Cisco Systems

CCIE 职业发展系列
CCIE Professional Development

ciscopress.com



TCP/IP 路由技术
（第一卷）（第二版）

Routing TCP/IP
Volume I, Second Edition

A detailed examination of interior routing protocols

[美] Jeff Doyle, CCIE #1919
Jennifer Carroll, CCIE #1402 著

葛建立　吴剑章　译

人民邮电出版社
POSTS & TELECOM PRESS

CCIE 职业发展系列

TCP/IP 路由技术（第一卷）（第二版）

[美] Jeff Doyle, CCIE #1919
Jennifer Carroll, CCIE #1402 著

葛建立　吴剑章　译

人 民 邮 电 出 版 社

图书在版编目（CIP）数据

TCP/IP 路由技术．第 1 卷：第 2 版/（美）多伊尔（Doyle，J. ），（美）卡罗尔（Carroll，J. ）著；葛建立，吴剑章译．—北京：人民邮电出版社，2007.1
（CCIE 职业发展系列）

ISBN 978-7-115-15429-3

Ⅰ．T… Ⅱ．①多…②卡…③葛…④吴… Ⅲ．计算机网络—通信协议—路由选择

Ⅳ．TN915.04

中国版本图书馆 CIP 数据核字（2006）第 125587 号

版 权 声 明

Jeff Doyle, Jennifer Carroll: Routing TCP/IP, Volume I, Second Edition (ISBN: 1587052024)

Copyright © 2006 Cisco Systems, 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。

CCIE 职业发展系列

TCP/IP 路由技术（第一卷）（第二版）

- 著 [美] Jeff Doyle, CCIE#1919
  Jennifer Carroll, CCIE#1402
  译 葛建立 吴剑章
  责任编辑 李 际
- 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 电子函件 315@ptpress.com.cn
  网址 http://www.ptpress.com.cn
  北京顺义振华印刷厂印刷
  新华书店总店北京发行所经销
- 开本：787×1092 1/16
  印张：48
  字数：1 208 千字 2007 年 1 月第 1 版
  印数：1 – 4 000 册 2007 年 1 月北京第 1 次印刷

著作权合同登记号 图字：01-2006-0305 号

ISBN 978-7-115-15429-3/TP・5783

定价：99.00 元

读者服务热线：（010）67132705 印装质量热线：（010）67129223

内容提要

本书是一本详细而又完整地介绍互连网络内部网关协议（IGP）的专业书籍，堪称有关 IGP 方面不可多得的经典之作。本书共分三个部分。第一部分主要介绍了网络和路由选择的基本知识，其中包括 IPv4 协议、IPv6 协议和路由技术。第二部分是本书的精华，这一部分详细、深入地讲述了各种常用的内部路由协议，如 RIP、RIPv2、RIPng、无类别路由选择、EIGRP、OSPFv2、OSPFv3、IS-IS 等协议，每一章除了对该协议的实现机制和参数详尽阐述，使读者对协议的实现原理有一个清晰的理解外，还通过在实际网络环境中的实例，详细地论述了该协议在 Cisco 路由器上的配置和故障处理方法，帮助读者获取大量解决实际问题的专业技能。第三部分介绍了如路由重新分配、缺省路由/按需路由选择、路由过滤、路由映射等多种重要而有效的路由控制工具，用来创建和管理多个 IP 路由选择协议的协调和互操作。附录部分讲述了二进制、十六进制转换、访问列表、CCIE 提示等内容。

相对于第一版，本书第二版具有以下更新：在第一版详细讲述 IPv4 协议中 IGP 的基础上，大量增加了相应协议在 IPv6 协议中的实现和配置，其中单独一章用来讲述 IPv6 中应用的 OSPFv3 协议，这是本书新版的一大亮点；同时本书根据 Internet 和 Cisco IOS 系统的最新发展，适当地删减了如网桥、IGRP 等过时的内容，并增加了许多新的 IOS 增强特性的讲解。

本书的读者不仅是那些准备 CCIE 考试的考生，也是任何需要完整理解 IPv4，特别是 IPv6 下 TCP/IP 内部路由选择协议实现的网络设计和工程人员。本书中对协议细节的讲解和对网络实例的探讨相信会让读者获益匪浅。

献　　词

我愿将本书献给我的妻子 Sara 和我的孩子们：Anna、Carol、James 和 Katherine。

——Jeff

我愿将本书献给我的丈夫 Mike 和我的儿子：Mitchell 和 Jonathan。他们的耐心和支持帮助我得以完成本书。

——Jennifer

致　　谢

感谢 Brett Bartow、Chris Cleveland、Andrew Cupp、San Dee Phillips，以及 Cisco Press 的所有工作人员，是他们的工作与努力使本书得以完成。

我们需要感谢本书的技术编辑 Steven Moore、Rena Yang 和 Frank Knox，感谢他们认真细致的编辑工作以及对本书提出的非常重要的忠告和建议。

我们还要感谢 Frank Knox、Carl Pike、Chris Tonini 和其他 Skylabs 网络的工作人员。Skylabs lab 为我们提供了设备安装和访问方便的实验室，使我们可以完成本书中所有配置和案例研究所需要的工作。

关于技术审稿人

Frank Knox，是 Skyline Computer 公司的首席技术执行官，已经供职 6 年多。他具有两个 CCIE 资格证书（CCIE#3698：SNA/IP 和路由选择/交换），同时也是一位 CCSI（Cisco 公司授权资深讲师）。除了负责 CTO 方面的职责外，Frank 教授过几门与 Cisco 有关的高级课程，其中包括为期一周的 CCIE 实验室考试准备专题培训等。他是与路由器相连的大型主机技术的权威，也是有关网络互连集成方面问题和技术的专家（例如，SNA/IP 和语音/数据）。他在 IBM 公司、GTE（Verizon）Directories 公司以及 Skyline Computer 公司积累了超过 37 年的网络组网经验。这些经验包括服务、技术支持、产品规划、管理，以及网络互连教育的各个方面。另外，Frank 还为 Dallas 大学电信 MBA 项目开发和讲授了一些课程。他在 Pace 大学获得了电信方面的硕士学位（4.0 GPA）。

Steven Edward Moore 在 Cisco 公司工作的 6 年半的时间里，作为一名工程师担任了不同的角色，目前转到了 IP Routing Protocol Scalability 团队（IP 路由选择协议扩展团队）。在这个团队，他主要是围绕对网络与协议的可扩展性进行扩展的各个方面进行工作：考虑协议架构的新特性和优化，设计对当前协议的可扩展性的测试和评估，帮助客户一起实现对客户网络的功能扩展，以及参加像 Networkers 会议这样的活动，来指导其他人怎样从路由选择的角度来增强他们的网络性能和可扩展性。

Rena Yang 是 Cisco 公司的一位软件工程师。她具有超过 6 年的实现 Cisco IOS 软件编码方面的经验。目前她从事 IS-IS 协议方面的工作。在这之前，她主要从事 IPv4、UDP、访问列表、策略路由和路由选择基础架构方面的工作。Rena 在 MIT 获得了理学学士和计算机科学工程硕士的学位。

Forward for Chinese-Language Edition

Since the publication of *Routing TCP/IP* Volumes I and II, I have had many opportunities to visit the People's Republic of China. In no other country have I received as many warm compliments on these books as I have in China. I am therefore delighted that PT Press is now offering a version of Volume I in Mandarin.

China is aggressively expanding its Internet infrastructure, and along with it services such as mobile IP and online gaming. In the next few years, I predict that China will become the world's leader in the commercial implementation of IPv6. Already China exceeds Japan in the number of PCs, and it will soon have the world's largest mobile network. All of this expansion means that over the coming decade, there will be an enormous increase in the demand for IP networking experts. The Cisco Certified Internet Expert program provides the opportunity for potential employers to discern the best networking engineers, and it provides those holding the certification a testament to their expertise. Therefore the CCIE program plays an increasingly important role in the growing Chinese networking industry. I hope that you, the reader, will find this book useful in your preparation to earn this coveted certification.

Best Regards,

Jeff Doyle



Foreword for the Second Chinese Edition

As I mentioned in the foreword of the first Chinese version of this volume, I make several trips each year to the People's Republic of China and have made many friends there; on every trip I meet people who have kind words about *Routing TCP/IP* Volumes I and II. I am, therefore, very pleased to introduce this Mandarin version of the second edition of Volume I.

Although the first edition of Volume I was first published more than eight years ago, the material it covers remains as relevant today as it was then. Understanding the concepts and protocols in this book is essential for building a foundation of knowledge before progressing to more advanced networking concepts. Too often, engineers try to tackle such difficult topics as IP multicast, complex inter-domain peering, routing policies, and MPLS traffic engineering without first acquiring a firm grasp of the routing basics underlying all of them all. This book, if you study it carefully, will give you that necessary foundation and help you to confidently move on to more advanced subjects.

In the foreword to the previous edition I made several predictions that have since become reality. For example, China now has the world's largest mobile network, which continues to grow at a tremendous rate. I also predicted that China would become the world leader in IPv6 development and deployment. Through the China Next-Generation Internet (CNGI) project, that prediction is quickly moving to reality. There is no other country in the world for which IPv6 is such an urgent and essential protocol, and therefore the coverage of IPv6 in this second edition is of particular importance to Chinese readers.

The translation of any book into another language is a daunting project, and I would like to offer my warmest gratitudeto Mr. Ge Jian Li for his hard work in making this Mandarin version available to you.

I would also like to thank Mr. Liu Tao and PT Press for making this translation possible.

And finally I would like to thank you, dear reader, and the thousands of other Chinese readers who have made *Routing TCP/IP* such a success in China.

Best Regards,

Jeff Doyle

Denver，Colorado



中文版第一版序

自从《TCP/IP 路由技术》第一卷和第二卷出版以来，我有了很多机会可以来到中国。这些书在其他国家都没有像在中国这样受到大家的热情关注。因此，我很高兴人民邮电出版社能够在中国出版本书第一卷的中文版本。

在中国，Internet 基础设施正在迅猛发展与普及，而且也提供了诸如移动 IP 和在线游戏之类的服务。在未来的几年，我预计中国将成为 IPv6 商业化部署的世界领导者。在 PC 机的拥有量上，中国已经超过了日本，并且在不远的将来，中国会拥有世界上最大的移动网络。所有这些巨大的发展都意味着在将来的 10 年里，中国对 IP 网络专家的需求量将会有很大的增长。Cisco 认证互联网络专家（CCIE）计划可以为企业管理者提供一个渠道，以便识别出最优秀的互联网络工程师；同时，它也为专业技能的证明提供了确实的依据。因此，CCIE 计划在中国互联网络产业的发展中将扮演日益重要的角色。我非常期待读者在获取令人羡慕的认证准备过程中，能从本书获益。

最诚挚的问候

Jeff Doyle



中文版第二版序

正如我在本卷第一版的中文版序言中所提到的，我每年都会到中国几次，并在中国结识了很多朋友；每次我都遇到一些提及有关《TCP/IP 路由技术》卷一和卷二的朋友。现在我非常高兴地把卷一第二版的中文版介绍给大家。

虽然本书第一版的首次出版距今已有 8 个年头，但它涵盖的一些内容到现在依然实用。了解本书中提及的概念和协议是建立基本知识架构的基础，这将有利于进一步学习更高级的网络知识。我们常常可以看到，一些网络工程师在未牢固掌握路由技术的基本知识之前，就试图去解决诸如 IP 多播路由、复杂的域间对等体、路由策略以及 MPLS 流量工程等困难的课题，而这些高级课题都是建立在前者基础之上的。读者如果认真学习本书，将会获取必要的基本知识，从而可以帮助你有信心进一步学习更高级的课题。

在本书第一版的序言中，我的几个预言现在已经变成了现实。例如，中国现在已经具有世界上最大的移动通信网络，并且仍在继续高速发展。我也预言了中国将成为发展和部署 IPv6 网络的全球领导者。通过中国下一代因特网的项目（CNGI）可以看出，这个预言也将很快变成现实。现今世界上没有哪个国家比中国更迫切需要 IPv6 协议，因此，在本书第二版中讲述的 IPv6 技术，对于中国的读者来说显得尤其重要。

把任何一本书翻译成另一种语言都是一件令人生畏的工作，我对本书译者葛建立先生致以最诚挚的谢意，是他所做的努力工作使本书的中文版本得以与读者见面。

我也要感谢人民邮电出版社和刘涛先生使本书中文版的翻译项目变成现实。

最后，我想感谢您，亲爱的读者朋友，以及使《TCP/IP 路由技术》在中国如此成功的其他中国读者。

最诚挚的问候

Jeff Doyle
美国科罗拉多州，丹佛



作者简介

Jeff Doyle（CCIE #1919）是专注于研究 IP 路由选择协议、MPLS 和 IPv6 方面的专家。他设计了或协助设计了遍及北美、欧洲、中国、韩国以及日本等很多地区的大型 IP 服务提供商的网络。Jeff 经常在为数众多的研究团体和会议上出现，并在 NANOG、JANOG、APRICOT 以及 IPv6 论坛会议上发言。Jeff 拥有 Memphis 州立大学的文学学士学位，并在新墨西哥大学学习了电气工程专业。Jeff 目前生活在科罗拉多州的丹佛地区。

Jennifer Carroll（CCIE #1402）是华盛顿州雷蒙德地区的独立网络顾问专家。在过去的 15 年中，她设计、实施和优化了许多 TCP/IP 网络，并开发和讲授了多种有关路由选择协议和 Cisco 公司路由器方面的网络组网和网间互连课程。读者可以通过电子邮件 jennifer.carroll@ieee.org 与 Jennifer 联系。

原书序

1976年，当我在数字设备公司（DEC）第一次看到Arpanet IMP 的时候，今天我们熟知的网络当时还处于发展初期。SNA、XNS 和 DECnet 等网络处于早期发展阶段，当时有关分组交换和电路交换的讨论还是热点话题。我们这些从事交换和路由选择算法设计的人处理的是具有 64KB 内存的路由器（虽然我们那时不这样称呼它们），56KB 的数据链路被认为是非常快的，具有 256 个节点的网络就已经足够大了。假如你是卖出了那 256 台计算机的销售人员，你将会拥有可观的财富并可以退休了。

30 年是很长的一段时间，如今，组成 Internet 的单个网络就包含了数以千计或数以万计的节点，整个 Internet 包含了数以亿计的计算机。在我们这一代的网络发展过程中，最显著的一点是基于 TCP/IP 协议簇的 Internet 基础并没有大的变化；虽然在这期间，计算机体系结构经历了四代或更多代，操作系统技术经历了整整 3 代，而传输速度也提高了 5 个数量级。

然而，我们仍然把分组交换网络中的路由选择看作一种“魔法”，为什么呢？

首先，设计强壮的、可扩展的分布式算法是困难的。虽然我们非常希望网络尽可能的简单，但是不可避免的要遇到一些特别的实际案例、网络优化、特殊的拓扑结构，以及链路技术等各种情况，因此网络的复杂性也在一点一点的增加。由于整个网络铲车式升级（fork lift upgrade）是几乎不可行的，因此，目前同时存在多种技术版本，我们必须维护一个向后兼容的、无缝连接的网络来提供网络服务。当控制数据包路由选择的策略变得越来越复杂的时候，我们设计用来自动发现和配置网络的能力就会受到限制，我们又不得不退回到依赖于手工配置和调整性能的技术。最后，当这些网络运行的环境已经从相互之间默认为信任关系的环境发展为会受到内外部攻击的环境时，设计和部署更加安全可靠的路由选择系统就变成一种迫切需要解决的优先问题。

本书完全解开了这个“魔法”之谜。新版的第一卷涵盖了

TCP/IP 网络所有必不可少的基础知识，并给出了理解在 Internet 的某个单一管理区域内如何完成路由选择所需要的所有工具。首先，在开始的有关地址和静态路由的章节中介绍了分组交换网络中路由技术的基本概念，然后深入地讨论了目前最流行的 IGP——RIP、EIGRP、OSPF 以及 IS-IS 协议。最后讲述了路由重新分配、路由过滤和策略路由选择等方面的高级课题。

这次出版的第二版也增加了有关 IPv6 方面的基本内容，并提供了 Cisco IOS 软件系统最新版本有关示例和配置的所有最新内容。

本书对于任何希望全面了解在 TCP/IP 网络中路由选择是如何实现的读者都是有帮助的：从路由选择算法的设计原则、地址规划的发展到设计和配置大型自主系统网络路由选择的实践等。

David Oran
Cisco Fellow

前　言

路由技术即使在最小的数据通信网络中也是基本的要素。在某种程度上，路由技术和路由器的配置是相当简单的。但是，当网络的规模越来越大，并且越来越复杂的时候，路由选择问题就变得比较突出和难以控制了。或许，有点不恰当地说，作为一名网络系统顾问，我应该感谢当前出现的大规模路由技术难题，这些问题给了我谋生的手段。假设没有它们，“你何以为生？”这句习语可能就会不幸地成为我每天生活词汇的一部分了。

Cisco 认证互联网专家（CCIE）在大型网络的设计、故障排除和管理能力方面得到广泛的认同。这种广泛的认同来自于这样一个事实：一个网络工作人员仅仅依赖参加一些课程的培训，并反复依赖记忆一些书面测试的内容是不可能成为一名 CCIE 的。一名 CCIE 必须通过一个众所周知、难度非常大的、并且需要亲自动手操作的实验室考试，从而使他或者她的专业技能得到提高。

本书的目标

本书是专门讨论 TCP/IP 路由问题的两卷书中的第一本。在早期撰写本书的第一版时，Cisco Systems 的前 CCIE 项目经理 Kim Lew 说过：“我们的目标是使人们成为 CCIE，而不是使人们通过 CCIE 实验室考试。”作者完全赞同这种观点，并且把它作为一种指导原则贯穿到本书的写作当中。虽然这本书包括了很多案例研究和练习可以帮助读者准备 CCIE 实验室考试，但是作者的主要目的还是提高读者对 IP 路由技术的理解——能有一个普通的水平并能够在 Cisco 的路由器上进行实现。

读者对象

本书的读者可以是任何需要完整理解 TCP/IP 内部路由选择协议的网络设计人员、管理人员或者工程人员。虽然本书的具体实践方面针对 Cisco IOS，但是本书的内容也可以应用于

任何路由选择平台。

这本书不仅仅是写给那些计划成为 Cisco 认证互联网专家的读者阅读的，而且是写给任何希望提高自己的 TCP/IP 路由技能的读者。这些读者可以划分为以下三类：

- “初学者”——具有基本的网络知识，并且希望开始深入学习网络互连的读者；
- 中级水平的网络专业人员——具有一定的路由器（Cisco 或其他厂商的产品）操作经验，并且计划提高自己的技能达到专家水平的读者；
- 经验丰富的网络专家——这些读者具有丰富和广泛的 Cisco 路由器的实践经验和专业技能，并且准备参加 CCIE 实验室考试。但是，这类读者需要自己制定一个复习表和一系列检验与确认自己技能的练习。

本书主要面向具有中级水平的网络专业人员。同时，对于初学者，本书提供了一个网络基本知识的概要。而对于网络方面的专家而言，本书也提供了一些磨炼他们的专业技能所需要的挑战性内容。

对第一版所做的改动

第二版所进行的改动受到几个因素的影响。第一个因素是 CCIE 本身。当笔者撰写本书的第一版的时候，CCIE——现在称为路由选择与交换专业的 CCIE——还是 Cisco 公司提供的惟一的认证考试。目前，已经形成了将 CCIE 作为塔尖的认证途径的一系列认证。此外，标准的网络互连专业人员也要比 1997 年具备更多的知识。考虑到这一点，我们删除了原书的第一章，其中包括网桥、路由器和网络地址的最基本的概念（读者在网络中看到网桥设备的最后时间是什么时候？）。

影响本书第二版所做变化的第二个因素是 Cisco 公司的 IOS 软件系统的变化。在撰写本书第一版的时候，IGRP 协议还经常使用，而现在它已经是一个过时的协议，存在的主要意义就是作为 EIGRP 协议的前身。因此，在第二版中删除了有关 IGRP 协议的章节，并在讲述 EIGRP 协议的章节中将 IGRP 协议作为历史回顾的一部分来讲述。IOS 软件系统的命令集本身也进行了扩展，以便适应新的功能和选项；我们也增加了在 20 世纪 90 年代后期还不存在的协议扩展和命令。

最后，IPv6 协议在 1997 年还主要是建议草案，现在却已经处于全球部署的早期阶段了。读者需要了解有关这个协议更为细节的知识，以及在不远的将来支持 IPv6 的 IP 路由选择协议扩展方面所需要的知识；如果读者还没有准备好，那么本书深入地探讨了 IPv6 的路由选择技术。

这个版本的其他一些变化是语法和语义方面的。例如，在第一版中，笔者对作为数据链路的“网络”和作为由路由器连接的网络集的“互连网络”进行了区分。虽然这些术语的确很精确，但也有些呆板，现在已经很少使用“互连网络”这样的术语了。相反，“网络”经常用来表示从本地链路到像 Level 3、NTT 和 Sprint 这样的全球自主系统的网络。我们尝试在这个新的版本中带给读者比较现代和通用的术语描述。

本书的内容组织

本书共有 14 章，分为 3 个部分。

第一部分“路由选择的基本知识”介绍了 IPv4 协议和 IPv6 协议的基本知识，以及路由技术的基本概念。虽然一些水平较高的读者可能希望跳过第 1 章，但是我建议这些读者至少应该浏览一下第 3 章“静态路由”和第 4 章“动态路由选择协议”的内容。当然，如果读者还不熟悉 IPv6 协议的话，也必须阅读第 2 章“IPv6 介绍”。

第二部分“内部路由选择协议”包括了 IP 路由选择的各种内部网关协议。针对具体协议的每一章都是从该协议的基本原理、实现机制以及参数讲解开始的，并在读者对该协议有了一个总体的了解后，接着通过多个不同的网络拓扑环境中的案例研究，详细地讲述了该协议在 Cisco 路由器上的配置和故障诊断方法。

外部网关协议，BGP 协议，还有组播路由选择、服务质量保证、路由器的安全与管理以及网络地址转换等一些主题，将在“TCP/IP 路由技术 第二卷”中介绍。

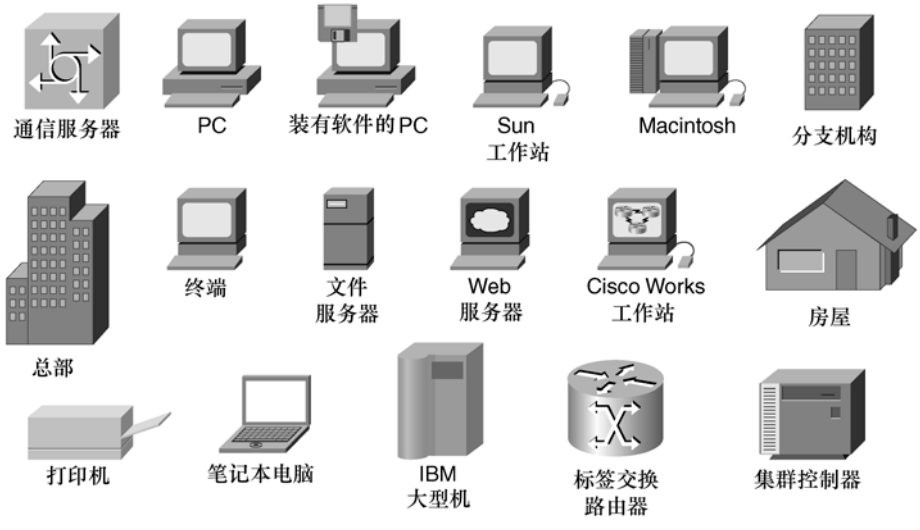
第三部分“路由控制和互操作性”介绍了多种有效的工具，用来创建和管理多个 IP 路由选择协议的互操作性，例如缺省路由、路由过滤等。同样地，最后部分的这些章节对创建复杂的路由选择策略所需的必要工具做了一个初步介绍，这些策略将会在本书卷二中详细介绍。这些章节和第二部分的章节一样，也是先从概念开始讲解，并以案例研究作为结束。

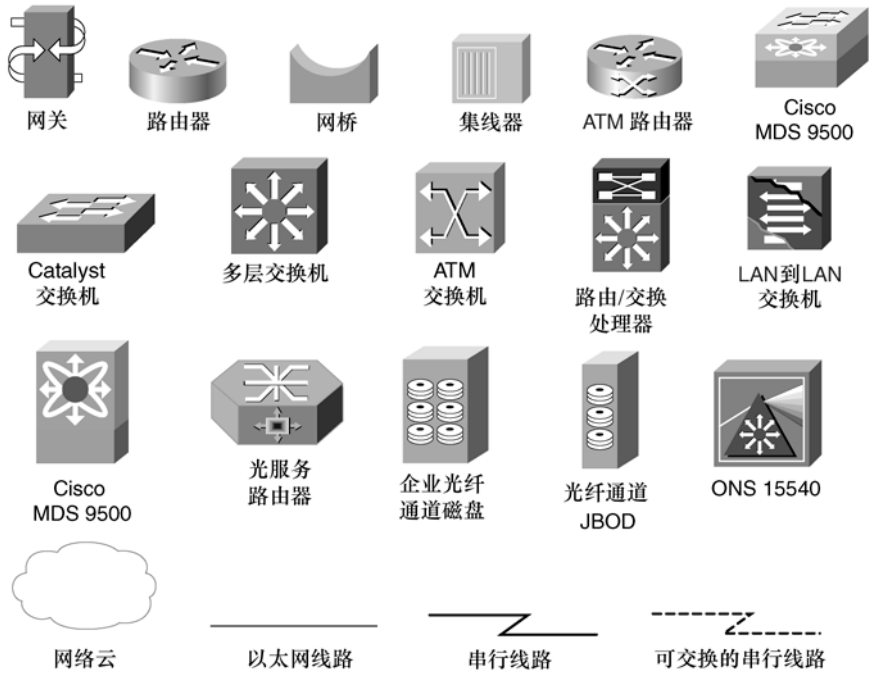
本书的风格

大多数章节在结束时都配有一组复习题、配置练习和故障诊断练习。复习题主要侧重于每章主题的基本理论方面，而配置和故障诊断练习主要侧重于每章主题在 Cisco 设备上的实际实现。

在每章末尾还列出了一张命令总结表，简要介绍了在这一章中使用到的 Cisco IOS 中的所有重要命令。这些命令使用的惯例和 Cisco IOS 命令参考中使用的惯例是一样的。

本书使用的图标





命令语法约定

本书在介绍命令语法时使用的约定与《IOS 命令参考手册》相同，这些约定如下：

- **粗体字**表示实际需要键入的命令和关键字，在实际的配置示例和输出信息（不是一般的命令语法）中，粗体字表示用户手工输入的命令（例如 **show** 命令）；
- *斜体字*表示需要用实际数值替换的参数；
- 竖线（|）表示在几个选项中选择一项，并且这些项是互相排斥的；
- 方括号[ ]表示可选的参数；
- 大括号{}表示一个必需的选项；
- 方括号内嵌大括号[{}]表示在一个可选项中的必选项。

译者注：本书第一版出版后，受到了广大读者的关注与厚爱，译者收到许多读者的来信，本书新版对第一版中出现的错误进行了修正，在此感谢读者的批评指正，欢迎广大读者来信交流，译者 E-mail：cfa35@126.com。

目　录

第一部分　路由选择的基本知识

第 1 章　TCP/IP 回顾 ····· 3

1.1　TCP/IP 协议层 ····· 3
1.2　IP 包头 ····· 5
1.3　IPv4 地址 ····· 12
1.3.1　首个八位组字节规则 ····· 13
1.3.2　地址掩码（Address Mask） ····· 15
1.3.3　子网和子网掩码 ····· 16
1.3.4　子网规划 ····· 18
1.3.5　打破八位组界线 ····· 19
1.3.6　子网掩码的故障诊断 ····· 21
1.4　地址解析协议（ARP） ····· 22
1.4.1　代理 ARP ····· 26
1.4.2　无故 ARP ····· 27
1.4.3　反向 ARP ····· 27
1.5　ICMP ····· 28
1.6　主机到主机层 ····· 31
1.6.1　TCP ····· 31
1.6.2　UDP ····· 33
1.7　展望 ····· 34
1.8　总结表：第 1 章命令总结 ····· 34
1.9　推荐读物 ····· 35
1.10　复习题 ····· 35
1.11　配置练习 ····· 36
1.12　故障诊断练习 ····· 36

第 2 章　IPv6 概述 ····· 39

2.1　IPv6 地址 ····· 40
2.1.1　地址表示法 ····· 41
2.1.2　IPv6 的地址类型 ····· 42

2.2 IPv6 包头格式 47
2.3 IPv6 扩展报头 49
2.4 ICMPv6 51
2.5 邻居发现协议（NDP） 52
2.5.1 NDP 消息 53
2.5.2 路由器发现（Router Discovery） 57
2.5.3 地址自动配置（Address Autoconfiguration） 58
2.5.4 地址冲突检测（Duplicate Address Detection） 60
2.5.5 邻居地址解析（Neighbor Address Resolution） 60
2.5.6 私有地址（Privacy Address） 62
2.5.7 邻居不可到达性的检测 63
2.6 展望 63
2.7 复习题 64

第 3 章 静态路由 67

3.1 路由表 67
3.2 配置静态路由 70
3.2.1 案例研究：简单 IPv4 静态路由 70
3.2.2 案例研究：简单 IPv6 静态路由 73
3.2.3 案例研究：汇总路由 77
3.2.4 案例研究：选择路由 78
3.2.5 案例研究：浮动静态路由（Floating Static Route） 79
3.2.6 案例研究：IPv6 浮动静态路由 81
3.2.7 案例研究：均分负载 83
3.2.8 案例研究：递归表查询 87
3.3 静态路由故障诊断 88
3.3.1 案例研究：追踪故障路由 88
3.3.2 案例研究：协议冲突 91
3.3.3 案例研究：被取代的路由器 94
3.3.4 案例研究：追踪 IPv6 故障路由 97
3.4 展望 99
3.5 总结表：第 3 章命令总结 100
3.6 复习题 100
3.7 配置练习 101
3.8 故障诊断练习 102

第 4 章 动态路由选择协议 107

4.1 路由选择协议基础 107
4.1.1 路径决策 108
4.1.2 度量 109
4.1.3 收敛 111
4.1.4 负载均衡 112

4.2　距离矢量路由选择协议 …… 112
4.2.1　通用属性 …… 112
4.2.2　依照传闻进行路由选择 …… 113
4.2.3　路由失效计时器 …… 115
4.2.4　水平分隔 …… 115
4.2.5　计数到无穷大 …… 117
4.2.6　触发更新 …… 118
4.2.7　抑制计时器 …… 118
4.2.8　异步更新 …… 118
4.3　链路状态路由选择协议 …… 119
4.3.1　邻居 …… 119
4.3.2　链路状态泛洪扩散 …… 120
4.3.3　链路状态数据库 …… 125
4.3.4　SPF 算法 …… 126
4.3.5　区域 …… 129
4.4　内部和外部网关协议 …… 130
4.5　静态或动态路由选择 …… 131
4.6　展望 …… 132
4.7　推荐读物 …… 132
4.8　复习题 …… 132

第二部分　内部路由选择协议

第 5 章　路由选择信息协议（RIP） …… 137

5.1　RIP 的基本原理与实现 …… 138
5.1.1　RIP 的计时器和稳定性 …… 138
5.1.2　RIP 消息格式（RIP Message Format） …… 140
5.1.3　请求消息类型（Request Message Type） …… 142
5.1.4　有类别路由选择（Classful Routing） …… 142
5.2　配置 RIP …… 145
5.2.1　案例研究：一种基本的 RIP 配置 …… 145
5.2.2　案例研究：被动接口（Passive Interface） …… 147
5.2.3　案例研究：配置单播更新（Unicast Update） …… 148
5.2.4　案例研究：不连续的子网 …… 150
5.2.5　案例研究：控制 RIP 的度量 …… 153
5.2.6　案例研究：最小化更新信息的影响 …… 155
5.3　RIP 故障诊断 …… 158
5.4　展望 …… 158
5.5　总结表：第 5 章命令总结 …… 158
5.6　推荐读物 …… 159
5.7　复习题 …… 159

5.8 配置练习 …… 159
5.9 故障诊断练习 …… 160

第 6 章 RIPv2、RIPng 和无类别路由选择 …… 165

6.1 RIPv2 的基本原理与实现 …… 165
6.1.1 RIPv2 的消息格式 …… 166
6.1.2 与 RIPv1 的兼容性 …… 168
6.1.3 无类别路由查找 …… 169
6.1.4 无类别路由选择协议 …… 169
6.1.5 可变长子网掩码（VLSM） …… 170
6.1.6 认证 …… 172
6.2 RIPng 的基本原理与实现 …… 175
6.3 RIPv2 的配置 …… 176
6.3.1 案例研究：RIPv2 的基本配置 …… 177
6.3.2 案例研究：与 RIPv1 的兼容性 …… 177
6.3.3 案例研究：使用 VLSM …… 179
6.3.4 案例研究：不连续的子网和无类别路由选择 …… 181
6.3.5 案例研究：认证 …… 183
6.4 RIPng 的配置 …… 185
6.4.1 案例研究：RIPng 的基本配置 …… 185
6.4.2 案例研究：RIPng 进程的定制 …… 192
6.4.3 案例研究：RIPng 的度量控制 …… 193
6.4.4 案例研究：路由汇总 …… 195
6.5 RIPv2 与 RIPng 的故障诊断 …… 195
6.6 展望 …… 200
6.7 总结表：第 6 章命令总结 …… 201
6.8 推荐读物 …… 201
6.9 复习题 …… 202
6.10 配置练习 …… 202
6.11 故障诊断练习 …… 203

第 7 章 增强型内部网关路由选择协议（EIGRP） …… 209

7.1 EIGRP 的前身：IGRP 协议回顾 …… 209
7.1.1 进程域 …… 210
7.1.2 IGRP 的计时器和稳定性 …… 212
7.1.3 IGRP 的度量 …… 213
7.2 从 IGRP 到 EIGRP …… 218
7.3 EIGRP 的基本原理与实现 …… 219
7.3.1 依赖于协议的模块（Protocol-Dependent Modules） …… 219
7.3.2 可靠传输协议 …… 220
7.3.3 邻居发现和恢复 …… 221
7.3.4 扩散更新算法 …… 222

7.3.5 EIGRP 的数据包格式 …… 241
7.3.6 地址聚合 …… 245
7.3.7 EIGRP 和 IPv6 …… 249
7.4 配置 EIGRP …… 249
7.4.1 案例研究：EIGRP 的基本配置 …… 249
7.4.2 案例研究：非等价负载均衡 …… 250
7.4.3 案例研究：设置最大的路径数 …… 253
7.4.4 案例研究：多个 EIGRP 进程 …… 255
7.4.5 案例研究：关闭自动路由汇总 …… 257
7.4.6 案例研究：末梢路由选择 …… 259
7.4.7 案例研究：地址汇总 …… 264
7.4.8 认证 …… 265
7.5 EIGRP 故障诊断 …… 266
7.5.1 案例研究：邻居丢失 …… 266
7.5.2 “卡”在活动状态的邻居 …… 271
7.6 展望 …… 275
7.7 总结表：第 7 章命令总结 …… 275
7.8 复习题 …… 276
7.9 配置练习 …… 277
7.10 故障排除练习 …… 278
第 8 章 开放最短路径优先协议（OSPFv2） …… 281
8.1 OSPF 的基本原理与实现 …… 282
8.1.1 邻居和邻接关系 …… 283
8.1.2 区域（Area） …… 309
8.1.3 链路状态数据库 …… 315
8.1.4 路由表 …… 328
8.1.5 认证 …… 331
8.1.6 按需电路上的 OSPF …… 332
8.1.7 OSPF 的数据包格式 …… 333
8.1.8 OSPF 的 LSA 格式 …… 339
8.1.9 可选项字段 …… 346
8.2 配置 OSPF …… 347
8.2.1 案例研究：一个基本的 OSPF 配置 …… 347
8.2.2 案例研究：使用 Loopback 接口设置路由器的 ID …… 350
8.2.3 案例研究：域名服务查询 …… 352
8.2.4 案例研究：OSPF 和辅助地址 …… 353
8.2.5 案例研究：末梢区域 …… 357
8.2.6 案例研究：完全末梢区域 …… 360
8.2.7 案例研究：NSSA 区域 …… 361
8.2.8 案例研究：地址汇总 …… 367
8.2.9 案例研究：在区域之间进行过滤 …… 372

8.2.10 案例研究：认证 373
8.2.11 案例研究：虚链路 375
8.2.12 案例研究：运行在 NBMA 网络上的 OSPF 377
8.2.13 案例研究：运行在按需电路上的 OSPF 384
8.3 OSPF 故障诊断 385
8.3.1 案例研究：孤立的区域 388
8.3.2 案例研究：路由汇总配置错误 392
8.4 展望 394
8.5 总结表：第 8 章命令总结 394
8.6 推荐读物 396
8.7 复习题 396
8.8 配置练习 397
8.9 故障排除练习 398

第 9 章 OSPFv3 403

9.1 OSPFv3 的基本原理与实现 403
9.1.1 OSPFv3 与 OSPFv2 的不同之处 404
9.1.2 OSPFv3 的消息 405
9.1.3 OSPFv3 的 LSA 概述 407
9.1.4 OSPFv3 的 LSA 格式 409
9.2 OSPFv3 的配置 416
9.2.1 案例研究：OSPFv3 的基本配置 416
9.2.2 案例研究：末梢区域 419
9.2.3 案例研究：一条链路上的多个实例 420
9.2.4 案例研究：运行在 NBMA 网络上的 OSPFv3 配置 422
9.3 OSPFv3 的故障诊断 427
9.4 展望 430
9.5 总结表：第 9 章命令总结 430
9.6 推荐读物 430
9.7 复习题 431
9.8 配置练习 431

第 10 章 集成 IS-IS 协议 433

10.1 集成 IS-IS 协议的基本原理与实现 435
10.1.1 IS-IS 区域 436
10.1.2 网络实体标题 438
10.1.3 IS-IS 的功能结构 440
10.1.4 IS-IS 的 PDU 格式 451
10.1.5 IS-IS 的扩展属性 469
10.1.6 泛洪扩散的时延 482
10.1.7 提高 SPF 的效率 484
10.2 集成 IS-IS 协议的配置 484

10.2.1　案例研究：IPv4 集成 IS-IS 的基本配置……485
10.2.2　案例研究：更改路由器的类型……489
10.2.3　案例研究：区域的迁移……491
10.2.4　案例研究：路由汇总……494
10.2.5　案例研究：认证……496
10.2.6　案例研究：IPv6 集成 IS-IS 的基本配置……500
10.2.7　案例研究：过渡到多拓扑模式……503
10.2.8　案例研究：层之间的路由泄漏……505
10.2.9　案例研究：多个 L1 区域运行在同一台路由器上……510
10.3　集成 IS-IS 协议的故障诊断……513
10.3.1　IS-IS 邻接关系的故障诊断……513
10.3.2　IS-IS 链路状态数据库的故障诊断……516
10.3.3　案例研究：运行于 NBMA 网络上的集成 IS-IS……519
10.4　展望……523
10.5　总结表：第 10 章命令总结……523
10.6　复习题……525
10.7　配置练习……526
10.8　故障诊断练习……527

