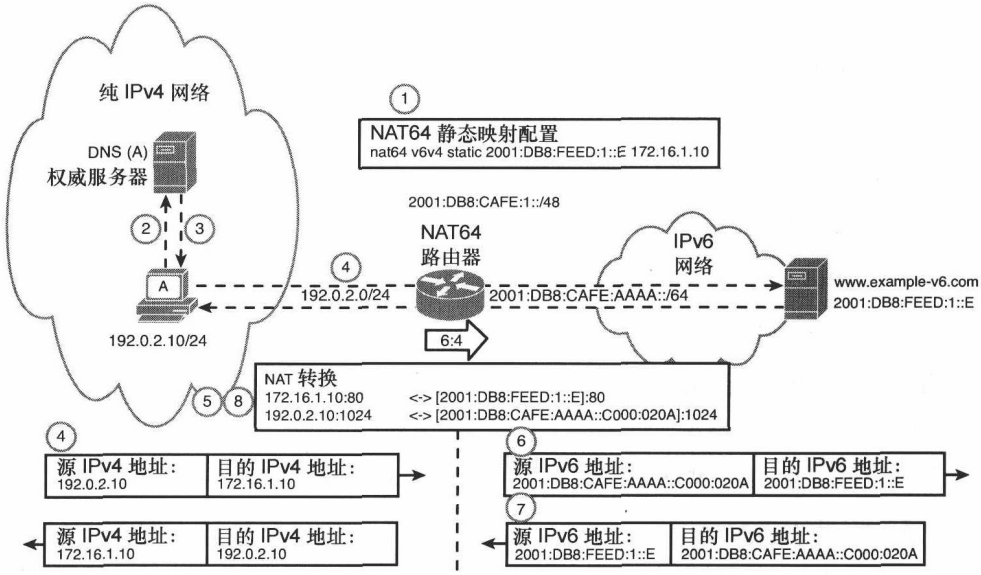



图 11-5 NAT64 路由器将 IPv4 流量转换为 IPv6 并负责返回流量

**第 1 步**：首先配置 IPv6-to-IPv4 静态映射，为 IPv4 地址 172.16.1.10 的主机提供对 IPv6 服务器 2001:DB8:FEED:1::E 的访问功能。此外，还需要在 DNS 服务器上将 IPv4 地址 172.16.1.10 注册为 www.example-v6.com 的 DNS 资源记录。使用以下命令即可创建静态 NAT64 映射。

```
NAT64-Router(config)# nat64 v6v4 static 2001:DB8:FEED:1::E 172.16.10
```

**第 2 步**：主机 A 是一台纯 IPv4 主机，希望与 www.example-v6.com 进行通信，因而向其 IPv4 DNS 权威服务器发送 DNS 查询（A: www.example-v6.com）。

**第 3 步**：DNS 服务器做出响应，将 www.example-v6.com 的 A 资源记录 172.16.1.10 发送给主机 A。

**第 4 步**：主机 A 此时就有了将 IPv4 包发送给 www.example-v6.com 所需的寻址信息：

- IPv4 目的地址：172.16.1.10；
- IPv4 源地址：192.0.2.10。

**第 5 步**：NAT64 路由器在其启用了 NAT64 功能的接口上收到 IPv4 包后，执行以下操作。

- 将 IPv4 报头转换为 IPv6 报头。
- 利用第 1 步状态配置中已经创建的 NAT64 转换状态将 IPv4 目的地址转换为 IPv6 目的地址，也就是将 IPv4 目的地址 172.16.1.10 转换为 IPv6 目的地址 2001:DB8:FEED:1::E。
- 通过将状态化 NAT64 前缀 2001:DB8:CAFE:AAAA::/96 附加到 IPv4 地址之前，将 IPv4 源地址转换为 IPv6 源地址，得到的 IPv6 源地址为 2001:DB8:CAFE:AAAA:: C000:020A（C000:020A 是 192.0.2.10 的十六进制形式）。

**第 6 步**：完成上述协议转换后，就可以采用常规的 IPv6 路由进程转发该 IPv6 包，最终将该 IPv6 包路由到位于 2001:DB8:FEED:1::E 的 www.example-v6.com 服务器。

**第 7 步**：服务器 www.example-v6.com 向主机 A 发送响应包。

**第 8 步**：NAT64 路由器在其启用了 NAT64 功能的接口上收到该 IPv6 服务器发送的 IPv6 响应包后，执行以下操作。

- 将 IPv6 报头转换为 IPv4 报头。
- 利用 NAT64 静态映射，将 IPv6 源地址 2001:DB8:FEED:1::E 转换为 IPv4 源地址 172.16.10。
- 通过移除状态化 NAT64 前缀 2001:DB8:CAFE:AAAA::/96，将 IPv6 目的地址转换为 IPv4 目的地址，IPv6 地址的低阶 32 比特 C000:020A 表示为点分十进制形式即为 IPv4 地址 192.0.2.10。

**第 9 步**：完成上述协议转移后，NAT64 路由器就采用常规的 IPv4 路由进程将该数据包转发到 192.0.2.10。

与上一个应用场景相似，本应用场景利用状态化 NAT64 也为纯 IPv4 主机与纯 IPv6 服务器之间提供了透明通信能力。两种应用场景的配置很相似，主要的区别在于第 1 步所说的静态映射命令。对于由纯 IPv6 客户端向 IPv4 服务器发起的通信过程中，DNS64 服务器扮演着非常重要的角色，但是对于本应用场景来说则不是如此。本场景需要的是在 NAT64 路由器上配置 IPv6-to-IPv4 地址的静态映射。

## 11.2 NAT-PT

本节将讨论 NAT-PT 机制。NAT-PT 类似于 IPv4 使用的 NAT 机制，NAT 的作用是将私有 IPv4 地址（RFC 1918）转换为公有 IPv4 地址。反之亦然，而 NAT-PT 则负责将 IPv4 地址转换为 IPv6 地址或者将 IPv6 地址转换为 IPv4 地址。本节将重点讨论静态和动态 NAT-PT 的配置。

**注**：IPv4 NAT 也被称为 NAPT（Network Address and Port Translation，网络地址和端口转换）、PAT（端口地址转换）、地址超量（address overloading）或者仅仅简称为 NAT。无论使用哪个名称，IPv4 NAT 的功能都是将一个 IPv4 地址转换为另一个 IPv4 地址。目前，为了与涉及 IPv6 地址转换的 NAT64 或 NAT-PT 相区分，人们也将 IPv4 NAT 称为 NAT44。