第三部分　路由控制和互操作性

第 11 章　路由重新分配……533

11.1　重新分配的原则……534
11.1.1　度量……534
11.1.2　管理距离……535
11.1.3　从无类别协议向有类别协议重新分配……540
11.2　配置重新分配……542
11.2.1　案例研究：重新分配 IGRP 和 RIP……544
11.2.2　案例研究：重新分配 EIGRP 和 OSPF……545
11.2.3　案例研究：重新分配和路由汇总……550
11.2.4　案例研究：重新分配 OSPFv3 和 RIPng……554
11.2.5　案例研究：重新分配 IS-IS 和 RIP/RIPng……559
11.2.6　案例研究：重新分配静态路由……563
11.3　展望……566
11.4　总结表：第 11 章命令总结……566
11.5　复习题……566
11.6　配置练习……566
11.7　故障诊断练习……567

第 12 章　缺省路由和按需路由选择……569

12.1　缺省路由基本原理……570

12.2 按需路由基本原理 …… 571
12.3 配置缺省路由和 ODR …… 573
12.3.1 案例研究：静态缺省路由 …… 574
12.3.2 案例研究：缺省网络命令 …… 578
12.3.3 案例研究：缺省信息始发命令 …… 581
12.3.4 案例研究：配置按需路由选择 …… 586
12.4 展望 …… 586
12.5 总结表：第 12 章命令总结 …… 587
12.6 复习题 …… 587

第 13 章 路由过滤 …… 589

13.1 配置路由过滤器 …… 590
13.1.1 案例研究：过滤特定路由 …… 590
13.1.2 案例研究：路由过滤和重新分配 …… 594
13.1.3 案例研究：协议迁移 …… 597
13.1.4 案例研究：多个重新分配点 …… 601
13.1.5 案例研究：使用管理距离设置路由器优先权 …… 606
13.2 展望 …… 607
13.3 总结表：第 13 章命令总结 …… 608
13.4 配置练习 …… 608
13.5 故障诊断练习 …… 610

第 14 章 路由映射 …… 613

14.1 路由映射的基本用途 …… 613
14.2 配置路由映射 …… 615
14.2.1 案例研究：策略路由选择 …… 617
14.2.2 案例研究：策略路由选择和服务质量路由选择 …… 624
14.2.3 案例研究：路由映射和重新分配 …… 626
14.2.4 案例研究：路由标记 …… 630
14.2.5 案例研究：从 OSPF 路由表中滤掉被标记的路由 …… 634
14.2.6 案例研究：使用路由映射重新分配 IPv6 路由 …… 636
14.3 展望 …… 637
14.4 总结表：第 14 章命令总结 …… 637
14.5 复习题 …… 638
14.6 配置练习 …… 639
14.7 故障诊断练习 …… 640

第四部分 附 录

附录 A 教程：二进制和十六进制 …… 645

A.1 二进制数 …… 646
A.2 十六进制数 …… 647

附录 B　教程：访问列表 …… 651

B.1　访问列表基础知识 …… 652
B.1.1　隐式拒绝一切 …… 652
B.1.2　顺序性 …… 653
B.1.3　访问列表类型 …… 654
B.1.4　编辑访问列表 …… 655
B.2　标准 IP 访问列表 …… 656
B.3　扩展 IP 访问列表 …… 659
B.3.1　TCP 访问列表 …… 662
B.3.2　UDP 访问列表 …… 663
B.3.3　ICMP 访问列表 …… 664
B.4　调用访问列表 …… 665
B.5　自反访问列表 …… 666
B.6　可供选择的关键字 …… 668
B.7　命名访问列表 …… 669
B.8　前缀列表 …… 670
B.9　对放置过滤器的考虑 …… 670
B.10　访问列表的监视和计费 …… 672

附录 C　CCIE 备考提示 …… 675

C.1　牢固的基础 …… 676
C.2　认证途径 …… 676
C.3　实践经验 …… 677
C.4　深入学习 …… 677
C.5　最后 6 个月 …… 678
C.6　参加考试 …… 678

附录 D　复习题答案 …… 681

附录 E　配置练习答案 …… 695

附录 F　故障诊断练习答案 …… 739



第一部分

路由选择的基本知识

第 1 章　ICP/IP 回顾

第 2 章　IPv6 概述

第 3 章　静态路由

第 4 章　动态路由选择协议

本章包括以下主题：

- TCP/IP 协议层；
- IP 包头（IP Packet Header）；
- IPv4 地址；
- 地址解析协议（ARP）；
- Internet 控制消息协议（ICMP）；
- 主机到主机层。

第 1 章

TCP/IP 回顾

考虑到这本书的书名是《TCP/IP 路由技术》，有必要从回顾 TCP/IP 的基本知识开始讲起，然后再讲述如何进行 TCP/IP 路由选择。如果读者正在准备 Cisco 认证互连网专家（Cisco Certified Internetwork Expert，CCIE）的考试，或者仅仅把本书作为路由选择技术方面的参考书，那么读者大概已经了解了本章所要讲述的大部分内容。但是，如果读者阅读一下本章，作为对基本知识的复习也不是一件坏事，有时还会有所帮助。

本章主要回顾了启用、控制或帮助 TCP/IP 路由选择的协议，对 TCP/IP 协议簇并不作深入讨论。本章最后列出的几本参考读物均对 TCP/IP 进行了深入详细的讲解，读者可以至少选择其中的一本进行阅读。

早在 20 世纪 70 年代初期，Vint Cerf 和 Bob Kahn 就对 TCP/IP 及其分层协议框架进行了构思，它的提出先于 ISO 的 OSI 参考模型。本章对 TCP/IP 协议层的简单回顾有助于读者理解 TCP/IP 的多种功能与服务是如何进行相互关联的。

1.1 TCP/IP 协议层

图 1-1 展示了 TCP/IP 协议簇与 OSI 参考模型的相互关系。[1] 在 TCP/IP 协议簇中，网络接口层对应于 OSI 的物理和数据链路层，但实际上在规范中并不存在这一层。如图 1-1 所示，作为对物理和数据链路层的表示，它已经成为事实上的一个层次。在本节中，我们将使用 OSI 的术语——物理和

[1] 除了少数例外，OSI 协议簇本身已经成为 Internet 历史早期的遗留产物。当前 OSI 协议对于网络技术的贡献看来主要是在对学习网络的学生讲述模块化的协议簇时，可以引用它的参考模型进行说明等有限的用途。当然，IS-IS 路由选择协议依然广泛地应用在大型服务提供商和运营商网络中。

数据链路层来描述它。

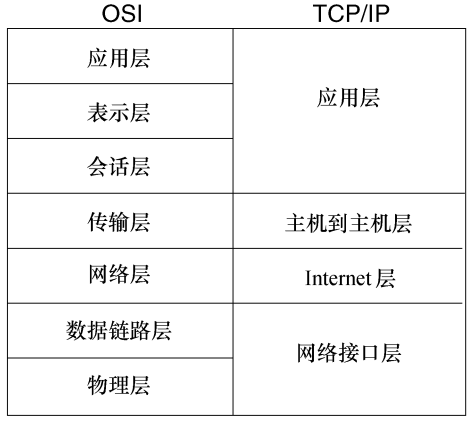


图 1-1 TCP/IP 协议簇

*物理层*包含了多种与物理介质相关的协议，这些物理介质用以支撑 TCP/IP 通信。物理层的协议按照正式的分类可以分为 4 类，这 4 类涵盖了物理介质的所有方面：

- **电子/光学协议**——描述了信号的各种特性。例如，电压或光强度、位定时、编码和信号波形。
- **机械协议**——规定了连接器的尺寸或导线的金属成分。
- **功能性协议**——描述了做什么。例如，在 EIA-232-D 连接器第 4 管脚上的功能描述是“请求发送”。
- **程序性协议**——描述了如何做。例如，在 EIA-232-D 导线上，二进制 1 表示电压小于−3V。

*数据链路层*包含了控制物理层的协议：如何访问和共享介质、怎样标识介质上的设备，以及在介质上发送数据之前如何完成数据成帧。典型的数据链路协议有 IEEE 802.3/以太网、帧中继、ATM 以及 SONET。

*Internet 层*与 OSI 的网络层相对应，主要负责定义数据包格式和地址格式，为经过逻辑网络路径的数据进行路由选择。当然，网络层也是本书内容涉及最多的一层。

与 OSI 传输层相对应的是*主机到主机层*，它指定了控制 Internet 层的协议，这就像数据链路层控制物理层一样。主机到主机层和数据链路层都定义了流控和差错控制机制。二者不同之处在于，数据链路层协议强调控制数据链路上的流量，即连接两台设备的物理介质上的流量；而传输层控制逻辑链路上的流量，即两台设备的端到端连接，这种逻辑连接可能跨越一连串数据链路。

*应用层*与 OSI 的会话层、表示层、应用层相对应。虽然一些路由选择协议使用这一层，例如边界网关协议（BGP）、路由选择信息协议（RIP）等，[1]但是应用层最常用的服务是向用户应用提供访问网络的接口。

对于图 1-1 中所示的协议簇和其他协议簇来说，各层之间多路复用是一个通用功能。许

[1] BGP 是一个应用层的协议，因为它使用 TCP 端口传送它的消息；而 RIP 协议也是应用层协议的原因是因为使用 UDP 接口传递它的消息。其他的路由选择协议如 OSPF，称为 Internet 层的协议是因为它们直接在 IP 数据包中封装它们的消息。

多应用可以使用主机到主机层的一个服务，同样许多主机到主机层的服务也可以使用 Intenret 层。多个协议簇（如 IP、IPX、AppleTalk）还可以通过数据链路协议共享一条物理链路。

1.2 IP 包头

图 1-2 给出了 IP 包头（Packet Header）的格式，相应标准见 RFC791。数据包中的大多数字段对路由选择都很重要。

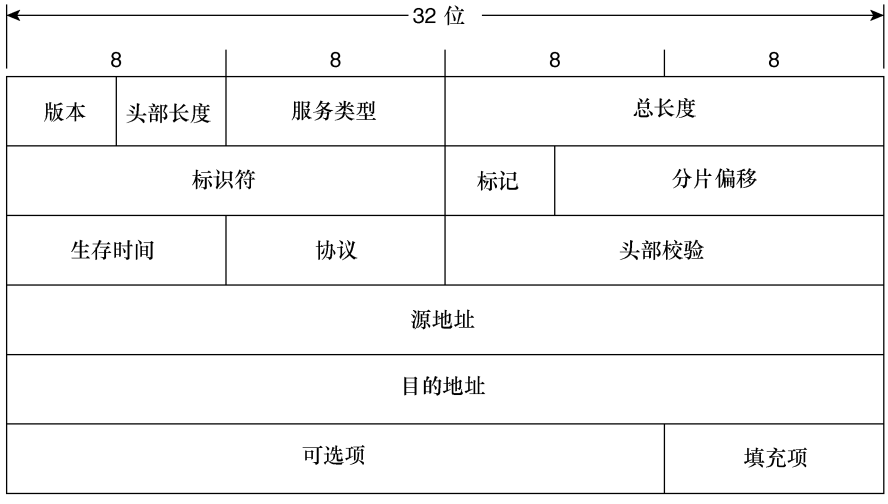


图 1-2 IP 数据包协议

- **版本（Version）**——标识了数据包的 IP 版本号。这个 4 位字段的值设置为二进制的 0100 表示 IP 版本 4（IPv4），设置为 0110 表示 IP 版本 6（IPv6）。本章主要涉及的是 IPv4，在第 2 章中主要讲述 IPv6。在表 1-1 中列出了所有已分配的现行版本号及相关 RFC。除 4 和 6（早期提出的简单 Internet 协议——即 SIP 协议，也使用版本号 6）之外，其他的所有版本号仅作为“历史产物”而存在，感兴趣的读者可以阅读相关的 RFC。

表 1-1 IP 版本号

版本号	版本	RFC
0	保留	
1～3	未分配	
4	Internet 协议版本 4（IPv4）	791
5	ST 数据报（Datagram）模式	1190
6	简单 Internet 协议（SIP）	
6	IPv6	1883
7	TP/IX	1475

续表

版本号	版本	RFC
8	P Internet 协议（PIP）	1621
9	使用更大地址的 TCP 和 UDP（TUBA）	1347
10～14	未分配	
15	保留	

- **报头长度（header length）**——字段长度为 4 位，正如字段名所示，它表示 32 位字长的 IP 报头长度。设计报头长度字段是因为数据包的可选项字段（在本节后面部分将会讨论）的大小会发生变化。IP 报头最小长度为 20 个八位组，最大可以扩展到 60 个八位组——通过这个字段也可以描述 32 位字的最大长度。
- **服务类型（Type of Service，ToS）**——字段长度为 8 位，它用来指定特殊的数据包处理方式。服务类型字段实际上被划分为两个子字段：优先权和 ToS。优先权用来设置数据包的优先级，这就像邮寄包裹一样，可以是平信、隔日送到或两日内送到。ToS 允许按照吞吐量、时延、可靠性和费用方式选择传输服务。虽然 ToS 字段通常不用（所有位均被设置为 0），但是在开放式最短路径优先（OSPF）协议的早期规范中还是称为 ToS 路由选择的。优先权位偶尔在服务质量（QoS）应用中使用。图 1-3 的 a 部分简要地说明了 8 个 ToS 位，更详细的信息可以参见 RFC1340 和 RFC1349。

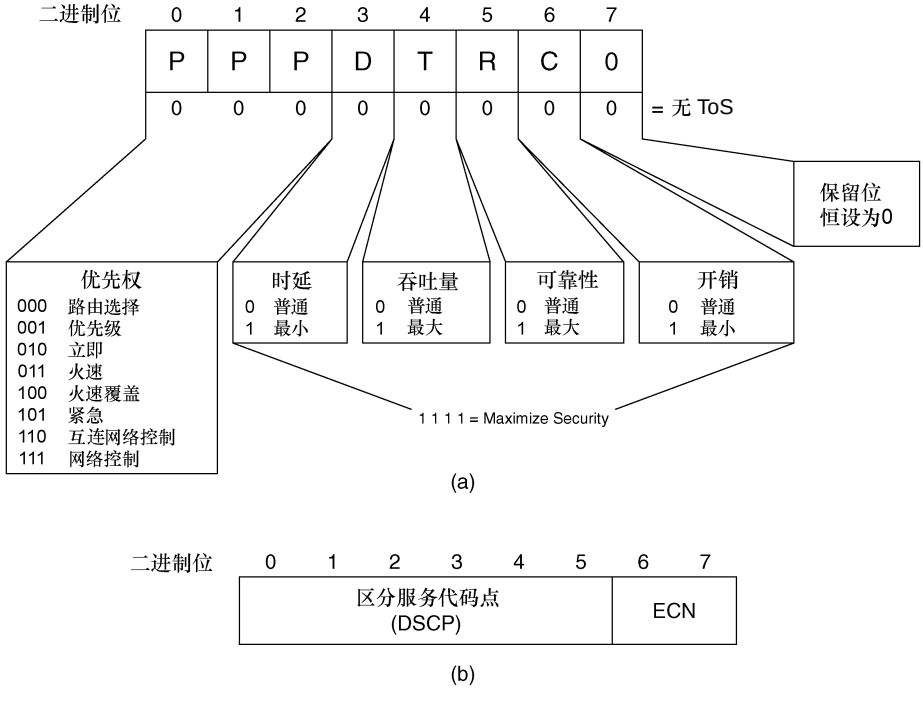


图 1-3 服务类型字段（a）或区分服务和 ECN（b）字段

在最近几年，ToS 字段已经作为区分服务（Diffserv）架构的一部分被重新定义了。[1] 区

[1] K.Nichols、S.Blake、F.Baker 和 D.Black，“Definition of the Differentiated Services Field （DS Field）in the IPv4 and IPv6 Headers”，RFC 2474，1998 年 12 月。

分服务架构为IP数据包所创建的处理比通过相对严格的ToS定义所允许的处理灵活得多。在DiffServ 下，我们能够在一台路由器上定义服务分类，将数据包归类到这些分类中去。路由器可以根据它们的分类使用不同的优先级对数据包进行排序和转发。每一个排序和转发的处理称为一个 *Per-Hop Behavior*（PHB）。虽然 DiffServ 定义了这个架构或体系，但这个机制本身称为区分服务类别或简单地称为服务类别（CoS）。

图 1-3 中的（b）部分显示了 ToS 字段是如何重新定义的，开始的 6 个位现在构成了区分代码点（DiffServ Code Point，DSCP）。利用这 6 个位，我们可以使用任意数值或根据在区分服务体系结构中预先定义的服务类别，最多可以定义 64 个不同的服务类别，并可整理到 PHB 中。请注意，在 IP 报头中的这个字段保留了 8 位；区分服务体系结构重新定义了路由器对这个字段中数值的解释。

- **显式拥塞通知（Explicit Congestion Notification，ECN）**——在图 1-3 中的显式拥塞通知是某些路由器用来支持显式拥塞通知的，当它支持该特性时，这些位可以用于拥塞信号（ECN＝11）。[1]
- **总长度（Total Length）**——数据包总长度字段的长度为 16 位，以八位组为单位计，其中包括 IP 报头。 接收者用 IP 数据包总长度减去 IP 报头长度，就可以确定数据包数据有效载荷的大小。16 位长的二进制数用十进制表示最大可以为 65 535，所以 IP 数据包的最大长度是 65 535。
- **标识符（Identifier）**——字段长度为 16 位，通常与标记字段和分段偏移字段一起用于数据包的分段。如果数据包原始长度超过数据包所要经过的数据链路的最大传输单元（MTU），那么必须将数据包分段为更小的数据包。例如，一个大小为 5 000 字节的数据包在穿过网络时，如果遇到一条 MTU 为 1 500 字节的数据链路，即数据帧最多容纳大小为 1 500 字节的数据包。路由器需要在数据成帧之前将数据包分段成多个数据包，其中每个数据包长度不得超过 1 500 字节；然后路由器在每片数据包的标识字段上打上相同的标记，以便接收设备可以识别出属于一个数据包的分段。[2]
- **标记字段（Flag）**——长度为 3 位，其中第 1 位没有使用。第 2 位是不分段（DF）位。当 DF 位被设置为 1 时，表示路由器不能对数据包进行分段处理。如果数据包由于不能被分段而未能被转发，那么路由器将丢弃该数据包并向源点发送错误消息。这一功能可以在网络上用于测试 MTU 值。参见示例 1-1 所示，在 IOS 软件系统中，使用扩展 Ping 工具可以对 DF 位进行设置。

示例 1-1 为了测试穿越网络的 MTU 值，IOS 软件中的扩展 Ping 工具允许设置 DF 位。在 ping 的输出信息中，到达目的地 172.16.113.17 的路径的最大 MTU 为 1 478 字节

```
Handy#ping
Protocol [ip]:
Target IP address: 172.16.113.17
Repeat count [5]: 1
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address:
```

（待续）

[1] K.Ramakrishnan，“The Addition of Explicit Congestion Notification （ECN）to IP，”RFC3168，2001 年 9 月。

[2] 被分段的数据包不会在数据链路的另一端被重组，而是一直保持分段状态，直至到达最终目的地时才会被重组。

```
Type of service [0]:
Set DF bit in IP header? [no]: y
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]: r
Number of hops [9]:
Loose, Strict, Record, Timestamp, Verbose[RV]:
Sweep range of sizes [n]: y
Sweep min size [76]: 500
Sweep max size [18024]: 2000
Sweep interval [1]: 500
Type escape sequence to abort.
Sending 4, [500..2000]-byte ICMP Echos to 172.16.113.17, timeout is 2 seconds:
Packet has IP options:  Total option bytes= 39, padded length=40
 Record route: <*> 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
         0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0

Reply to request 0 (16 ms) (size 500).  Received packet has options
 Total option bytes= 40, padded length=40
 Record route: 172.16.192.5 172.16.113.18 172.16.113.17 172.16.113.17
         172.16.192.6 172.16.192.5 <*> 0.0.0.0 0.0.0.0 0.0.0.0
 End of list

Reply to request 1 (24 ms) (size 1000).  Received packet has options
 Total option bytes= 40, padded length=40
 Record route: 172.16.192.5 172.16.113.18 172.16.113.17 172.16.113.17
         172.16.192.6 172.16.192.5 <*> 0.0.0.0 0.0.0.0 0.0.0.0
 End of list

Reply to request 2 (28 ms) (size 1500).  Received packet has options
 Total option bytes= 40, padded length=40
 Record route: 172.16.192.5 172.16.113.18 172.16.113.17 172.16.113.17
         172.16.192.6 172.16.192.5 <*> 0.0.0.0 0.0.0.0 0.0.0.0
 End of list

Unreachable from 172.16.192.6, maximum MTU 1478 (size 2000).
 Received packet has options
 Total option bytes= 39, padded length=40
 Record route: <*> 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
         0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0

Success rate is 75 percent (3/4), round-trip min/avg/max = 16/22/28 ms
Handy#
```

第 3 位表示还有更多分段（MF）位，当路由器对数据包进行分段时，除了最后一个分段的 MF 位设置为 0 外，其他所有分段的 MF 位均设置为 1，以便接收者直到收到 MF 位为 0 的分段为止。

- **分段偏移（Fragment Offset）**——字段长度为 13 位，以 8 个八位组为单位，用于指明分段起始点相对于报头起始点的偏移量。[1]由于分段到达时可能错序，所以分段偏移字段可以使接收者按照正确的顺序重组数据包。

  请注意，如果一个分段在传输中丢失，那么必须在网络中同一点对整个数据包重新分段并重新发送。因此，容易发生故障的数据链路会造成时延不成比例。另外，如果由于网络拥塞而造成分段丢失，那么重传整组分段会进一步加重网络拥塞。
- **生存时间（Time To Live，TTL）**——字段长度为 8 位，在最初创建数据包时 TTL 即被设置为某个特定值。当数据包逐个沿路由器被传输时，每台路由器都会降低 TTL

[1] 为了使 13 位长的分段偏移字段可以表示的最大数据包长度为 65 535 字节，所以使用 8 个八位组作为本字段的单位。

的数值。当 TTL 值减为 0 时，路由器将会丢弃该数据包并向源点发送错误信息。这种方法可以防止数据包在网络上无休止地被传输。

按照最初构想，TTL 值以 s（秒）为单位。如果数据包在路由器上被延迟的时间超过 1s，路由器将会相应地调整 TTL 值。然而，这种方法实施起来十分困难，从来没有被广泛地支持。现代的路由器不管实际时延是多少，统统将 TTL 值减 1，所以 TTL 实际上是表示跳数。[1]虽然 TTL 常见的值为 15 和 32，但是建议的缺省值是 64。

像 IOS 软件中的 **trace** 命令这样的一些追踪工具使用的是 TTL 字段。如果路由器被告知需要追踪到达主机地址为 10.11.12.13 的路径，路由器将发送 3 个数据包，其中 TTL 值被设置为 1；第 1 台路由器将会把 TTL 值减少到 0，而且在丢弃数据包的同时向源点发送错误信息。源点路由器通过阅读错误信息从而得知发送错误信息的路由器即为路径上的第 1 台路由器。再一次被路由器发送的 3 个数据包的 TTL 值被设置为 2。第 1 台路由器将 TTL 值减 1，第 2 台路由器将 TTL 值再减 1 后为 0，此时源点路由器将会接收到第 2 台路由器发送来的错误信息。第 3 次发送的数据包 TTL 值被设置为 3，依此类推，直到目的地被发现。最终，沿着网络路径所有的路由器都会被标识出来。示例 1-2 中显示了 IOS 软件中路由追踪的输出结果。

示例 1-2　追踪工具使用 TTL 字段来标识沿途路由器。星号表示超时数据包

```
Elvis#traceroute www.cisco.com

Type escape sequence to abort.
Tracing the route to cio-sys.Cisco.COM (192.31.7.130)

  1 172.18.197.17  4 msec  4 msec  4 msec
  2 ltlrichard-s1-13.hwy51.com  (172.18.197.1)  36 msec  44 msec  2536 msec
  3 cperkins-rtr-fr2.hwy51.com  (10.168.204.3)  104 msec  64 msec *
  4 cberry.hwy51.com  (10.168.193.1)  92 msec  *
  5 jllewis-inner.hwy51.com  (10.168.207.59)  44 msec  *   44 msec
  6 bholly-fw-outer-rt.hwy51.com  (10.168.207.94)  44 msec  *   48 msec
  7 sl-stk-14-S10/0:6-512k.sprintlink.net  (144.228.214.107)  92 msec  *
  8 sl-stk-2-F1/0/0.sprintlink.net  (144.228.40.2)  52 msec  1156 msec  *
  9 sl-mae-w-H1/0-T3.sprintlink.net  (144.228.10.46)  100 msec  124 msec  2340 msec
 10 sanjose1-br1.bbnplanet.net  (198.32.136.19)  2264 msec  164 msec  *
 11 paloalto-br2.bbnplanet.net  (4.0.1.10)  64 msec  60 msec  *
 12 su-pr2.bbnplanet.net  (131.119.0.218)  76 msec  76 msec  76 msec
 13 cisco.bbnplanet.net  (131.119.26.10)  2560 msec  76 msec  936 msec
 14 sty.cisco.com  (192.31.7.39)  84 msec  72 msec  *
 15 cio-sys.Cisco.COM  (192.31.7.130)  60 msec  *   64 msec
Elvis#
```

- **协议（Protocol）**——字段长度为 8 位，它给出了主机到主机层或传输层协议的“地址”或协议号，协议字段指定了数据包中信息的类型。当前已分配了 100 多个不同的协议号，表 1-2 给出了其中一些较常用的协议号。

表 1-2　一些众所周知的协议号

协议号	主机到主机层协议
1	Internet 消息控制协议（ICMP）
2	Internet 组管理协议（IGMP）
4	被 IP 协议封装的 IP

[1] 正如读者将在第 2 章中所读到的，在 IPv6 报头中等价的字段已经重新命名为 Hop Limit，以便更加确切地反映它的真正用途。

续表

协议号	主机到主机层协议
6	传输控制协议（TCP）
17	用户数据报协议（UDP）
45	域间路由选择协议（IDRP）
46	资源预留协议（RSVP）
47	通用路由选择封装（GRE）
54	NBMA 下一跳解析协议（NHRP）
88	Cisco Internet 网关路由选择协议（IGRP）
89	开放式最短路径优先（OSPF）

- **报头校验和（Header Checksum）**——是针对 IP 报头的纠错字段。校验和不计算被封装的数据，UDP、TCP 和 ICMP 都有各自的校验和。报头校验和字段包含一个 16 位二进制补码和，这是由数据包发送者计算得到的。接收者将连同原始校验和重新进行 16 位二进制补码和计算。如果数据包传输中没有发生错误，那么结果应该 16 位全部为 1。回忆前面所述内容，由于每台路由器都会降低数据包的 TTL 值，所以每台路由器都必须重新计算校验和。RFC1141 讨论了一些简化计算的策略。
- **源地址和目的地址（Source and Destination Address）**——字段长度为 32 位，分别表示发送者数据包源点和目的地的 IP 地址。IP 地址的格式将会在 1.3 节中讨论。
- **可选项（Options）**——是一个长度可变的字段，并像其名字所表示的，它是可选的。可选项被添加在包头中，包括源点产生的信息和其他路由器加入的信息；可选项字段主要用于测试。常用的可选项如下：
  - **松散源路由选择（Loose Source Routing）**——它给出了一连串路由器接口的 IP 地址序列。数据包必须沿着 IP 地址序列传送，但是允许在相继的两个地址之间跳过多台路由器。
  - **严格源路由选择（Strict Source Routing）**——它也给出了一系列路由器接口的 IP 地址序列。不同于松散源路由选择，数据包必要严格按照路由转发。如果下一跳不再列表中，那么将会发生错误。
  - **记录路由（Record Route）**——当数据包离开时为每台路由器提供空间记录数据包的出站接口地址，以便保存数据包经过的所有路由器的记录。记录路由选项提供了类似于路由追踪的功能，但是不同点在于这里记录了双向路径上的出站接口信息。
  - **时间戳（Timestamp）**——除了每台路由器还会记录一个时间戳之外，时间戳选项十分类似于记录路由选项，这样数据包不仅可以知道自己到过哪里，而且还可以记录到达的时间。

在 Cisco 路由器上使用扩展的 Ping 工具可以调用所有这些选项。示例 1-1 中使用了记录路由选项，示例 1-3 使用了松散源路由选择和时间戳选项，严格源路由选择选项在示例 1-4 中被使用。

示例 1-3 使用 Cisco 的扩展 Ping 工具来设置 IP 报头中的可选项字段的各项参数。在这个例子中，用到了松散源路由选择选项和时间戳选项

```
Handy#ping
Protocol [ip]:
Target IP address: 172.16.113.9
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address:
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]: l
Source route: 172.16.113.14 172.16.113.10
Loose, Strict, Record, Timestamp, Verbose[LV]: t
Number of timestamps [ 6 ]: 2
Loose, Strict, Record, Timestamp, Verbose[LTV]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.113.9, timeout is 2 seconds:
Packet has IP options: Total option bytes= 23, padded length=24
  Loose source route: <*> 172.16.113.14 172.16.113.10
  Timestamp: Type 0. Overflows: 0 length 12, ptr 5
    >>Current pointer<<
    Time= 0
    Time= 0

Request 0 timed out
Reply to request 1 (76 ms). Received packet has options
  Total option bytes= 24, padded length=24
  Loose source route: 172.16.113.13 172.16.192.6 <*>
  Timestamp: Type 0. Overflows: 6 length 12, ptr 13
    Time= 80FF4798
    Time= 80FF4750
    >>Current pointer<<
  End of list

Request 2 timed out
Reply to request 3 (76 ms). Received packet has options
  Total option bytes= 24, padded length=24
  Loose source route: 172.16.113.13 172.16.192.6 <*>
  Timestamp: Type 0. Overflows: 6 length 12, ptr 13
  Time= 80FF4FC0
  Time= 80FF4F78
  >>Current pointer<<
End of list

Request 4 timed out
Success rate is 40 percent (2/5), round-trip min/avg/max = 76/76/76 ms
Handy#
```

示例 1-4 这里使用扩展 Ping 在 ping 数据包中设置严格源路由选择选项

```
Handy#ping
Protocol [ip]:
Target IP address: 172.16.113.10
Repeat count [5]: 2
Datagram size [100]:
```

（待续）

```
Timeout in seconds [2]:
Extended commands [n]: y
Source address:
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]: s
Source route: 172.16.192.6 172.16.113.17 172.16.113.10
Loose, Strict, Record, Timestamp, Verbose[SV]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 2, 100-byte ICMP Echos to 172.16.113.10, timeout is 2 seconds:
Packet has IP options: Total option bytes= 15, padded length=16
  Strict source route: <*> 172.16.192.6 172.16.113.17 172.16.113.10

Reply to request 0 (80 ms). Received packet has options
  Total option bytes= 16, padded length=16
  Strict source route: 172.16.113.10 172.16.113.17 172.16.192.6 <*>
  End of list

Reply to request 1 (76 ms). Received packet has options
  Total option bytes= 16, padded length=16
  Strict source route: 172.16.113.10 172.16.113.17 172.16.192.6 <*>
  End of list

Success rate is 100 percent (2/2), round-trip min/avg/max = 76/78/80 ms
Handy#
```

- **填充（Padding）**——该字段通过在可选项字段后面添加 0 来补足 32 位，这样保证报头长度是 32 位的倍数。

示例 1-5 显示了协议分析器捕获到的 IP 报头的信息。请与图 1-2 中的信息作一下比较。

示例 1-5　在协议分析器的窗口中，可以看到 IP 包头各字段及每个字段的值

```
Internet Protocol, Src Addr: 172.16.1.102 (172.16.1.102), Dst Addr: 224.0.0.5
(224.0.0.5)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6; ECN: 0x00)
    Total Length: 64
    Identification: 0x6e61 (28257)
    Flags: 0x00
    Fragment offset: 0
    Time to live: 1
    Protocol: OSPF IGP (0x59)
    Header checksum: 0xbcc8 (correct)
    Source: 172.16.1.102 (172.16.1.102)
    Destination: 224.0.0.5 (224.0.0.5)
```

1.3　IPv4 地址

IPv4 地址长度为 32 位。像所有其他网络层地址一样，IPv4 地址也包括网络号和主机号两部分。网络号部分惟一地标识了一条物理链路或逻辑链路，对于与该链路相连的所有设备来说网络号部分是共同的。而主机号部分惟一地标识了该链路上连接的具体设备。

有几种方式可以表示 IP 地址的 32 位。举例来说，32 位的 IP 地址 00001010110101100101011110000011 可以用十进制表示为 181 819 267。

由此可见用二进制表示 IP 地址十分麻烦，而全部 32 位数字用十进制格式表示计算起来又很耗时。图 1-4 给出了一种更好的表示方法。

32 位的地址包含 4 个字节，每个字节均可以用 0~255 之间的十进制数表示，而每个十进制数之间用点号分隔。在图 1-4 中，将 32 位的地址映射到用点分十进制法表示的地址上。[1]

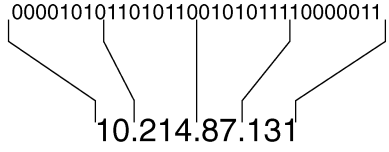


图 1-4 虽然使用点分十进制表示法书写 IP 地址十分方便，但是不要将它与路由器（或主机）所看到的 32 位字符串混淆起来

在使用 IPv4 地址时需要记住一点，点分十进制表示法便于人们阅读和书写，而路由器更适合使用 32 位的二进制串，而不是按照 4 个八位组的方式读取地址。牢记这一点可以避免许多易犯的错误。如果读者没有接触过二进制数，特别是二进制与十进制之间的转换方法，那么建议读者在继续学习这一章的内容之前可以阅读一下附录 A 的内容。

IP 地址与众不同的特性可能就是，IPv4 地址不像其他网络层地址的网络号和主机号是固定不变的，IP 地址的网络号和主机号可以在 32 位的界线内发生变化。也就是说，IP 地址的网络号和主机号都有可能占据 32 位中的大多数位，也可能两者平分 32 位。例如 NetWare 和 AppleTalk 协议，由于它们主要用于相对较小的网络，所以协议的网络层地址的网络号和主机号长度是固定。这样的安排的确使得工作更加容易，接收设备可以从地址中读入固定的位来获取网络号，剩下的位便是主机号。

然而，TCP/IP 从最初设计出来到现在可以灵活地应用于任何网络，从很简单的几个功能发展成为一个庞大的协议簇。TCP/IP 这种适应性使得 IP 地址的管理更加困难。本节仅介绍了 IP 地址管理的一些基本内容，在第 6 章中将会介绍一些更高级的技术。

1.3.1 首个八位组字节规则

如果不对网络作太过精确的划分，那么网络可以按照主机数量分为 3 类：大型网络、中型网络和小型网络。

- **大型网络**——可以定义为包含大量主机的网络。大型网络的数量相对很少。
- **小型网络**——作为大型网络的对照，它仅仅包含很少数量的主机，但小型网络的数目很多。
- **中型网络**——相对于大型和小型网络来说，包含的主机数量中等，而且中型网络的数量也中等。

对于这 3 种规模的网络，高层的地址划分要求有 3 种类型的网络地址。面向大型网络的地址需要有能力为大量的主机编址，但是由于大型网络的数量有限，所以大型网络仅需要少量的网络地址。

而对小型网络来说情况又颠倒过来了，因为小型网络数量庞大，所以需要大量的小型网络的网络地址。但是小型网络主机有限，所以仅需要少量主机地址。

[1] 点分十进制表示法只用于 IPv4 地址。在第 2 章读者将会看到，IPv6 地址的表示法完全不同。

对于中等规模的网络来说，网络地址和主机地址的需求量均趋于中等水平。

图 1-5 显示了 3 类 IPv4 地址的网络号和主机号是怎样划分的。

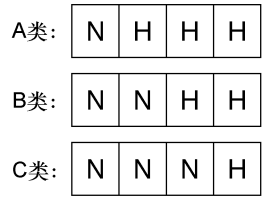


图 1-5 A、B 和 C 类 IPv4 地址格式

迄今为止，对于所描述的大型、中型和小型网络，是按照如下方式映射到各类地址的：

- **A 类地址**——用于大型网络，第 1 个八位组是网络号，后 3 个八位组是主机号。8 位的网络号最多可以表示 256 个网络，而每个网络地址的主机号可以提供的主机数量为 2^{24} 或 16 777 216。
- **B 类地址**——用于中型网络。前 2 个八位组表示网络号，后 2 个八位组表示主机号。网络号和主机号的数量均为 2^{16} 或 65 536 个。
- **C 类地址**——对应于 A 类 IP 地址。前 3 个八位组表示网络号，最后 1 个八位组表示主机号。

因为所有的 IPv4 地址都是 32 位二进制字符串，所以需要某种方法来区分一个特定地址到底是属于哪一类地址。表 1-3 所示的首个八位组规则提供了这种方法，如下所述：

- 对于 A 类地址，首个八位组的第 1 位，即 32 位字符串最左边的 1 位，总是被设置为 0。因此，通过设置首个八位组的剩余位为 0（最小）或为 1（最大），我们可以找到 A 类地址范围中的最小数和最大数。于是我们可以得到最小数和最大数分别为 0 和 127，但是这里有几个例外：0 被保留作为缺省地址部分（参见第 12 章），127 被保留为内部回送地址。[1]剩下的十进制数则是 1~126。因此任何首个八位组落在 1 和 126 之间的 IP 地址均属于 A 类地址。

表 1-3 首个八位组字节规则

规则	最小值与最大值	十进制范围
A 类：第一位恒为 0	**0**0000000=0 **0**1111111=127	1～126*
B 类：第一、二位恒为 10	**10**000000=128 **10**111111=191	128～191
C 类：第一、二、三位恒为 110	**110**00000=192 **110**11111=223	192～223

* 0 和 127 保留。

- B 类地址总是把左边的第 1 位设置为 1，第 2 位设置为 0。那么再次通过设置首个八位组的剩余位为 0 或为 1，我们依然可以找到最小数和最大数。在图 1-4 中，我们可以看到首个八位组落在 128 和 191 之间的 IP 地址属于 B 类地址。
- 在 C 类地址中，前 2 位均被设置为 1，第 3 位被设置为 0。这样设置的结果是首个

[1] 设备使用环回地址（典型的是 127.0.0.1）向自己发送流量。发送到该地址的数据将会被直接送回给发送进程，而不会离开此设备。

八位组在 192 和 223 之间。[1]

到目前为止，IPv4 的编址看上去并不是十分困难。路由器和主机通过首个八位组字节规则能够很容易地确定 IP 地址的网络号。如果第 1 位是 0，需要读取前 8 位才能获取网络地址；如果前两位是 10，那么需要读取 16 位；如果前 3 位是 110，则需求读取 24 位才能获取网络号。不幸的是，事情并不会这样简单。

1.3.2 地址掩码（Address Mask）

表示整个数据链路的地址——非特指某台主机的网络地址，可以用 IP 地址的网络部分来表示，其中主机位全部为 0。例如，IP 地址管理机构 [2]可以将 172.21.0.0[3]分配给一个申请者。因为 172 在 128 和 191 之间，所以这是一个 B 类地址，其中后两个八位组作为主机位，全部被设置为 0。虽然前 16 位（172.21.）已经被指定，但是地址所有者有权决定后 16 位主机位的使用。