NAT-PT 适用于纯 IPv6 网络中的设备与纯 IPv4 网络进行通信的应用场景（如图 11-6 所示）。NAT-PT 不是双栈和隧道等过渡机制的替代技术，而是在没有其他方法可用的情况下，才使用 NAT-PT 技术为纯 IPv6 设备提供与纯 IPv4 网络进行通信的能力。

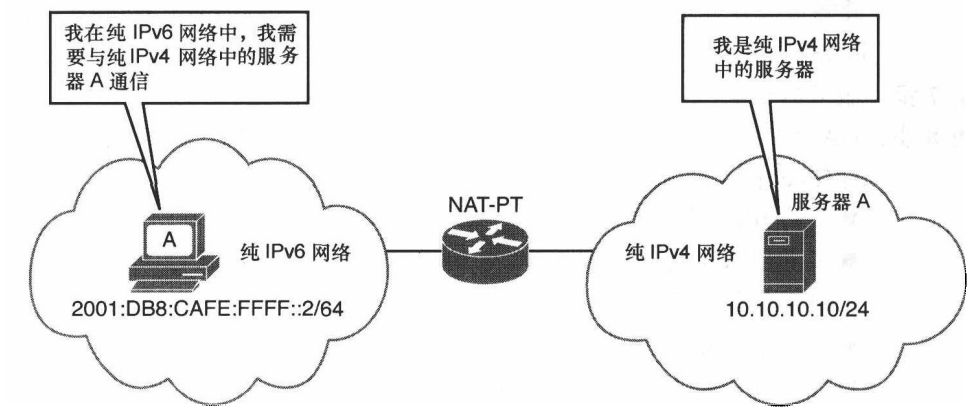



图 11-6 纯 IPv4 与纯 IPv6 网络中的设备需要进行通信

NAT-PT 定义在 RFC 2766“Network Address Translation - Protocol Translation，NAT-PT”中。NAT-PT 使用的转换算法定义在 RFC 6145“IP/ICMP Translation Algorithm”中。SIIT（Stateless IP/ICMP Translation，无状态 IP/ICMP 转换）算法负责 IPv4 报头与 IPv6 报头之间的转换（包括 ICMP 报头）。NAT-PT 路由器负责完成双向的源地址、目的地址和其他三层字段的转换，由于 NAT-PT 源于 IPv4 NAT，因而也具有第 1 章所说的各种应用限制。虽然目前 NAT-PT 仍然被用作两种协议之间的转换机制，但是 RFC 2766 已经被 RFC 4966“Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status”所废止了。只是包括 Cisco 在内的很多厂商还都支持 NAT-PT，也仍然认为 NAT-PT 是一种可选的转换技术。有关 NAT-PT 的局限性将在本章后面进行讨论。

### 11.2.1 应用层网关

在向 IPv6 迁移的过渡阶段，纯 IPv6 网络中的设备可能需要与纯 IPv4 网络中的设

备进行通信。NAT-PT 用来允许这些设备在网络层进行通信——负责在 IPv6 地址与 IPv4 地址之间进行转换。顾名思义，ALG（Application Level Gateway，应用层网关）运行在 OSI 参考模型的应用层。由于 NAT-PT 不分析 IP 包的净荷，对应用不感知，因而 NAT-PT 有时可能需要 ALG，才能实现 IPv6 设备上的 IPv6 应用与 IPv4 设备上的 IPv4 应用进行通信，反之亦然。此时，ALG 可以与 NAT-PT 协同工作，来满足应用之间通信需求。

如前所述，纯 IPv6 与纯 IPv4 网络中的设备可能需要进行通信。例如，IPv6 客户端可能需要通过 HTTP 连接到 IPv4 Web 服务器，或者 IPv6 客户端可能需要利用 SMTP（Simple Mail Transfer Protocol，简单邮件传输协议）向纯 IPv4 邮件服务器发送电子邮件。

DNS 就是一个需要 ALG 的例子，既可以在 NAT-PT 路由器上配置 DNS ALG，也可以将 DNS ALG 作为一台单独的服务器。例 11-7 解释了 IPv6 客户端与 IPv4 服务器通过 DNS ALG 进行通信时的 DNS 处理过程，并详细列出了相应的处理步骤（第 1 步～第 5 步）。

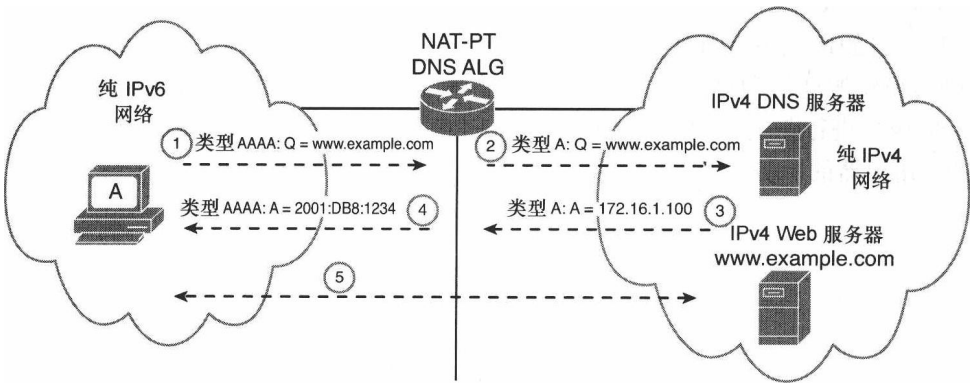



图 11-7　DNS 应用层网关

**第 1 步**：主机 A 发起 AAAA 记录的 DNS 查询，以查找 www.example.com 的 IPv6 地址。DNS 查询被转发给 NAT-PT 路由器，然后又由 NAT-PT 路由器进行协议转换后转发给 IPv4 DNS 服务器。有关 NAT-PT 的操作将在下一小节进行讨论。

**第 2 步**：NAT-PT/ALG 路由器转换该 AAAA 记录之后，将该 DNS 查询以 A 记录形式转发给 IPv4 DNS 服务器（仍然是查询 www.example.com）。相应的源 IPv6 地址和目的 IPv6 地址都被 NAT-PT/ALG 路由器转换为 IPv4 地址。

**第 3 步**：IPv4 DNS 服务器对该查询做出响应，将携带 IPv4 地址（172.16.1.100）的 DNS 响应返送给 NAT-PT/ALG 路由器。

**第 4 步**：NAT-PT/ALG 路由器将该 DNS 响应从 A 记录转换为 AAAA 即可，并将 DNS 响应中的 IPv4 地址更改为 IPv6 地址（2001:DB8:1234）。

**第 5 步**：至此主机 A 就知道了与 www.example.com 进行通信的 IPv6 地址，NAT-PT/ALG 路由器负责将 IPv6 地址转换为 IPv4 地址，反之亦然。

有关应用层网关的细节以及 Cisco 路由器的实现情况已超出了本书写作范围，详细信息请参考 Cisco 白皮书 *Network Address Translator - Protocol Translator*：www.cisco.com/en/US/prod/collateral/iosswrel/ps8802/ps6969/ps1835/prod_white_paper09186a008011ff51_ps6640_Products_White_Paper.html。

### 11.2.2 使用 NAT-PT

NAT-PT 包括网络层地址转换和协议转换。NAT-PT 对所有需要由 NAT-PT 路由器转换的 IPv6 流量使用已配置的 96 比特前缀，所有需要由 NAT-PT 路由器转换为 IPv4 地址的 IPv6 流量都将该/96 前缀用作自己的目的地址。图 11-8 给出了一个纯 IPv6 网络示例 2001:DB8:CAFE::/48。所有流量都被转发给路由器 R1（是 NAT-PT 路由器），R1 收到 IPv6 目的地址中带有指定前缀 2001:DB8:FEED::/96 的 IPv6 数据包后，会根据映射规则转换将源地址和目的地址转换为 IPv4 地址。由于前缀 2001:DB8:FEED::/96 应该在 IPv6 域中进行路由，因而带有网络 2001:DB8:CAFE::/48 的所有路由器都可以使用 NAT-PT 路由器的服务。

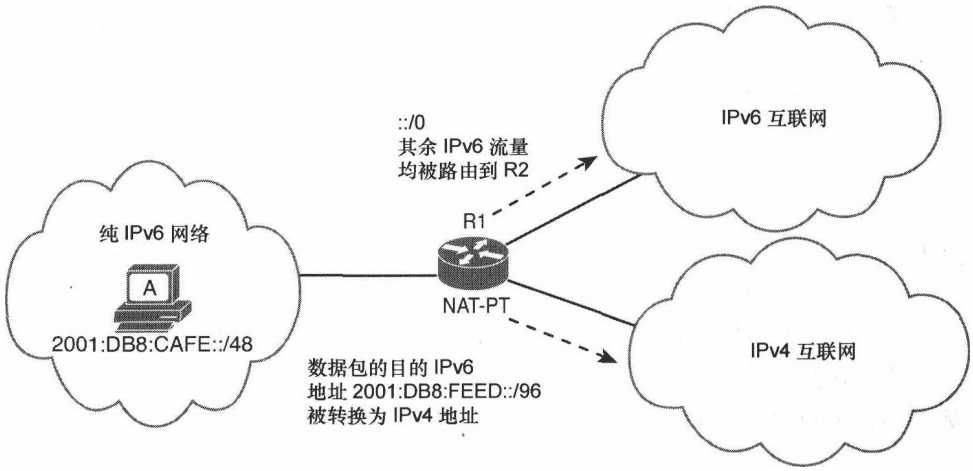



图 11-8 执行 NAT-PT 服务的路由器 R1

图 11-9 给出了另一种应用场景，此时的 NAT-PT 路由器与纯 IPv6 转发路由器完全独立，所有目的地址包含该前缀的 IPv6 数据包都被发送给 NAT-PT 路由器 R1 并被转换为 IPv4 地址，而其他 IPv6 数据包则作为纯 IPv6 包被转发给路由器 R2 进行转发。

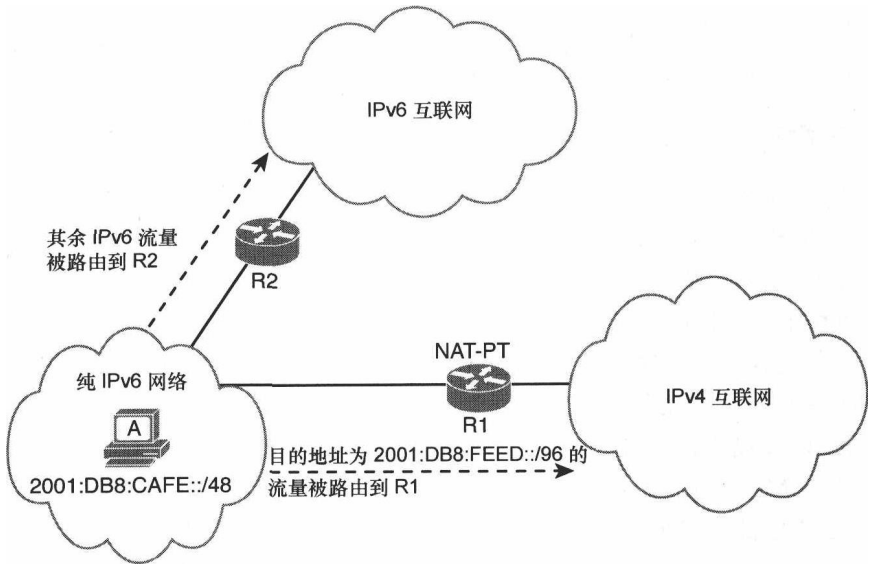



图 11-9 NAT-PT 路由器与 IPv6 路由器分离

NAT-PT 包括以下类型。

- **静态 NAT-PT**：静态 NAT-PT 使用静态转换规则将 IPv6 地址转换为 IPv4 地址，这种一对一的 IPv6 地址与 IPv4 地址映射关系都是静态配置在 NAT-PT 路由器上的，对需要进行通信的纯 IPv6 主机与纯 IPv4 主机来说，需要为每台主机都配置静态 NAT-PT 映射。当应用或服务器需要访问稳定的 IPv4 地址的时候（如访问外部 IPv4 DNS 服务器），这种静态 NAT-PT 是非常有用的。
- **动态 NAT-PT**：动态 NAT-PT 映射虽然也提供类似于静态 NAT-PT 的一对一映射，但是动态 NAT-PT 使用的是 IPv4 地址池。地址池中的源 IPv4 地址数量决定了能够同时进行的 IPv6-to-IPv4 并发转换数量。
- **带有端口地址转换的 NAT-PT**：也被称为 NAPT-PT（Network Address Port Translation - Protocol Translation，网络地址端口转换-协议转换），可以提供多对一的动态映射功能，也就是可以将多个 IPv6 地址映射为单个 IPv4 地址。通过在不同端口号上的复用机制，单个 IPv4 地址可以同时用于多个会话，从而将多个 IPv6 用户与单个 IPv4 地址相关联。类似于 NAT-PAT 机制，NAPT-PT 也使用了 TCP/UDP 端口转换技术，可以通过指定接口或地址池来实现 NAPT-PT。
- **带有 IPv4 映射操作的 NAT-PT**：动态 NAT-PT 可以与 DNS ALG 协同使用。在路由器上配置了 IPv4 映射之后，当 DNS ALG 的 IPv4 地址被转换为 IPv6 地址时，就会处理该 IPv6 地址，并且来自 IPv4 网络的 DNS 包就会得到已被转换