每一台设备和接口都将被分配一个惟一的、主机号明确的地址，例如 172.21.35.17。不管设备是路由器还是主机，显然都需要知道自身的地址，而且它还需要能够确定它所属的网络，在这个案例中，它属于 172.21.0.0。

这一任务通常由地址掩码来完成。地址掩码是一个 32 位的字符串，与 IPv4 地址的每一位相对应。掩码也可以像 IPv4 地址一样用点分十进制表示。这种表示方法会成为某些初学者的绊脚石。虽然地址掩码可以用点分十进制书写，但是它并不是一个地址。表 1-4 给出了对应于 3 类 IPv4 地址的标准地址掩码。

表 1-4　　A 类、B 类和 C 类 IPv4 地址的地址掩码

类	掩码	点分十进制表示
A	11111111000000000000000000000000	255.0.0.0
B	11111111111111110000000000000000	255.255.0.0
C	11111111111111111111111100000000	255.255.255.0

对于每一位 IPv4 地址位，设备会拿它与地址掩码的对应位进行布尔（逻辑）AND 操作。AND 函数表述如下：

比较两位并得出结果。当且仅当两位全部为 1 时，结果为 1。如果两位中任意一位为 0，则结果为 0。

对于一个指定的 IPv4 地址，图 1-6 给出了怎样用地址掩码确定网络地址。地址掩码值为 1 的位对应于地址的网络位，值为 0 的位对应于主机位。因为 172.21.35.17 是 B 类地址，所以掩码前两个八位组必须全部设置为 1，后两个八位组，即主机号的所有位必须设置为 0。参见表 1-4，这个掩码的点分十进制表示为 255.255.0.0。

在 IPv4 地址和地址掩码的每一位上执行逻辑“与”（AND）操作，结果如图 1-6 所示。在结果中，网络位不变，所有主机位则变为 0。通过向接口分配地址 172.21.35.17 和掩码

[1] 注意，223 并没有用完第一个八位组中所有可用的数。参见本章最后的配置练习 1。

[2] 负责管理和分配 IP 地址的高级管理机构是亚洲的 APNIC、北美的 ARIN、中美与南美州的 LACNIC，以及 EMEA 的 RIPE。

[3] 事实上，这个地址决不会被分配，因为它属于私有的保留地址；本书中所用到的大多数地址都是保留地址，见 RFC1918。保留地址包括：10.0.0.0-10.255.255.255、 172.16.0.0-172.31.255.255 和 192.168.0.0-192.168.255.255。

255.255.0.0，设备将会知道接口属于网络 172.21.0.0。对 IPv4 地址和掩码应用 AND 操作总能够得到网络地址。

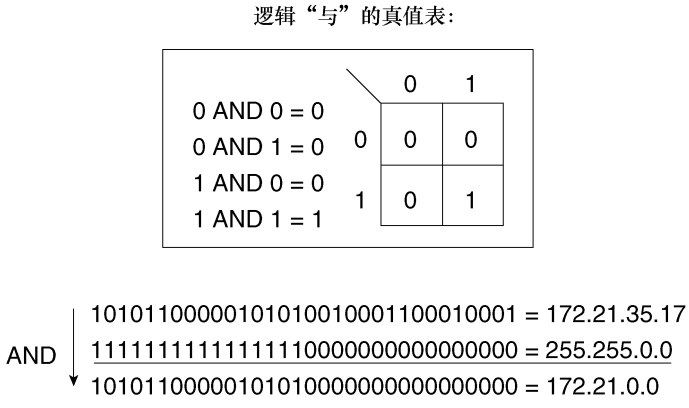


图 1-6 B 类地址的每一位与地址掩码的对应位进行 AND 操作，然后得到网络地址

通过下面命令可以向 Cisco 路由器的接口分配地址和掩码（本例中接口为 E0）：

```
Smokey(config)# interface ethernet 0
Smokey(config-if)# ip address 172.21.35.17 255.255.0.0
```

但是为什么要使用地址掩码？到目前为止，使用首个八位组字节规则看上去更简单一些。

1.3.3 子网和子网掩码

首先，决不要忽略网络层地址的必要性。为了完成路由选择，每个数据链路（网络）都必须有一个惟一的地址；另外，数据链路上的每台主机也必须有一个地址，这个地址不仅标识主机为一个网络成员，还可以把主机与网络上的其他主机区分开来。

到目前为止的定义中，一个 A 类、B 类或 C 类地址仅仅能用在一个单一网络中；为了建立一个网络，每个数据链路都必须使用不同的地址，以便这些网络可以被惟一地标识。如果每一个数据链路都使用一个单独的 A 类、B 类或 C 类地址，那么即使用尽所有的 IPv4 地址，也只能给少于 1700 万个数据链路分配地址。显然，这种方法是不切实际的，[1]在前面的例子中，如果充分地使用主机地址空间，那么在数据链路 172.21.0.0 中的设备数目可以超过 65 000！

使 A 类、B 类或 C 类地址实用化的惟一方法是对主网地址进行划分，例如将 172.21.0.0 划分为子网地址。请回忆两个事实：

- IPv4 地址的主机部分可以随意使用。
- IPv4 地址的网络号由分配给接口的地址掩码确定。

如图 1-7 所示，分配给网络的地址为 B 类地址 172.21.0.0。5 个数据链路将主机和路由器互连起来，每个数据链路都需要一个网络地址。按照目前的情况，172.21.0.0 必须分配给其中的一个数据链路，那么另外 4 个数据链路还需要 4 个地址。

注意图 1-7 所示，地址掩码并不是标准的 16 位 B 类地址掩码；而是被扩展了 8 位，以便 IP 地址的前 24 位都被解释为网络位。换句话说，掩码使路由器和主机把读取的前 8 位主

[1] 1700 万个数据链路看上去很多，但是你要考虑到，一个中等规模的企业就可能有许多数据链路。

机位作为网络地址的一部分。结果是，主网络地址应用于整个网络，而每一个数据链路则变为一个子网（subnet）；一个子网是一个主 A 类、B 类或 C 类地址空间的一个子集。

现在，IPv4 地址包括 3 个部分：网络部分、子网部分和主机部分。地址掩码现在变为子网掩码，或比标准地址掩码长的掩码。地址的前两个八位组依然是 172.21，但是第 3 个八位组——主机位已经由子网位代替——的变化范围为 0~255。在图 1-6 中的网络有子网 1、2、3、4 和 5（172.21.1.0~172.21.5.0）。在单一 B 类地址下最多可以有 256 个子网，对应的掩码如图 1-7 所示。

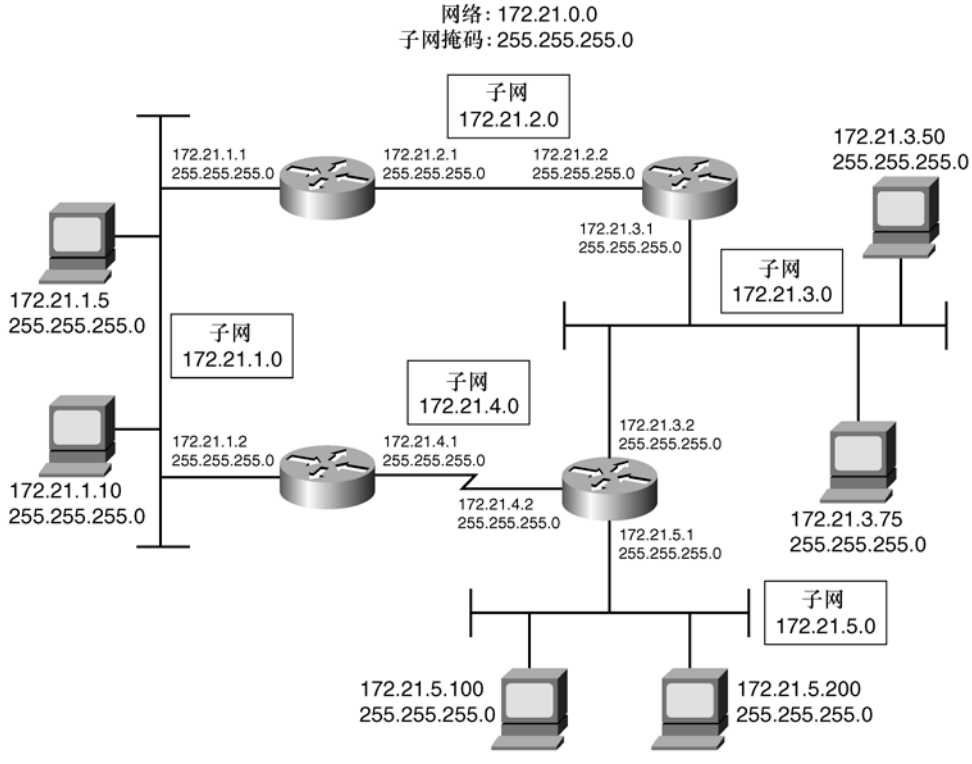


图 1-7 通过向主机位借位用作子网位，子网掩码使一个单一的网络地址可以用于多个数据链路

下面给出两点告诫。首先，并不是所有路由选择协议都支持子网地址，即子网位全 0 或全 1。因为这些协议是有类别化协议，它们不能区分一个全 0 子网和主网络号。例如，在图 1-7 中子网 0 为 172.21.0.0；而主网 IP 地址也为 172.21.0.0。没有更多信息将无法区分二者。

同样的，有类别路由选择协议也不能区分全 1 子网的广播地址和一个所有子网的广播地址。[1] 例如，图 1-7 中的全 1 子网为 172.21.255.0。对于这个子网，广播地址是 172.21.255.255，但是这也是在主网 172.21.0.0 的所有子网上所有主机的广播地址。没有更多的信息也无法区分二者。第 1 版 RIP 协议和 IGRP 协议都是有类别路由选择协议；第 7 章将会介绍无类别路由选择协议，这种路由选择协议才可以真正地使用全 0 或全 1 子网。

其次是与子网及其掩码的口头表述有关。在图 1-7 中，对 B 类地址的第 3 个八位组进行子网划分是非常普遍的，但还常常听到人们这样表述子网设计：“B 类地址使用 C 类地址掩码”，或者“将 B 类地址划分为 C 类地址”。这两种表述都是错误的。它们常常会对子网设计

[1] 所有主机的 IP 广播地址是所有位全为 1：255.255.255.255。特定子网的广播地址是所有主机位全为 1；例如，子网 172.21.1.0 的广播地址是 172.21.1.255。最后，对于所有子网的所有主机来说，广播地址是子网为和主机位均为 1：172.21.255.255。

引起误解或者是不准确的理解。对于图 1-6 中所示的子网划分图解的正确表述应该是“一个使用 8 位进行子网划分的 B 类地址”或者“一个带有 24 位掩码的 B 类地址”。

可以用以下 3 种格式中的任何一种表示子网掩码：

点分十进制：255.255.255.0

位计数：172.21.0.0/24

十六进制：0xFFFFFF00

虽然位计数格式变得渐渐流行起来，但是点分十进制暂时一段时期仍旧经常使用在一些软件里面。与点分十进制相比，位计数格式更容易书写（地址后面是/，/后面紧跟着是网络部分的位计数）。另外，位计数格式可以更清楚地描述掩码的实际作用，因而可以避免前面段落出现的语义误解问题。某些 UNIX 系统使用十六进制格式。

虽然在 Cisco 路由器中必须使用点分十进制方式表示地址掩码，但是在行配置模式下使用命令 **ip netmask-format**[**decimal**|**hexadecimal**|**bit-count**]，可以设置使用 3 种格式中的任何一种格式显示掩码。例如，为使路由器以位计数格式显示掩码，配置如下：

```
Gladys(config) # line vty 0 4
Gladys(config-line)# ip netmask-format bit-count
```

1.3.4 子网规划

如前面部分所述，在有类别地址环境中，子网位不能全部为 0 或全部为 1。同样的，一个主机的 IPv4 地址也不能将主机位全部设置为 0，这种用法是为路由器保留的，用于表示网络和子网自身。当然 IPv4 地址的主机位也不能全部被设置为 1，因为它用于表示广播地址。所有这些限制无一例外地适用于 IP 地址的主机位，并且这也是子网规划的起点。除了这些限制，网络设计人员还需要根据地址空间与网络详细的匹配程度来选择最合理的子网划分方案。

在规划子网和子网掩码时，可以使用相同的公式计算一个主网地址下可用的子网数以及每个子网内可用的主机数，公式为：2^n-2，其中 n 表示子网位数或主机空间，2 表示减去全 0 和全 1 两个不可用地址。例如，给定一个 A 类地址 10.0.0.0，子网掩码 10.0.0.0/16（255.255.0.0）意味着有 8 位子网空间，也就是可以产生 2^8-2 ＝254 个子网，每个子网可以有 $2^{16}-2$＝65 534 个主机地址。 另一方面，掩码 10.0.0.0/24（255.255.255.0）表示有 16 位子网空间，可以产生 65 534 个子网，其中 8 位主机空间可以在某个子网中产生 254 个主机地址。

下面是 IPv4 地址子网划分的步骤：

步骤 1：确定需要多少个子网，每个子网需要多少台主机。

步骤 2：为了满足第 1 步提出的需求，使用公式 2^n-2 确定子网位数和主机位数。如果存在多个子网掩码可以满足第 1 步需求，那么选择最能够符合未来需求的一个。例如，如果网络最有可能通过增加子网发展起来，那么选择子网位最多的掩码；如果网络最有可能借助增加现有子网内的主机数发展起来，则选择主机位最多的掩码。为了避免所选择的方案中的子网及子网内的主机地址被迅速地用完，需要为将来的发展预留一些空间。

步骤 3：使用二进制进行计算，在子网空间中确定所有的位组合方式。在每种组合方式中，将所有主机位都设置为 0，将得到的子网地址转换为点分十进制格式。最终结果就是子网地址。

步骤 4：对于每一个子网地址，再次使用二进制，在保持子网位不变的情况下写出所有主机位组合，并将结果转换成点分十进制格式。最终结果就是每个子网的可用主机地址。

这里没有过分强调在最后两步中使用二进制的重要性。当进行子网划分时，最主要的惟一错误根源就是，在没有理解在二进制上会发生什么的情况下试图使用点分十进制方法。此外，点分十进制表示法对于人们读写 IPv4 地址十分方便。但是路由器和主机却把地址看作 32 位二进制字符串；为了顺利地完成地址操作，必须采用路由器和主机处理地址的方式。

就目前给出的例子而言，作者在前面的段落中似乎有点多虑了。在没有限定必须使用二进制方式表示地址和掩码的时候，子网模式和主机地址看上去还是十分清楚的。下面小节将讨论使用 4 个步骤完成子网规划，在那里点分十进制表示法将十分不明确。

1.3.5 打破八位组界线

到目前为止，在给出的例子中，子网空间都是以八位组为界线的。但这并不总是最实用或最有效的选择。例如，如果你需要对一个 B 类地址进行子网划分，并满足以下需求：数据链路数为 500，每个数据链路内主机数不超过 100 台，应该怎么办？这样的需求很容易得以满足，只要使用 9 位子网位，就可以得到 $2^9-2=510$ 个子网，剩下 7 位做主机位，每个子网的可用主机数为 $2^7-2=126$。除此不再有其他位组合可以满足上面的需求。

请注意，如果还是以八位组为界线的话，那么将无法对 C 类地址进行子网划分。如果要这样做就会占用最后 1 个八位组，那么就没有更多主机位了。因此，如下面的例子所示，子网位和主机位必须共享最后 1 个八位组。

图 1-8 与图 1-7 中显示的网络除了分配的地址是 C 类地址 192.168.100.0 之外，其他完全相同。

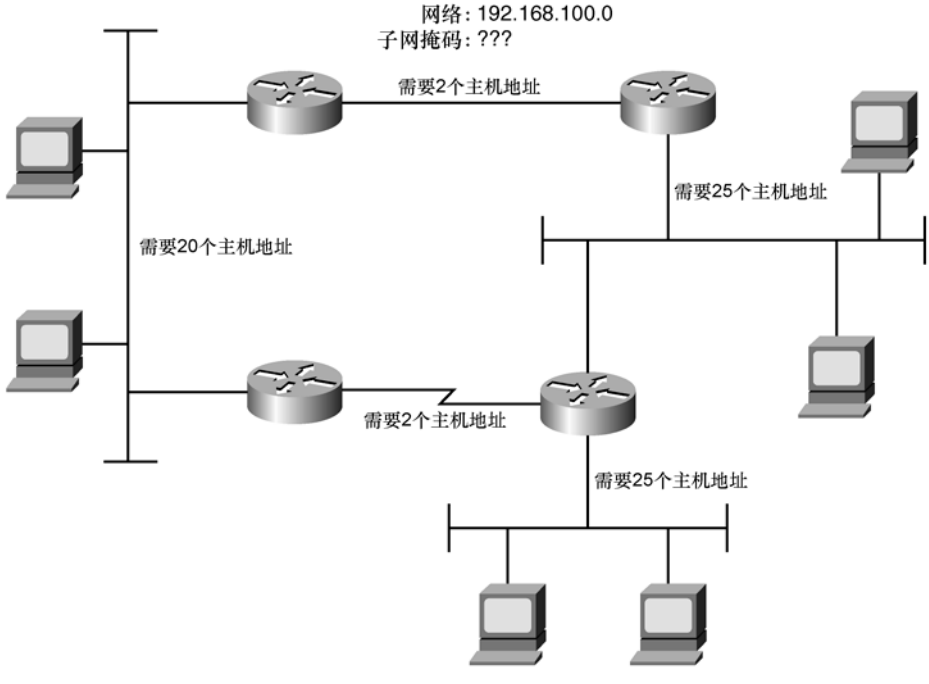


图 1-8 除了分配 C 类地址之外，这里沿用图 1-7 给出的网络。如果子网位占用整个八位组，那么将无法进行划分，因为主机位将没有空间

在这个网络中共有 5 条数据链路，因此至少需要划分出 5 个子网地址。图中还指明了每个子网需要分配的主机数（包括路由器接口）。其中两个以太网最多需要 25 个主机地址。所以完整的子网划分最小需求是 5 个子网，每个子网至少需要 25 个主机地址。

使用公式 2^n-2 可以计算出，3 个子网位和 5 个主机位即可以满足需求：$2^3-2=6$，$2^5-2=30$。带有 3 位子网位的 C 类地址掩码可以用点分十进制表示为 255.255.255.224。

图 1-9 给出了子网位的推导过程。用二进制表示第 2 步计算出的子网掩码，子网掩码下面是 IP 地址。垂直线标记了子网空间，从二进制 0 开始计数，在这一空间中的所有位组合均被写出。

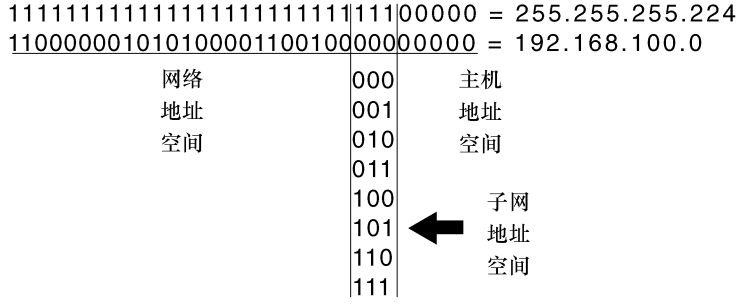


图 1-9 从标记的子网空间中可以得到子网位，接着从二进制 0 开始计数，给出子网空间中所有可能的位组合

在图 2-10 中，不发生变化的网络位填写在子网空间的左边，全部为 0 的主机位填写在子网位的右边。结果被转换为点分十进制表示后，得到 6 个子网地址（记住，第一个和最后一个地址，即在子网空间中全部位为 0 和全部位为 1 的地址不能使用）。

```
11111111111111111111111111100000 = 255.255.255.224
11000000101010000110010000000000 = 192.168.100.0
11000000101010000110010000000000 = 192.168.100.0
11000000101010000110010000100000 = 192.168.100.32
11000000101010000110010001000000 = 192.168.100.64
11000000101010000110010001100000 = 192.168.100.96
11000000101010000110010010000000 = 192.168.100.128
11000000101010000110000010100000 = 192.168.100.160
11000000101010000110000011000000 = 192.168.100.192
11000000101010000110000011100000 = 192.168.100.224
```

图 1-10 将网络地址填写在子网空间的左边，全部为 0 的主机位置填写在子网空间的右边，并将结果用点分十进制表示后得到子网地址

最后一步是计算每个子网内的可用主机地址。这一步通过以下方式完成：首先选择一个子网地址，保持其中的网络位和子网位不变，从二进制 0 开始计数，写出主机空间中所有的位组合。图 1-11 给出了针对子网 192.168.100.32 的计算过程。

注意结果的模式：第一个地址所有主机位全部为 0，这是子网地址。最后一个地址主机位全部为 1，这是子网 192.168.100.32 的广播地址。主机地址从子网地址起到广播地址为止。按照顺序，下一个子网地址是 192.168.100.64。

现在，在二进制层次上理解子网划分的重要性就显而易见了。给出一个地址，如192.168.100.160，你不能确定它是否是一个主机地址、子网地址或广播地址。甚至在子网掩码已知情况下，结论也并不总是明显的。

这里我们鼓励读者计算例子中所有余下子网的主机地址，并且仔细观察产生地址的模式，理解这些模式对下一部分的内容会有帮助。

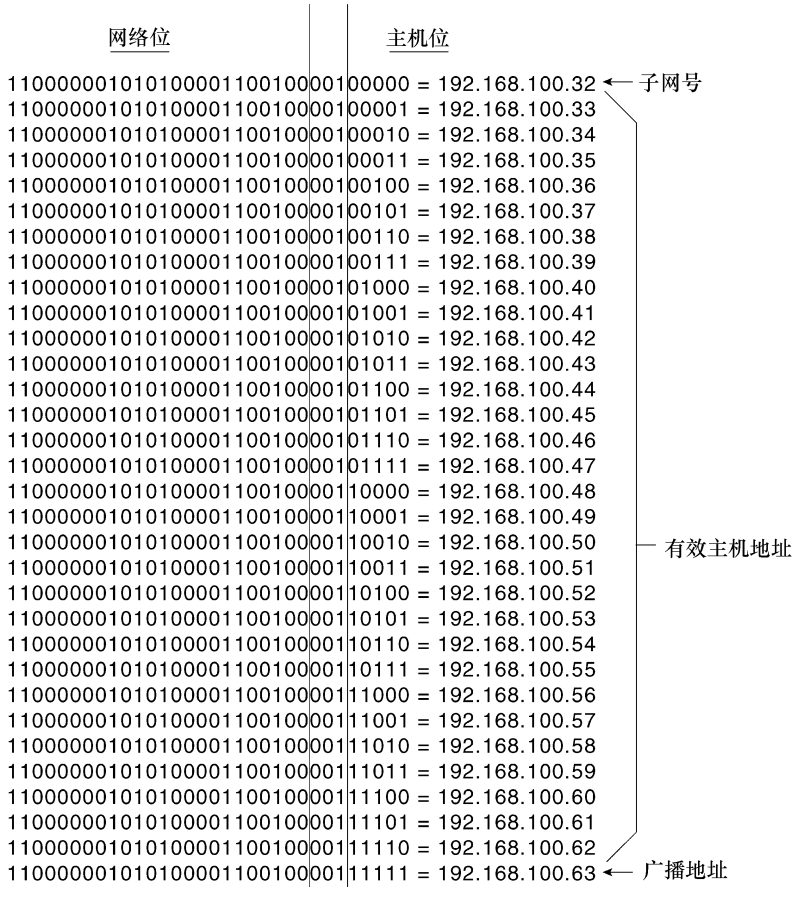


图 1-11 写出主机空间中所有位组合可以得到子网内的主机地址。这里是子网 192.168.100.32 的主机位

1.3.6 子网掩码的故障诊断

在“解剖”一个给定的主机地址和掩码时，常常需要确定地址属于哪个子网。例如，如果在一个接口上配置了地址，一个很好的实践就是首先验证对于接口连接的子网来说该地址是否合法。

使用下面的步骤逆推一个 IP 地址：

步骤 1：用二进制写下一个给定的子网掩码。

步骤 2：用二进制写下一个主机 IPv4 地址。

步骤 3：在知道一个地址的类别后，掩码的子网位便是显然的了。根据掩码位，在最后网络位和第 1 个子网位之间画一条线，在最后子网位和第 1 台主机之间也画另一条线。

步骤 4：写下地址的网络位和子网位，设置所有的主机位为 0。最终的结果就是主机地址所属的子网地址。

步骤 5：再次写下地址的网络位和子网位，这次设置所有主机位为 1。结果就是本子网的广播地址。

步骤 6：按照顺序可以知道第一个地址是子网地址，最后一个地址是广播地址。而且还可以知道在这两个地址之间的所有地址都是合法的主机地址。

对于地址 172.30.141/25，图 1-12 给出了以上步骤的示例。

这个地址是 B 类地址，所以前 16 位是网络位，25 位掩码中的后 9 位是子网位。可以发现子网地址是 172.30.0.128，广播地址是 172.30.0.255。在这两个地址之间的主机地址对于这个子网来说都是合法的，如对子网 172.30.0.128 来说，172.30.0.129~172.30.0.254 都是主机地址。

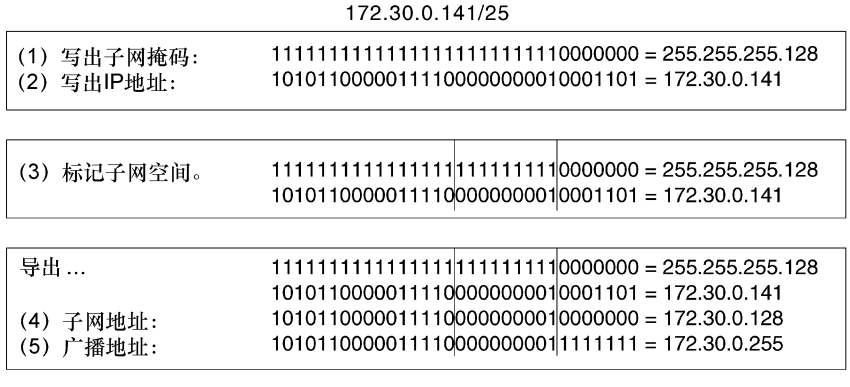


图 1-12 给定一个 IPv4 地址和子网掩码，按照以上这些步骤可以找出子网地址、广播地址和主机地址

在这个例子中，初次进行子网划分的人可能会受到以下几种情况的干扰。一种是地址的第 3 个八位组所有位都为 0。另一种是最后一个八位组仅一个子网位。一些人可能会认为广播地址看上去不合法。所有这些不舒服的感觉都源自地址的点分十进制表示法。当使用二进制表示地址和掩码时，这些疑虑会被打消，任何事看上去都一切正常，掩码设定了 9 位子网空间——包括第 3 个八位组和第 4 个八位组的第 1 位。这个案例说明了如果使用二进制表示法时一切正常，那么就不必担心看上去有些奇怪的点分十进制表示法。

1.4 地址解析协议（ARP）

第 1 章解释了通过读取和操作数据包的网络地址，路由器可以沿逻辑路径传送数据包，其中逻辑路径包括多个数据链路。沿独立的数据链路传送数据包时，需要把数据包封装在帧中，并且使用数据链路标识（如 MAC 地址）让帧可以从链路的源点到达目的地。本书的主题之一是为了进行路由选择，路由器利用何种机制发现并共享地址信息。类似的，数据链路上的设备也需要一种方法发现邻居的数据链路标识，以便将数据帧传送到正确的目的地。

有几种机制可以提供这些信息；[1] IPv4 使用地址解析协议（ARP），详见 RFC826。图 1-13

[1] 例如，NetWare 把设备的 MAC 地址作为网络层地址的主机部分，这是一种明智之举。

给出了 ARP 的工作机制。当一台设备需要发现另一台设备的数据链路标识符时，它将建立一个 ARP 请求数据包。这个请求数据包中包括目标设备的 IPv4 地址以及请求设备（发送者）的源点 IPv4 地址和数据链路标识符（MAC 地址）。然后 ARP 请求数据包被封装在数据帧中，其中带有作为源的发送者的 MAC 地址和作为目标的广播地址（参见示例 1-6）。[1]

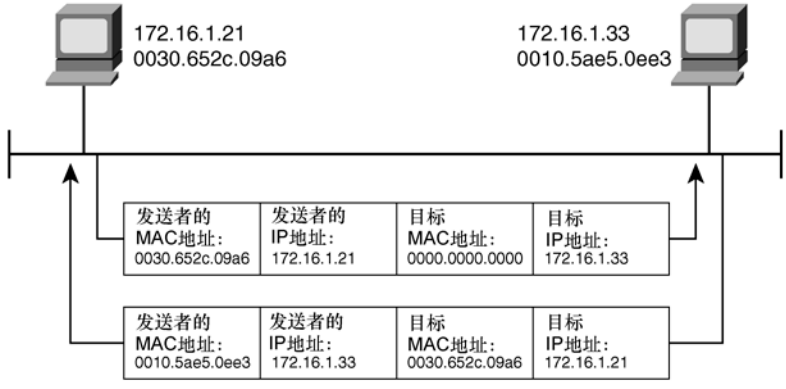


图 1-13 ARP 用于把设备的数据链路标识符映射到它的 IP 地址上

示例 1-6 协议分析器捕捉到图 1-13 所描述的 ARP 请求数据包及封装帧

```
Ethernet II, Src: 00:30:65:2c:09:a6, Dst: ff:ff:ff:ff:ff:ff
    Destination: ff:ff:ff:ff:ff:ff (Broadcast)
    Source: 00:30:65:2c:09:a6 (AppleCom_2c:09:a6)
    Type: ARP (0x0806)
Address Resolution Protocol (request)
    Hardware type: Ethernet (0x0001)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (0x0001)
    Sender MAC address: 00:30:65:2c:09:a6 (AppleCom_2c:09:a6)
    Sender IP address: 172.16.1.21 (172.16.1.21)
    Target MAC address: 00:00:00:00:00:00 (00:00:00_00:00:00)
    Target IP address: 172.16.1.33 (172.16.1.33)
```

广播地址意味着数据链路上的所有设备都将收到该帧，并且要检查帧内封装的数据包。除了目标机可以识别此数据包外，其他所有设备都会丢弃此数据包。目标机将向源地址发送 ARP 响应数据包，提供它的 MAC 地址（参见示例 1-7）。

示例 1-7 协议分析器捕捉的图 1-13 所描述的 ARP 响应数据包

[1] 类似于 IP 的广播地址，MAC 的广播地址也是所有位全部为 1：ffff.ffff.ffff。

```
Ethernet II, Src: 00:10:5a:e5:0e:e3, Dst: 00:30:65:2c:09:a6
    Destination: 00:30:65:2c:09:a6 (AppleCom_2c:09:a6)
    Source: 00:10:5a:e5:0e:e3 (3com_e5:0e:e3)
    Type: ARP (0x0806)
    Trailer: 151515151515151515151515151515...
Address Resolution Protocol (reply)
    Hardware type: Ethernet (0x0001)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (0x0002)
    Sender MAC address: 00:10:5a:e5:0e:e3 (3com_e5:0e:e3)
    Sender IP address: 172.16.1.33 (172.16.1.33)
    Target MAC address: 00:30:65:2c:09:a6 (AppleCom_2c:09:a6)
    Target IP address: 172.16.1.21 (172.16.1.21)
```

当调用调试功能 **debug arp** 时，Cisco 路由器可以显示 ARP 的活动情况，参见示例 1-8。

示例 1-8　路由器 Aretha（172.21.5.1）响应来自主机 172.19.35.2 的 ARP 请求

```
Aretha#debug arp
IP ARP: rcvd req src 172.19.35.2 0002.6779.0f4c, dst 172.21.5.1 Ethernet0
IP ARP: sent rep src 172.21.5.1 0000.0c0a.2aa9,
                dst 172.19.35.2 0002.6779.0f4c Ethernet0
Aretha#
```

图 1-14 给出了 ARP 数据包的格式。这里可以把图中描述的各字段同示例 1-6 和示例 1-7 的 ARP 数据包相对照。

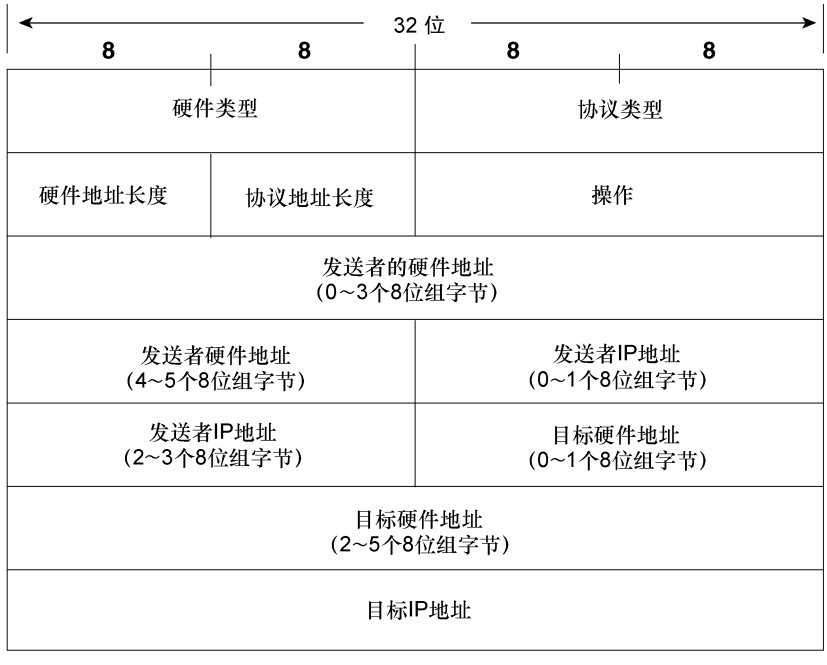


图 1-14　ARP 数据包格式

- **硬件类型（Hardware Type）**——指定了硬件的类型，详见 IETF 的规范说明。[1]一些

[1] 在整个 TCP/IP 协议簇中各字段使用的所有号码都来源于这里：J.Postel 和 J.Reynolds，“Assigned Numbers，”RFC1700，1994 年 10 月。这本大型文档（230 页）是一本很有价值的参考文献，但目前有点过时了。现在有关号码分配的最新列表可以在 www.iana.org 上查到。

常用的类型编号如表 1-5 所示。

表 1-5 常用的硬件类型码

编号	硬件类型
1	以太网
3	X.25
4	Proteon ProNET Token Ring
6	IEEE 802 网络
7	ARCnet
11	Apple LocalTalk
14	SMDS

续表

编号	硬件类型
15	帧中继
16	异步传输模式（ATM）
17	高速数据链路控制（HDLC）
18	光纤信道
19	异步传输模式（ATM）
20	串行链路

- **协议类型（Protocol Type）**——指定了发送者映射到数据链路标识符的网络层协议的类型；IP 对应 0x0800。
- **硬件地址长度（Hardware Adress Length）**——指定了数据链路标识符的长度，单位是八位组。MAC 地址的长度为 6。
- **协议地址长度（Protocol Address Length）**——指定了网络层地址的长度，单位是八位组。IPv4 地址的长度为 4。
- **操作（Operation）**——指明了一个数据包是 ARP 请求（1）还是 ARP 响应（2）。这里还可以发现有其他的值表明 ARP 数据包的其他用途。如反向 ARP 请求（3）、反向 ARP 响应（4）、反转 ARP 请求（8）、反转 ARP 响应（9）。

最后 20 个八位组是发送者和目标机的数据链路标识符和 IPv4 地址。

在示例 1-9 所示屏幕的最上面，命令 **show arp** 用于检查 Cisco 路由器内的 ARP 表。

示例 1-9 连接到相同网络上的 3 台设备的 ARP 表：Cisco 路由器，Windows 主机和 Linux 主机

```
Martha#show arp
Protocol  Address          Age (min)     Hardware Addr   Type   Interface
Internet  10.158.43.34           2      0002.6779.0f4c  ARPA   Ethernet0
Internet  10.158.43.1            -      0000.0c0a.2aa9  ARPA   Ethernet0
Internet  10.158.43.25          18      00a0.24a8.a1a5  ARPA   Ethernet0
Internet  10.158.43.100          6      0000.0c0a.2c51  ARPA   Ethernet0
Martha#
```

```
C:\WINDOWS>arp -a

Interface: 148.158.43.25
  Internet Address      Physical Address      Type
  10.158.43.1           00-00-0c-0a-2a-a9     dynamic
  10.158.43.34          00-02-67-79-0f-4c     dynamic
  10.158.43.100         00-00-0c-0a-2c-51     dynamic
```

```
Linux:~# arp -a
Address                 HW type         HW address              Flags   Mask
10.158.43.1             10Mbps Ethernet 00:00:0C:0A:2A:A9       C       *
10.158.43.100           10Mbps Ethernet 00:00:0C:0A:2C:51       C       *
10.158.43.25            10Mbps Ethernet 00:A0:24:A8:A1:A5       C       *
Linux:~#
```

请注意年龄一栏，这一栏表明为了防止陈旧信息充满 ARP 表，每经过一个特定的实际间隔，ARP 信息将会被刷新。Cisco 路由器保存 ARP 表项的时间为 4 个小时（14 400s）；这个缺省值可以修改。下面的例子就是将 ARP 的超时值修改为 30min（1 800s）：

```
Martha(config)# interface Ethernet 0
Martha(config-if)# arp timeout 1800
```

示例 1-9 所示屏幕的中间给出了 Windows PC 的 ARP 表，屏幕底部给出了 Linux 机器的 ARP 表。虽然它们的格式不同于 Cisco 路由器的 ARP 表，但是 3 个表中的实质性信息是相同的。

ARP 表项还可以永久地保存在表中。为了实现地址 172.21.5.131 到硬件地址 0000.00a4.b74c 的静态映射，并且采用 SNAP（Subnetwork Access Protocol）封装类型，可以使用以下命令完成：

```
Martha(config)# arp 172.21.5.131 0000.00a4.b74c snap
```

命令 **clear arp-cache** 可以从 ARP 表中强制删除所有动态表项。并且此命令也可以清除快速交换高速缓冲区和 IP 路由高速缓冲区中的内容。

ARP 还有几种变形，其中至少有一种对路由选择十分重要，它就是代理 ARP。

1.4.1 代理 ARP

代理 ARP 有时也被叫做混杂 ARP，详见 RFC925 和 RFC1027，代理 ARP 被路由器作为向主机表明自身可用的一种手段。例如，主机 192.168.12.5/24 需要向主机 192.168.20.101/24 发送数据包，但是它没有配置缺省网关信息，因而也就不知道如何到达路由器。这时它可以向 192.168.20.101 发送一个 ARP 请求；本地路由器收到这一请求，并且路由器知道如何到达网络 192.168.20.0，因此路由器将回复以上请求，其中把自己的数据链路标识符作为 ARP 回复数据包中的硬件地址。事实上，路由器欺骗了本地的主机，让它认为路由器的接口就是 192.168.20.101 的接口。最终所有发向 192.168.20.101 的数据包都被送往路由器。

图 1-15 给出了代理 ARP 的另一种用途。这里特别关注的是地址掩码。路由器配置的掩码是 28 位掩码（4 个子网位的 C 类地址），而主机配置的是标准的 C 类地址掩码（24 位）。

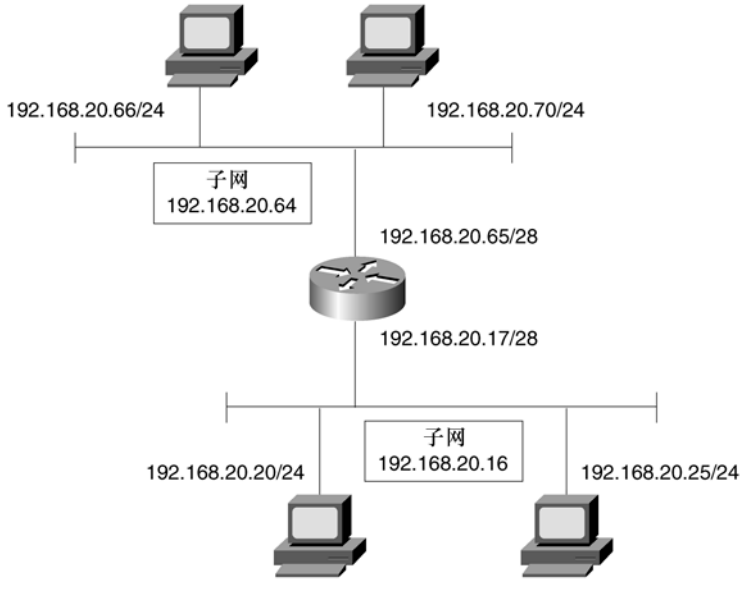


图 1-15 代理 ARP 实现了子网划分的透明性

其结果是主机并不知道子网的存在。当主机 192.168.20.66 想发送数据包到 192.168.20.25 时，它首先将发送 ARP 请求。这时路由器识别出数据包的目标地址属于另一个子网，因而向请求主机回复自己的硬件地址。这种代理 ARP 使得子网化网络拓扑结构对主机来说是透明的。

示例 1-10 所示的 ARP 高速缓冲暗示了代理 ARP 的又一用途。注意，有多个 IPv4 地址映射到单一的 MAC 标识符；其中 IP 地址对应着主机，而硬件 MAC 标识符属于路由器接口。

示例 1-10 图 1-15 中主机 192.168.20.66 的 ARP 表显示出多个 IPv4 地址映射到单一 MAC 标识符，这说明正在使用代理 ARP

```
C:\WINDOWS>arp -a

Interface: 192.168.20.66
  Internet Address      Physical Address      Type
  192.168.20.17         00-00-0c-0a-2a-a9     dynamic
  192.168.20.20         00-00-0c-0a-2a-a9     dynamic
  192.168.20.25         00-00-0c-0a-2a-a9     dynamic
  192.168.20.65         00-00-0c-0a-2c-51     dynamic
  192.168.20.70         00-02-67-79-0f-4c     dynamic
```

在 IOS 系统中，缺省情况下代理 ARP 功能是打开的，当然也可以在每个接口上使用命令 **no ip proxy-arp** 关闭此功能。

1.4.2 无故 ARP

主机偶尔也会使用自己的 IPv4 地址作为目标地址发送 ARP 请求。这种 ARP 请求称为无故 ARP，主要有两个用途：

- 无故 ARP 可以用于检查重复地址。一台设备可以向自己的 IPv4 地址发送 ARP 请求，如果收到 ARP 响应则表明存在重复地址。
- 无故 ARP 还可以用于通告一个新的数据链路标识符。当一台设备收到一个 ARP 请求，如果 ARP 高速缓冲中已有发送者的 IPv4 地址，那么与此 IPv4 地址相对应的硬件地址将会被发送者新的硬件地址所更新。这种无故 ARP 用途正是基于此事实。
- 某个子网内运行热备份路由器协议（HSRP 协议）的路由器如果从其他路由器变成了主路由器，它就会发出一个无故 ARP 来更新该子网上主机的 ARP 缓存。

许多 IP 实现中都没有实现无故 ARP 功能，但是读者应该知道它的存在。在 IOS 系统中缺省情况下是关闭的，但可以通过命令 **ip gratuitous-arps** 激活它。

1.4.3 反向 ARP

代替映射硬件地址到已知 IPv4 地址，反向 ARP（RARP）可以实现 IPv4 地址到已知硬件地址的映射。某些设备，如无盘工作站在启动时可能不知道自己启动时的 IPv4 地址。嵌入这些设备固件中的 RARP 程序可以允许它们发送 ARP 请求，其中硬件地址为设备的硬件编入地址。RARP 服务器将会向这些设备回复相应的 IPv4 地址。

RARP 在很大程度上正在被动态主机配置协议（DHCP）和自举协议（BOOTP）的扩展协议所替代，不同于 RARP，这两种协议都可以提供 IPv4 地址以外的更多信息，而且还可以跨越本地数据链路。

1.5 ICMP

Internet 消息控制协议（ICMP）指定了多种消息类型，这些消息的共同目的就是管理网络，详见 RFC792。ICMP 的消息可以分为错误消息、请求消息和响应消息。图 1-16 给出了一般的 ICMP 数据包格式。数据包可以通过类型来标识，许多数据包类型都有多个指定的类型，可以用代码字段来标识它们。表 1-6 列出了多种 ICMP 的数据包类型和代码，详见 RFC1700。

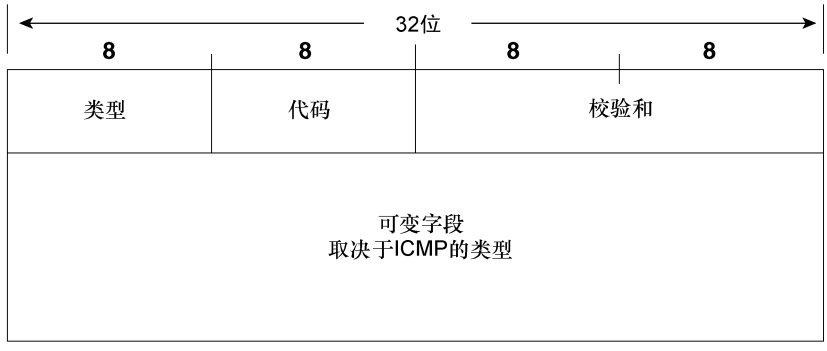


图 1-16 ICMP 数据包头包括类型字段，进一步标识某些类型的代码字段和校验和。剩余的字段依赖于特定类型和代码

表 1-6 ICMP 数据包类型字段和代码字段

类型	代码	名称
0	0	回应应答
3		目的地不可达
	0	网络不可达
	1	主机不可达
	2	协议不可达
	3	端口不可达
	4	需要分段和不需要分段标记置位
	5	源路由失败
	6	目的网络未知
	7	目的主机未知
	8	源主机被隔离
	9	与目的网络的通信被禁止
	10	目的主机的通信被禁止
	11	对请求的服务类型，目的网络不可达
	12	对请求的服务类型，目的主机不可达
4	0	源抑制（Source Quench）
5		重定向
	0	为网络（子网）重定向数据报
	1	为主机重定向数据报
	2	为网络和服务类型重定向数据报

续表

类型	代码	名称
	3	为主机和服务类型重定向数据报
6	0	选择主机地址
8	0	回应
9	0	路由器通告
10	0	路由器选择
11		超时
	0	传输中超出 TTL
	1	超出分段重组时间
12		参数问题
	0	指定错误的指针
	1	缺少需要的选项
	2	错误长度
13	0	时间戳
14	0	时间戳回复
15	0	信息请求（废弃）
16	0	信息回复（废弃）
17	0	地址掩码请求（即将废弃）
18	0	地址掩码回复（即将废弃）
30	—	路由跟踪

示例 1-11 和示例 1-12 给出了协议分析器捕捉到的两种众所周知的 ICMP 消息——Echo 请求和 Echo 回复，它们常用在 ping 命令的功能中。

示例 1-11 ICMP 的 Echo 消息及其 IPv4 头部

```
Internet Protocol, Src Addr: 172.16.1.21 (172.16.1.21),
    Dst Addr: 198.133.219.25 (198.133.219.25)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 84
    Identification: 0xabc3 (43971)
    Flags: 0x00
    Fragment offset: 0
    Time to live: 64
    Protocol: ICMP (0x01)
    Header checksum: 0x8021 (correct)
    Source: 172.16.1.21 (172.16.1.21)
    Destination: 198.133.219.25 (198.133.219.25)
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xa297 (correct)
    Identifier: 0x0a40
    Sequence number: 0x0000
    Data (56 bytes)

0000  40 fd ab c2 00 0e 73 57 08 09 0a 0b 0c 0d 0e 0f   @.....sW........
0010  10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f   ................
0020  20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f    !"#$%&'()*+,-./
0030  30 31 32 33 34 35 36 37                          01234567
```

示例 1-12 ICMP 的 Echo 回复消息

```
Internet Protocol, Src Addr: 198.133.219.25 (198.133.219.25),
    Dst Addr: 172.16.1.21 (172.16.1.21)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 84
    Identification: 0xabc3 (43971)
    Flags: 0x00
    Fragment offset: 0
    Time to live: 242
    Protocol: ICMP (0x01)
    Header checksum: 0xce20 (correct)
    Source: 198.133.219.25 (198.133.219.25)
    Destination: 172.16.1.21 (172.16.1.21)
Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0xaa97 (correct)
    Identifier: 0x0a40
    Sequence number: 0x0000
    Data (56 bytes)

0000  40 fd ab c2 00 0e 73 57 08 09 0a 0b 0c 0d 0e 0f   @.....sW........
0010  10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f   ................
0020  20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f    !"#$%&'()*+,-./
0030  30 31 32 33 34 35 36 37                          01234567
```

虽然大部分 ICMP 类型都与路由选择功能有关，但是有 3 个类型特别重要：

- **路由器通告（Router Advertisement）和 路由器选择（Router Selection）**——分别是类型 9 和类型 10，它们用于 ICMP 路由器发现协议（IRDP）。IRDP 协议用于某些操作系统发现本地的路由器（例如微软 Windows 操作系统的大多数版本）。
- **重定向（Redirection）**——是 ICMP 的类型 5，被路由器用于通知主机去往指定目标的网关，是数据链路上的另一台路由器。假设路由器 A 和路由器 B 连接在相同的以太网上，主机 X 也在以太网上，而且 X 还把路由器 A 配置为自己的缺省网关。如果主机向路由器 A 发送数据包，而路由器 A 发现该数据包目的地址需通过路由器 B 才可以到达（即路由器 A 必须在接收此数据包的端口再次转发此数据包）。路由器 A 不仅要向路由器 B 转发数据包，而且还要向主机 X 发送 ICMP 重定向消息，通知它如果继续向特定的目标发送数据包，那么请直接将数据包发送给路由器 B。示例 1-13 显示出路由器发送了一个重定向消息。

示例 1-13 使用调试功能 **debug ip icmp**，可以看到路由器向主机 10.158.43.25 发送了一个重定向消息，通知它到达目的地 10.158.40.1 的正确网关应该是路由器 10.158.43.10

```
Pip#debug ip icmp
ICMP packet debugging is on
ICMP: redirect sent to 10.158.43.25 for dest 10.158.40.1, use gw 10.158.43.10
0
Pip#
```

当数据链路上连接多台路由器时，避免数据包重定向的一个窍门是将每一台主机的缺省网关设置为主机自己的 IPv4 地址。于是主机对任何目的地址都会发送 ARP 请求，当目的地址不属于本地数据链路时，合适的路由器将通过代理 ARP 功能回复请求。使用这种策略避免

重定向是有争议的，因为重定向会被减少或消除，但是 ARP 的流量又增加了。

在 IOS 软件系统中，缺省状态下重定向功能是打开的。在接口上使用命令 **no ip redirects** 可以关闭此功能。

1.6 主机到主机层

TCP/IP 协议的主机到主机层的命名恰如其分。尽管网络层负责网络之间的逻辑路径，但主机到主机层是负责两个在完全不同网络[1]上的主机之间的全程逻辑路径。从另一个角度看，主机到主机层向应用提供了一个到协议簇下一层的接口，使应用不必关心它们的数据实际上是如何被传送的。

可以把这种服务比喻为公司的信件收发室。一个包裹被送到收发室，并附有邮寄要求（平信或隔日送到）。提出邮寄要求的人不需要知道或可能不关心实际是怎样邮寄此包裹的。收发室的工作人员将会安排合适的邮寄方式来满足其要求（送邮局邮寄、FedEx、交给骑自行车横穿城镇送快件的人）。

主机到主机层提供两个主要的服务：TCP 和 UDP。

1.6.1 TCP

传输控制协议（TCP），向应用提供了可靠的、面向连接的服务，详见 RFC793。换句话说，TCP 提供了一个类似于点到点的连接。

点到点连接有两个特点：

- 仅存在一条到达目的地的路径。进入连接的数据包不会丢失，因为数据包惟一可去的地方就是连接的另一端。
- 数据包到达的顺序与发送顺序相同。

TCP 提供了一条看似点到点的连接，虽然实际上这条连接并不存在。TCP 利用的 Internet 层可以提供无连接的、尽力而为转发的服务。这类似于邮政服务。一叠信一旦交给邮递员后，谁也不能保证信件将按照原先叠放的顺序依次送达，也不能保证信件都将在同一天送到，甚至不能保证全部送到。邮政服务仅仅能承诺尽最大努力邮寄这些信件。

同样的，Internet 层不保证所有的数据包使用相同的路径，因而也不保证数据包到达时仍旧保持发送时的顺序和间隔或者全部到达。

另一方面，电话呼叫是一个面向连接的服务。数据必须顺序、可靠地到达，否则数据就会作废。像电话呼叫一样，TCP 首先必须建立连接，然后是传送数据，当数据传送完成后要拆除连接。

在无连接服务之上，TCP 使用了 3 种基础的机制实现面向连接的服务：

- 使用序列号对数据包进行标记，以便 TCP 接收服务在向目的应用传递数据之前修正错序的数据包排序。
- TCP 使用确认、校验和定时器系统提供可靠性。当接收者按照顺序识别出数据包未能到达或发生错误时，接收者将通知发送者，或者接收者在特定时间内没有发送确

[1] 类似的，主机到主机层也可以看作在传输层之上，功能上等同于 OSI 的会话层，在两个跨网络的应用之间提供逻辑的、端到端的路径。

认信息，那么发送者就认为在发送结束后数据包没有到达接收方。在这两种情况下，发送者都会考虑重传数据包。

- TCP 使用窗口机制调整数据包的流量；窗口机制可以减少因接收方缓冲区满而造成丢失数据包的可能性。

TCP 在应用层数据上附加了一个报头，报头包括序列号字段和这些机制的其他一些必要信息，如叫做端口号的地址字段，该字段可以标识数据的源和目标应用程序。为了传送数据，应用数据及附加的 TCP 报头被封装在 IP 数据包内。图 1-17 显示了 TCP 数据报头字段，示例 1-14 显示了协议分析器获取的 TCP 报头信息。

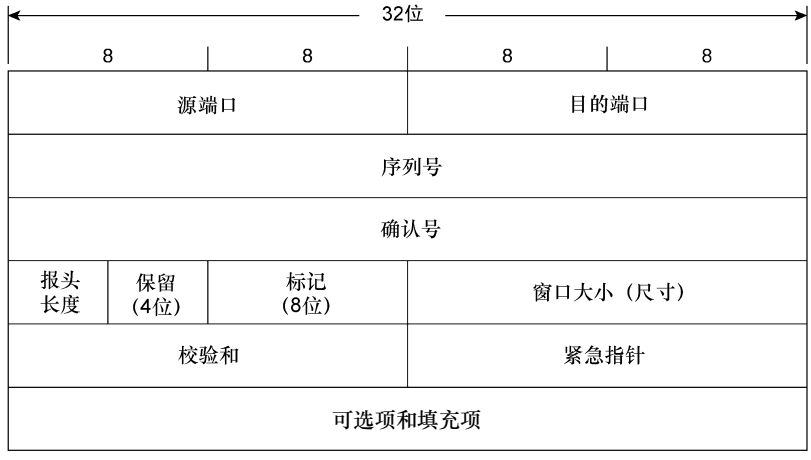


图 1-17 TCP 报头格式

示例 1-14 协议分析器显示出的 TCP 报头

```
Ethernet II, Src: 00:0c:41:3c:2b:18, Dst: 00:30:65:2c:09:a6
Internet Protocol, Src Addr: 66.218.71.112 (66.218.71.112),
    Dst Addr: 172.16.1.21 (172.16.1.21)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 52
    Identification: 0xc0b7 (49335)
    Flags: 0x04
    Fragment offset: 0
    Time to live: 50
    Protocol: TCP (0x06)
    Header checksum: 0x509d (correct)
    Source: 66.218.71.112 (66.218.71.112)
    Destination: 172.16.1.21 (172.16.1.21)
Transmission Control Protocol, Src Port: http (80),
    Dst Port: 60190 (60190), Seq: 288, Ack: 811, Len: 0
    Source port: http (80)
    Destination port: 60190 (60190)
    Sequence number: 288
    Acknowledgement number: 811
    Header length: 32 bytes
    Flags: 0x0010 (ACK)
    Window size: 66608
    Checksum: 0xb32a (correct)
    Options: (12 bytes)
        NOP
        NOP
```

(待续)

```
        Time stamp: tsval 587733966, tsecr 1425164062
    SEQ/ACK analysis
        This is an ACK to the segment in frame: 17
        The RTT to ACK the segment was: 0.047504000 seconds
```

- **源端口（Source Port ）和目的端口（Destination Port ）**——字段长度各为 16 位，它们为封装的数据指定了源和目的应用程序。像 TCP/IP 使用其他编号一样，RFC1700 描述了所有常用和不常用的端口号。应用程序的端口号加上应用程序所在主机的 IP 地址统称为套接字（socket）。在网络上套接字惟一地标识了每一个应用程序。
- **序列号（Sequence Number）**——字段长度为 32 位，序列号确定了发送方发送的数据流中被封装的数据所在位置。例如，如果本段数据的序列号为 1343，且数据段长 512 个八位组，那么下一数据段的序列号应该为 1343＋512＋1＝1856。
- **确认号（Acknowledgment Number）**——字段长度为 32 位，确认号确定了源点下一次希望从目标接收的序列号。如果主机收到的确认号与它下一次打算发送（或已发送）的序列号不符，那么主机将获悉丢失的数据包。
- **报头长度（Header Length）** ——又叫数据偏移量，长度为 4 位，报头长度指定了以 32 位字为单位的报头长度。由于可选项字段的长度可变，所以这一字段标识出数据的起点是很有必要的。
- **保留（Reserved）**——字段长度为 6 位，通常设置为 0。
- **标记（Flag）**——包括 8 个 1 位的标记，用于流和连接控制。它们从左到右分别是：拥塞窗口减少（Congestion Window Reduced，CWR）、ECN-Echo（ECE）、紧急（URG）、确认（ACK）、弹出（PSH）、复位（RST）、同步（SYN）和结束（FIN）。
- **窗口大小（Window Size）**——字段长度为 16 位，主要用于流控制。窗口大小指明了自确认号指定的八位组开始，接收方在必须停止传输并等待确认之前发送方可以接收的数据段的八位组长度。
- **校验和（Checksum）**——字段长度为 16 位，它包括报头和被封装的数据，校验和允许错误检测。
- **紧急指针（Urgent Pointer）**——字段仅当 URG 标记置位时才被使用。这个 16 位数被添加到序列号上用于指明紧急数据的结束。
- **可选项（Options）**——字段用于指明 TCP 的发送进程要求的选项。最常用的可选项是最大段长度，最大段长度通知接收者发送者愿意接收的最大段长度。为了保证报头的长度是 32 个八位组的倍数，所以使用 0 填充该字段的剩余部分。

1.6.2 UDP

用户数据报协议（UDP）提供了一种无连接、尽力而为传送的数据包转发服务，详见 RFC 768。起初，对应用程序宁愿使用不可靠的转发服务，而不用面向连接的 TCP 服务，感觉很有疑问。然而 UDP 的优点是不花时间建立连接，直接发送数据。用 UDP 代替 TCP，可以使发送小数据量的应用取得更好的性能优势。

图 1-18 中给出了 UDP 的另一个优点：UDP 报头长度远远小于 TCP 报头长度。UDP 报头中的源端口和目的端口字段与 TCP 完全相同；UDP 的长度指明了以八位组为单位的整个

段长度。校验和包括整个段，但是不同于 TCP，在这里，校验和是可选的；当不使用校验和时，此字段全部设置为 0。在示例 1-15 中显示出协议分析器捕捉到的 UDP 报头。

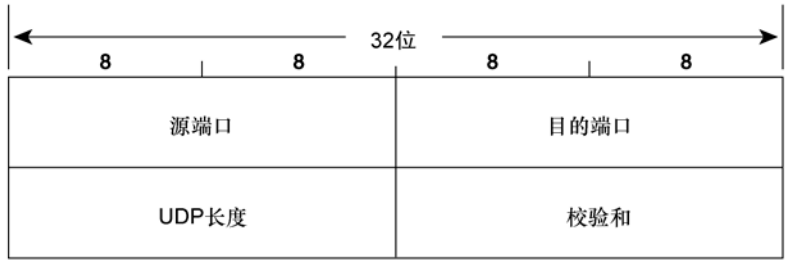


图 1-18 UDP 报头格式

示例 1-15 协议分析器显示的 UDP 报头

```
Ethernet II, Src: 00:30:65:2c:09:a6, Dst: 00:0c:41:3c:2b:18
Internet Protocol, Src Addr: 172.16.1.21 (172.16.1.21),
    Dst Addr: 198.133.219.25 (198.133.219.25)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 40
    Identification: 0x8a4d (35405)
    Flags: 0x00
    Fragment offset: 0
    Time to live: 1
    Protocol: UDP (0x11)
    Header checksum: 0xe0b3 (correct)
    Source: 172.16.1.21 (172.16.1.21)
    Destination: 198.133.219.25 (198.133.219.25)
User Datagram Protocol, Src Port: 35404 (35404), Dst Port: 33435 (33435)
    Source port: 35404 (35404)
    Destination port: 33435 (33435)
    Length: 20
    Checksum: 0x0000 (none)
Data (12 bytes)

0000  01 01 00 00 40 fd ac 74 00 00 d2 45                ....@..t...E
```

1.7 展　　望

本章重点讲述了设备的 Intenret 层（或 OSI 网络层）的自我标识机制以及 Intenret 层是怎样映射到网络接口层（OSI 数据链路）的。本章还分析了像 ARP、ICMP 这样的 Intenret 层协议对路由选择的重要性。下一章将研究 IP 协议的新版本——IP 版本 6，它与 IPv4 的不同之处，以及为什么需要新版本的 IP 协议的原因。

1.8 总结表：第 1 章命令总结

命令	描述
arp *ip-address hardware-address*〔**alias**〕	静态地映射 IP 地址到硬件地址

续表

命令	描述
arp timeout *seconds*	设置 Cisco 路由器保留 ARP 表项的时间值
clear arp-cache	强制从 ARP 表中删除所有动态表项
debug ip icmp	显示在路由器上出现的 ICMP 事件
ip address *ip-address mask*〔**secondary**〕	为接口分配 IP 地址和掩码
ip gratuitous-arp	启动无故 ARP 特性
ip netmask-format{bit-count|decimal|hexadecimal}	配置路由器，使路由器可以用位计数、点分十进制和十六进制方式显示 IP（地址，掩码）对
ip proxy-arp	启用代理 ARP
ip redirects	启用 ICMP 重定向功能

1.9 推荐读物

Baker, F., ed.“Requirements for IP Version 4 Routers,”RFC 1812, June 1995.

这篇文章给出了对要运行 IP 协议的路由器的要求和建议。

Braden, R., ed.“Requirements for Internet Hosts—Communication Layers,”RFC 1122, October 1989.

RFC 1812 的姊妹篇，中心内容是主机。

Comer, D. E. *Internetworking with TCP/IP*, Vol. 1. Englewood Cliffs, New Jersey: Prentice-Hall; 1991.

这本书，就像 Perlman 的著作一样，是一本经典书。尽管你不一定要把 Comer 和 Stevens 的书都读，但是如果都能阅读的话，对你绝不会有坏处。

Stevens, W. R. *TCP/IP Illustrated*, Vol. 1. Reading, Massachusetts: Addison-Wesley; 1994.

这是一本关于 TCP/IP 的好书。Stevens 在深入地介绍协议的同时，针对封二上的网络图提供了大量现实网络的细节。

1.10 复 习 题

1．TCP/IP 协议簇的 5 个层次是什么？每一层的目的是什么？

2．目前最常用的 IP 版本是什么？

3．什么是分段？IP 报头的什么字段用于分段？

4．IP 报头中的 TTL 字段的用途是什么？TTL 过程是如何工作的？

5．什么是首个八位组字节规则？

6．怎样识别点分十进制表示的 A 类、B 类和 C 类地址？怎样识别二进制表示的地址？

7．什么是地址掩码？它是如何工作的？

8．什么是子网？在 IP 环境中为什么使用子网？

9．为什么在有类别路由选择环境中子网位不能全部为 0 或 1？

10．什么是 ARP？

11．什么是代理 ARP？

12．什么是重定向？

13．TCP 和 UDP 的本质区别是什么？

14．TCP 提供面向连接服务的机制是什么？

15．为了替代 ARP，Novell NetWare 用设备的 MAC 地址作为网络地址中的主机部分。为什么 IP 不能这样做？

16．UDP 通过在无连接服务之上提供无连接服务的目的是什么？

1.11 配置练习

1．首个八位组字节规则指出最高的 C 类地址是 223，而我们知道八位组的最大十进制数是 255。因而还有两类地址，一类是 D 类地址，用于组播，另一类是 E 类地址，用于实验。其中 D 类地址的前 4 位为 1110。请问 D 类地址首个八位组的十进制数的范围是什么？

2．为 10.0.0.0 选择一个子网掩码以便至少可以划分出 16 000 个子网，并且每个子网至少拥有 700 个主机地址。为 172.27.0.0 选择子网掩码以便至少可以划分出 500 个子网，并且每个子网至少拥有 100 个主机地址。

3．如果 C 类地址有 6 个子网位，那么可以划分出多个子网？每个子网有多少个主机地址？这样的子网规划有实际用途吗？

4．对地址 192.168.147.0 进行子网划分，子网掩码为 28 位，请写出所有子网。试给出每个子网的可用主机地址。

5．对地址 192.168.147.0 进行子网划分，子网掩码为 29 位，请写出所有子网。试给出每个子网的可用主机地址。

6．对地址 172.16.0.0 进行子网划分，子网掩码为 20 位，试给出每个子网的可用主机地址（地址按照最低到最高顺序给出）。

1.12 故障诊断练习

1．根据以下主机地址和子网掩码，试找出每个地址所属的子网，并且找出该子网中的广播地址和可用主机地址的范围：

```
10.14.87.60/19
172.25.0.235/27
172.25.16.37/25
```

2．请问接口上配置 IP 地址 192.168.13.175，掩码 255.255.255.240，会有问题吗？如果有，问题是什么？

本章包括以下主题：

- IPv6 地址；
- IPv6 包头格式；
- IPv6 扩展报头（Header）；
- Internet 消息控制协议第六版（ICMPv6）；
- 邻居发现协议（NDP）。

第 2 章

IPv6 概述

在网络最初开始发展，还未最终发展成为我们现在所称的 Internet 时，那时的网络还仅仅只限于学术和研究领域。而且，当时在 Vint Cerf 和 Bob Kahn 创建有关这些网络的 TCP/IP 协议簇的时候，更是没有人会预料到 Internet 能发展为今天这样。在当时看来，所设计的 32 位地址空间可以提供大约 43 亿个地址，这看上去好像是不可能耗尽的。

但是，当那些在学院里就使用网络的孩子们到了“现实的世界中”时，他们自身就很欣赏基于开放标准上的对等网络创造出的种种可能。从而，越来越多的有价值的网络应用不断地涌现出来；公司连接到某个公共网络的价值也日益得到重视，并开始促成一个商用的 Internet 网络。在所有这些发生的同时，桌面计算机的使用也变得普及起来，不再是仅仅在办公室使用，更引人注目地是，计算机的使用也在家庭普及开来。但是，那些早期的家庭计算机还没有普遍到把调制解调器作为计算机通用的附属配件，因为那时很少有家庭计算机用户能够意识到个人计算机连接到一个公共网络上的价值和好处。

随着万维网（World Wide Web）的出现，这种情况开始发生改变。突然间，对于非技术领域的用户，轻松地获取和共享信息极大地提高了他们把桌面计算机作为工具的价值。结果，不到 20 年的时间，Internet 已经变成了我们通信、商务活动以及学习的途径和手段。Internet 使我们的世界变得更小，同时也对世界政治和经济产生了意义深远的影响。

随着“Internet 的大家庭”变得越来越大，它的大小和差异性都呈现出爆炸性的增长，同时也出现了平常令人非常讨厌的东西，例如垃圾邮件、病毒，等等。这样就产生了一个比较严重的技术性问题需要考虑：原来认为可以提供所谓无穷无尽多的 IPv4 地址现在看来显然已经很有限了。

早在 20 世纪 90 年代初期，人们就意识到 IPv4 地址可能消耗殆尽的问题，当时各方面的专家预测显示，如果 IPv4 地址的分配按照目前的增长率继续下去，那么在未来短短几年间就将耗尽所有的地址空间。于是，人们就提出了一个新的 IP 地址版本来解决这个问题，以前在开发阶段这个新的 IP 地址版本被称为 IP 下一代版本（或者称为 IPng），而现在一般称为 IP 协议第六版（或称为 IPv6）。但是众所周知，发展一种新的标准需要时间逐步部署，因此，在发展新标准的过程中还需要一种解决 IPv4 地址耗尽问题的短期方案。

这种短期的解决方案就是网络地址转换（Network Address Translation——NAT），它允许多台主机共享一个或较少的公用 IP 地址。在 NAT 设备上，相对外部公共网络的内部网络使用私有 IP 地址，私有 IP 地址的使用规则在 RFC1918（请求注释）中有详细的描述。读者后面会注意到，在本书的大多数例子中都会使用这种私有 IP 地址。NAT 技术在减缓 IPv4 地址耗尽问题方面显然非常成功，并在大多数网络设计中已经成为一个标准部分。因而，至今仍然有很多人对发展 IP 协议新版本的必要性提出质疑。但是，NAT 技术的广泛使用把原来具有开放、透明、对等特点的 Internet 变成了看上去更像一个具有客户-服务器（Client-Server）结构的网络的巨大集合。而用户则只在外围连接到 Internet 的“边缘层”，Internet 向他们提供服务。用户很少对 Internet 的整体资源作出贡献。更多的从某种经济的角度看，Internet 的用户仅仅成为了消费者，而不是生产者。

虽然大多数 IPv6 协议标准在多年前就已经完成了，但对从 IPv4 到 IPv6 协议迁移的巨大兴趣也只是最近才显现出来。这种日益重视使用 IPv6 协议的背后来自于两个基本的推动力。第一个基本的推动力是对使用诸如移动 IP 协议（Mobile IP）、服务质量保证、端到端的安全、网格计算（grid computing），以及点到点网络互连等核心概念的新型应用的先见之明。NAT 技术遏制了这些领域的创新，因而摒弃 NAT 技术的惟一手段就是提供充足的并且易于使用的公共 IP 地址。

促进 IPv6 协议发展的第二个基本推动力就是拥有众多人口的国家快速的现代化发展，例如中国和印度。一个引人注目的统计数字显示，目前剩余的未分配的 IPv4 地址数目几乎和中国的人口一样多：大约还有 13 亿个地址。随着中国国家 Internet 网络基础设施的急剧扩展，在不久的将来，仅中国就会对现有已经十分紧张的 IPv4 地址池带来难以忍受的压力。在印度，其人口也接近于中国的人口，不得不继续保留一个具有 4～5 层 NAT 技术的网络层次架构，以支持对 IP 地址的需求。

IPv6 协议使用 128 位的地址替代 32 位的 IPv4 地址，这样大约可以产生 340 万亿亿亿亿（3.4×10^{38}）个可用的地址。在可预见的未来，这个地址数目将可以满足公共 IP 地址的需求，也可以解决上面讨论的两个基本推动力需要的地址需求。[1]

2.1 IPv6 地址

IPv6 地址与 IPv4 地址的不同之处不仅仅在于它们的地址长度不同，而在更多的方面都有所不同。快捷地表示它们的“速记”方式也是不同的，它们的表示格式差别非常大，而且它们的功能组织也是不同的。本节我们将为读者介绍这些不同之处。

[1] 考虑到在 IPv4 地址分配时所没有预见到的情况——当时 IPv4 所拥有的 43 亿个地址被认为可以无限的提供给所有的实际需求，再也不会有人认为 IPv6 协议所提供的几乎不可思议的巨大地址空间是不可耗尽的了。

2.1.1 地址表示法

读者一定已经了解 32 位的 IPv4 地址的表示方式了，IPv4 地址被分割为 4 个 8 位段，其中每个 8 位段的数字大小在 0~255 之间，并且每个 8 位段之间使用英文符号句点“.”来分开，因此有时也使用术语“点分十进制表示法”来专指 IPv4 地址的这种表示法。

而 128 位的 IPv6 地址则被分割成 8 个 16 位段来表示，其中每个 16 位段书写为大小在 0x0000~0xFFFF 之间的十六进制的数字表示，并且每个 16 位段之间使用英文符号冒号“:”来分开。例如，下面就是一个 IPv6 地址的书写方式：

3ffe:1944:0100:000a:0000:00bc:2500:0d0b

要想记住更多一些像这样表示的地址实际上是几乎不可能的，当然书写这些地址也不是一件令人愉快的事情。幸运地是，有两条规则可以用来简化 IPv6 地址书写的大小。第一条规则是：

任何一个 16 位段中起始的 0 不必写出来；任何一个 16 位段如果少于 4 个十六进制的数字，就认为忽略书写的数字是起始的 0。

在前面提到的地址例子中，第 3、4、5、6 和 8 个分段都包含有起始的 0。利用这个地址压缩简化规则，该地址可以书写为：

3ffe:1944:100:a:0:bc:2500:d0b

这里要注意的是，只有起始的 0 才可以被忽略掉；末尾的 0 是不能忽略的，因为这样做会使16位分段变得不确定，你无法确切地判断所省略的0是在所写的数字之前还是在其之后。

另外，还有一个值得注意的地方是，上述的地址例子中的第 5 个分段全部是 0，并且被书写为单个 0。事实上，有许多 IPv6 地址中具有一长串的 0。举例如下：

ff02:0000:0000:0000:0000:0000:0000:0005

这个地址可以简写为以下形式：

ff02:0:0:0:0:0:0:5

然而，利用第二个规则可以进一步地简化这个地址的书写格式：

任何由全 0 组成的 1 个或多个 16 位段的单个连续的字符串都可以用一个双冒号“::”来表示。

利用这条规则，上面例子中的地址可以表示成如下格式：

ff02::5

使用这样的方式书写上面这样的地址显然可以增加很多便利。但是在这里要注意的是，这条规则强调的是仅仅对于单个连续不间断的全 0 字符串分段部分能够用一个双冒号“::”来表示，在一个 IPv6 地址中使用多于一个以上的双冒号会引起含混不清。下面举一个这样的地址例子作为说明：

2001:0d02:0000:0000:0014:0000:0000:0095

对于上面这个地址，以下两种地址的缩写方式都被认为是正确的，因为它们都只使用了一次双冒号：

2001:d02::14:0:0:95

2001:d02:0:0:14::95

但是，请读者注意，下面这个缩写方式是不正确的，因为它使用了两次双冒号：

2001:d02::14::95

之所以认为上面这个缩写方式是错误的，是因为它中间的两个全 0 字符串的长度是含混

不清的，从而无法确定它们的长度；它可以表示成下面的任何一种可能的 IPv6 地址：

2001:0d02:0000:0000:0014:0000:0000:0095

2001:0d02:0000:0000:0000:0014:0000:0095

2001:0d02:0000:0014:0000:0000:0000:0095

不像 IPv4 协议的前缀（即地址的网络部分）可以通过点分十进制或十六进制地址掩码标识，或可以通过位计数（bitcount）来标识，IPv6 协议的前缀始终通过位计数的方式来标识。更确切地说，通过在 IPv6 地址后面加一个斜线“/”，随后再跟一个十进制的数字来标识一个 IPv6 地址的起始位有多少位是前缀位。举一个例子，下面这个地址的前缀就是起始的 64 位：

3ffe:1944:100:a::bc:2500:d0b/64

当读者需要书写一个 IPv6 地址的前缀时，也可以使用和 IPv4 地址一样的书写方式将所有的主机位设置为 0。例如：

3ffe:1944:100:a::/64

一个由全 0 组成的 IPv6 地址能够被简单地写成一个双冒号。在本书中，存在两种实例使用了全 0 的地址。第一个实例就是缺省地址，这将在第 12 章中讨论，在那里缺省地址表示为全 0 的形式，并且它的前缀长度也是 0：

::/0

第二个使用全 0 的 IPv6 地址的实例是未指定地址（unspecified address）。未指定地址使用在某些邻居发现协议过程中，邻居发现协议将在本章后面的章节中讲述。一个未指定地址就像一个填充器，用来标识一个还未确定的实际 IPv6 地址。在书写一个未指定地址的时候要注意，它与缺省地址的书写方式是有区别的，它们的前缀长度不同：

::/128

2.1.2 IPv6 的地址类型

IPv6 地址存在以下三种类型：

- 单播（Unicast）；
- 任意播（Anycast）；
- 多播（Multicast）。

和 IPv4 相比，IPv6 地址有一个重要的不同，IPv6 地址协议中没有广播地址。但是，IPv6 地址协议提供了一个包含“全部节点”的多播地址，用来实现与 IPv4 地址协议中广播地址同样的目的。

1. 全球单播地址

单播地址用来表示单台设备的地址。一个全球单播地址是指这个单播地址是全球惟一的。IPv6 单播地址的通用格式如图 2-1 所示。该格式在 RFC3587 中有详细地描述，并取代和简化了早期的格式版本，早期的格式将 IPv6 单播地址分成了顶级聚合（Top Level Aggregator，TLA）、次级聚合（Next-Level Aggregator，NLA）和其他字段。但是，读者应该了解的是，这个被取代的格式其实相对来说是最近才被替代的，因而，读者在一些书籍和资料中可能仍然会碰到对这个旧的 IPv6 地址格式的描述。



图 2-1 IPv6 通用的单播地址格式

地址的主机部分被称作接口 ID（Interface ID）。之所以取这个名字是因为一台主机可以拥有不止一个的 IPv6 接口，因而使用这样的地址标识主机的一个接口比标识一台主机本身更加准确。但是，它的精确性也就仅仅到此为止：单个接口也能够拥有多个 IPv6 地址，并且能够拥有一个附加的 IPv4 的地址，在这样的实例中，接口 ID 仅仅表示该接口的几个标识符的其中一个。