为 IPv6 网络的 ALG。然后由 NAT-PT 路由器将 DNS AAAA 记录查询转换为 A 记录查询，并以 A 记录作为 AAAA 记录的响应。

注：本章仅讨论静态和动态 NAT-PT 的配置。

NAT-PT 存在很多衍生于 IPv4 NAT 的限制条件。如前所述，NAT-PT 应该仅应用于无其他解决方案的场合。RFC 4966 “Reasons to Move the Network Address Translator - Protocol Translator [NAT-PT] to Historic Status” 提到 “这些问题非常严重，以至于建议将 RFC 2766 作为一种通用过渡机制也完全不适用，本文档建议 IETF 应该将 RFC 2766 “NAT-PT” 从建议标准归类为历史状态”，因此，RFC 2766 目前已被 “废止”。在部署 NAT-PT 时存在的限制和约束情形如下。

- **ALG 支持有限**：NAT-PT 提供了有限的 ALG 支持能力，对 ALG 的支持仅限于 ICMP、FTP 和 DNS。
- **无端到端的安全性**：与 IPv4 NAT 一样，NAT-PT 也有同样的局限性，不能提供端到端的安全性。由于 NAT-PT 会在转换过程中修改数据包的报头，会使 IP 报头检查失败，因而传输层和应用层安全机制对携带了 IP 地址的应用程序来说是不可能实现的，这是 NAT 技术本身的局限性。需要使用 IPSec 网络层安全的端节点必须支持 IPv4 或 IPv6，但无需同时支持 IPv4 和 IPv6。
- **单点故障**：NAT-PT 路由器是网络中的单点故障源。
- **不支持多播**：NAT-PT 无法处理多播流量。
- 不支持 PMTUD（Path MTU Discovery，路径 MTU 发现）：PMTUD 无法与 NAT-PT 协同工作。
- **不支持 CEF（Cisco Express Forwarding，Cisco 快速转发）**：NAT-PT 不支持 CEF，使用 NAT-PT 时必须关闭 IPv4 CEF 和 IPv6 CEF；
- **可用性有限**：NAT-PT 仅可用于低端平台。

### 11.2.3 静态 NAT-PT

静态 NAT-PT 映射是 IPv6 地址与 IPv4 之间的一对一映射。图 11-10 给出了示例拓扑结构以及这两种协议之间的静态映射关系，随后将解释相关配置情况。NAT-PT 路由器配置了 NAT 前缀 2001:DB8:FEED::/96，所有需要发送到纯 IPv4 网络的 IPv6 包都必须在其 IPv6 目的地址中使用前缀 2001:DB8:FEED::/96。NAT-PT 路由器会将目的 IPv6 地址中包含该前缀的数据包根据已配置的映射关系转换为相应的 IPv4 地址。

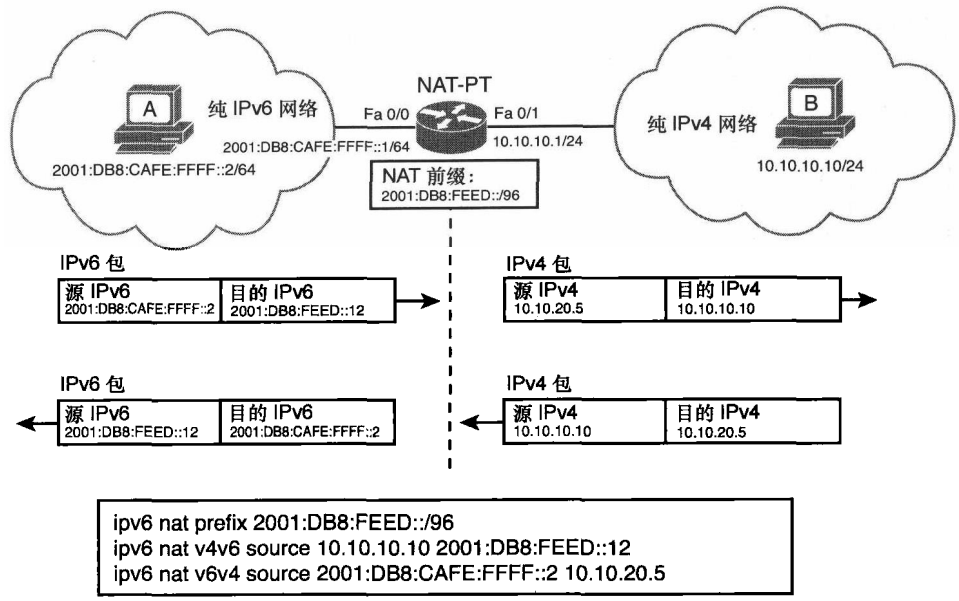



图 11-10 静态 NAT-PT 映射

表 11-2 给出了在路由器上配置 NAT-PT 的相关命令。

**表 11-2 静态 NAT-PT 配置命令**

| 命令 | 描述 |
|---|---|
| Router(config)# **interface** *type number* | 指定接口类型和接口号并进入路由器的接口配置模式，该接口是面向纯 IPv6 网络的接口，需要配置 IPv6 地址 |
| Router(config-if)# **ipv6 address** *ipv6-address/prefix-length* | 指定分配给该接口的 IPv6 地址和前缀长度 |
| Router(config-if)# **ipv6 nat** | 在接口上启用 NAT-PT 转换功能 |
| Router(config)# **interface** *type number* | 指定接口类型和接口号并进入路由器的接口配置模式，该接口是面向纯 IPv4 网络的接口，需要配置 IPv4 地址 |
| Router(config-if)# **ip address** *ipv4-address subnet-mask* | 指定分配给该接口的 IPv4 地址和子网掩码 |
| Router(config-if)# **ipv6 nat** | 在接口上启用 NAT-PT 转换功能 |
| Router(config)# **ipv6 nat prefix** *ipv6-prefix/prefix-length* | 为 IPv6 域定义用做 NAT-PT 前缀的 IPv6 前缀，所有需要 NAT-PT 路由器进行转换的去往纯 IPv4 网络的数据包都要使用该前缀，唯一支持的前缀长度是 96 |
| Router(config)# ipv6 **nat v4v6 source** *ipv4-address ipv6-address* | 数据包的源 *ipv4-address* 会被转换为源 *ipv6-address*，*ipv6-address* 是 IPv6 主机到达 IPv4 主机的目的 IPv6 地址 |
| Router(config)# **ipv6 nat v6v4 source** *ipv6-address ipv4-address* | 数据包的源 *ipv6-address* 会被转换为源 *ipv4-address*，*ipv4-address* 是 IPv4 主机到达 IPv6 主机的目的 IPv4 地址 |

理解这些配置命令的最好方式就是分析图 11-10 的配置情况。例 11-2 给出了 NAT-PT 路由器的完整配置，并且后面还列出了相应的命令解释和配置步骤。

### 例 11-2 静态 NAT-PT 配置

```
NAT-PT-Router(config)# no ip cef
NAT-PT-Router(config)# no ipv6 cef
NAT-PT-Router(config)# interface FastEthernet0/0
NAT-PT-Router(config-if)# ipv6 address 2001:DB8:CAFE:FFFF::1/64
NAT-PT-Router(config-if)# ipv6 nat
NAT-PT-Router(config-if)# exit
NAT-PT-Router(config)# interface FastEthernet0/1
NAT-PT-Router(config-if)# ip address 10.10.10.1 255.255.255.0
NAT-PT-Router(config-if)# ipv6 nat
NAT-PT-Router(config-if)# exit

NAT-PT-Router(config)# ipv6 nat prefix 2001:DB8:FEED::/96
NAT-PT-Router(config)# ipv6 nat v4v6 source 10.10.10.10 2001 :DB8 :FEED::12
NAT-PT-Router(config)# ipv6 nat v6v4 source 2001:DB8:CAFE:FFFF::2 10.10.20.5
```

```
NAT-PT-Router(config)# no ip cef
```

该命令禁用 IPv4 CEF。

```
NAT-PT-Router(config)# no ipv6 cef
```

该命令禁用 IPv6 CEF。

```
NAT-PT-Router(config)# interface FastEthernet0/0
```

该接口是路由器将要启用 NAT-PT 功能的 IPv6 接口。

```
NAT-PT-Router(config-if)# ipv6 address 2001:DB8:CAFE:FFFF::1/64
```

该命令为接口配置 IPv6 全局单播可路由地址。

```
NAT-PT-Router(config-if)# ipv6 nat
```

该命令在 IPv6 接口上启用 NAT-PT 功能。

```
NAT-PT-Router(config)# interface FastEthernet0/1
```

该接口是路由器将要启用 NAT-PT 功能的 IPv4 接口。

```
NAT-PT-Router(config-if)# ip address 10.10.10.1 255.255.255.0
```

该命令配置路由器的 IPv4 地址。

```
NAT-PT-Router(config-if)# ipv6 nat
```

该命令在 IPv4 接口上启用 NAT-PT 功能。

```
NAT-PT-Router(config)# ipv6 nat prefix 2001:DB8:FEED::/96
```

该命令配置 NAT-PT 前缀，所有目的 IPv6 地址为 2001:DB8:FEED::/96 的数据包都被转换为 IPv4 包（包括源地址和目的地址）。真实地址使用 **ipv6 nat v4v6 source** 和 **ipv6 nat v6v4 source** 命令进行静态配置。与此相反，所有去往 IPv6 网络的 IPv4 数据包的源地址都要被转换为 IPv6 地址。

```
NAT-PT-Router(config)# ipv6 nat v4v6 source 10.10.10.10 2001:DB8:FEED::12
```

路由器收到源 IPv4 地址为 10.10.10.10 的数据包后，会将其源地址转换为 IPv6 地址 2001:DB8:FEED:::12。

路由器收到目的 IPv6 地址为 2001:DB8:FEED::12 的数据包后，会将目的地址转换为 IPv4 地址 10.10.10.10。

```
NAT-PT-Router(config)# ipv6 nat v6v4 source 2001:DB8:CAFE:FFFF::2 10.10.20.5
```

路由器收到源 IPv6 地址为 2001:DB8:CAFE:FFFF::2 的数据包后，会将源地址转换为 IPv4 地址 10.10.20.5。

路由器收到目的 IPv4 地址为 10.10.20.5 的数据包后，会将目的地址转换为 IPv6 地址 2001:DB8:CAFE:FFFF::2。

下面通过主机 A 向主机 B 发起 ping 操作来测试上述配置（如例 11-3 所示）。命令 **debug ipv6 nat** 可以验证图 11-10 中的地址转换情况。

**例 11-3 验证静态 NAT-PT 转换**

```
PCA> ping 2001:DB8:FEED::12

Pinging 2001:DB8:FEED::12 from 2001:db8:ffff::2 with 32 bytes of data:

Reply from 2001:DB8:FEED::12: time=1ms
Reply from 2001:DB8:FEED::12: time=1ms
Reply from 2001:DB8:FEED::12: time=1ms
Reply from 2001:DB8:FEED::12: time=1ms

Ping statistics for 2001:DB8:FEED::12:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 1ms, Average = 1ms

PCA>
```

```
NAT-PT-Router# debug ipv6 nat
IPv6 NAT-PT debugging is on
*Apr  1 21:36:28.631: IPv6 NAT: icmp src (2001:DB8:CAFE:FFFF::2) ->
(10.10.20.5), dst (2001:DB8:FEED::12) -> (10.10.10.10)
```

```
*Apr  1 21:36:28.631: IPv6 NAT: icmp src (10.10.10.10) ->
(2001:DB8:FEED::12), dst (10.10.20.5) -> (2001:DB8:CAFE:FFFF::2)
NAT-PT-Router#
```

主机 A 发送源 IPv6 地址为 2001:DB8:CAFE:FFFF::2 且目的 IPv6 地址为 2001:DB8:FEED::12 的 ICMPv6 回显请求消息。由于所有目的地址前缀为 2001:DB8:FEED::/96 的数据包都要被路由到 NAT-PT 路由器（对本例来说，主机 A 与路由器位于同一条链路上），因而 NAT-PT 路由器收到该 IPv6 包后，发现目的地址 2001:DB8:FEED::12 与已配置的 NAT 前缀 2001:DB8:FEED::/96 相匹配，因而使用其静态 **ipv6 nat v6v4 source** 映射表项，即 NAT-PT 路由器将源 IPv6 地址 2001:DB8:CAFE:FFFF::2 转换为源 IPv4 地址 10.10.20.5。利用静态 **ipv6 nat v4v6 source** 映射表项，目的 IPv6 地址 2001:DB8:FEED::12 被转换为目的 IPv4 地址 10.10.10.10。