除了长度不同外，IPv6 地址与 IPv4 地址协议之间最显著的不同就是子网标识符的位置不同。IPv6 地址的子网标识符的位置是地址的网络域的一部分，而不是该地址的主机域的一部分。在 IPv4 地址分类体系结构的传统概念中，一个地址的子网部分来自于该地址的主机部分，减少了地址的主机位。结果是，IPv4 地址的主机部分不仅仅使它的分类产生变化，而且导致用于子网标识的位数产生变化。

使用地址的网络部分作为 IPv6 子网 ID 的一个直接的好处就是，所有 IPv6 地址的接口 ID 都有大小一致的位数，这就大大地简化了地址的解析复杂度。而且，使用地址的网络部分作为子网 ID，会产生一个更加清楚的分工，功能更加清晰：网络部分提供了一台设备到下行专用数据链路的定位，而主机部分提供这条数据链路上该设备的标识。

除了极少数的例外，全球 IPv6 地址的接口 ID 都是 64 位二进制位的长度。同样，除了极少数的例外，子网 ID 字段都是 16 位二进制位（如图 2-2 所示）。一个 16 位的子网 ID 字段可以提供 65 536 个不同的子网。使用固定长度大小的子网 ID 看起来好像有些浪费，因为在大多数实例中远没有使用到这么大容量的子网数。但是，考虑到使用 IPv6 地址空间的总长度和容易分配、设计、管理以及解析地址的好处，使用固定长度大小的子网 ID 所带来的浪费也是可接受的。

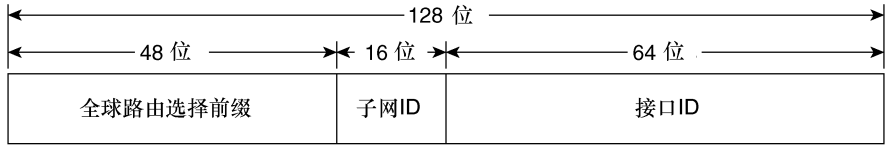


图 2-2 全球单播 IPv6 地址的标准字段大小

Internet 地址授权委员会（Internet Assigned Numbers Authority，IANA）和地区 Internet 注册机构（Regional Internet Registries，RIR）[1]通常把长度为/32 或/35 的 IPv6 前缀分配给本地 Internet 注册机构（Local Internet Registries，LIR）。然后，本地 Internet 注册机构 LIR——通常是大型的 Internet 服务提供商（ISP），他们再把更长的前缀分配给他们各自的客户。在大多数的实例中，本地 Internet 注册机构 LIR 分配的前缀长度都是/48。正如前面所提及的，有一些例外的情况，LIR 也可能会分配不同长度的前缀：

- 如果一个客户非常庞大，那么可以分配一个长度小于/48 的前缀。

[1] 在写本书时，共有 5 个 RIR 机构：RIPE（Réseaux IP Européens）——服务于欧洲、中东地区和中亚地区；LACNIC（拉丁美洲和加勒比海 Internet 地址注册机构，Latin American and Caribbean Internet Address Registry ）——服务于中美、南美以及加勒比海地区；ARIN（美国 Internet 编号注册机构，American Registry for Internet Numbers）——服务于北美地区和部分加勒比海地区；AfriNIC——服务于非洲地区；而 APNIC（亚太地区网络信息中心，Asia Pacific Network Information Centre）——则服务于亚洲和太平洋地区的国家。

- 如果有一个并且仅有一个子网需要编址，那么可以分配一个长度为/64 的前缀。
- 如果有一台并且仅有一台设备需要编址，那么可以分配一个长度为/128 的前缀。

2. 标识 IPv6 的地址类型

IPv6 地址起始的一些二进制位指明了该地址的类型。例如，目前所有的全球单播地址的前 3 位是 001。因此，识别全球单播地址的十六进制表示是相当容易的：所有的全球单播地址都是以 2 或 3 开头的，这根据全球路由选择前缀的第 4 位的值而定。举一个例子说明，当前指定分配给 6Bone（公共 IPv6 研究网络，the public IPv6 research network）使用的前缀开始于 3ffe，而目前由 RIR 分配的 IPv6 地址则开始于 2001。

二进制的数字 001 预期可以充分满足在未来一段时间内全球单播地址的需要。而其他的一些二进制位的组合则分配用来定义其他的地址类型，其中大多数前导位的组合都是保留的。表 2-1 列出了目前已经分配的前导位的组合，在后面的章节中将会陆续介绍一些其他主要的 IPv6 地址类型。

表 2-1 IPv6 地址类型的高位数字组合

地址类型	高位数字（二进制）	高位数字（十六进制）
未指定	00…0	::/128
环回地址	00…1	::1/128
多播地址	11111111	FF00::/8
链路本地单播地址	1111111010	FE80::/10
地区本地单播地址（目前存在争议）	1111111011	FEC0::/10
全球单播地址（当前分配的）	001	2xxx::/4 或者 3xxx::/4
保留的类型（未来全球单播地址的分配）	其他所有的	

3. 本地单播地址

当我们谈论全球单播地址的时候，我们就认为这个地址是全球范围使用的。也就是说，这样的地址是全球惟一的，并且能够在全球范围内被路由而无需进行更改。

IPv6 也拥有链路本地单播地址（link-local unicast address），这种地址是使用范围限定在单条链路上的地址。它的惟一性是仅仅限于所在的链路，并且相同的地址也可能存在于另一条链路上，因此这样的地址离开所在的链路是不可路由的。正如读者在表 2-1 中所看到的，链路本地单播地址的起始 10 位永远是 1111111010（FE80::/10）。

读者在本章后面的章节中将会看到，链路本地单播地址在像邻居发现协议等功能中是很有用的，邻居发现协议只能在单条链路上进行通信，这在后面的章节中会讲述。链路本地单播地址也允许链路上的设备直接创建 IPv6 地址和该链路上的其他设备进行通信，而不必给它们分配全球前缀，或者说它们无需知道所在链路的全球前缀。在 2.5.3 小节中将会讲述在这种情形下如何使用链路本地前缀。

除了链路本地单播地址外，IPv6 协议最初还定义了一个地区本地单播地址。地区本地单播地址（site-local unicast address）是指仅仅在一个给定的地区区域内该地址是惟一的。在其他地区区域内的设备可以使用相同的地址。因此，一个地区本地单播地址只能在分配该地址的那个地区区域内是可路由的。在 IPv6 协议中的地区本地地址在功能上和 RFC1918 中定义的私有 IPv4 地址有些类似。

地区本地地址的支持者举出这样几个应用。一个非常突出的应用就是，对于那些即使在使用 IPv6 地址时也希望使用 NAT 技术的网络人员来说，是为了维持他们自己的地址架构独立于他们的服务提供商。另外，地区本地地址也是几个被提议的 IPv6 多归路机制的关键。

但是，IETF（Internet 工程任务组）的 IPv6 工作组发现地区本地单播地址也带来了一些困难。一种明显的困难就是这样一个事实：地区区域的定义是含糊的，因为对于不同的网络管理者来说地区的含义是不同的。另一个问题是，当这样的地址被错误地“泄漏到”网络管理者设定的地区范围边界之外时，也像 RFC1918 定义的 IPv4 地址一样会涉及到一些管理上的困难。其他一些潜在的问题包括给应用程序和路由器增加了复杂性，因为它们必须识别和复制这些地区本地地址。基于以上这些考虑，并经过一些激烈的争论后，结果是，IPv6 工作组在 RFC3879 中不赞成使用地区本地地址。为了给那些认为地区本地地址有很多优点的人一些信心，又引入了另外一个计划，它具有类似于“比链路本地地址的范围大一些，而比全球地址的范围小一些”的特点，但是到撰写本书时为止还未见到这样一种替代的计划。

如表 2-1 所示，地区本地单播地址的起始 10 位是 1111111011（FEC0::/10）。

4. 任意播地址

一个任意播地址（Anycast address，也可称为任播地址或泛播地址）表示的更像一种服务，而不是一台设备，并且相同的地址可以驻留在提供相同服务的一台或多台设备中。如图 2-3 所示，某些服务是由 3 台服务器提供的，但却是通过 IPv6 地址 3ffe:205:1100::15 来进行该服务的所有通告的。

接收到包含该地址通告的路由器不会知道是由 3 台不同的设备通告给它的。相反，路由器会假定有 3 条路由到达相同的目的地，并会选择一条代价最低的路由。如图 2-3 所示，这条路由是到达服务器 C 的，它的代价是 20。

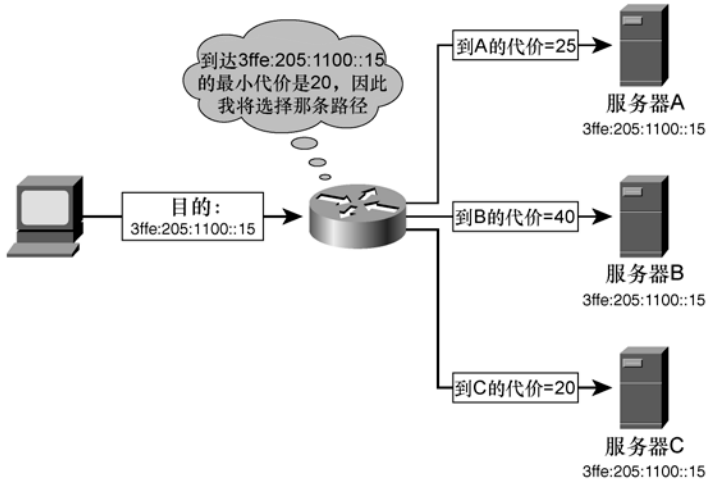


图 2-3 一个任意播地址表示的是一种服务，它可能是多台设备

使用任意播地址的好处就是，路由器总是选择到达“最近的”或“代价最低的”服务器的路由。[1]因此，提供一些通用服务的服务器能够通过一个大型的网络进行传播，并且流量可以由本地传送到最近的服务器，这样就可以使网络中的流量模型变得更有效。而且，如果其

[1] 路由器在到达同一个目的地的多条路由中选择路由的方法将在第 4 章中讲述。

中一台服务器变得不可用时，路由器能够把路由指向下一台最近的服务器。举例来说，如图 2-3 所示，如果服务器 C 因为网络或服务器本身出现故障而变得不可用了，那么路由器就会选择到达服务器 A 的路径，因为到达服务器 A 是倒数第二个代价最低的路由。从路由器的角度来看，它选择的是到达同一个目的地最优的路由。

任意播地址仅是根据它们提供的服务功能而定义的，而不是根据它们的格式，而且理论上来说可能是任何范围内的任何一个 IPv6 单播地址。但是，在 RFC 2526 中定义了一个保留的任意播地址的格式。任意播地址在 IPv4 协议的网络中已经使用了一段时间，但是在 IPv6 协议中它们的定义才被正式化。

5. 多播地址

多播地址标识的不是一台设备，而是一组设备——一个多播组（multicast group，或称为多播群）。发送给一个多播组的数据包可以由单台设备发起。因此，一个多播数据包通常包括一个单播地址作为它的源地址，一个多播地址作为它的目的地址。在一个数据包中，多播地址从来不会作为源地址出现。

一个多播组的成员可能只有一台单个的设备，也可能甚至是该网络上所有的设备。事实上，IPv6 协议并不像 IPv4 协议那样有一个保留的广播地址，而是有一个保留的包含所有节点的多播组，实际上做相同的事情：所有接收它的设备都是属于该多播组。

多点传送实际上是 IPv6 协议的一个基本的操作，特别是对于即插即用特性的一些功能，例如路由器发现和地址自动配置等，这些功能是邻居发现协议的一部分，邻居发现协议将在本章后面讲述。

IPv6 多播地址的格式如图 2-4 所示。多播地址起始的 8 位总是全 1，并且后跟的 4 位被指定作为标记位。这些标记位的前 3 位目前没有使用，全部设置为 0。第 4 位用来指出这个地址是一个永久的、公认的地址（设为 0），还是一个管理分配使用的暂时性的地址（设为 1）。接下来的 4 位数字表示该地址的范围，如表 2-2 所示。表 2-3 中显示了几个保留的、公认的 IPv6 多播地址，所有这些地址都属于链路本地的范围。由于多播组总是一组独立的节点，因而在多播地址中的子网字段是不需要的，或者说是没有意义的。而最后的 112 位用来作为组 ID（Group ID），标识各个不同的多播组。目前的用法是设置前面的 80 位为 0，而只使用后面的 32 位。

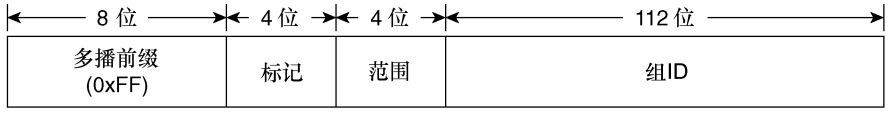


图 2-4 IPv6 多播地址的格式

表 2-2 多播地址的范围

范围字段的值	范围
0x0	保留
0x1	节点本地（Node-Local）
0x2	链路本地
0x5	地区本地
0x8	组织本地（Organizaion Local）
0xE	全球的
0xF	保留

表 2-3 公认的 IPv6 多播地址举例

地址	多播组
FF02::1	所有的节点
FF02::2	所有的路由器
FF02::5	OSPFv3 路由器
FF02::6	OSPFv3 指定路由器
FF02::9	RIPng 路由器
FF02::A	EIGRP 路由器
FF02::B	移动代理（Mobile Agents）
FF02::C	DHCP 服务器/中继代理
FF02::D	所有的 PIM 路由器

6. 嵌入的 IPv4 地址

有几种转换技术——将 IPv4 地址的网络转换成 IPv6 地址的技术，或者另外一种让两者共存的技术——要求 IPv4 地址在 IPv6 地址环境中进行通信。这些不同的技术详细说明了 IPv4 地址如何嵌入到 IPv6 地址中的细节，以及如何实现在 128 位的 IPv6 地址中寻找 32 位的 IPv4 地址的技术。但是，读者也会发现，多数技术使用惟一的格式来表示它们的地址，以便允许读者识别嵌入的 IPv4 地址。例如，一个嵌入了 IPv4 地址 10.23.1.5 的 IPv6 地址是：

FE80::5EFE:10.23.1.5 （一个 ISATAP 地址）

::FFFF:10.23.1.5 和::FFFF:0:10.23.1.5（SIIT 地址）

FEC0:0:0:1::10.23.1.5（TRT 地址）

在上述的每个例子中，IPv4 地址都是位于 IPv6 地址的最后 32 位，并使用点分十进制表示法表示。

其他使用嵌入 IPv4 地址的转换技术都不是使用点分十进制表示，而是把 IPv4 地址转换成十六进制编码表示。例如，6to4 技术就是使用十六进制表示。地址 10.23.1.5 在十六进制中的表示是 0A17:0105，所以嵌入了地址 10.23.1.5 的一个 6to4 前缀表示为：

2002:0A17:0105::/48

本卷书的内容并不涵盖这些转换技术，所以读者在本书中不太可能会看到这些地址表示方式。在这里之所以提到它们，仅仅是因为读者如果从事 IPv6 协议方面的工作时很可能会遇到像这样的地址。

2.2 IPv6 包头格式

IPv6 包头（Paeket Header）的格式如图 2-5 所示。这和 IPv4 的数据包头部有些明显的相像，也有些或明显的或细微的不同之处，IPv4 的数据包头部请参见前面章节中的图 1-2 所示。

- **版本（Version）**——和 IPv4 的报头（Header）一样，是一个 4 位的字段，用来指出 IP 协议的版本。当然，在这里它被设置为 0110，表明是版本 6。
- **流量类别（Traffic Class）**——是一个 8 位的字段，这相当于 IPv4 协议中的 ToS 字段。但是，考虑到 ToS 字段这些年的发展，现在都用来做区分服务等级（Differentiated Class of Service，DiffServ）了。所以，即使这个字段和旧的 ToS 字段有些相似，它

们的名字要比所传送的值更能确切地反映目前的用处。

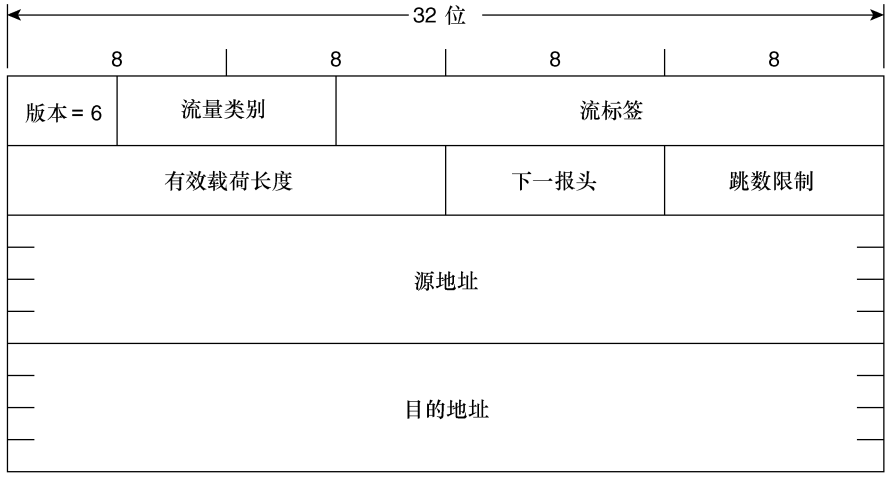


图 2-5 IPv6 的报头

- **流标签（Flow Label）**——是 IPv6 协议独有的字段，长度为 20 位。这个字段的设置目的是允许为特定的业务流打上标签；也就是说，数据包不仅仅始发于相同的源和到达相同的目的地，而且在源和目的地都属于相同的应用。区分不同的流可以带来几方面的优点，从可以提供更精细的服务类别区分的颗粒，到在平衡业务流量通过多条路径时可以确保属于同一个流的数据包能够总是转发到相同的路径上去，以便避免对数据包进行重新排序。流（或更精确地称为微分流，microflows）可以用源地址、目的地址加上源和目的端口的组合来确定。

  但是，为了识别源和目的端口，路由器必须能够看到 IP 报头的上层——TCP 或 UDP（或其他传输层协议）报头——这就增加了转发处理的复杂性，并可能会影响路由器的性能。由于 IPv6 的扩展报头，在 IPv6 的数据包里搜索传输层报头尤其会产生问题，扩展报头将在下一小节中讲到。一台 IPv6 的路由器必须分步通过可能存在的多个扩展报头来搜索传输层报头。

  在发起一个数据包时，加上合适的流标签字段，路由器就能够识别一个流，而不必进一步地查找数据包头部。但是，在撰写本书时，有关如何使用流标签字段的完整说明仍然还在争论，因此目前路由器忽略这个字段。尽管如此，它还是作出了这样一个许诺——允许 IPv6 可以为像 IP 语音（VoIP）这样的应用提供比 IPv4 更好的服务质量保证（QoS）。
- **有效载荷长度（Payload Length）**——用来指定数据包所封装的有效载荷的长度，以字节计数。请读者回忆一下第 1 章，因为 IPv4 的报头包含可选字段和填充字段，因而它的报头长度是可变的。因此，为了得到 IPv4 数据包的有效载荷长度，必须用总长度字段的值减去报头长度字段的值。而另一方面，IPv6 数据包头部长度总是固定的 40 字节，而且单从有效载荷长度字段就足以能得到有效载荷的起始和结尾了。

在这里也请读者注意，尽管 IPv4 的总长度字段是 16 位的，但是 IPv6 的有效载荷长度字段却是 20 位。这意味着该字段能够指定更长的有效载荷（1 048 575 字节，相对 IPv4 中只有 65 535 字节）。因此，IPv6 数据包本身在理论上来说具有传送更大的有效载荷的能力。

- **下一报头（Next Header）**——指出了跟随该 IPv6 数据包头部后面的报头。在这里，

这个字段和 IPv4 协议报头中的协议字段非常类似。事实上，当下一报头字段是一个上层协议报头时，其将用于同样的目的。与 IPv4 中的字段一样，这个字段也是 8 位的。但是，在 IPv6 协议中，跟随在该数据包头部后面的报头可能不是一个上层协议报头，而是一个扩展的头部（在下一节中讲述）。因此，从下一头部字段的命名上就可以反映出它具有更广泛的功能范围。

- **跳数限制（Hop Limit）**——这个字段和 IPv4 协议中生存时间（TTL）字段在长度（都是 8 位）和功能上都是非常一致的。正如第 1 章中所讲到的，TTL 字段的最初设想是，在数据包转发过程期间，当在路由器上排队等待时就用该字段的值减去相应的以等待的秒数为单位的计数值，但这一功能从来没有实现过。相反，路由器直接对 TTL 值进行减 1，而不管该数据包在这台路由器上排队等待了多长时间（况且，在现代网络中，数据包在任何一个地方排队等待接近 1s 的时间都是非常不正常的）。因此，TTL 事实上总是被用来作为衡量一个数据包到达目的地的路径中所能跨越的最大路由器跳数的工具。如果 TTL 值减少为 0，那么该数据包就被路由器丢弃。跳数限制的功能也完全相同，但它的命名更加贴近了它的功能特点。
- **源地址和目的地址（Source and Destination Address）**——这和 IPv4 协议中的源地址和目的字段是一样的，当然，在 IPv6 协议中，这些字段是 128 位的长度而已。

在 IPv6 报头中，很明显缺少的一个字段是 IPv4 报头中包含的校验和字段。在现代传输介质的可靠性全面提高的今天——当然无线传输或许是例外——由于上层协议通常携带它们自己的错误校验和恢复机制，IPv6 报头本身的校验和就体现不出太多的价值了，因而就被去掉了。

2.3 IPv6 扩展报头

对比图 2-5 中的 IPv6 报头和图 1-2 中的 IPv4 报头，读者可以看出，虽然源地址和目的地址字段的长度都是报头的 4 倍，但是 IPv6 报头本身的长度并不比 IPv4 报头长：IPv6 报头长度为 40 字节，而 IPv4 报头最小长度为 20 字节。如果 IPv4 的可选项字段也用来进行扩展应用（虽然这不常见），那么 IPv4 报头的长度实际上比 IPv6 报头大。

读者会注意到，除了可选项字段，其他的字段并不是很常用到，例如那些与分段有关的字段，从而在 IPv6 报头里就去掉了那些字段。因此，给定了固定长度并且排除了所有不携带每个数据包转发时所必要信息的字段，IPv6 报头变得更加简洁和有效。

但是，如果我们需要使用这些 IP 特性的某个可选项，例如分段、源路由选择或认证，我们又该怎么做呢？于是就在 IPv6 协议中，提供了一项可选的功能——扩展报头（extension header），在需要提供这些功能时可以添加在报头之后。例如，如果需要使用源路由选择、分段和认证等可选功能，那么就可以把它们各自需要增加的功能信息加载到 3 个扩展报头当中，就像图 2-6 所显示的那样。因为这些报头，IPv6 数据包可以在以下两个方面提高效率：

- 数据包仅仅需要传送各自数据包所需要的信息，不需要传送用不到的字段。
- 可以通过定义新的扩展报头添加到 IPv6 数据包中来增加新的可选功能。

每一个扩展报头都像 IPv6 报头一样，有一个下一报头字段。因此，每一个报头都会告知是哪一个报头跟在它的后面。表 2-4 中显示了当前定义的扩展报头和它们下一报头的值。例

如，如图 2-7 所示，IPv6 报头中下一报头字段的值表明它的下一个报头是一个路由选择扩展报头（43），这个报头的下一报头字段表示它的下一个报头是一个分段扩展报头（44），依此类推。最后一个扩展报头 AH 表示它的下一个报头是一个 TCP 报头（协议号为 6）。

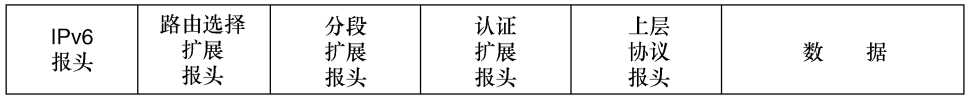


图 2-6 扩展报头允许 IPv6 数据包传送该数据包需要的所有信息，但传送的是该数据包仅仅需要传送的信息

表 2-4 下一报头的值

报 头	下一报头的值
逐跳可选项	0
路由选择	43
分段	44
封装安全有效载荷（ESP）	50
认证报头（AH）	51
目的地可选项	60
TCP/IP 协议	由协议定义的协议号的值（例如 TCP＝6，UDP＝17，OSPF＝89，等等）
没有下一报头	59

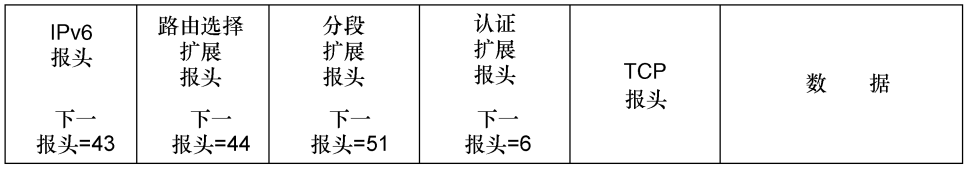


图 2-7 IPv6 报头中的下一报头字段和每一个扩展报头指定的跟在其后的报头

在 RFC 1883 中描述了每一个扩展报头的格式。但是概括来说，每个扩展报头的功能如下：

- **逐跳可选项（Hop-By-Hop Options）**——传送必须被转发路径中的每一个节点都检验处理的信息。例如，路由器告警和超大包有效载荷选项等。
- **路由选择（Routing）**——通过列出在到达目的地的路径中数据包所要经过的节点列表来提供源路由选择的功能。
- **分段（Fragment）**——是指在一个数据包被分段时用来为接收节点重组数据包提供必要的信息。在 IPv4 和 IPv6 数据包中有一个重要的不同是，只有发起该数据包的节点能够对数据包进行分段；而 IPv6 路由器对数据包并不分段。因此，发起该数据包的节点要么必须使用路径 MTU 发现（Path MTU Discovery，PMD）来得到该数据包到达目的地的路径上最小的 MTU 值，要么就从不发出大于 1 280 字节的数据包。PMD 将在下一小节中讲述。IPv6 协议规定运行 IPv6 的所有链路都必须能够支持最小 1 280 字节大小的数据包。因此，发起数据包的节点如果可以选择的话，可以利用最小长度大小选项，而不用 PMD。
- **封装安全有效载荷（Encapsulating Security Payload，ESP）**——用于有效载荷的加

密封装。

- **认证报头（Authentication Header，AH）**——用于数据包必须在源与目的节点之间进行认证的情况。
- **目的地可选项（Destination Options）**——用于传送仅仅被目的节点，或者可能是路由选择报头中列出的节点检验处理的信息。

如果使用扩展报头的话，在 RFC 1883 中也规定了它们应该出现的顺序。这里严格规定，如果使用逐跳可选项的话，它必须直接跟在 IPv6 报头的后面，以便于它能够被必须处理它的传输节点很容易的发现。建议的扩展报头顺序如下：

1. IPv6 报头。
2. 逐跳可选项。
3. 目的地可选项（只有在路由选择报头中指定的中间路由器才必须处理这个报头）。
4. 路由选择。
5. 分段。
6. 认证。
7. 封装安全有效载荷。
8. 目的地可选项（只有最后的目的节点必须处理这个报头）。
9. 上层报头。

2.4 ICMPv6

正像 IPv4 一样，IPv6 也需要一个控制协议来交换与处理错误和信息的消息。并且和 IPv4 一样，它也使用 ICMP 来实现这一功能。但是在 IPv6 中使用 ICMP 和在 IPv4 中使用 ICMP 并不一样。在 IPv4 协议中 ICMP 使用的协议号是 1，而在 IPv6 协议中 ICMPv6 使用的下一报头的值为 58。

ICMPv6（Internet 消息控制协议第六版）是在 RFC 2463 中规定的。在 RFC 中定义的很多功能都和 IPv4 中 ICMP 的定义相同；但是，在 ICMPv6 中也有很多 ICMP 消息意义并不相同，例如源抑制（Source Quench）和时间戳（Timestamp）消息。

比较图 2-8 中显示的 ICMPv6 报头和图 1-28 显示的 ICMP 报头，读者能够看到它们基本上是相同的。并且 ICMPv6 像 ICMP 一样使用一组类型与代码值的组合来标识一般的类型和它们的子类型。在 RFC 1885 中给出了这些值的定义，如表 2-5 中所列。

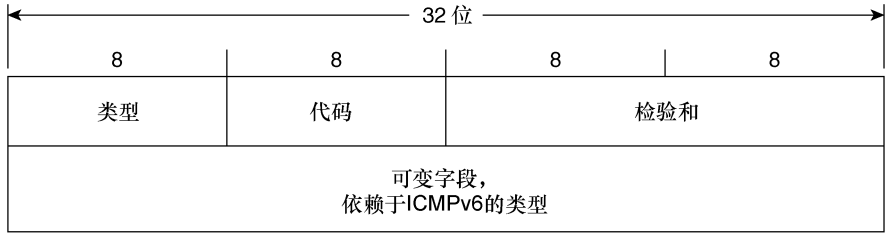


图 2-8 ICMPv6 报头的格式

表 2-5 ICMPv6 消息的类型域和代码域

类型	代码	消息
1		目的地不可达
	0	没有到达目的地的路由
	1	和目的地的通信被管理员禁止
	2	不是邻居
	3	地址不可达
	4	端口不可达
2	0	数据包长度太大
3		超时
	0	在传输时超出跳数限制
	1	分片的重组时间超时
4		参数问题
	0	遇到错误的报头字段
	1	遇到不可识别的下一报头类型
	2	遇到不可识别的 IPv6 选项
128	0	ECHO 请求
129	0	ECHO 答复
130	0	组成员请求
131	0	组成员报告
132	0	组成员减少

ICMPv6 协议除了基本的错误和信息控制功能外，还包括一些使用 ICMPv6 消息的机制。例如，在前面章节里提到的路径 MTU 发现机制会发送长度越来越大的数据包到目的节点。当一个给定的数据包长度超过到达目的节点的路径上最小的 MTU 时，该数据包将被丢弃，并发送一个数据包长度太大的消息给源地址，从而源地址节点就可以知道这条路径上最小的 MTU 大小。和 IPv4 一样，Echo 请求和 Echo 答复消息用于 Ping 的功能当中。

然而，ICMPv6 除了基本的错误和信息消息外，还使用了一组单独的由基本的 IPv6 协议 ICMPv6 消息：邻居发现协议，这将在下面讲述。

2.5 邻居发现协议（NDP）

IPv6 协议除了它显著地增加了地址空间外，一个最显著的特征就是它的即插即用特性。邻居发现协议（Neighbor Discovery Protocol，NDP）就是使用以下的功能来实现这些即插即用特性的协议：

- **路由器发现（Router Discovery）**——当一个节点连接到一个 IPv6 的链路上时，它能够发现本地的路由器，而不必借助动态主机配置协议（DHCP）。
- **前缀发现（Prefix Discovery）**——当一个节点连接到一个 IPv6 的链路上时，它能够发现分配给该链路的前缀。
- **参数发现（Parameter Discovery）**——节点能够发现它所相连的链路的参数，例如链路的 MTU 和跳数限制等。

- **地址自动配置（Address Autoconfiguration）**——节点能够确定它的完整地址，同样也不需要借助 DHCP 协议。
- **地址解析（Address Resolution）**——节点不需要利用地址解析协议（ARP）就能够发现所连接链路上其他节点的链路层地址。
- **下一跳确定（Next-Hop Determination）**——一条链路上的节点能够确定到达目的节点的下一跳链路层节点，或者是本地的目的节点，或者是到达目的节点的路由器。
- **邻居不可达检测（Neighbor Unreachability Detection）**——节点能够检测到链路上的邻居何时不再可达，在这里邻居可能是其他主机也可能是一台路由器。
- **地址冲突检测（Duplicate Address Detection）**——节点能够检测到它所要使用的地址是否已经被所在链路上的其他节点占用。
- **重定向（Redirect）**——对于非连接（off-link）的目的节点，路由器能够通过重定向消息通知主机存在比它自己更好的下一跳路由。该重定向功能在 IPv4 协议中是 ICMP 基本功能的一部分，但是在 IPv6 协议里被重新定义为邻居发现协议的一部分。

NDP 消息通常应该在链路本地的范围内收发，因此，封装 NDP 消息的数据包也始终使用 IPv6 链路本地地址，或者链路本地范围内的多播地址。为了增加更进一层的安全性，承载所有 NDP 消息的 IPv6 数据包的跳数限制为 255。如果所收到的数据包中有跳数限制的值小于 255 的，那么就说明这个数据包最少已经经过了一台路由器，因而该数据包将被丢弃。这种做法可以阻止 NDP 受到来自不与本地链路相连的源节点的攻击或欺骗。

2.5.1 NDP 消息

NDP 是在 RFC 2461 中定义的，为了完成某些功能，它使用 ICMPv6 协议来交换一些必要的消息，具体来说，在 RFC 2461 中详细说明了以下 5 个新的 ICMPv6 消息：

- **路由器通告（Router Advertisement，RA）消息**——路由器通告消息由路由器发起，用来通告这些路由器的存在和链路细节的参数，例如，链路前缀、链路 MTU，以及跳数限制等。这些消息周期性地发送，也用于答复路由器请求消息。
- **路由器请求（Router Solicitation，RS）消息**——路由器请求消息由主机发起，用来请求路由器发送一个 RA。
- **邻居请求（Neighbor Solicitation，NS）消息**——邻居请求消息由节点主机发起，用来请求另一台主机的链路层地址，也用来实现诸如地址冲突检测和邻居不可达检测的功能。
- **邻居通告（Neighbor Advertisement，NA）消息**——节点发送一个邻居通告消息来响应邻居请求消息。如果一个节点改变了它的链路层地址，那么它能够通过发送一个未请求的邻居通告消息来通告这个新地址。
- **重定向（Redirect）消息**——重定向消息的使用和 IPv4 协议中 ICMP 的用法是相同的，它们只不过从基本的 ICMPv6 协议中抽取了一部分移到了 NDP 当中。

如图 2-9 所示，图中显示了路由器通告消息的格式。它的 ICMPv6 类型是 134，代码是 0。封装这个 RA 消息的 IPv6 数据包的源地址总是始发这个数据包的接口的 IPv6 链路本地地址。假如这个 RA 消息周期性地发送，它的目的地址就是所有节点的多播地址（FF02::1），或者

如果这个 RA 消息是为了响应一台路由器请求消息，则它的目的地址是发起请求的节点的链路本地地址。

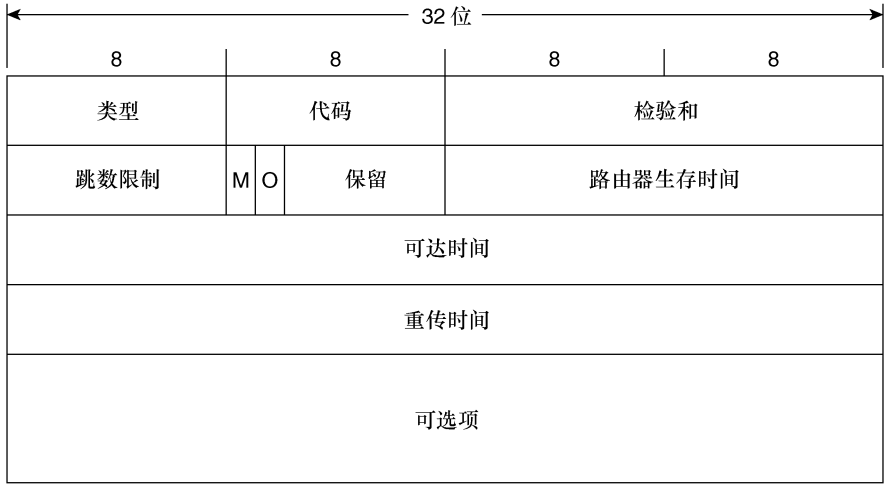


图 2-9 路由器通告消息的格式

跳数限制（Hop Limit）——如果设定了跳数限制字段的值，那么链路上的节点将在它始发到链路上的所有数据包里都设置该值。如果这台路由器没有指定跳数限制，那么该字段设置为全 0。

M——是管理地址的配置标记。如果设置了该位，始发路由器就会利用 DHCPv6 协议来告诉链路上的主机使用有状态地址自动配置。如果不设置该位，链路上的主机应该使用无状态地址自动配置。地址自动配置将在本章后面的章节中讲述。

O——是其他的有状态配置标记。当设置该位时，始发路由器就会告诉所在链路上的主机使用 DHCPv6 协议来获取其他的链路信息。M 标记和 O 标记可以一起使用。例如，不设置 M 标记但设置 O 标记，那么路由器将会告诉链路上的主机使用无状态地址自动配置，但对于其他的配置参数则不考虑 DHCPv6 服务器的存在。

路由器生存时间（Router Lifetime）——只有在始发路由器是一台缺省路由器时，该字段才设置为非 0 的值。如果是这种情况，则该字段指定为该缺省路由器的存活时间，以秒为单位，它的最大值为 18.2h。

可达时间（Reachable Time）——是指用于实现 NDP 协议中邻居不可达性检测功能的字段。当一个节点确认它的邻居是可到达的后，可达时间会指定一个时间值，在该时间内，这个节点假定它的邻居是可达的，以毫秒为单位。

重传计时（Retransmit Timer）——用于实现 NDP 协议里有关地址解析和邻居不可达性检测功能的字段。它指定了重传的邻居请求消息之间的最小时间，以毫秒为单位。

在路由器通告消息的可选项字段里携带的可能选项包括以下内容：

- 发起路由器通告消息（RA）的接口的链路层地址。
- 路由器通告消息所在链路的 MTU 说明。
- 分配给链路的一个或多个前缀。该信息是无状态地址自动配置的基本信息，它告诉链路上的主机该链路的前缀信息。

如图 2-10 所示，图中说明了路由器请求消息的格式。路由器请求消息的 ICMPv6 类型值

为 133，而代码值为 0。封装路由器请求消息的 IPv6 数据包的源地址，要么是始发该消息的接口所分配的 IPv6 地址，要么是一个用“::”（全部为 0）表示的未指定地址，这种情况出现在没有分配地址的情况（在发起的主机节点刚刚开始进行地址自动配置时会出现没有地址的情况）。目的地址是所有路由器多播地址（FF02::2）。

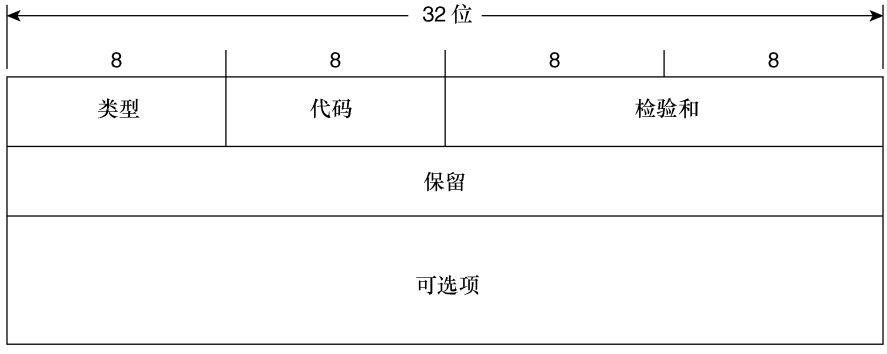


图 2-10 路由器请求消息的格式

这里的可选项字段包括了始发该消息接口的链路层地址，如果可以知道的话。但是，如果封装该数据包的源地址是未指定的，那么就不一定包括源链路层地址，例如在地址自动配置期间始发主机正在请求路由器的时候。