主机 B 收到 ICMPv4 回显请求消息后以 ICMPv4 回显应答消息进行响应。主机 B 发送源 IPv4 地址为 10.10.10.10 且目的 IPv4 地址为 10.10.20.5（是 ICMPv4 回显请求消息的源 IPv4 地址）的 IPv4 包。

NAT-PT 路由器收到来自主机 B 的 ICMPv4 回显应答消息后，执行反向的 NAT-PT 转换操作，利用 **ipv6 nat v4v6 source** 命令将源 IPv4 地址 10.10.10.10 转换为源 IPv6 地址 2001:DB8:FEED::12，利用 **ipv6 nat v6v4 source** 命令将目的 IPv4 地址转换为 2001:DB8:CAFE:FFFF::2。例 11-3 通过命令 **debug ipv6 nat** 验证了上述转换结果。

大家可能会疑惑为何 10.10.20.5 会被用作 **ipv6 nat v4v6 source** 命令中的 IPv4 地址。该地址与 NAT-PT 路由器的 IPv4 地址 10.10.10.1/24 不在同一个子网上。如果我们的映射使用了与路由器接口位于同一个子网上的 IPv4 地址，那么主机 B 在向主机 A 发送数据包的时候，就能识别出这两个地址都是位于同一子网上的主机。主机 B 在发送回显应答响应消息之前，需要目的 IPv4 地址（是回显请求消息的源 IPv4 地址）的 MAC 地址。如果该地址位于主机 B 的子网中，那么就会发送 ARP 请求，但是由于该地址不是路由器的接口地址，因而路由器不会发送 ARP 应答消息，此时就会出现问题。相反，如果使用位于不同子网中的 IPv4 地址 10.10.20.5，那么当主机 B 需要将回显请求消息封装到以太网帧中时，需要的就是其默认网关 10.10.10.1（NAT-PT 路由器）的 MAC 地址，而不是数据包的目的 IPv4 地址的 MAC 地址。

例 11-4 给出了命令 **show ipv6 nat translation** 和 **show ipv6 nat statistics** 的输出结果。其中，命令 **show ipv6 nat translation** 用于验证路由器上所有处于激活状态的 NAT-PT 转换情况，而命令 **show ipv6 nat statistics** 则用于显示与转换相关的统计信息。

例 11-4 命令 show ipv6 nat translation

```
NAT-PT-Router# show ipv6 nat translation
Prot  IPv4 source              IPv6 source
      IPv4 destination         IPv6 destination
---   ---                      ---
      10.10.10.10              2001:DB8:FEED::12

---   10.10.20.5               2001:DB8:CAFE:FFFF::2
      10.10.10.10              2001:DB8:FEED::12

---   10.10.20.5               2001:DB8:CAFE:FFFF::2
      ---                      ---

NAT-PT-Router# show ipv6 nat statistics

Total active translations: 2 (2 static, 0 dynamic; 0 extended)
NAT-PT interfaces:
  FastEthernet0/0, FastEthernet0/1, NVI0
Hits: 4  Misses: 0
Expired translations: 0

NAT-PT-Router#
```

为 NAT-PT 选择/96 前缀时，一定要考虑该前缀的可达性。该前缀对于纯 IPv6 网络中的其他路由器来说必须是可达的。通常 NAT-PT 路由器会使用 OSPFv3 或 IPv6 EIGRP 等 IGP 协议来宣告该前缀。

### 11.2.4 动态 NAT-PT

静态 NAT-PT 适用于 IPv6-to-IPv4 映射数量有限且地址稳定（不变）的应用场景。而动态 NAT-PT 则是从地址池中分配 IPv4 地址，类似于动态 IPv4 NAT。

对于动态 NAT-PT 来说，NAT-PT 路由器收到目的 IPv6 地址使用了已分配/96 NAT 前缀的数据包后，会执行与静态 NAT-PT 相同的处理进程。只是此时不再通过静态配置的 IPv4 地址来转换 IPv6 地址，而是通过 IPv4 地址池来转换 IPv6 地址。

配置动态 NAT-PT 的命令与配置动态 IPv4 NAT 的命令相似，表 11-3 给出了与配置动态 NAT-PT 相关的命令信息。

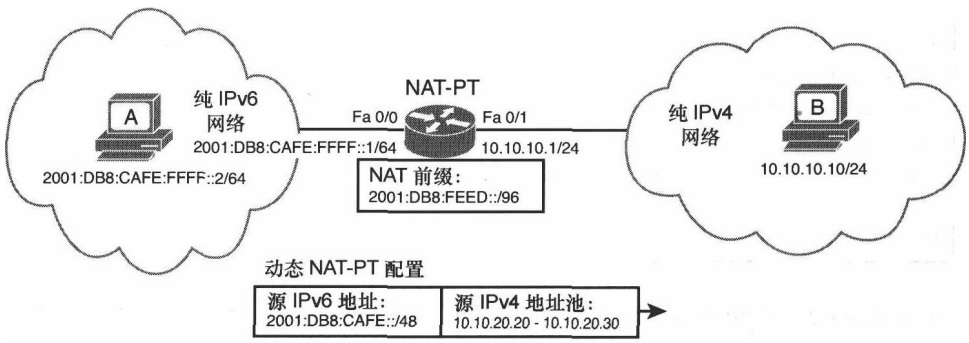
表 11-3 动态 NAT-PT 配置命令

| 命令 | 描述 |
| --- | --- |
| Router(config)# **ipv6 access-list** *ipv6-acl-name* | 定义 IPv6 访问列表并进入路由器的 IPv6 访问列表配置模式 |

续表

| 命令 | 描述 |
| --- | --- |
| Router(config-ipv6-acl)# **permit** *source-ipv6-address/prefix-length destination-ipv6-address/prefix length* | 定义可以被转换的 IPv6 地址区间，在 ACL 后面有一个隐式的 **deny any any** 语句 |
| Router(config)# **ipv6 nat v6v4 source list** *ipv6-acl-name* pool *ipv4-pool-name* | 在允许的 IPv6 源地址与 IPv4 地址池之间定义动态映射，*ipv6-acl-name* 的作用是标识用于确定需要转换的 IPv6 地址的 IPv6 ACL，*ipv4-pool-name* 的作用是标识用于转换的 IPv4 地址池 |
| Router(config)# **ipv6 nat v6v4 pool** *ipv4-pool-name ipv6-acl-name startipv4-address end-ipv4-address* | 为转换定义源 IPv4 地址池 |