在图 2-11 中，显示了邻居请求消息的格式。邻居请求消息的 ICMPv6 类型值为 135，而代码值为 0。封装邻居请求消息的 IPv6 数据包的源地址要么是始发该消息的接口所分配的 IPv6 地址，要么是一个用“::”（全部为 0）表示的未指定地址，这种情况出现在为地址冲突检测而发送的邻居请求消息的时候。目的地址是对应于目标地址的一个被请求节点的多播地址，或者就是目标地址。

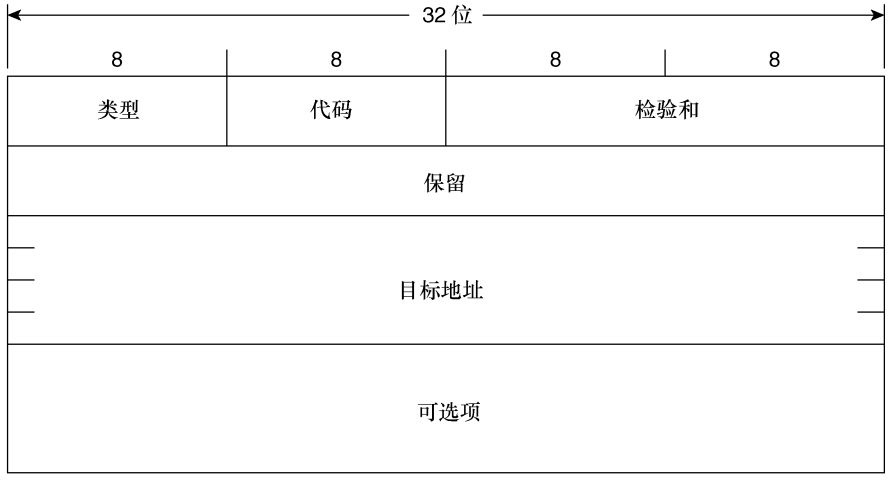


图 2-11 邻居请求消息的格式

目标地址（Target Address）——是指请求目标的 IPv6 地址。目标地址永远不可能是一个多播地址。

邻居请求消息的可选项字段可以包括始发该请求消息的接口的链路层地址。

如图 2-12 所示，图中显示了邻居通告消息数据包的格式。邻居通告消息数据包的 ICMPv6 类型值为 136，而代码值同样为 0。封装邻居通告消息的 IPv6 数据包的源地址总是由始发该消息的接口所分配的（或者自动分配的）IPv6 地址。目的地址要么是包含该通告消息所要答复的邻居请求数据包的源地址，要么是所有节点的多播地址（FF02::1）。

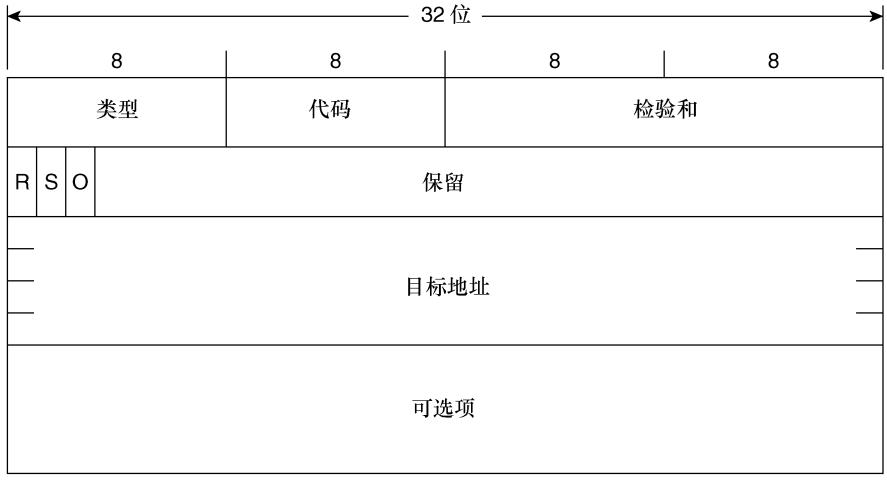


图 2-12 邻居通告消息的格式

R——是路由器标记。如果设置了该位，表示始发该消息的节点是一台路由器。这个位字段用于在邻居可达性检测期间检查已经变为主机的路由器。

S——是一个请求标记。在为答复一个邻居请求消息而发送的邻居通告消息中设置该位。

O——过载标记。当设置该位时，它表示邻居通告消息中的信息超出了任何现有邻居的缓存条目，需要更新所缓存的链路层地址。当该位没有被设置时，说明邻居通告消息没有超出现有邻居的缓存条目。

目标地址——如果为了响应一个邻居请求而发送的邻居通告，那么目标地址就是邻居请求消息中的目标地址字段。如果邻居通告消息是未请求的（也就是说，发送通告始发节点的链路层地址的变化），那么目标地址就是始发节点的地址。

邻居通告消息的可选项字段可包含目标链路层地址——也就是说，是始发邻居通告消息节点的链路层地址。

如图 2-13 所示，图中显示了一个重定向消息的格式。重定向消息的 ICMPv6 类型值为 137，而代码值同样为 0。封装重定向消息的 IPv6 数据包的源地址总是始发该消息的接口的 IPv6 链路本地地址。而它的目的地址总是触发重定向的数据包的源地址。

目标地址——是指更好的第一跳地址——通常是链路上另一台路由器的链路本地地址。

目的地址——是指被重定向到目标地址的 IPv6 目的地址。

重定向消息的可选项字段包括目标节点的链路层地址，这和触发该重定向的报头大小相同，不会使它的数据包长度超过 1 280 字节。

所有以上 5 种类型的消息中的可选项字段，无论包含什么信息，都是由一个或多个类型/长度/数值（TLV）这 3 个参数组合构成的。如图 2-14 所示，每一个 TLV 都是由一个 8 位的类型字段指定在数值字段中携带的信息的类型，一个 8 位的长度字段指定该数值字段的长度大小（用 8 位组作为单位，即字节数），以及一个长度可变的数值字段组成。

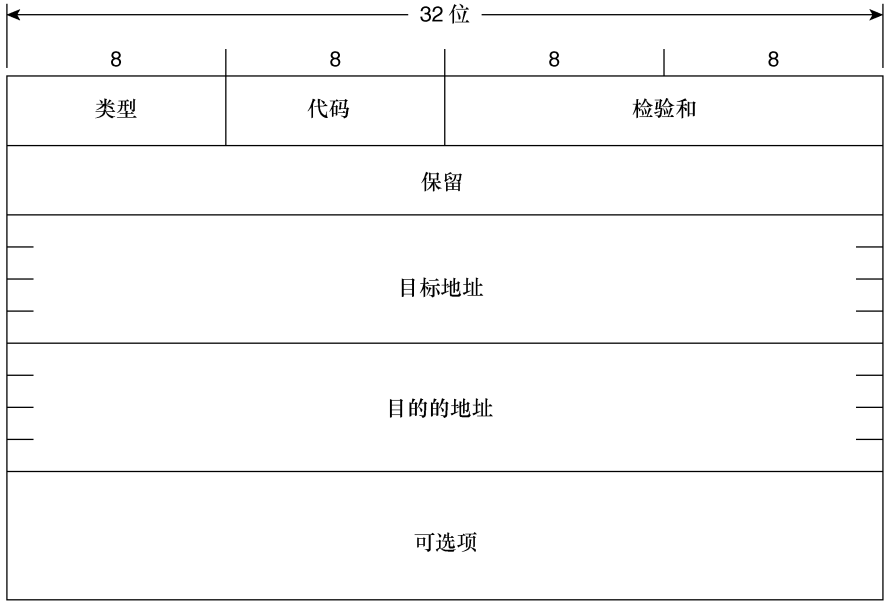


图 2-13 重定向消息的格式



图 2-14 用于 NDP 消息的可选项字段的 TLV 的格式

在表 2-6 中显示了可选的数值和它们相对应的类型号。本章并不讲述各个数值字段的格式，有关数值字段的详细描述，读者可以参考 RFC 2461。

表 2-6 数值字段和相应的类型

类型	数值
1	源链路层地址
2	目标链路层地址
3	前缀信息
4	重定向的报头
5	MTU

2.5.2 路由器发现（Router Discovery）

路由器通过在相连的链路上周期性地发送路由器通告消息，表明它的存在，并通过路由器通告消息通告所配置的所有参数。大概用到路由器消息最多的链路就是广播链路了，例如以太网，链路上的主机可以收到路由器通告消息，因而可以学到有关链路的必要信息。

在 RFC 2461 中规定，路由器通告消息发送的周期是 4~1800s 之间，缺省值为 600s。同时也规定了路由器通告消息通告的最小周期缺省为 200s。 这些通告将在最大值和最小值之间波动以防止链路上的同步。

一些未经请求的路由器通告消息的源地址是发送该消息的路由器接口的链路本地 IPv6 地址，而目的地址是所有节点的多播地址（FF02::1）。

在 Cisco 公司的路由器上，只要使用命令 **ipv6 unicast-routing** 使 IPv6 有效，Cisco 路由器就可以在以太网和 FDDI 接口上自动地发送路由器通告消息。缺省的间隔值是 200s，并且该值可以通过命令 **ipv6 nd ra-interval** 进行改变。缺省条件下，所传送的路由器通告消息的路由器生存时间是 1800s，并可以通过命令 **ipv6 nd ra-lifetime** 进行改变。如果读者不希望链路上的某台路由器作为缺省路由器，就可以使用该命令将该路由器的生存时间设置为 0。路由器通告消息的可达时间缺省为 0（意味着是未指定的），并可以通过命令 **ipv6 nd reachable-time** 进行改变。缺省条件下，重传计时字段设置为 0ms（未指定的），并可以通过命令 **ipv6 nd ns-interval** 进行改变。M 标记与 O 标记字段的值可以分别通过命令 **ipv6 nd managed-config-flag** 与 **ipv6 nd other-config-flag** 来设定。如果读者根本不希望某个接口传送路由器通告消息，可以通过命令 **ipv6 nd suppress-ra** 使其无效。

在缺省的情况下，Cisco 路由器在路由器通告消息中包含了始发该消息的接口配置的所有 IPv6 前缀。可以通过命令 **ipv6 nd prefix** 来控制前缀的通告，以及与这些前缀相关的参数。

刚刚连接到某个链路接口的主机需要等待一个路由器通告消息，以便能够发现链路上的路由器和学习到链路的参数，显然，要等待 200s 的时间显得过长了。因此，当链路上的一台主机开始激活时，就会发送一个路由器请求消息去请求立即得到一个路由器通告消息。这个路由器请求消息的源地址可以是未指定的地址（::），也可以是该主机的链路本地 IPv6 地址。它的目的地址则始终是所有路由器的多播地址（FF02::2）。

当一台路由器收到一条路由器请求消息时，它就会发送（有一个 5s 的延时）一条路由器通告消息作为响应。如果触发某个路由器通告消息的路由器请求消息的源地址是一台主机的链路本地地址，那么这条路由器通告消息就会使用它的链路本地地址以单播方式被传送给该主机。如果路由器请求消息的源地址未指定，那么它所请求的路由器通告消息就会以多播方式被传送给所有的节点地址。

当一台主机收到一条路由器通告消息时，它就会将这台路由器添加到它的缺省路由器列表中（除非这条路由器通告消息指定它的路由器生存时间为 0 而不能作为缺省路由器使用）。如果一台主机在它的缺省路由器列表中具有多台路由器条目，那么主机就应该明确地给出如何选定一个缺省路由器的方法。它要么在整个缺省路由器列表中依次轮询，要么选择和保持单台路由器作为缺省路由。当出现与主机所选择的不同的缺省路由时，不管上述两种情况的哪一种，重定向功能对于用来更新主机的缺省路由都是重要的。

2.5.3 地址自动配置（Address Autoconfiguration）

当一台 IPv6 的主机第一次连接到链路上时，它能够自我配置其自己的接口地址。这个过程的第一步是确定地址的 64 位接口 ID 部分。对于广播型的接口（大多数主机是这种情况），使用一种称为 MAC-to-EUI64 转换法的机制。简单地讲，这个机制采用接口的 48 位介质存取

控制（MAC）地址——通常认为它是全球惟一的，在这个 MAC 地址中间插入一个保留的 16 位数值 0xFFFE，并把它的全局/本地（Universal/Local，U/L）位翻转设置为 1（全局的，事实上是指 IEEE 指定），这样就把它转换成了一个 64 位的接口 ID。

在图 2-15 中演示了这个转换过程，图中被转换的地址是 0000:0B0A:2D51。为了更易于理解所发生的变化，在图中把 MAC 地址表示成二进制格式。第一步，把 MAC 地址从中间的位置分开成两个 24 位的对等部分，并在这两个 24 位对等部分之间插入 0xFFFE。这样这个 MAC 地址就变成了 64 位的长度。接着，把原来 MAC 地址中的 U/L 位——始终是第 7 位——从 0 翻转为 1。最后得到的结果就是一个有效的 64 位接口 ID，即 0200:0BFF:FE0A:2D51。

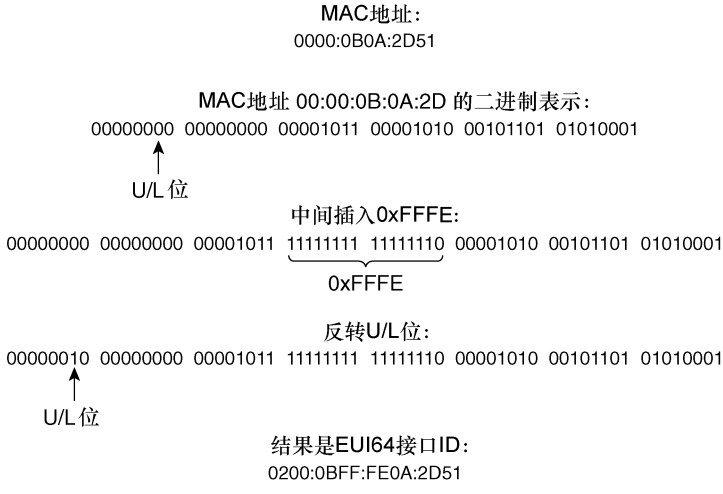


图 2-15 MAC 到 EUI64 转换法用来基于接口的 48 位 MAC 地址创建一个 64 位的接口 ID

当然，这个接口 ID 也仅仅是一个 IPv6 地址的一半，还需要一个 64 位的前缀。读者回想一下表 2-1 中链路本地前缀是一个保留的、公认的数值 0xFF80::/10。使用它作为一个完整的 64 位前缀（0xFF80::/64），再加上转换导出的接口 ID，这样就构成了一个完整的 IPv6 地址，可以用来和同一链路上其他设备进行通信。例如，把链路本地前缀和图 2-15 中转换导出的接口 ID 组合在一起就得到了一个链路本地 IPv6 地址：FF80::0200:0BFF:FE0A:2D51。下面显示了一个链路本地地址的例子，在该例中链路地址来自于一台 Macintosh OS X 主机的以太网接口“en1”。利用链路本地前缀 FF80::/10 和 MAC 到 EUI64 转换法，它的 IPv6 接口不用借助任何其他设备的帮助就可以导出自己的链路本地地址：

```
[Jeff-Doyles-Computer:~] jdoyle% ifconfig en1
en1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1300
        inet6 fe80::211:24ff:fe23:334e prefixlen 64 scopeid 0x5
        inet 10.10.24.13 netmask 0xffffff00 broadcast 10.10.24.255
        ether 00:11:24:23:33:4e
        media: autoselect status: active
        supported media: autoselect
[Jeff-Doyles-Computer:~] jdoyle%
```

如果一台主机仅仅需要和所在链路上的设备通信，那么它自动配置的链路本地地址就已经足够了。但是如果主机需要和链路以外的设备进行通信，那么它就需要一个更大范围的地址——通常是一个全球 IPv6 地址。这可以通过两种途径获取该地址：有状态或无状态的地址自动配置。

如果一台主机使用有状态地址自动配置方式，它将会借助 DHCPv6 服务器来获取必要的地址信息。它要么根据预先的配置去查找 DHCPv6 服务器，要么由收到的可能设置了 M 标

记的路由器通告消息来告诉它使用 DHCPv6 服务器。DHCPv6 协议在 RFC 3315 中描述，它与 IPv4 的 DHCP 协议在最终结果上没有太多的不同。

更有趣的是无状态的自动配置。这是一个非常简单的过程，主机从它所收到的路由器通告消息中获取一个或多个链路前缀，然后加上它先前确定的接口 ID，这样就得到了一个全球惟一的 IPv6 地址。举例来说，假设图 2-15 中的主机收到了一个路由器通告消息通告的前缀为 3FFE:1104:404:1::/64，把这个前缀加上它的接口 ID 就构成了它的全球地址：3FFE:1104:404:1:0200:0BFF:FE0A:2D51。

2.5.4 地址冲突检测（Duplicate Address Detection）

虽然利用 MAC 地址转换导出一个接口 ID 的方法，大多数情况下通常可以保证在任何范围下得到一个惟一的地址，但是确保地址的惟一性无疑是明智的。所以不管一台设备是如何获取一个地址的，在使用这个地址之前都必须进行地址冲突检测。地址冲突检测不考虑地址是通过有状态或无状态配置方式获取的，还是人工静态配置获取的。这个规则只有一个例外情况就是任意播地址，因为定义的任意播地址可以出现在不止一台的设备上。在这里存在这样一个假设：如果一个链路本地地址具有通过 MAC 到 EUI64 转换法导出的接口 ID，通过了地址冲突检测并被认为是惟一的，那么其他使用相同的接口 ID 的地址也被认为是惟一的，而不需要重新执行地址冲突检测。

已经获取一个新地址的节点会把这个新的地址归类为临时状态的地址。在地址冲突检测操作没有完成并确认链路上没有其他节点使用这个地址之前，该地址不能被使用。这个节点会发送一个把目标地址字段设置为该地址的邻居请求消息来验证。邻居请求消息的源地址是未指定的地址，而它的目的地址是被请求节点的多播地址。

一个被请求节点的多播地址由前缀 FF02:0:0:0:0:1:FF00::/104 前导，加上目标地址的最后 24 位组成。举例来说，对于图 2-15 中导出的接口 ID，这里被请求节点的多播地址是 FF02::1:FF0A:2D51。这样做的原因是如果一个节点自动配置了多个接口地址，它所有地址的最后 24 位都应该是相同的。因此，一个带有被请求节点多播地址的邻居请求消息应该匹配它所有的接口地址。更重要的是，使用被请求节点的多播地址可以确保在两个节点试图同时对同一个地址执行地址冲突检测操作的时候，它们可以互相检测到。

如果一个节点收到一个邻居请求消息，并且它的目标地址与该节点所分配的其中一个地址匹配，它就会发送一个目标地址和目的地址都为试探地址的邻居通告消息。发起这个邻居请求消息的节点就会知道这个试探地址是冲突的，并且不能使用。

2.5.5 邻居地址解析（Neighbor Address Resolution）

读者在第 1 章中已经了解到，当一个 IPv4 节点希望和本地链路上的另一个 IPv4 节点通信时，它必须首先发现目的节点的链路层地址（或称为数据链路地址）；然后这个地址被作为封装 IP 数据包到那个节点的数据帧的目的地址。举例来说，一个节点可能希望发送一个数据包到 examplehost.com，DNS 的查询会返回地址 3FFE:521:2400:15:211:24FF:FE23:334E，发送端节点这时必须发现链路层地址用作本地链路帧的目的地址。正如前面章节所讨论的，IPv4 协议使用 ARP 获取这个地址，而 IPv6 则使用 NDP 获取这个地址。

当一个节点检验由 DNS 返回的 IPv6 地址的前缀后，它就可以判断出目的节点是它所在的本地链路上的邻居，还是一个本地链路以外的节点并因此需要通过缺省路由器才能可达。如果是属于后一种情况，这个节点应该已经知道来自路由器通告消息的缺省路由器的链路层地址。但是，如果目的节点位于本地链路上，那么节点就会首先查找邻居缓存看一下是否已经学到该地址。IPv6 协议中的邻居缓存与 IPv4 协议中的 ARP 缓存非常类似，它记录了网络层的地址和与之相关联的链路层地址。下面显示了一个微软 WindowsXP 操作系统主机的邻居缓存。这个邻居缓存保存了主机已知的 IPv6 地址和与它们相对应的链路层地址：

```
C:\Documents and Settings\Jeff Doyle>ipv6 nc
5: fe80::202:2dff:fe25:5e4c 00-02-2d-25-5e-4c permanent
4: fe80::260:83ff:fe7b:2df3 00-60-83-7b-2d-f3 stale (router)
4: fe80::210:a4ff:fea0:bc97 00-10-a4-a0-bc-97 permanent
4: 3ffe:3700:1100:1:210:a4ff:fea0:bc97 00-10-a4-a0-bc-97 permanent
4: 3ffe:3700:1100:1:d9e6:b9d:14c6:45ee 00-10-a4-a0-bc-97 permanent
4: 2001:468:1100:1:210:a4ff:fea0:bc97 00-10-a4-a0-bc-97 permanent
4: 2001:468:1100:1:d9e6:b9d:14c6:45ee 00-10-a4-a0-bc-97 permanent
3: 2002:c058:6301::c058:6301 192.88.99.1       permanent
3: 2002:836b:213c::836b:213c 131.107.33.60     permanent
3: 2002:4172:a85b::4172:a85b 127.0.0.1         permanent
3: 2002:836b:213c:1:e0:8f08:f020:6 131.107.33.60     permanent
3:  2001:708:0:1::624                  incomplete
2:    ::65.114.168.91 127.0.0.1        permanent
2: fe80::5efe:65.114.168.91 127.0.0.1         permanent
2: fe80::5efe:169.254.113.126 127.0.0.1         permanent
1:           fe80::1                   permanent
1:               ::1                   permanent
```

如果一个地址不在邻居缓存中，它也显示在表中但被标记成不完全的（Incomplete），这表示正在进行该地址的地址解析处理。这样，主机会发送一个邻居请求消息到与目标节点相关的被请求节点的多播地址。邻居请求消息应该包含源链路层可选项（类型 1），这样被请求的节点就会了解到请求节点的链路层地址，并因此知道发送作为答复的邻居通告消息到哪一个节点。如果在路由器通告消息中包含了不为 0 的值，那么根据指定的时间间隔可以发送多个邻居请求消息。如果一个路由器通告消息中的重传计时字段的值为未指定的（0），那么邻居请求消息将会每 1 000ms 重传一次，直到收到一个邻居通告消息为止。如果发送了 3 个邻居请求消息后，还是没有收到来自被请求节点的邻居通告消息，那么邻居地址解析就会失败，并会为每一个排队等待传送到目前还未知的目的地的数据包返回一个类型 1/代码 3（目的地不可达/地址不可达）的 ICMP 消息。

假如被请求的节点存在并且邻居请求消息是有效的，那么它将答复一个邻居通告消息。这个邻居通告消息的目标地址字段设置为触发该邻居通告的那个邻居请求消息的目标地址字段的值。直到接收到回应的邻居通告消息，发起请求的节点才能够将目标节点的链路层地址添加到邻居缓存的条目中，并把该条目从不完全的（Incomplete）状态更改成可达的（Reachable）状态。

在 Cisco 路由器上，可以通过命令 **show ipv6 neighbors** 来查看邻居缓存，参见示例 2-1。

示例 2-1　在 Cisco 路由器上，可以通过命令 show ipv6 neighbors 来显示邻居缓存

```
Confucius# show ipv6 neighbors

IPv6 Address                              Age  Link-layer Addr  State  Interface
2001:201:1502:1:210:a4ff:fea0:bc97          0  0010.a4a0.bc97   REACH  Ethernet0
fe80::210:a4ff:fea0:bc97                    0  0010.a4a0.bc97   REACH  Ethernet0
fe80::260:83ff:fe4c:5df2                    0  0060.834c.5df2   REACH  Ethernet0
3ffe:1300:a47:20:d9e6:b9d:14c6:45ee         0  0002.2d25.5e4c   REACH  Ethernet1
```

2.5.6 私有地址（Privacy Address）

无状态地址自动配置在涉及到以下的一些情况时会产生一种安全问题：即使一台设备从一个子网移到另一个子网，或者从一个主网络转移到另一个主网络，它的接口 ID 也始终保持不变。如果它的接口 ID 保持不变，那么它就能够被追踪。至少，这成为了一个私密的问题。例如，假设读者正在使用 IPv6 的地址和其所在公司的网络相连。记录和分析来自网络某部分的数据包就能够通过不变的接口 ID 来识别读者的身份。而且，你的同事通过进一步分析前置在你的接口 ID 前的不同前缀，推断出任何时候你所在的位置：在公司工作，在家里，旅行，或者任何活动。更为严重的是，他们可以利用这些追踪以及保存有关你的活动和位置的记录，来进行犯罪交易。

在 RFC 3041 中，通过定义 IPv6 私有地址来解决该安全问题。私有地址就是通过利用伪随机数字算法生成的接口 ID。接口 ID 大约一天变化一次（或根据所配置的周期变化），也会在节点获取一个新的 IPv6 地址时改变。这一点非常重要，也符合合理的保密要求。

当然，一个经常变化的地址对于可达性而言是不实际的。希望与你通信的节点，以及 DNS 服务器必须通过一个或一些静态地址了解你所在位置。因此，标准的无状态配置的 IPv6 地址保留了你的“公共”地址；任何一个需要向你发送数据包的节点都使用这个地址作为目的地址。但当你发送返回的数据包时，你使用的却是私有地址。这有点像你家里的主叫 ID（Caller ID），但它会阻止你的号码显示在其他人的主叫 ID 上。你能够看到是谁正在呼叫你，但是，当你呼叫其他人的时候他们却不能看到你的号码。

下面显示了分配给一台微软 Windows XP 系统的地址。在这里，有两个公共的 IPv6 地址分配给这台机器的接口，读者可以看到它们的前缀虽然不同，但它们的由 MAC 到 EUI64 转换得到的接口 ID 却是相同的。可以通过插入到地址中间的 0xFFFE 来容易地识别出公共接口 ID。但是，对于这两个公共地址，也有一个私有地址（Windows 的标签是作为“anonymous”），这些私有地址是通过在随机生成的接口 ID 前加上路由器通告消息发现的 IPv6 前缀生成的。公共和私有的（这里称为“anonymous”）地址一起用来确保主机的隐匿并同时维持主机的可达性：

```
C:\Documents and Settings\Jeff Doyle>ipv6 if 4
Interface 4: Ethernet: Local Area Connection 2
  uses Neighbor Discovery
  uses Router Discovery
  link-layer address: 00-10-a4-a0-bc-97
    preferred global 2001:484:1200:1:d9e6:b9d:14c6:45ee,
     life 6d21h14m26s/21h12m4s (anonymous)
    preferred global 2001:468:1200:1:210:a4ff:fea0:bc97,
     life 29d23h59m25s/6d23h59m25s (public)
    preferred global 3ffe:3705:1200:1:d9e6:b9d:14c6:45ee,
     life 6d21h14m26s/21h12m4s (anonymous)
    preferred global 3ffe:3705:1200:1:210:a4ff:fea0:bc97,
     life 29d23h59m25s/6d23h59m25s (public)
    preferred link-local fe80::210:a4ff:fea0:bc97, life infinite
    multicast interface-local ff01::1, 1 refs, not reportable
    multicast link-local ff02::1, 1 refs, not reportable
    multicast link-local ff02::1:ffa0:bc97, 3 refs, last reporter
    multicast link-local ff02::1:ffc6:45ee, 2 refs, last reporter
  link MTU 1500 (true link MTU 1500)
  current hop limit 64
  reachable time 22000ms (base 30000ms)
  retransmission interval 1000ms
  DAD transmits 1
```

2.5.7 邻居不可到达性的检测

在前面的章节里讨论邻居地址解析时，已经提到了标记为 Incomplete 或 Reachable 的邻居缓存条目。事实上，一个邻居缓存条目可以具有以下 5 种状态：

- **Incomplete**——该状态说明正在进行邻居地址的解析处理。一个邻居请求消息已经为该条目发送被请求节点的多播地址，但是还没有收到相应的邻居通告消息。
- **Reachable**——该状态说明目前地址已经被确认为是可到达的。“目前被确认”的意思是指在路由器通告消息中 Reachable Time 字段指定的时间内，已经收到了某些可达性的信息。如果在路由器通告消息里没有指定 Reachable Time 时间，那么将使用 30s 作为缺省的 Reachable Time 时间。
- **Stale**——该状态说明自从收到最新的有关到达目的地的可达性肯定的确认后，已经经历了 Reachable Time 所指定的时间。
- **Probe**——该状态是指每经过 Retransmit Time 时间或每 1 000ms（如果没有指定 Retransmit Time 时间的话），节点就会通过向目的节点发送邻居请求消息来搜索可达性的确认。
- **Delay**——当一个数据包发送到一个处于 Stale 状态的目的节点时，这个地址就进入到该状态。如果该地址在 Delay 状态等待了 5s，并在这段时间内还没有收到有关可达性的确认，那么该状态就转换到 Probe 状态。这个状态是在节点发出搜索的邻居请求消息前，给上层协议一个确认可达性的机会的优化措施。

对于邻居可达性的确认，一般可通过以下两种方法来确认：

- 来自上层协议的“提示（Hints）”，例如像一个 TCP 消息的 ACK 等。
- 通过请求一个路由器通告消息或邻居通告消息得到有关对目的地址搜索的响应。这是一种必要的措施，因为对于某些上层协议，例如 UDP 协议，它本身并不对所传送消息的接收进行主动的确认。

邻居不可达性的检测不仅仅要确认从邻居到本地节点一方的可达性，而且还要确认从本地节点到邻居方向的可达性，确保双向的可达性。基于这个原因，一个未经请求的路由器通告消息或邻居通告消息，不能改变邻居缓存条目的状态到 Reachable；所收到的消息仅仅可以表明从始发节点到本地节点的单向可达性。只有收到一个传输层消息的远程响应消息（例如一个 TCP 数据包的 ACK），或者收到一个响应请求消息的路由器通告消息或邻居通告消息后，才能确认双向的可达性。

2.6 展　望

讲述本章和前面一章的目的是为了阐明基于两个版本的 IP 协议的基础概念。理解 IP 地址的基本概念和有关 IP 的基础知识可以作为下一步掌握 IP 路由选择知识的基础。本书后面的章节将开始研究一台路由器能够成功地和精确地转发数据包到达它的目的地所需要的信息。

2.7 复 习 题

1．一个 IPv6 地址的长度是多少？
2．怎样表示一个 IPv6 地址？
3．用来压缩和简化 IP 地址表示的两条规则是什么？
4．为什么在一个 IPv6 地址里使用多个双冒号是不允许的？
5．IPv6 地址::/0 和::/128 有什么不同之处？
6．在单播 IPv6 地址中，用来指定主机的部分是什么？它的长度是多少？
7．一个单播 IPv6 地址的子网 ID 部分的长度是多少？
8．假设一个 IPv6 地址的起始 10 位是 FF80::/10，那么它是什么类型的地址？
9．3FFE:204:100:90::1 是什么地址类型？
10．什么是任意播地址？
11．什么是多播地址？
12．一个 IPv6 报头的长度是多少？
13．在 IPv6 报头中设置流标签字段的目的是什么？
14．在 IPv4 报头中，什么字段和 IPv6 的下一报头字段相对应的？
15．在 IPv4 报头中，什么字段和 IPv6 的跳数限制字段相对应的？
16．在 IPv6 的下一报头字段中，有哪些方面像 IPv4 的协议号字段?又有哪些地方不同？
17．怎样扩展报头以便使 IPv6 的数据包更加富有效率？
18．ICMPv6 中下一报头的值是什么？
19．IPv4 的分段和 IPv6 的分段有哪些重要的不同之处？
20．用于邻居发现协议的 5 种 ICMPv6 消息是什么？
21．在一个路由器通告消息中，M 标记和 O 标记的用途是什么？
22．在一个路由器通告消息中，可达时间字段的用途是什么？
23．在一个路由器通告消息中，重传计时字段的用途是什么？
24．在一个路由器通告消息中，如果它的路由器生存时间字段被设置为 0 代表什么意思？
25．在一个邻居通告消息中，S 标记的用途和作用是什么？
26．有状态地址自动配置和无状态地址自动配置有什么不同？
27．MAC 到 EUI64 转换使用哪两个步骤可以导出一个接口 ID？
28．在一台设备获取一个单播 IPv6 地址时，它必须执行地址冲突检测操作，但是有一个例外情况，这个例外情况是什么？
29．前缀 FF02:0:0:0:0:1:FF00::/104 表示什么意思？
30．在 IPv6 中，使用什么代替 ARP 和 ARP 缓存？
31．什么是私有地址？
32．在一个邻居缓存中，Incomplete 状态的条目表示什么意思？
33．在一个邻居缓存中，Probe 状态的条目表示什么意思？
34．邻居不可达性检测使用哪两种方法来确认一个邻居双向的可达性？

本章包括以下主题：

- 路由表；
- 配置静态路由；
- 静态路由故障诊断。

第 3 章

静态路由

第 1 章中有一点非常重要，就是在 OSI 模型定义中，数据链路层/物理层和传输层/网络层执行的任务非常相似：它们都提供了数据传输的手段，即沿某条路径将数据从源点发往目的地的方法。不同之处在于，数据链路层/物理层提供跨物理路径的通信服务，而传输层/网络层则提供跨由一连串的数据链路组成的逻辑路径或虚拟路径的通信服务。

此外，第 1 章中还阐述了沿物理路径进行通信，必须获得有关数据链路标识和数据封装的信息，并且这些信息要保存在数据库中，如 ARP 高速缓冲。同样，传输层/网络层也需要获取和保存所涉及到的相关信息。有区别的是，这些信息被保存在路由表中，路由表又叫路由选择信息库（RIB）。

本章所要研究的内容包括：需要何种信息路由数据包，怎样在路由表中保存这些信息，如何向数据库中输入这些信息，以及通过向正确的路由器的路由表中输入正确的信息来建立一个可路由网络的相关技术。

3.1 路由表

为了理解路由表中的信息种类，我们需要先考虑数据包到达路由器接口时会发生什么，这是非常有用的。首先，路由器会检查数据帧目标地址字段中的数据链路标识。如果它包含了路由器接口标识符或广播标识符，那么路由器将从帧中剥离出数据包并传递给网络层。在网络层，路由器将检查数据包的目标地址。如果目标地址是路由器接口的 IP 地址或是所有主机的广播地址，那么需要进一步检查数据包的协议字段，然后再把被封装的数据发送给适当的内部

进程。[1]

除此之外，所有其他目标地址都需要进行路由选择。这里的目标地址可能是另一个网络上的主机地址，该网络或者与路由器相连（包括与那个网络相连接的路由器接口），或者不直接连接到路由器上。目标地址还可能是一个定向的广播地址，这种地址有明确的网络地址或子网地址并且主机位全部为 1。这些地址也是可以路由的。

如果数据包是可以被路由的，那么路由器将会查找路由表获得一个正确的路径。在数据库中的每个路由表项最少必须包括下面两个项目：

- **目标地址**——这是路由器可以到达的网络地址。正像本章所解释的，路由器可能会有多条路径到达相同的地址或是相同主网 IP 地址下的一组等长或变长的子网。
- **指向目标的指针**——指针不是指向路由器的直连目标网络就是指向直连网络内的另一台路由器地址，或者是到这个链路的本地接口。更接近目标网络一跳的路由器叫下一跳（next hop）路由器。

路由器将会尽量地进行最精确的匹配。[2] 按精确程度递减的顺序，可选地址排列如下：

- 主机地址（主机路径）；
- 子网；
- 一组子网（一条汇总路由）；
- 主网号；
- 一组主网号（超网）；
- 缺省地址。

本章将提供有关前 4 类的例子。超网将会在第 6 章中讨论。缺省地址是最不明确的地址，只有当所有匹配都失败时才被使用。缺省地址会在第 12 章中讨论。

如果数据包的目标地址不能匹配到任何一条路由表项，那么数据包将被丢弃，同时一个“目标网络不可达”的 ICMP 消息将会被发送给源地址。

如图 3-1 所示，这是一个简单的网络，图中给出了每台路由器需要的路由表项。这里最重要的是这些路由表将如何作为一个整体运行并能准确高效地传输数据包。路由表的网络栏列出了路由器可达的网络地址。指向目标网络的指针在下一跳栏中。

在图 3-1 中，如果路由器 Carroll 收到一个源地址为 10.1.1.97、目标地址为 10.1.7.35 的数据包，路由表查询的结果是：目标地址的最优匹配是子网 10.1.7.0，可以从 S0 接口出站经下一跳地址 10.1.2.2 去往目的地。数据包被发送给路由器 Dahl，Dahl 查找自己的路由表后发现数据包应该从 S1 接口出站经下一跳 10.1.4.2 去往目标网络 10.1.7.0。此过程将一直持续到数据包到达路由器 Baum。当 Baum 在接口 S0 接收到数据包时，Baum 通过查找路由器，发现目的地是连接在端口 E0 的一个直连网络。最终结束路由选择过程，数据包被传递给以太网链路上的主机 10.1.7.35。

上面说明的路由选择过程是假设路由器可以将下一跳地址同它的接口进行匹配。例如，路由器 Dahl 必须知道通过接口 S1 可以到达 Lewis 的地址 10.1.4.2。首先 Dahl 从分配给接口 S1 的 IP 地址和子网掩码可以知道子网 10.1.4.0 直接连接在接口 S1 上；那么 Dahl 就可以知道

[1] 还有一种特殊情况，那就是组播地址，它表示一组设备而不是所有设备。D 类地址 224.0.0.5 就是一个组播地址，这个地址为所有 OSPF 路由器保留。

[2] 寻找最优匹配有两个基本过程，它们依赖于路由器是否表现为有类别或无类别。有类别路由表的查找过程参见第 5 章的相关内容，无类别路由表的查找参见第 6 章的相关内容。