图 11-11 给出了动态 NAT-PT 的示例拓扑结构。




```
ipv6 access-list cafe-only
     permit 2001:db8:cafe::/48 any
ipv6 nat v6v4 source list cafe-only pool ipv4-pool
ipv6 nat v6v4 pool ipv4-pool 10.10.20.20 10.10.20.30 prefix-length 24
ipv6 nat v4v6 source 10.10.10.10 2001:DB8:FEED::12
```

图 11-11 动态 NAT-PT 映射

例 11-5 给出了动态 NAT-PT 路由器的完整配置信息。

### 例 11-5 动态 NAT-PT 配置

```
NAT-PT-Router(config)# interface FastEthernet0/0
NAT-PT-Router(config-if)# ipv6 address 2001:DB8:CAFE:FFFF::1/64
NAT-PT-Router(config-if)# ipv6 nat
NAT-PT-Router(config-if)# exit
NAT-PT-Router(config)# interface FastEthernet0/1
NAT-PT-Router(config-if)# ip address 10.10.10.1 255.255.255.0
NAT-PT-Router(config-if)# ipv6 nat
NAT-PT-Router(config-if)# exit
```

```
NAT-PT-Router(config)# ipv6 nat prefix 2001:DB8:FEED::/96
NAT-PT-Router(config)# ipv6 nat v4v6 source 10.10.10.10 2001:DB8:FEED::12

NAT-PT-Router(config)# ipv6 access-list cafe-only
NAT-PT-Router(config-ipv6-acl)# permit 2001:db8:cafe::/48 any
NAT-PT-Router(config-ipv6-acl)# exit
NAT-PT-Router(config)# ipv6 nat v6v4 source list cafe-only pool ipv4-pool
NAT-PT-Router(config)# ipv6 nat v6v4 pool ipv4-pool 10.10.20.20 10.10.20.30 prefix-
length 24
NAT-PT-Router(config)# end
NAT-PT-Router#
```

可以看出，接口命令、**ipv6 nat v4v6** 命令以及 **ipv6 nat prefix 2001:DB8:FEED::/96** 等命令与前面的静态 NAT-PT 配置完全一样，不过动态 NAT-PT 配置中没有使用命令 **ipv6 nat v6v4**，而是代之以下述命令：

```
NAT-PT-Router(config)# ipv6 access-list cafe-only
```

该命令的作用是定义 IPv6 访问列表 **cafe-only**。

```
NAT-PT-Router(config-ipv6-acl)# permit 2001:db8:cafe::/48 any
```

仅允许 NAT-PT 路由器转换源 IPv6 地址包含前缀 2001:db8:cafe::/48 的数据包。

```
NAT-PT-Router(config)# ipv6 nat v6v4 source list cafe-only pool ipv4-pool
```

该命令标识名为的 IPv6 访问列表 **cafe-only**。该访问列表定义了可以使用 **ipv4-pool** 定义的 IPv4 地址池进行映射的源 IPv6 地址。

```
NAT-PT-Router(config)# ipv6 nat v6v4 pool ipv4-pool 10.10.20.20 10.10.20.30
prefix-length 24
```

该命令列出了 NAT-PT 使用的源 IPv4 地址池的起始地址（10.10.20.20）和结束地址（10.10.20.30）。

例 11-6 通过 **ping** 命令和 **debug** 命令验证了网络的可达性。

**例 11-6 验证动态 NAT-PT 转换**

```
PCA> ping 2001:DB8:FEED::12

Pinging 2001:DB8:FEED::12 from 2001:db8:ffff::2 with 32 bytes of data:

Reply from 2001:DB8:FEED::12: time=1ms
Reply from 2001:DB8:FEED::12: time=1ms
Reply from 2001:DB8:FEED::12: time=1ms
Reply from 2001:DB8:FEED::12: time=1ms

Ping statistics for 2001:DB8:FEED::12:
```

```
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 1ms, Average = 1ms

PCA>
```

```
NAT-PT-Router# debug ipv6 nat
IPv6 NAT-PT debugging is on
*Apr  2 02:47:25.367: IPv6 NAT: icmp src (2001:DB8:CAFE:FFFF::2) ->
(10.10.20.20), dst (2001:DB8:FEED::12) -> (10.10.10.10)
*Apr  2 02:47:25.367: IPv6 NAT: icmp src (10.10.10.10) ->
(2001:DB8:FEED::12), dst (10.10.20.20) -> (2001:DB8:CAFE:FFFF::2)
NAT-PT-Router#
```

再次从主机 A 向主机 B 发起 ping 操作以验证相关配置（如例 11-6 所示）。可以看出，只允许对 IPv6 源地址为 2001:db8:cafe::/48 的主机进行转换，并且从 IPv4 地址池（10.10.20.20～10.10.20.30）分配源 IPv4 地址。命令 **debug ipv6 nat** 的输出结果证实了 NAT-PT 转换正在使用 **ipv6 nat v6v4 pool** 命令中列出的源 IPv4 地址池。对于本例来说，使用的是地址池中的第一个地址 10.10.20.20。

## 11.3 其他转换技术

NAT64 和 NAT-PT 是两种最常见的 IPv4-to-IPv6 转换方法，但并不是唯一的两种转换方法。IETF 还定义了其他转换技术，允许纯 IPv6 网络中的设备能够与纯 IPv4 网络中的设备进行通信。本节将简要介绍这些可用的转换技术。

- **TRT（Transport Relay Translation，传输中继转换）**：TRT 类似于 NAT-PT，通过 TRT 系统在 IPv4 域与 IPv6 域之间中继流量。TRT 允许纯 IPv6 主机与纯 IPv4 主机交换 TCP 和 UDP 流量。TRT 定义在 RFC 3142“An IPv6-to-IPv4 Transport Relay Translator”中。
- **6RD（IPv6 Rapid Deployment，IPv6 快速部署）**：类似于 6to4 隧道，6RD 通过将 IPv6 无状态地封装到 IPv4 中，从而能够在纯 IPv4 网络中传输 IPv6 包。与 6to4 隧道使用固定的 6to4 前缀不同的是，6RD 服务提供商使用自己的 IPv6 前缀。6RD 定义在 RFC 5569“IPv6 Rapid Deployment on IPv4 Infrastructures (6rd)”中。
- **DS-Lite（Dual-Stack Lite，轻量级双栈）**：服务提供商利用 DS-Lite 将其使用私有 IPv4 地址的客户连接到 IPv4 互联网（如图 11-12 所示）。IPv4 客户与服务提供商之间是 IPv6 链路，客户使用私有源 IPv4 地址（RFC 1918）发送 IPv4

包时，路由器会将该数据包封装到 IPv6 包中并通过 IPv6 网络进行传输，由隧道对端负责解封装出 IPv4 包，并利用 IPv4 NAT 将私有 IPv4 地址转换为公有 IPv4 地址。DS-Lite 和 6RD 都是常常与转换机制共同使用的隧道机制。

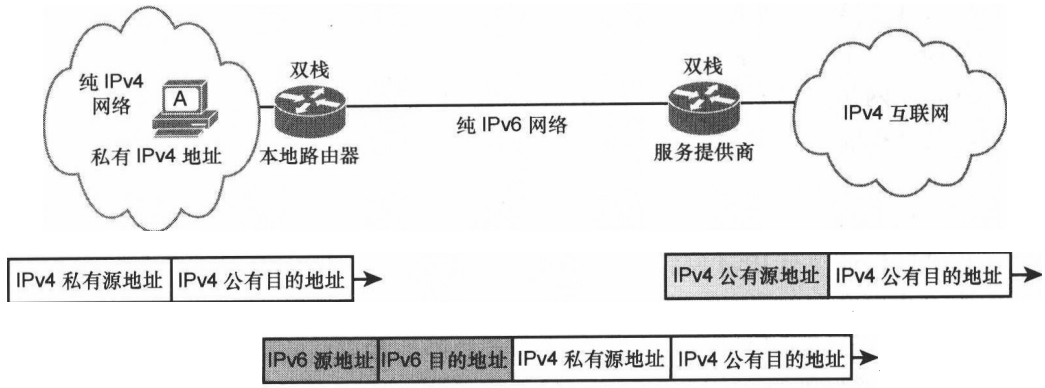



图 11-12　DS-Lite

## 11.4　本章小结

虽然存在很多从 IPv4 向 IPv6 迁移的过渡技术和共存技术，但是纯 IPv6 网络基础设施应该是首选目标也是终极目标。第 10 章讨论了双栈和隧道技术，本章则重点讨论了 NAT64 和 NAT-PT 等转换技术。虽然双栈网络和隧道也都有各自的约束条件，但是与各种形式的转换技术相比，它们的局限性则要小得多，因而只有无法实现纯 IPv6 或者无其他过渡方法可用的情况下，才考虑使用 NAT64 和 NAT-PT。

本章主要讨论了两种形式的 NAT：NAT64 和 NAT-PT。其中，NAT64 是建议的 IPv4-to-IPv6 过渡技术和 IPv4 与 IPv6 共存技术，通过 DNS64，NAT64 允许纯 IPv6 客户端向纯 IPv4 服务器发起通信进程或者纯 IPv4 客户端向纯 IPv6 服务器发起通信进程（利用静态或手工绑定）。

DNS64 服务器是提供 IPv6 AAAA 记录的普通 DNS 服务器。如果没有 4A（AAAA）记录，那么 DNS64 服务器将试图定位 IPv4 A 记录。找到了相应的 A 记录后，DNS64 服务器就利用 NAT64 前缀将该 IPv4 A 记录转换为 IPv6 AAAA 记录，从而让纯 IPv6 主机认为自己可以使用 IPv6 与服务器进行通信。NAT64 负责在 IPv4 和 IPv6 两种协议之间进行 IP 报头的转换操作。

NAT-PT 允许纯 IPv6 网络中的设备与纯 IPv4 网络中的设备进行通信。NAT-PT 路由器负责在 IPv6 地址与 IPv4 地址之间进行转换（类似于 IPv4 NAT 在公有 IPv4 地址与私有 IPv4 地址之间的转换操作）。对于 NAT-PT 来说，不仅要在两种协议之间进行地址转换，而且还要执行其他网络层字段的转换操作。

静态 NAT-PT 和动态 NAT-PT 都执行 IPv6 地址与 IPv4 地址的一对一映射操作。对于静态 NAT-PT 来说，每对 IPv6-to-IPv4 地址都必须是单独映射的，而动态 NAT-PT 则是从源 IPv4 地址池中提供源地址。

虽然双栈、隧道以及转换机制都不是过渡到 IPv6 的理想网络环境，但它们是简化过渡操作的必要工具。IETF 提供了大量相关过渡机制来帮助全球互联网向 IPv6 的迁移。由于全球互联网包含了大量不同的网络应用环境，因而没有任何一种工具能够适应所有网络环境的迁移需求。

无论何时，无论何地，纯 IPv6 都是终极目标，但现实却并非总能实现。目前是网络管理员开始熟悉并掌握 IPv6 的最佳时机，将设备（主机、路由器等）配置为双栈方式，不但能够保证现有 IPv4 网络的安全稳定运行，而且还能促使大家尽快熟悉 IPv6。

在我过去研究和部署 IPv6 的岁月里，IPv6 一直都处于演进变化之中，因而随时关注最新的 RFC、厂商实现以及 IPv6 的部署讨论将是非常重要的环节。过去我们曾经期待 IPv4 设备能够提供的许多功能特性，目前都已经在 IPv6 设备上得以实现了。

## 11.5 参考文献

RFC：

RFC 2766, Network Address Translation - Protocol Translation (NAT-PT) , G.Tsirtsis, Campio Communications, www.ietf.org/rfc/rfc2766 , February 2000

RFC 3142, An IPv6-to-IPv4 Transport Relay Translator , J. Hagino, IIJ Research Laboratory, www.ietf.org/rfc/rfc3142 , June 2001

RFC 4966, Reasons to Move the Network Address Translator – Protocol Translator (NAT-PT) to Historic Status , C. Aoun, Energize Urnet, www.ietf.org/rfc/rfc4 966 , July 2007

RFC 5569, IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) , R. Despres,RD-IPtech, www.ietf.org/rfc/rfc5569 , January 2010

RFC 6145, IP/ICMP Translation Algorithm , X. Li, CERNET Center/Tsinghua,www.ietf.org/rfc/rfc6145 , April 2011

RFC 6146, Stateful NAT64: Network Address and Protocol Translation , M.Bagnulo, UC3M, www.ietf.org/rfc/rfc6146 , April 2011

网站：

Network Address Translator - Protocol Translator, www.cisco.com/en/US/prod/collateral/iosswrel/ps8802/ps6969/ps1835/prod_white_paper09186a008011ff51_ps6640_Products_White_Paper.htmlImplementing NAT-PT for IPv6, www.cisco.com/en/US/d ocs/ios/ ipv6/configuration/guide/ip6-nat_trnsln_ps6350_TSD_Products_Configuration_Guide_Chapter.html