10.1.4.2 是子网 10.1.4.0 的成员，而且一定被连接到该子网上。

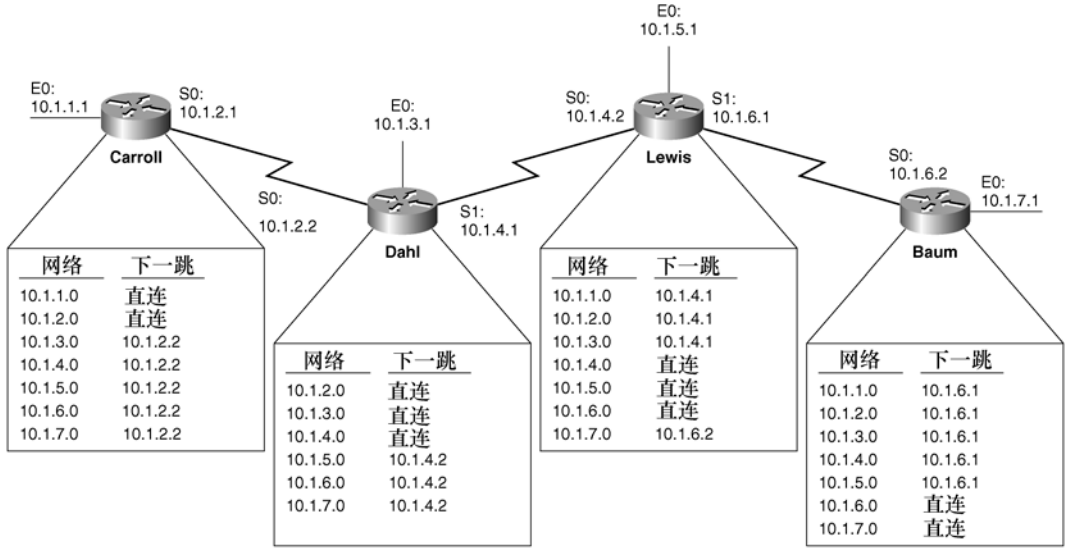


图 3-1 每一个路由表项目需要的信息至少应该包括目标网络地址和指向目标网络地址的指针

注意，为了正确地进行数据包交换，每台路由器都必须保持信息的一致性和准确性。例如，在图 3-1 中，路由器 Dahl 的路由表中丢失了关于网络 10.1.1.0 的表项。从 10.1.1.97 到 10.1.7.35 的数据包将被传送，但是当 10.1.7.35 向 10.1.1.97 回复数据包时，数据包从 Baum 到 Lewis 再到 Dahl 传递。Dahl 查找路由表后发现没有关于子网 10.1.1.0 的路由表项，因此丢弃此数据包，同时 Dahl 向主机 10.1.7.35 发送目标网络不可达的 ICMP 信息。

示例 3-1 给出了图 3-1 中路由器 Lewis 的路由表。在 Cisco 路由器中查看路由表的 IOS 命令是 **show ip route**。

检查数据库的内容并把它与图 3-1 中路由器 Lewis 的一般路由表相比较。可以看到，表最上方的关键字是对路由表左侧的一列字母的解释。这些字母指明了每个路由表项是如何学习到的。在示例 3-1 中，标记为 C 的路由表示直连网络，标记为 S 的路由表示静态路由。声明“gateway of last resort is not set”指的是缺省路由。

示例 3-1 图 3-1 中路由器 Lewis 的路由表

```
Lewis#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP,
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route, o - ODR
Gateway of last resort is not set
     10.0.0.0/24 is subnetted, 7 subnets
S       10.1.3.0 [1/0] via 10.1.4.1
S       10.1.2.0 [1/0] via 10.1.4.1
S       10.1.1.0 [1/0] via 10.1.4.1
S       10.1.7.0 [1/0] via 10.1.6.2
C       10.1.6.0 is directly connected, Serial1
C       10.1.5.0 is directly connected, Ethernet0
C       10.1.4.0 is directly connected, Serial0
Lewis#
```

表头有一句声明主网络地址 10.0.0.0 有 7 个已知子网，掩码为 24 位。在 7 个路由表项中，每一个都给出了目标子网。对于不是直连网络的表项——数据包必须转发到下一跳路由器——置于括号内的元组指明了路由的［管理距离/度量］。管理距离将会在本章后面介绍，在第 11 章中还将详细讨论。

度量是通过优先权评价路由的一种手段，度量越低，路径越短，也就是该路径更理想。在第 4 章中将会详细讨论度量。注意，在示例 3-1 中静态路由的度量为 0。最后，路由表还给出了下一跳路由器的接口地址或连接直连目标网络的接口。

3.2 配置静态路由

路由表可以用下面 3 种方式之一获取信息：

- 基于路由器的直连子网；
- 以静态路由表项的方式手工输入信息；
- 通过某种自动信息发现和共享系统（动态路由选择协议）自动地获取信息。

本书将花大量篇幅讲述动态 IP 路由选择协议，这里讨论一下静态路由配置有助于读者理解后继章节。

除了上述目的外，在某些场合，人们宁愿选用静态路由选择，而不是动态路由选择。对于任何过程而言，自动化程度越高，可控程度就越差。虽然动态（自动）路由选择要求更少的人为干涉，但静态路由选择允许在网络的路由选择行为上实施非常精确的控制。然而为此付出的代价是，每当网络拓扑结构发生变化时都需要重新进行手工配置。

3.2.1 案例研究：简单 IPv4 静态路由

如图 3-2 所示，网络有 4 台路由器和 6 个数据链路。注意，网络 10.0.0.0 的几个子网是不连续的——有一个不属于 10.0.0.0 的子网（在 Tigger-to-Piglet 链路上的 192.168.1.192）将子网 10.1.0.0 与 10.0.0.0 的其他子网分离开来。10.0.0.0 子网是可变长子网——在整个网络中子网掩码长度不一致：子网 10.1.0.0 有 16 位子网掩码，而 10.4.0.0 是 24 位。最后要说的是，路由器 Pooh 的以太网链路上的子网地址是一个全 0 子网。在后面章节可以看到这种编址方式对于更简单的有类别路由选择协议，如 RIP 和 IGRP，将会产生一些问题；而静态路由则工作得很正常。

网络的静态路由选择过程共有 3 步：

步骤 1：为网络中的每个数据链路确定子网或网络地址。

步骤 2：为每台路由器标识所有非直连的数据链路。

步骤 3：为每台路由器写出关于每个非直连地址的路由语句。

这里没有必要写出有关直连数据链路的路由描述，因为在路由器接口上配置的地址和掩码可以使这些直连网络被记录在路由表中。

例如，在图 3-2 中的 6 个子网是：

- 10.1.0.0/16
- 10.4.6.0/24
- 10.4.7.0/24

- 192.168.1.192/27
- 192.168.1.64/27
- 192.168.1.0/27

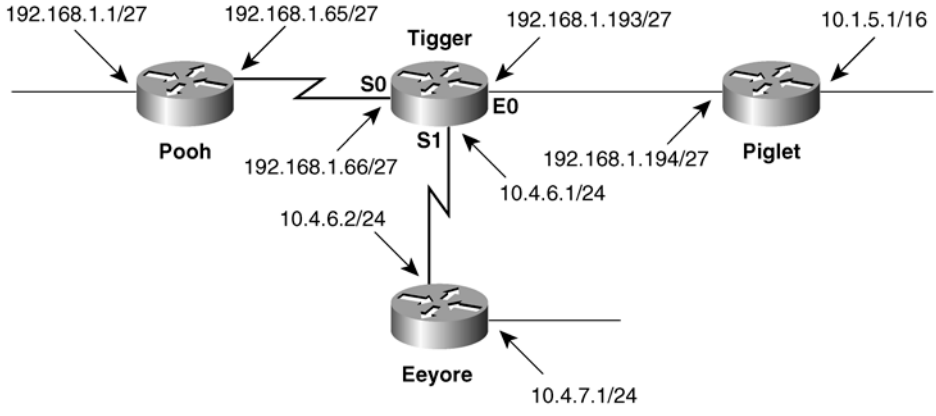


图 3-2 路由选择协议 RIP 和 IGRP 对于含有非连续和可变长子网的网络来说不能进行正常的路由，而静态路由没有问题

为了在路由器 Piglet 上配置静态路由，将那些非直连的子网标识如下：

- 10.4.6.0/24
- 10.4.7.0/24
- 192.168.1.64/27
- 192.168.1.0/27

对于静态路由来说，这些子网必须被记录下来。在路由器 Piglet 上配置静态路由[1]的命令见示例 3-2。

示例 3-2 配置 Piglet 的静态路由表项

```
Piglet(config)# ip route 192.168.1.0 255.255.255.224 192.168.1.193
Piglet(config)# ip route 192.168.1.64 255.255.255.224 192.168.1.193
Piglet(config)# ip route 10.4.6.0 255.255.255.0 192.168.1.193
Piglet(config)# ip route 10.4.7.0 255.255.255.0 192.168.1.193
```

对其他 3 台路由器也执行相同的步骤，示例 3-3 给出了其他 3 台路由器的具体路由配置表项。

示例 3-3 路由器 Pooh、Tigger 和 Eeyore 的路由配置

```
Pooh(config)# ip route 192.168.1.192 255.255.255.224 192.168.1.66
Pooh(config)# ip route 10.1.0.0 255.255.0.0 192.168.1.66
Pooh(config)# ip route 10.4.6.0 255.255.255.0 192.168.1.66
Pooh(config)# ip route 10.4.7.0 255.255.255.0 192.168.1.66
Tigger(config)# ip route 192.168.1.0 255.255.255.224 192.168.1.65
Tigger(config)# ip route 10.1.0.0 255.255.0.0 192.168.1.194
Tigger(config)# ip route 10.4.7.0 255.255.255.0 10.4.6.2
Eeyore(config)# ip route 192.168.1.0 255.255.255.224 10.4.6.1
Eeyore(config)# ip route 192.168.1.64 255.255.255.224 10.4.6.1
Eeyore(config)# ip route 192.168.1.192 255.255.255.224 10.4.6.1
Eeyore(config)# ip route 10.1.0.0 255.255.0.0 10.4.6.1
```

[1] 要使本例及本章后续例子中的静态路由正常工作，必须附加两个全局配置命令：**ip classless** 和 **ip subnet-zero**，在 IOS 11.3 及更高版本中，**ip classless** 功能在缺省状态下是打开的。提及这两个命令主要考虑有读者希望在实验室中尝试这个配置实例，第 6 章将介绍这些命令。

如果读者记住每条命令描述一个路由表项的话，路由配置命令本身是很容易阅读的。命令 **ip route** 用于 IPv4，它后面跟着的是将要被输入到路由表中的地址、确定地址网络号的掩码及下一跳路由器的链接口地址。

配置 IPv4 静态路由还可以选择另一种命令，这种命令用出站接口代替下一跳路由器的接口地址，其中通过出站接口可以到达目标网络。例如，如示例 3-4 所示，可以这样配置路由器 Tigger 的路由表：

示例 3-4 另一种配置 Tigger 的路由表项的方法

```
ip route 192.168.1.0 255.255.255.224 S0
ip route 10.1.0.0 255.255.0.0 E0
ip route 10.4.7.0 255.255.255.0 S1
```

在配置静态路由时，必须满足以下条件。首先 IP 路由选择必须启动，如果使用下一跳地址，那么该地址必须可达；其次出战接口必须配置 IP 地址，接口必须正常工作。

示例 3-5 比较了两种配置方法所产生的路由表。注意，这里引入了一个错误，所有用静态路由指明的网络，如果静态路由参照出站接口，那么它们将被作为直连网络输入到路由表中。这里所涉及到路由重新分配的问题将在第 11 章中讨论。

示例 3-5 上面的路由表是指向下一跳路由器的静态路由表项产生的结果，下面是指向出站接口到达目标网络的静态路由表项产生的路由表 [1]

```
Tigger#show ip route
Gateway of last resort is not set
     10.0.0.0 is variably subnetted, 3 subnets, 2 masks
C       10.4.6.0 255.255.255.0 is directly connected, Serial1
S       10.4.7.0 255.255.255.0 [1/0] via 10.4.6.2
S       10.1.0.0 255.255.0.0 [1/0] via 192.168.1.194
     192.168.1.0 255.255.255.224 is subnetted, 3 subnets
C       192.168.1.64 is directly connected, Serial0
S       192.168.1.0 [1/0] via 192.168.1.65
C       192.168.1.192 is directly connected, Ethernet0
Tigger#

Tigger#show ip route
Gateway of last resort is not set
     10.0.0.0 is variably subnetted, 3 subnets, 2 masks
C       10.4.6.0 255.255.255.0 is directly connected, Serial1
S       10.4.7.0 255.255.255.0 is directly connected, Serial1
S       10.1.0.0 255.255.0.0 is directly connected, Ethernet0
     192.168.1.0 255.255.255.224 is subnetted, 3 subnets
C       192.168.1.64 is directly connected, Serial0
S       192.168.1.0 is directly connected, Serial0
C       192.168.1.192 is directly connected, Ethernet0
Tigger#
```

在示例 3-5 中，令人感兴趣的一点是，在 10.0.0.0 子网标题中指明了网络中使用了可变长子网掩码。可变长子网掩码（VLSM）是一种很有用的工具，我们会在第 6 章中讨论 VLSM。

配置静态路由的第三种选择是联合使用出站接口和下一跳地址，即下一跳地址加上指定的出站接口。如果出站接口失效，即使下一跳地址通过替代路由递归可达，路由依然会被删除。这样可以把与下一跳地址相关联的出站接口查询减到最小，同时使相应的路由表项不再是直连网络，而是距离为 1 的静态路由。

[1] 为了表达清楚，删除了路由器上方的关键字。

如果直接把静态路由指向一个广播型出站接口，而不使用下一跳地址，可能会导致广播网络上出现过多的流量，而且还可能耗尽路由器的内存。例如，如果我们在路由器 Tigger 上输入命令 **ip route 10.1.0.0 255.255.0.0**，路由器会认为 10.1.0.0 是直连网络。当路由器向 10.1.0.0/16 中的目标主机转发数据包时，路由器会发送 ARP 请求以便获取目标主机的 MAC 地址。如果广播网络上的一台路由器（ARP 代理）代表网络 10.1.0.0 回复 ARP 响应，那么不管数据包的目标地址是否有效，每次到达都会触发一个 ARP 请求和响应，并需要路由器配备大容量的 ARP 高速缓冲。例如 **ip route 10.1.0.0 255.255.0.0 E0 192.168.1.194**，如果在静态路由表项后面附加下一跳地址，那么路由器就不再认为目标网络是直连网络。这时，ARP 的请求对象只可能是下一跳地址，而且仅当第一个去往 10.1.0.0 的数据包才会触发 ARP 请求。

指定出站接口和下一跳地址可以最小化与下一跳地址关联的出站接口查询，并且把广播网络上的流量减到最小。

如示例 3-6 所示，当联合使用出站接口和下一跳地址时，静态路由表项会有所不同。

示例 3-6　在静态路由中指定出站接口，而不指定下一跳地址将会在广播网络上产生过多流量

```
Tigger#show ip route static
     10.0.0.0/16 is subnetted, 1 subnets
S       10.1.0.0 is directly connected, Ethernet0

Tigger#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  192.168.1.193            -   0004.c150.f1c0  ARPA   Ethernet0
Internet  10.1.8.1                0   0010.7b38.37d5  ARPA   Ethernet0
Internet  192.168.1.194           24   0010.7b38.37d5  ARPA   Ethernet0
Internet  10.1.5.5                 0   0010.7b38.37d5  ARPA   Ethernet0
Internet  10.1.1.1                 0   0010.7b38.37d5  ARPA   Ethernet0
Tigger#


Tigger#show ip route static
     10.0.0.0/16 is subnetted, 1 subnets
S       10.1.0.0 [1/0] via 192.168.1.194, Ethernet0

Tigger#show arp
Protocol  Address                Age (min)  Hardware Addr   Type   Interface
Internet  192.168.1.193                  -   0004.c150.f1c0  ARPA   Ethernet0
Internet  192.168.1.194                 22   0010.7b38.37d5  ARPA   Ethernet0
```

第一组路由表和 ARP 高速缓冲没有使用下一跳地址，网络 10.1.0.0 是直连网络，并且有多个 ARP 高速缓冲表项与之相关联，其中硬件地址相同，且都是路由器 192.168.1.194 的 MAC 地址。该路由器代替网络 10.1.0.0 中的所有主机回复 ARP 请求。在第 1 章中曾经讲述过，ARP 代理功能在 IOS 缺省状态下是打开的。

第二组表显示同时使用了出站接口和下一跳地址。注意，网络不再被直连连接了，而是通过 192.168.1.194 到达，出站接口是 E0。ARP 高速缓冲中没有关于 10.1.0.0 网络的表项，仅有实际存在的直连网络地址，其中包含 192.168.1.194 的。

3.2.2　案例研究：简单 IPv6 静态路由

IPv6 静态路由的配置方法和 IPv4 基本相同，惟一的不同的是 IPv4 网络掩码使用点分十

进制形式，而 IPv6 使用目标网络的前缀长度。不像 IPv4，IPv6 路由选择缺省情况下是关闭的，所以在输入 IPv6 静态路由前，必须使用命令 **ipv6 unicast-routing** 启动 IPv6 路由选择。同 IPv4 一样，在向路由表中添加 IPv6 路由选择之前，出站接口必须有效，并且接口上已配置好一个 IPv6 地址。创建静态路由的命令是 **ipv6 route**，该命令后面跟随的参数是目标网络、前缀长度（单位是比特）和下一跳路由器地址或去往目标网络的出站接口。

当我们需要确定静态路由表项中下一跳地址时，详细的网络图表会有所帮会，但是地址接口 ID 部分的动态特性又使得图表的信息容易变得过。在为 IPv6 网络分配地址时，要想预先指定下一跳地址，就必须手工指定接口 ID，而不能使用自动构建的 EUI-64 格式的地址。如果数据链路上接口已经配置了 EUI-64 的接口 ID，那么就只能指定地址的前 64 位。路由器将基于 MAC 地址确定后 64 位。如果路由器被新路由器替换，那么相应的 IPv6 地址也不同（前 64 位保持相同，后 64 位将不同）。为了标识邻接路由器的 128 位 IPv6 地址，可使用 Cisco 发现协议（CDP）的统计信息。CDP 可以给出关于邻接路由器的信息，例如路由器的主机名、路由器类型、IOS 和链路远端的 IP 地址。命令 **show cdp** 的显示格式如示例 3-7 所示。

示例 3-7　Cisco 发现协议可以给出大量关于邻居设备的信息

```
Honeybee#show cdp neighbor detail
-------------------------
Device ID: Honeytree
Entry address(es):
  IP address: 10.4.6.2
  IPv6 address: FE80::2B0:64FF:FE30:1DE0  (link-local)
  IPv6 address: FEC0::1:2B0:64FF:FE30:1DE0  (site-local)
Platform: cisco 2610,  Capabilities: Router
Interface: Serial0/0.2,  Port ID (outgoing port): Serial0/0.2
Holdtime : 146 sec

Version :
Cisco Internetwork Operating System Software
IOS (tm) C2600 Software (C2600-J1S3-M), Version 12.3(6), RELEASE SOFTWARE (fc3)
Copyright (c) 1986-2004 by cisco Systems, Inc.
Compiled Wed 11-Feb-04 19:24 by kellythw

advertisement version: 2
```

示例 3-7 给出了大量关于邻居设备的信息，包括路由器类型、IOS、主机名和 IP 地址。在命令中使用关键字 **detail** 可以获取所有显示信息。

另一种确定链路 IPv6 地址的方法是使用 **show ipv6 interface** 命令。该命令可以显示相关接口的 IPv6 信息。在路由器 Honeybee 上使用该命令后的输出结果如示例 3-8 所示。

示例 3-8　命令 **show ipv6 interface** 给出相关接口的 IPv6 信息，其中 IPv6 地址是 EUI-64 格式

```
Honeybee#show ipv6 interface serial0/0.1
Serial0/0.1 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::204:C1FF:FE50:F1C0
  Description: Link to Piglet
  Global unicast address(es):
    FEC0::3:204:C1FF:FE50:F1C0, subnet is FEC0:0:0:3::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF30:1DE0
```

图 3-3 是一个使用 IPv6 地址的简单网络。[1]

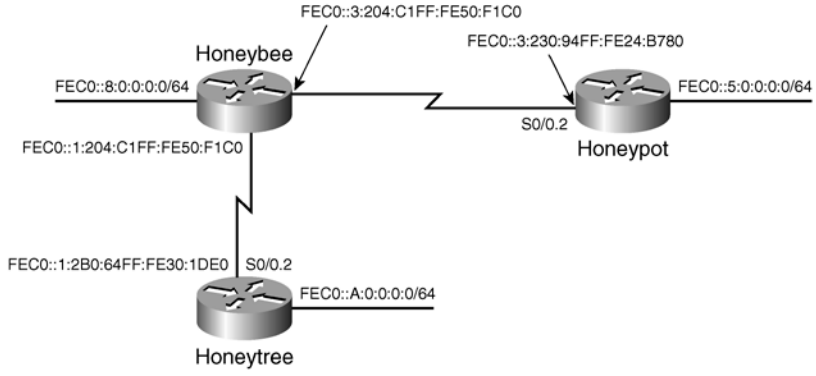


图 3-3 静态路由选择使用 IPv6 也可以工作

示例 3-9 给出了在 Honeypot 上配置静态 IPv6 路由的命令。

示例 3-9 在路由器 Honeypot 上配置静态 IPv6 路由

```
ipv6 unicast-routing
interface serial 0/0.2 point-to-point
 ipv6 address fec0:0:0:3::/64 eui-64
ipv6 route fec0::1:0:0:0:0/64 fec0::3:204:c1ff:fe50:f1c0
ipv6 route fec0::a:0:0:0:0/64 fec0::3:204:c1ff:fe50:f1c0
ipv6 route fec0::8:0:0:0:0/64 fec0::3:204:c1ff:fe50:f1c0
```

示例 3-10 和示例 3-11 分别给出了路由器 Honeytree 和 Honeybee 的路由表项。

示例 3-10 为 Honeytree 配置 IPv6 静态路由

```
ipv6 route fec0::8:0:0:0:0/64 fec0::1:204:c1ff:fe50:f1c0
ipv6 route fec0::3:0:0:0:0/64 fec0::1:204:c1ff:fe50:f1c0
ipv6 route fec0::5:0:0:0:0/64 fec0::1:204:c1ff:fe50:f1c0
```

示例 3-11 为 Honeybee 配置 IPv6 静态路由

```
ipv6 route fec0::a:0:0:0:0/64 fec0::1:2b0:64ff:fe30:1de0
ipv6 route fec0::5:0:0:0:0/64 fec0::3:230:94ff:fe24:b780
```

让我们看一下 Honeypot 和 Honeytree 的路由表项的下一跳地址。 Honeypot 每一条路由的下一跳地址是 fec0::3:204:c1ff:fe50:f1c0，Honeytree 路由的下一跳地址是 fec0::1:204:c1ff:fe50:f1c0。 这两个地址分别是 Honeybee 到 Honeypot 和 Honeytree 的接口地址。注意，Honeybee 接口地址的后 64 位都是相同的；因为路由器使用第一个遇到的 MAC 地址来生成它的每个串行接口 EUI-64 格式的 IPv6 地址的后 64 位。

与 IPv4 一样，IPv6 静态路由也可以使用出站接口代替下一跳地址。一种选择是在接口后面输入地址，你可以输入链路本地地址，或者一个配置的地址。当出站接口是广播接口时，例如以太网，应该使用下一跳地址。

在示例 3-12 中，Honeypot 在 **ipv6 route** 语句中仅指明了下一跳地址，因此得到示例的 IPv6 路由表。使用命令 **show ipv6 route** 可以显示出 IPv6 的路由表，其中包括前缀、前缀长度、下一跳地址或出站接口、管理距离和路由度量。

[1] 接口上配置的地址是 EUI-64 格式的地址，因而对路由器来说，这些地址结合 MAC 地址是惟一的。

示例 3-12 与 IPv4 一样，IPv6 静态路由表包括目标网络和去往目标网络的下一跳地址

```
Honeypot#show ipv6 route
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

L   FE80::/10 [0/0]
     via ::, Null0
C   FEC0:0:0:3::/64 [0/0]
     via ::, Serial0/0.2
L   FEC0::3:230:94FF:FE24:B780/128 [0/0]
     via ::, Serial0/0.2
S   FEC0:0:0:A::/64 [1/0]
     via FEC0::3:204:C1FF:FE50:F1C0
S   FEC0:0:0:8::/64 [1/0]
     via FEC0::3:204:C1FF:FE50:F1C0
S   FEC0:0:0:1::/64 [1/0]
     via FEC0::3:204:C1FF:FE50:F1C0
C   FEC0:0:0:5::/64 [0/0]
     via ::, Ethernet0/0
L   FEC0::5:230:94FF:FE24:B780/128 [0/0]
     via ::, Ethernet0/0
L   FF00::/8 [0/0]
     via ::, Null0
```

在示例 3-12 中，所示静态路由是使用下一跳 IPv6 地址建立的。同 IPv4 一样，路由器必须递归查找与该 IPv6 地址相关联的出站接口。目标网络 FEC0:0:0:A::/64 表项的下一跳地址是 FEC0::3:204:C1FF:FE50:F1C0，如果再进一步观察路由表，可以看到网络 FEC0:0:0:3::/64 连接到接口 Serial0/0.2 上。注意，使用下一跳 IPv6 地址建立的静态路由的管理距离是 1，路由测度是 0，这和 IPv4 是相同的。

当然，我们还可以使用去往目标网络的出站接口或再结合下一跳地址来建立静态路由。如示例 3-13 所示，按照此方法对 Honeypot 的静态路由配置进行了修改。

示例 3-13 另一种配置 Honeypot 静态路由的方法

```
ipv6 route fec0::a:0:0:0:0/64 serial 0/0.2
ipv6 route fec0::8:0:0:0:0/64 serial 0/0.2
ipv6 route fec0::1:0:0:0:0/64 serial 0/0.2
ipv6 route fec0::20:0:0:0:0/62 Ethernet0/0 FE80::2B0:64FF:FE30:1DE0
```

其中最后一项使用了出站接口和下一跳地址，这样有助于阐明两种命令形式的区别。示例 3-14 显示了以此方法新生成的路由表。

示例 3-14 在 Honeypot 上，使用出站接口替代下一跳地址后生成的路由表

```
Honeypot#show ipv6 route static

S   FEC0:0:0:A::/64 [1/0]
     via ::, Serial0/0.2
S   FEC0:0:0:8::/64 [1/0]
     via ::, Serial0/0.2
S   FEC0:0:0:1::/64 [1/0]
     via ::, Serial0/0.2
S   FEC0:0:0:20::/62 [1/0]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet0/0
```

另一个需要注意的问题是，虽然使用出站接口后静态路由的管理距离仍然为 1，但是这

和使用相同方法配置的 IPv4 路由却不相同，因为路由显示目标网络不是直接连接的。

除非同时指定出站接口和下一跳地址，否则在输入出站接口时下一跳地址是不确定的。在示例 3-14 的路由表中，语句 1 说目标网络通过 ::可达，出站接口是 Serial0/0.2。“::”意味着下一跳没有被指定，但出站接口是 Serial0/0.2。在点到点的串行接口上，不指定下一跳地址不会出现问题，因为在点到点网络的另一端仅会有一台设备，所有从出站接口发出的数据包一定能到达那台设备。

而在广播网络接口上，路由器必须找到邻居才可以发送数据包。路由器会在以太网上组播邻居请求消息，并等待下一跳设备的邻居通告。不同于移动 IPv6 节点，这里没有代理地址解析机制。因而对于以太网上具有去往目标网络路由的路由器来说，它不会代表其他设备响应邻居请求消息。

因此，在广播网络上使用出站接口配置静态路由时，必须指定下一跳地址。这里建议采用本地链路地址作为下一跳地址。这样做有两个原因，一是除非网卡或路由器被替换，否则本地链路地址不会发生改变，即使使用不同的 IPv6 全局前缀重新编号站点也是这样。二是可以与路由器通告信息中的地址保持一致，使用这些地址的进程可以按要求运行。例如 ICMPv6 重定向进程。

路由器向广播网络上的所有设备通告自己的存在及本地链路地址。主机使用这些通告信息建立路由器列表，并使用这些列表确定如何向网络转发数据包。如果主机把一个数据包转发给路由器，并且该路由器知道一个更适合转发该数据包的路由器，那么路由器将向主机发送重定向信息。重定向信息包括另一台路由器的本地链路 IPv6 地址。当主机在处理重定向信息时，如果路由器列表中包含这台更合适的路由器，那么主机将向它转发数据包，否则（或列表中这台路由器使用了不同的 IPv6 地址），主机将丢弃数据包。

3.2.3 案例研究：汇总路由

汇总路由（Summary Route）是一个包含路由表中几个更加精确地址的地址。正是由于路由表项与地址掩码联合使用，使得静态路由的使用如此灵活。通过使用合适的子网掩码，有时可以为多个目标地址生成一条单一的汇总路由。

例如，在前面的案例研究中，为每个数据链路都使用了一条单独的路由。每个路由表项的掩码与连接数据链路的设备接口的地址掩码是一致的。再回想一下图 3-2，读者可以发现对于路由器 Piglet 来说，可以使用经路由器 Tigger 可达 10.4.0.0/16 的单一表项来完成对子网 10.4.6.0/24 和 10.4.7.0/24 的说明。同样的，也可以用指向 192.168.1.0/24 的单一表项替代路由表中的子网 192.168.1.0/27 和 192.168.1.64/27。10.4.0.0/16 和 192.16.1.0/24 两个路由表项就是汇总路由。

使用汇总路由技术，路由器 Piglet 的静态路由配置如示例 3-15 所示。

示例 3-15 Piglet 的静态路由被汇总为两条路由

```
ip route 192.168.1.0 255.255.255.0 192.168.1.193
ip route 10.4.0.0 255.255.0.0 192.168.1.193
```

因为从路由器 Pooh 看，10.0.0.0 网络的所有子网经过路由器 Tigger 都可以到达，所以仅需要包括主网地址和相应掩码的单一路由表项就够了（参见示例 3-16）。

示例 3-16 Pooh 把所有关于 10.0.0.0 子网的路由汇总为一条路由

```
ip route 192.168.1.192 255.255.255.224 192.168.1.66
ip route 10.0.0.0 255.0.0.0 192.168.1.66
```

对路由器 Eeyore，所有以 192 开头的目标地址都可以经过 Tigger 到达。如示例 3-17 所示，这个汇总路由中的地址掩码可以不指明所有的 C 类地址位。[1]

示例 3-17 Eeyore 把所有以 192 开头的路由都汇总为一条路由

```
ip route 192.0.0.0 255.0.0.0 10.4.6.1
ip route 10.1.0.0 255.255.0.0 10.4.6.1
```

汇总路由还可以应用到图 3-3 中所示的 IPv6 目标地址。通过把前缀长度由 64 改为 62，可以把 Honeypot 的两条静态路由汇总成一条路由，这条路由包括从 fec0:0:0:8::到 fec0:0:0:b::的一组目标地址（参见示例 3-18）。

示例 3-18 Honeypot 汇总 IPv6 静态路由

```
ipv6 route fec0::8:0:0:0:0/62 fec0::3:204:c1ff:fe50:f1c0
```

通过对一组子网甚至主网汇总，可以使静态路由项的数目迅速减小——在本示例中减少了三分之一。但是，在对地址进行汇总时需要小心，当汇总不正确时，可能会有意想不到的路由行为发生（见本章后面的 3.3.1 小节案例研究的内容）。第 7 章和第 8 章将会深入分析路由汇总及由此带来的问题。

3.2.4 案例研究：选择路由

如图 3-4 所示，在 Pooh 和 Eeyore 之间新增加了一条链路。除了去往主机 10.4.7.25 的数据包外，所有从 Pooh 到网络 10.0.0.0 的数据包都将使用这条新的路径。下面通过一条策略可以使这些数据包改经 Tigger 去往目的地。在 Pooh 上，相应的静态路由命令见示例 3-19。

示例 3-19 Pooh 上的静态路由命令可以实现一条策略，使流量经过指定的路由器

```
ip route 192.168.1.192 255.255.255.224 192.168.1.66
ip route 10.0.0.0 255.0.0.0 192.168.1.34
ip route 10.4.7.25 255.255.255.255 192.168.1.66
```

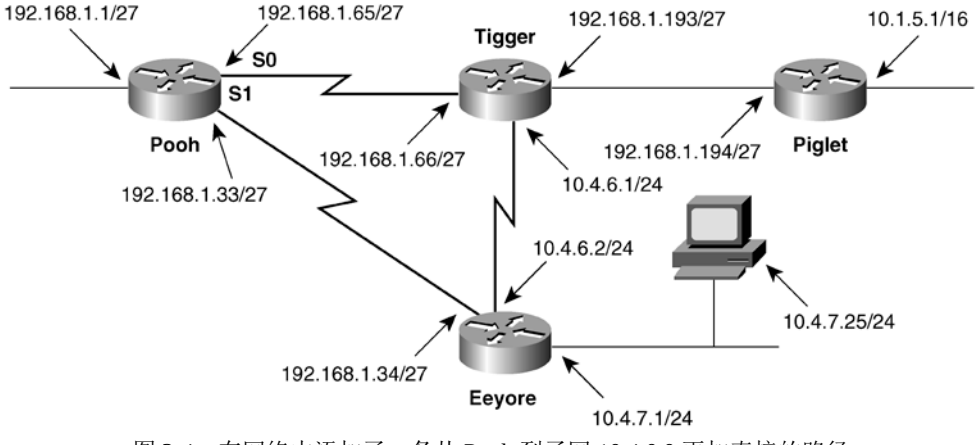


图 3-4 在网络中添加了一条从 Pooh 到子网 10.4.0.0 更加直接的路径

[1] 用小于特定类别地址标准掩码长度的掩码汇总一组主网地址的方法叫做超网化，详见第 6 章。

目前，除了第 2 条路由指向 Eyeore 上新的接口 192.168.1.34 外，前两个路由表项没有其他变化。第 3 条路由是一条指向单一主机 10.4.7.25 的主机路由，通过把地址掩码所有位都设置为 1 便可以实现。注意，不同于 10.0.0.0 子网的其他条目，这条主机路由指向 Tigger 的接口 192.168.1.66。

为了观察添加新路由条目之后数据包采用的路径，可以在路由器 Pooh 上打开调试功能 **debug ip packet**（参见示例 3-20）。这时从主机 192.168.1.15 有一个数据包被发向主机 10.4.7.25。前两条调试捕获信息显示了从接口 E0 进入路由器的数据包被路由到出站接口 S0 后再送往下一跳路由器 192.168.1.66（Tigger），按照要求，在接口 S0 接收到的回复数据包被路由到出站接口 E0 后再发往主机 192.168.1.15。

示例 3-20 调试信息证实了在 Pooh 上的新路由表项工作正常

```
Pooh#debug ip packet
IP packet debugging is on
Pooh#
IP: s=192.168.1.15 (Ethernet0), d=10.4.7.25 (Serial0), g=192.168.1.66, forward
IP: s=10.4.7.25 (Serial0), d=192.168.1.15 (Ethernet0), g=192.168.1.15, forward
Pooh#
IP: s=192.168.1.15 (Ethernet0), d=10.4.7.100 (Serial1), g=192.168.1.34, forward
IP: s=10.4.7.100 (Serial0), d=192.168.1.15 (Ethernet0), g=192.168.1.15, forward
Pooh#
```

下一个调试信息显示一个数据包从主机 192.168.1.15 发往主机 10.4.7.100。除了主机 10.4.7.25 外，去往 10.0.0.0 子网上所有主机的数据包都将沿新链路到达 Eeyore 的接口 192.168.1.34。第 3 个调试信息证实了这一点。然而，第 4 个调试信息最初看上去有点让人惊讶。从 10.4.7.100 去往 192.168.1.15 的响应数据包自 Tigger 到达 Pooh 的接口 S0。

还记得在其他路由器中的路由表项与原来例子中路由表项相比没有发生改变。不管是否期望有这样的结果，但是它体现出了静态路由的两个特性。

- 第一，如果网络拓扑结构发生变化，那么需要知道这些变化的路由器必须被重新配置。
- 第二，可以用静态路由建立非常精确的路由选择行为。在本例中，也许是来去流量使用的路径。

关于本例最后要说的一点是，由于使用了从 Pooh 到 Eeyore 再到 Tigger 的路径，而不是直接从 Pooh 到 Tigger 的路径，所以从 Pooh 到子网 10.1.5.0 的路由不是最优路径。示例 3-21 给出了一个更有效的配置。

示例 3-21 在路由器 Pooh 配置更加有效的静态路由

```
ip route 192.168.1.192 255.255.255.224 192.168.1.66
ip route 10.0.0.0 255.0.0.0 192.168.1.34
ip route 10.1.0.0 255.255.0.0 192.168.1.66
ip route 10.4.7.25 255.255.255.255 192.168.1.66
```

现在，第 3 条路由将把去往子网 10.1.5.0 的所有数据包直接发送到 Tigger。

3.2.5 案例研究：浮动静态路由（Floating Static Route）

浮动路由与其他静态路由不同，路由表中的其他路由总是优选于浮动静态路由，仅在一种特殊的情况下，即在一条首选路由发生失败的时候，浮动路由才会出现在路由表中。

在图 3-5 中，一台新的路由器（Rabbit）通过两条并行链路连接到 Piglet 上，一条链路连

接它们各自的接口 S0，另一跳连接各自的接口 S1。添加第二条链路主要是考虑到线路冗余：如果主链路 10.1.10.0 发生故障，浮动静态路由将会指引流量经过备份链路 10.1.20.0。

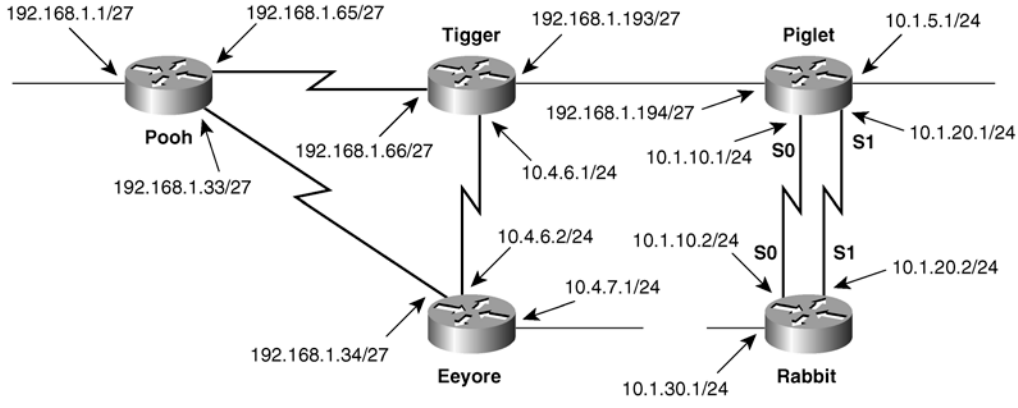


图 3-5 Piglet 上连接了一台新的路由器。这里使用了两条串行链路——一条作为主链路，另一条作为备份链路

此外，Piglet 以太网接口上的掩码由 10.1.5.1/16 改为 10.1.5.1/24。这一改变使得路由器 Tigger 上的一条路由不仅可以指向子网 10.1.5.0，而且还可以指向新路由器的所有新子网。

```
ip route 10.1.0.0 255.255.0.0 192.168.1.194
```

为了建立浮动静态路由，示例 3-22 和示例 3-23 分别给出了在 Piglet 和 Rabbit 上的路由表项。

示例 3-22 为 Piglet 建立浮动静态路由的路由表项

```
ip route 192.168.1.0 255.255.255.0 192.168.1.193
ip route 10.4.0.0 255.255.0.0 192.168.1.193
ip route 10.1.30.0 255.255.255.0 10.1.10.2
ip route 10.1.30.0 255.255.255.0 10.1.20.2 50
```

示例 3-23 为 Rabbit 建立浮动静态路由的路由表项

```
ip route 10.4.0.0 255.255.0.0 10.1.10.1
ip route 10.4.0.0 255.255.0.0 10.1.20.1 50
ip route 10.1.5.0 255.255.255.0 10.1.10.1
ip route 10.1.5.0 255.255.255.0 10.1.20.1 50
ip route 192.168.0.0 255.255.0.0 10.1.10.1
ip route 192.168.0.0 255.255.0.0 10.1.20.1 50
```

在 Piglet 上有两条路由指向 Rabbit 的网络 10.1.30.0；一条指定 Rabbit 接口 S0 的地址作为下一跳地址，另一条指定 Rabbit 接口 S1 的地址作为下一跳地址。对每个目标网络 Rabbit 都有类似的双重表项。

注意，在所有使用子网 10.1.20.0 的静态路由后面都跟有 50。该数字指定了管理距离，管理距离是一种优选度量。当存在两条路径到达相同的网络时，路由器将会选择管理距离较低的路径。这种思想初听起来有点像度量，但度量指明了路径的优先权，而管理距离指明了发现路由方式的优先权。

例如，指向下一跳地址的 IPv4 静态路由的管理距离为 1，而指向出站接口的静态路由的管理距离为 0。如果有两条静态路由指向相同的目标网络，一条指向下一跳地址，一条指向

出站接口，那么后一条路由——管理距离值较低的路由——被选择。

将经过子网 10.1.20.0 的静态路由的管理距离提高到 50，可以使经过子网 10.1.10.0 的静态路由成为首选路由。示例 3-24 反复 3 次给出了 Rabbit 的路由表。在第 1 个路由表中，所有指向非直连网络的路由都使用下一跳地址 10.1.10.1。在每条路由表项中，括号内的数字指定了管理距离为 1，度量值为 0（因为静态路由没有度量）。

示例 3-24 当主链路 10.1.10.0 失败时，备份链路 10.1.20.0 被启用。当主链路恢复时，再次启用主链路

```
Rabbit#show ip route
10.0.0.0 is variably subnetted, 5 subnets, 2 masks
C 10.1.10.0 255.255.255.0 is directly connected, Serial0
S 10.4.0.0 255.255.0.0 [1/0] via 10.1.10.1
S 10.1.5.0 255.255.255.0 [1/0] via 10.1.10.1
C 10.1.30.0 255.255.255.0 is directly connected, Ethernet0
C 10.1.20.0 255.255.255.0 is directly connected, Serial1
S 192.168.0.0 255.255.0.0 [1/0] via 10.1.10.1
Rabbit#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to down
%LINK-3-UPDOWN: Interface Serial0, changed state to down
Rabbit#show ip route
10.0.0.0 is variably subnetted, 4 subnets, 2 masks
S 10.4.0.0 255.255.0.0 [50/0] via 10.1.20.0
S 10.1.5.0 255.255.255.0 [50/0] via 10.1.20.1
C 10.1.30.0 255.255.255.0 is directly connected, Ethernet0
C 10.1.20.0 255.255.255.0 is directly connected, Serial1
S 192.168.0.0 255.255.0.0 [50/0] via 10.1.20.1
Rabbit#
%LINK-3-UPDOWN: Interface Serial0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to up
Rabbit#show ip route
10.0.0.0 is variably subnetted, 5 subnets, 2 masks
C 10.1.10.0 255.255.255.0 is directly connected, Serial0
S 10.4.0.0 255.255.0.0 [1/0] via 10.1.10.1
S 10.1.5.0 255.255.255.0 [1/0] via 10.1.10.1
C 10.1.30.0 255.255.255.0 is directly connected, Ethernet0
C 10.1.20.0 255.255.255.0 is directly connected, Serial1
S 192.168.0.0 255.255.0.0 [1/0] via 10.1.10.1
Rabbit#
```

接着，陷阱消息（trap message）通知连接到接口 S0 的主链路状态变为“down”，表明链路发生故障。查看第 2 次重复给出的路由表迭代，可以发现所有非直连网络路由都指向下一跳地址 10.1.20.1。由于原来的首选路由不再可用，所以路由器切换到管理距离为 50 的备份链路。而且因为子网 10.1.10.0 发生故障，所以路由表中不再把它作为直连网络。

在第 3 次给出路由表之前，陷阱消息提示主链路状态恢复为“链路正常（up）”，路由表中再次显示子网 10.1.10.0，而且路由器也再次使用 10.1.10.1 作为下一跳地址。

第 11 章将结合多种动态路由选择协议讨论管理距离，可以说动态路由选择协议的管理距离远远大于 1。因此对于相同的目标网络，缺省情况下，到相同目标网络的静态路由总是优于动态路由。

3.2.6 案例研究：IPv6 浮动静态路由

IPv6 浮动静态路由的工作方式和 IPv4 一样。在图 3-3 中，如果 Honeypot 和 Honeybee 之间的主链路发生故障，那么第二条链路将被添加到 IPv6 网络中（参见图 3-6）。

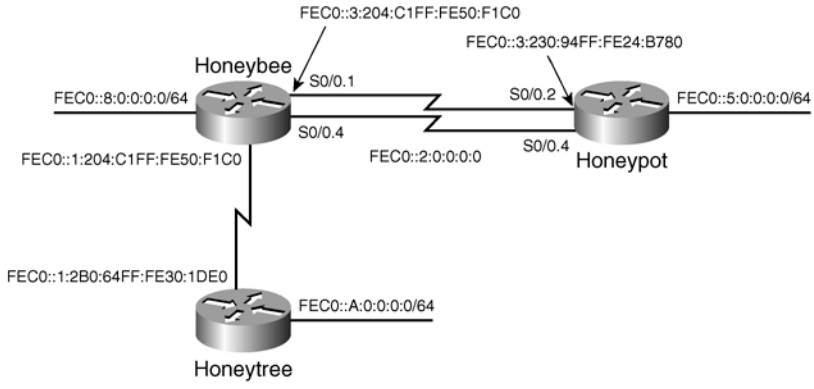


图 3-6 使用浮动静态路由，路由器之间的备份链路可以在主链路中断时恢复网络畅通

如示例 3-25 所示，新的静态路由表项被添加到 Honeypot 的配置中，其中路由的管理距离大于 1。在示例 3-26 中，类似的静态路由也被添加到 Honeybee 上。

示例 3-25 为 Honeypot 和 Honeybee 之间新的冗余并行链路配置浮动静态路由

```
ipv6 route FEC0::/62 FEC0::3:204:C1FF:FE50:F1C0
ipv6 route FEC0::/62 FEC0::2:204:C1FF:FE50:F1C0 50
ipv6 route FEC0:0:0:8::/62 FEC0::3:204:C1FF:FE50:F1C0
ipv6 route FEC0:0:0:8::/62 FEC0::2:204:C1FF:FE50:F1C0 50
```

示例 3-26 为 Honeybee 和 Honeypot 之间新的冗余并行链路配置浮动静态路由

```
ipv6 route FEC0:0:0:5::/64 FEC0::3:230:94FF:FE24:B780
ipv6 route FEC0:0:0:5::/64 FEC0::2:230:94FF:FE24:B780 50
ipv6 route FEC0:0:0:A::/64 FEC0::1:2B0:64FF:FE30:1DE0
```

除非链路发生故障，否则 IPv6 流量将从 Honeypot 的 Serial0/0.2 接口出站。

示例 3-27 给出了通过子网 fec0::3:0:0:0:0/64 安装到 Honeypot 路由表中的两条路由，这两条路由的管理距离都为 1。当接口 S0/0.2 发生故障时，管理距离为 50 的备份路由替代了原来的路由；如果接口 S0/0.2 恢复正常后，路由选择进程将会学习去往目标网络的更优路由，并重新将其安装到路由表中。

示例 3-27 仅当管理距离小的路由被删除时，管理距离大的静态路由才会被添加到 IPv6 的路由表中

```
Honeypot#show ipv6 route static
S   FEC0::/62 [1/0]
     via FEC0::3:204:C1FF:FE50:F1C0
S   FEC0:0:0:8::/62 [1/0]
     via FEC0::3:204:C1FF:FE50:F1C0
Honeypot#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/2, changed state to down
%LINK-3-UPDOWN: Interface Serial0/2, changed state to down

Honeypot#show ipv6 route static
S   FEC0::/62 [50/0]
     via FEC0::2:204:C1FF:FE50:F1C0
S   FEC0:0:0:8::/62 [50/0]
     via FEC0::2:204:C1FF:FE50:F1C0
Honeypot#
%LINK-3-UPDOWN: Interface Serial0/2, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/2, changed state to up
```

不管是使用出站接口还是下一跳地址，IPv6 静态路由管理距离的缺省值都为 1。使用这两种方式指定的静态路由是等价的，并且会均分负载（在 3.2.7 小节中学习）。

3.2.7 案例研究：均分负载

上一小节所用配置方法的弊病是在正常情况下备份链路不能被利用；备份链路的可用带宽资源被浪费了。均分负载（Load Sharing），又叫负载均衡，允许路由器利用多路径的优点，在所有可用的路径上发送数据包。

均分负载可以是等价或非等价的，这里的代价（cost）是一个通用术语，指与路由相关联的度量。

- **等价均分负载（Equal-Cost Load Sharing）**——将流量均匀地分布到多条度量相同的路径上。在这种情况下，均分负载又叫负载平衡。
- **非等价均分负载（Unequal-Cost Load Sharing）**——将数据包分布到度量不同的多条路径上。各条路径上分布的流量与路由代价成反比。也就是说，代价越低的路径分配的流量越多，代价越高的路径分配的流量越少。

一些路由选择协议可以支持等价和非等价负载均衡两种方式，而其他一些路由选择协议仅支持等价方式。静态路由没有度量，所以仅支持等价负载均衡。

为了使用静态路由实现负载均衡，在图 3-5 中配置了并行链路，示例 3-28 和示例 3-29 分别给出了 Piglet 和 Rabbit 相应的路由表项。

示例 3-28 为实现均分负载使用静态路由配置并行链路： Piglet 上相应的路由表项

```
ip route 192.168.1.0 255.255.255.0 192.168.1.193
ip route 10.4.0.0 255.255.0.0 192.168.1.193
ip route 10.1.30.0 255.255.255.0 10.1.10.2
ip route 10.1.30.0 255.255.255.0 10.1.20.2
```

示例 3-29 为实现均分负载使用静态路由配置并行链路： Rabbit 上相应的路由表项

```
ip route 10.4.0.0 255.255.0.0 10.1.10.1
ip route 10.4.0.0 255.255.0.0 10.1.20.1
ip route 10.1.5.0 255.255.255.0 10.1.10.1
ip route 10.1.5.0 255.255.255.0 10.1.20.1
ip route 192.168.0.0 255.255.0.0 10.1.10.1
ip route 192.168.0.0 255.255.0.0 10.1.20.1
```

除了两条链路使用缺省管理距离 1 以外，这些路由表项已在上一节的浮动静态路由中被用过了。如示例 3-30 所示，在 Rabbit 的路由表中，对于每个目标网络都存在两条路由。

示例 3-30 这个路由表显示出对于每个目标网络，都存在两条路径。路由器将会在多条路径上平衡负载

```
Rabbit#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
      U - per-user static route
Gateway of last resort is not set
     10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
```

（待续）

```
C       10.1.10.0/24 is directly connected, Serial0
S       10.1.5.0/24 [1/0] via 10.1.10.1
                    [1/0] via 10.1.20.1
S       10.4.0.0/16 [1/0] via 10.1.10.1
                    [1/0] via 10.1.20.1
C       10.1.20.0/24 is directly connected, Serial1
S       192.168.0.0/16 [1/0] via 10.1.10.1
                       [1/0] via 10.1.20.1
Rabbit#
```

IPv6 和 IPv4 工作方式一样。在示例 3-31 中，Honeypot 的路由配置将会产生示例 3-32 所示的路由表。

示例 3-31 为了使 Honeypot 的 IPv6 静态路由可以实现均分负载，对每个地址前缀使用不同的下一跳地址

```
ipv6 route FEC0::/62 FEC0::2:204:C1FF:FE50:F1C0
ipv6 route FEC0::/62 FEC0::3:204:C1FF:FE50:F1C0
ipv6 route FEC0:0:0:8::/62 FEC0::2:204:C1FF:FE50:F1C0
ipv6 route FEC0:0:0:8::/62 FEC0::3:204:C1FF:FE50:F1C0
```

示例 3-32 路由器在去往相同目标网络的两条路径上均分负载

```
Honeypot#show ipv6 route static
S   FEC0::/62 [1/0]
     via FEC0::3:204:C1FF:FE50:F1C0
     via FEC0::2:204:C1FF:FE50:F1C0
S   FEC0:0:0:8::/62 [1/0]
     via FEC0::3:204:C1FF:FE50:F1C0
     via FEC0::2:204:C1FF:FE50:F1C0
```

负载均衡有两种方式：基于目标网络和基于数据包。

1. 负载均衡和 Cisco 急速转发

基于目标网络的负载均衡是根据目标地址分配负载。假设到一个网络存在两条路径，那么发往该网络中第一个目标的数据包从第一条路径通过，发往网络中第二个目标的数据包通过第二条路径，发往此网络中第三个目标的所有数据包还通过第一条路径，依此类推。这就是 Cisco 急速转发（CEF）使用的缺省负载均衡方式。在大部分平台上，IPv4 的缺省交换模式是 CEF，但是 IPv6 却不是。

CEF 是一种非常有效的交换方法。它事先从路由表中获取信息并把信息存储在转发信息库（FIB）中，当任何数据包需要这些信息时可以立即使用。FIB 包括路由表中的所有目标网络，如果路由表稳定且不发生改变，那么 FIB 也不会变化。CEF 使用一个单独的数据表－邻接关系表，为 FIB 的每个表项维护第二层转发信息。邻接关系表由第二层信息构成，例如，这些信息可以通过 IP ARP 或 IPv6 邻居发现协议学习到。FIB 和邻接关系表是在数据包转发之前建立的。

CEF 在缺省情况下是基于目标进行负载均衡。这实际上是按照源目地址对进行负载均衡。所有发往特定目标地址的流量只要源地址相同都会从相同的接口出站，而不同源目地址对的流量可能会从下一个接口出站。

基于数据包的负载均衡是交换 IPv4 数据包的另一种方式。对于 IPv6，CEF 仅支持基于目标网络的负载均衡方式。基于数据包的负载均衡方式意味着在不同的链路上发送数据包，即使在路径等代价、目标相同的情况下也是这样。如果路径代价不同，那么可能会在高、低代价路径上按照代价比率进行分流。基于数据包的负载均衡方式可以更加均匀地布流量，这

取决于不同源目地址对的数量。但是数据包选择不同的路径去往目标网络会引起非顺序到达。对于某些应用来说，这是不能接受的，例如 VoIP。

为了确定 CEF 功能是否在路由器上被全局的打开，可以使用命令 **show ip cef** 和 **show ipv6 cef**。如果缺省情况下 CEF 没有被打开，针对 IPv4 可以使用命令 **ip cef**，而对 IPv6 来说，必须先打开 IPv4 的 CEF，然后使用命令 **ipv6 cef** 打开此功能。

在 IPv4 下，命令 **ip load-sharing per-packet** 可以打开基于数据包的负载均衡功能，如果需要打开基于目标地址的负载均衡，可以用 **ip load-sharing per-destination** 命令。你可以使用命令 **show cef interface** 来检查使用了哪一种负载均衡模式，该命令可以给出在这个接口上配置的 CEF 信息。

路由器通常根据入站接口和源与目的地址类型确定是否使用 CEF 交换。对于考虑使用 CEF 的路由器来说，出站接口必须配置为 CEF 交换模式，如果接口上配置了 CEF，那么 CEF 将尝试交换数据包。否则，CEF 将会把数据包交付给仅次于最好的可用交换方法去处理。对于 IPv4，这种方法是快速交换，而在 IPv6 中叫进程交换（process switching）。

你可以使用命令 **show cef interface**{*interface*} 和 **show pv6 cef**{*interface*} **detail** 来验证在接口上 CEF 功能是否被打开。

2. 基于目标网络的负载均衡和快速交换

IOS 在配置了快速交换的出站接口上执行基于目标网络的负载均衡，在一些路由器上，IOS 的缺省交换模式是快速交换。

快速交换的工作方式如下：

（1）当路由器为第一个去往特定目标的数据包进行交换处理时，路由器将执行路由表查询并选择出站接口。

（2）然后获取有关被选接口的数据链路信息（例如从 ARP 表），这些信息对数据包成帧是必需的，最后封装数据包并发送。

（3）前面获取的路由和数据链路信息被输入到快速交换的高速缓冲内。

（4）一旦去往相同目的地的后继数据包进入路由器，高速缓冲中的信息使路由器不必查找路由表和 ARP 高速缓冲，就可以立即交换数据包。

快速交换意味着所有去往指定目的地的数据包都从相同的接口被发送出去，因此交换时间和处理器的占用率会大大降低。当去往相同网络内不同主机的数据包进入路由器且还存在一条可选路由时，路由器会在另一条路径上发送数据包到目的地。因此路由器能够做得最好的就是基于目标网络的均衡负载。

3. 基于数据包的负载均衡和过程交换

过程交换（process switching）就是对于每个数据包，路由器都要进行路由表查询和接口选择，然后再查询数据链路信息。因为每一次为数据包确定路由的过程都是相互独立的，所以不会强制去往相同目标网络的所有数据包使用相同的接口。为了在接口上打开过程交换功能，可以在 IPv4 下使用命令 **no ip route-cache**。对于 IPv6 什么也不需要做，因为缺省情况下该功能是打开的。

在示例 3-33 中，主机 192.168.1.15 向主机 10.1.30.25 发送了 6 个 ping。在 Piglet 上使用 **debug ip packet** 可以观察到 ICMP 的回应请求和回应应答数据包。通过查看出站接口和转发地址可以发现 Piglet 和 Rabbit 都在交替使用接口 S0 和 S1。注意命令 **debug ip packet** 仅允许

观察过程交换的数据包，快速交换的数据包将不被显示出来。

注意：命令 **debug ip packet** 仅显示过程交换的数据包。

示例 3-33 路由器交替使用接口 S0 和 S1 向相同目标网络发送数据包。注意，在两条链路另一端的路由器也以同样的方式回复数据包

```
Piglet#debug ip packet
IP packet debugging is on
Piglet#
IP: s=192.168.1.15 (Ethernet0), d=10.1.30.25 (Serial0), g=10.1.10.2, forward
IP: s=10.1.30.25 (Serial0), d=192.168.1.15 (Ethernet0), g=192.168.1.193, forward
IP: s=192.168.1.15 (Ethernet0), d=10.1.30.25 (Serial1), g=10.1.20.2, forward
IP: s=10.1.30.25 (Serial1), d=192.168.1.15 (Ethernet0), g=192.168.1.193, forward
IP: s=192.168.1.15 (Ethernet0), d=10.1.30.25 (Serial0), g=10.1.10.2, forward
IP: s=10.1.30.25 (Serial0), d=192.168.1.15 (Ethernet0), g=192.168.1.193, forward
IP: s=192.168.1.15 (Ethernet0), d=10.1.30.25 (Serial1), g=10.1.20.2, forward
IP: s=10.1.30.25 (Serial1), d=192.168.1.15 (Ethernet0), g=192.168.1.193, forward
IP: s=192.168.1.15 (Ethernet0), d=10.1.30.25 (Serial0), g=10.1.10.2, forward
IP: s=10.1.30.25 (Serial0), d=192.168.1.15 (Ethernet0), g=192.168.1.193, forward
IP: s=192.168.1.15 (Ethernet0), d=10.1.30.25 (Serial1), g=10.1.20.2, forward
IP: s=10.1.30.25 (Serial1), d=192.168.1.15 (Ethernet0), g=192.168.1.193, forward
Piglet#
```

正如许多设计选择一样，基于数据包的均分负载也是要付出代价的。这种方式虽然使流量的分布比前一种方式更均匀，但是快速交换的较低交换时间和处理器占用的优点也随之丧失了。

4. 哪一种交换方法会被用到

IOS 首先基于入站接口的配置来决定交换模式；如果接口上配置了 CEF，不管出站接口的配置是什么，数据包都会被 CEF 交换。

如果入站接口上没有配置 CEF，那么 IOS 会处理并转发数据包，并根据出站接口的配置，后继的数据包或者被快速交换，或者被过程交换。表 3-1 给出了交换方法与出/入站接口配置的对应关系表。

表 3-1 IOS 根据入站和出站接口的配置确定交换方法

入站配置	出站配置	所用的交换方法
CEF	过程	CEF
CEF	快速	CEF
过程	CEF	快速（如果是 IPv6 为过程）
过程	快速	快速
快速	CEF	快速（如果是 IPv6 为过程）
快速	过程	过程

如果入站接口的 CEF 功能被打开，IOS 将只使用 CEF 交换数据包。否则，出站接口的配置会确定交换方法。注意，如果在出站接口上打开 CEF 功能的同时又在入站接口上配置了过程交换或快速交换，那么快速交换将被使用。只有在入站接口上配置了 CEF，它才会起作用。对于 IPv4，尽管在出站接口上打开了 CEF 功能，但是起作用的还是快速交换。

有些时候即使打开了 CEF，但是并没有使用 CEF 交换数据包（例如，如果访问列表的日

志功能被打开和数据包将被记录下来）。那么数据包将被送交仅次于最快的交换方法，例如在 IPv4 下使用快速交换，在 IPv6 下使用过程交换。

3.2.8 案例研究：递归表查询

所有路由表项不必一定指向下一跳路由器。图 3-7 是图 3-5 中网络的简化版。在这个网络中，Pooh 配置见示例 3-34。

示例 3-34 Pooh 的路由表项使用了多种地址作为下一跳参数。这些地址不需要是下一跳路由器接口的实际地址

```
ip route 10.1.30.0 255.255.255.0 10.1.10.2
ip route 10.1.10.0 255.255.255.0 192.168.1.194
ip route 192.168.1.192 255.255.255.224 192.168.1.66
```

如果 Pooh 需要向主机 10.1.30.25 发送数据包，Pooh 将查找路由表并发现经过 10.1.10.2 可以到达这个子网。因为这个地址不在直连网络中，所以 Pooh 必须再次查找路由表并发现去往 10.1.10.0 需要途经 192.168.1.194。由于这个子网也不是直连子网，因而需要进行第 3 次路由表查找。这次 Pooh 发现途经 192.168.1.66 可以到达 192.168.1.192，并且 192.168.1.66 在一个直连子网中。于是现在可以转发数据包了。

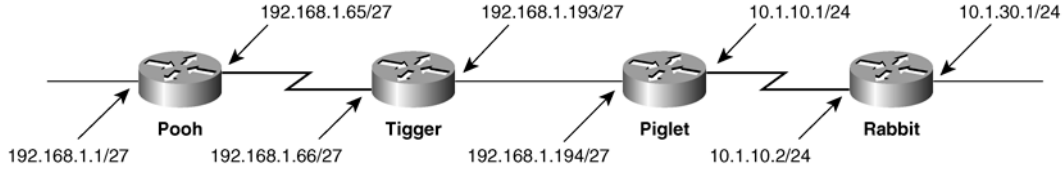


图 3-7 为了到达网络 10.1.30.0，Pooh 必须进行 3 次路由表查找

因为每次路由表查询都会花费处理器的时间，所以在正常情况下强制路由器进行多次路由表查询是一种不好的设计决策。快速交换对递归查询进行了限制，仅对去往每个目标网络的第 1 个进行递归查询，从而有效地降低了这些不利的影响；但是在使用这种设计方法之前仍然需要充分的理由。

图 3-8 给出了一个递归路由表查询的实例，例子中的递归查询也许是有用的。在这里，Sanderz 途经 Heffalump 可以到达所有网络。然而，网络管理员计划弃用 Heffalump，改用 Woozle。Sanderz 中的前 12 条路由不再指向 Heffalump，而是指向被连接到子网 10.87.14.0 上的合适路由器。最后一条路由指明经 Heffalump 可以到达子网 10.87.14.0。

使用下面的配置，仅需要改动最后一条静态路由，便可以使 Sanderz 的所有路由都重新指向 Woozel，详见示例 3-35。

示例 3-35 仅需改动最后一条静态路由，便可以使 Sanderz 的所有路由都重新指向 Woozel

```
Sanderz(config)# ip route 10.87.14.0 255.255.255.0 10.23.5.95
Sanderz(config)# no ip route 10.87.14.0 255.255.255.0 10.23.5.20
```

如果以 10.23.5.20 作为所有静态路由条目的下一跳地址，那么需要删除 13 条路由，再重

新输入 13 条新的路由。不过，读者要仔细斟酌一下，省去重新输入静态路由的麻烦与递归查询带给路由器的额外负担到底谁更重要。

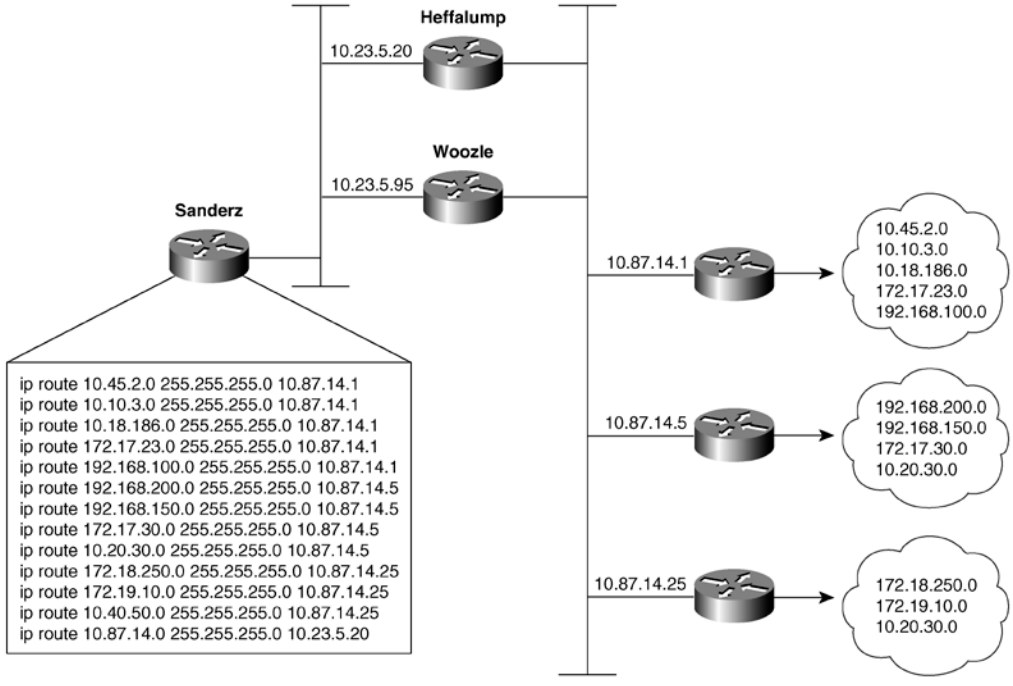


图 3-8 为 Sanderz 配置递归查询，使网络管理员仅需要修改一条路由，便可把从 Sanderz 到 Heffaoump 的流量重定向到 Woozle

3.3 静态路由故障诊断

也许你听过这样的问题，“他知道什么？他什么时候知道的？”这些问题对于网络调查员也同样适用。当排除路由故障时，首要的一步就是检查路由表。路由器知道什么？这些信息在路由表中存在多久了？路由器知道怎样如何到达讨论中的目标网络？路由表中的信息准确吗？为了顺利地排除网络的故障，了解如何跟踪路由是十分必要的。

3.3.1 案例研究：追踪故障路由

图 3-9 所示的网络在前面已经作过相应的配置，其中包括每台路由器相应的静态路由。现在发现一个问题，在连接到 Pooh 以太网接口的子网 192.168.1.0/27 上，设备与子网 10.1.0.0/16 上的设备可以正常地通信。然而，当从 Pooh 向子网 10.1.0.0/16 发送 ping 时，结果出现 ping 失败（参见示例 3-36）。这看上去很奇怪。如果 Pooh 可以成功地路由数据包到达目标网络，那么为什么从 Pooh 发的数据包却传送失败呢？

为了解决这个问题，需要跟踪 ping 所经过的路径。首先，检查 Pooh 的路由表（参见示例 3-37）。（根据路由表）目标地址 10.1.5.1 可以匹配到路由表项 10.0.0.0/8，该子网经下一跳地址是 192.168.1.34，192.168.1.34 是 Eeyore 的一个接口。

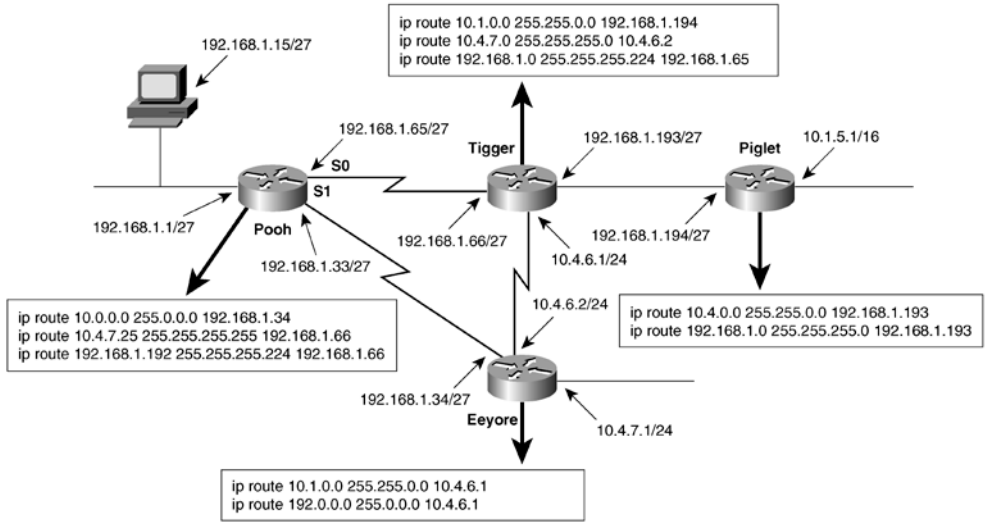


图 3-9 从子网 192.168.1.0/27 到子网 10.1.0.0/16 数据包可以被正确地路由，但从 Pooh 却不能 ping 通子网 10.1.0.0/16 中的任何设备

示例 3-36 子网 192.168.1.0/27 上的设备可以 ping 通 Piglet 的以太网接口，但 Pooh 却 Ping 不通

```
C:\WINDOWS>ping 10.1.5.1
Pinging 10.1.5.1 with 32 bytes of data:
Reply from 10.1.5.1: bytes=32 time=22ms TTL=253
Reply from 10.1.5.1: bytes=32 time=22ms TTL=253
Reply from 10.1.5.1: bytes=32 time=22ms TTL=253
Reply from 10.1.5.1: bytes=32 time=22ms TTL=253

Pooh#ping 10.1.5.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echoechoes to 10.1.5.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
Pooh#
```

示例 3-37 去往目标地址 10.1.5.1 的数据包在匹配到路由表项 10.0.0.0/8 之后被转发到下一跳路由器 192.168.1.34

```
Pooh#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
     10.0.0.0 is variably subnetted, 2 subnets, 2 masks
S       10.0.0.0 255.0.0.0 [1/0] via 192.168.1.34
S       10.4.7.25 255.255.255.255 [1/0] via 192.168.1.66
     192.168.1.0 255.255.255.224 is subnetted, 4 subnets
C       192.168.1.64 is directly connected, Serial0
C       192.168.1.32 is directly connected, Serial1
C       192.168.1.0 is directly connected, Ethernet0
S       192.168.1.192 [1/0] via 192.168.1.66
Pooh#
```

然后，需要检查 Eeyore 的路由表（参见示例 3-38）。目标地址 10.1.5.1 可以匹配到路由

表项 10.1.0.0/16，该表项的下一跳地址为 10.4.6.1，它是 Tigger 的一个接口地址。

示例 3-38 10.1.5.1 在匹配到路由表项 10.1.0.0/16 之后被转发到 10.4.6.1

```
Eeyore#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
     10.0.0.0 is variably subnetted, 3 subnets, 2 masks
C       10.4.6.0 255.255.255.0 is directly connected, Serial1
C       10.4.7.0 255.255.255.0 is directly connected, Ethernet0
S       10.1.0.0 255.255.0.0 [1/0] via 10.4.6.1
     192.168.1.0 255.255.255.224 is subnetted, 1 subnets
C       192.168.1.32 is directly connected, Serial0
S       192.0.0.0 255.0.0.0 [1/0] via 10.4.6.1
Eeyore#
```

示例 3-39 给出了 Tigger 的路由表。目标地址在匹配到路由表项 10.1.0.0/16 之后将被转发给 Piglet（192.168.1.194）。

示例 3-39 10.1.5.1 在匹配到路由表项 10.1.0.0/16 后将被转发到 192.168.1.194

```
Tigger#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
     10.0.0.0 is variably subnetted, 3 subnets, 2 masks
C       10.4.6.0 255.255.255.0 is directly connected, Serial1
S       10.4.7.0 255.255.255.0 [1/0] via 10.4.6.2
S       10.1.0.0 255.255.0.0 [1/0] via 192.168.1.194
     192.168.1.0 255.255.255.224 is subnetted, 3 subnets
C       192.168.1.64 is directly connected, Serial0
S       192.168.1.0 [1/0] via 192.168.1.65
C       192.168.1.192 is directly connected, Ethernet0
Tigger#
```

Piglet 的路由表（参见示例 3-40）显示出目标网络 10.1.0.0 是一个直连网络。换言之，数据包已经到达目的地了，因为目标地址 10.1.5.1 就是 Piglet 自己的接口地址。因为去往目标地址的路径经过验证是正确的，所以我们可以认为来自 Pooh 的 ICMP 回应数据包可以到达目标网络。

下一步是跟踪 ICMP 回应应答数据包所经过的路径。为了跟踪这一路径，读者需要知道回应数据包的源地址——这个地址将是回应应答数据包的目标地址。从路由器发出数据包的源地址就是发送数据包的接口地址。[1] 在本例中，最初由 Pooh 向 192.168.1.34 转发回应数据包。在图 3-9 中，显示出数据包的源地址为 192.168.1.33。所以 Piglet 将要发送的回应应答数据包的目标地址就是 192.168.1.33。

参考示例 3-34 中 Piglet 的路由表，可以发现 192.168.1.33 将会匹配到路由表项 192.168.1.0/24 并被转发到 192.168.1.193，它是 Tigger 的另一个接口。重新检查示例 3-39 中 Tigger 的路由表，首先使我们想起还有一条关于 192.168.1.0 的路由。可是，如果根据这些信息准确地解释那里发生的实际情况，那么需要十分小心。

[1] 除非用扩展 ping 工具将源地址设置为其他地址。

示例 3-40 目标网络 10.1.0.0 直接连接在 Piglet 上

```
Piglet#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
     10.0.0.0 255.255.0.0 is subnetted, 2 subnets
C       10.1.0.0 is directly connected, Ethernet1
S       10.4.0.0 [1/0] via 192.168.1.193
     192.168.1.0 is variably subnetted, 2 subnets, 2 masks
S       192.168.1.0 255.255.255.0 [1/0] via 192.168.1.193
C       192.168.1.192 255.255.255.224 is directly connected, Ethernet0
Piglet#
```

比较一下 Tigger 路由表中有关 10.0.0.0 子网和 192.168.1.0 子网的路由表项。10.0.0.0 的标题表明其子网大小各不相同；换句话说，指向子网 10.4.7.0 Tigger 的静态路由使用了 24 位掩码，指向子网 10.1.0.0 的静态路由使用了 16 位掩码。该路由表为每个子网记录了正确的掩码。

192.168.1.0 的标题则不同，它表明 Tigger 知道 192.168.1.0 有 3 个子网，且掩码都为 255.255.255.224。用这个掩码可以确定目标地址 192.168.1.33 所属的目标网络为 192.168.1.32/27。但是路由表中只有关于 192.168.1.64/27、192.168.1.0/27 和 192.168.1.192/27 的路由表项，而没有关于 192.168.1.32/27 的路由表项，因此路由器不知道该如何到达这个子网。

这样问题就很清楚了，ICMP 的回应应答数据包是在 Tigger 处被丢弃的。一种解决办法是，另外创建一条指向网络 192.168.1.32 的静态路由，掩码为 255.255.255.224，指向下一跳的地址为 192.168.1.65 或 10.4.6.2。还有一种解决办法是，将关于 192.168.1.0 的路由表项的掩码由 255.255.255.224 改为 255.255.255.0。

此案例的精髓就是当你跟踪路由时，你必须考虑完整的通信过程。

注意：不仅要验证去往目标网络的路由是正确的，还要验证返回的路径也是正确的。

3.3.2 案例研究：协议冲突

如图 3-10 所示，两台路由器被两个以太网连接起来，其中一个以太网包括一个简单的网桥。该网桥同时还负责处理其他几条没有画出的链路的流量，所以有时网桥会变得十分拥挤。主机 Milne 是一台承担重要任务的服务器，网络管理员担心网桥会延误 Milne 的流量，所以在 Roo 上添加了一条指向 Milne 的主机路由，该路由指引数据包使用图上方的以太网以避开该网桥。

这种解决方案看上去是合理的，但是实际并非如此。在添加上面那条静态路由后，数据包经 Roo 不但不能被路由到服务器，而且经 Kanga 路由的数据包也不能到达服务器，尽管没有对 Kanga 作过改动。

第一步首先检查路由表。Roo 的路由表（参见示例 3-41）指明目标地址为 172.16.20.75 的数据包实际上被转发到 Kanga 的接口 E1 上，这正是我们所期望的。由于目标网络直接连接到 Kanga 上，所以不再需要进一步的路由。经过快速检查确定在 Kanga 和 Milne 上的两个

以太网接口都工作正常。

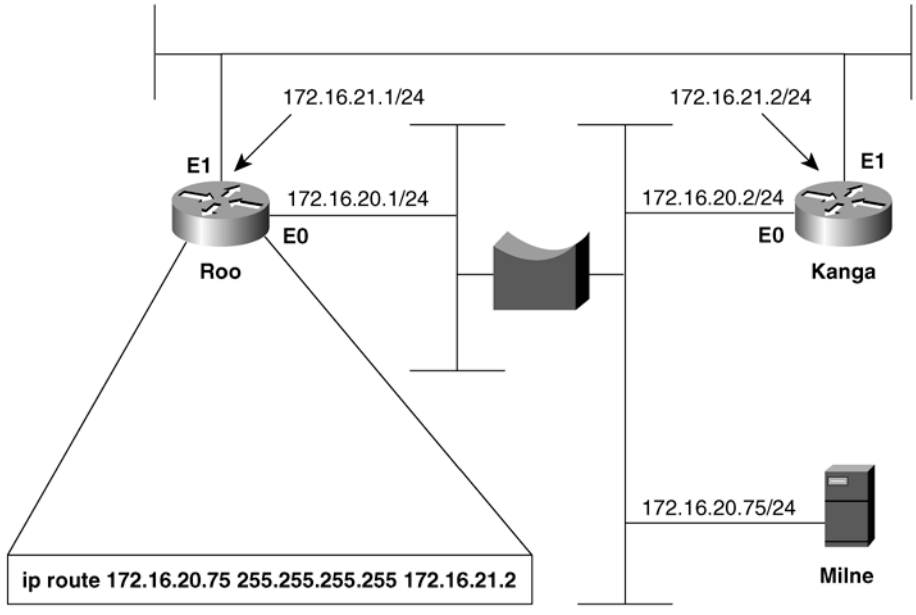


图 3-10 主机路由指引从 Roo 到 Milne 的数据包经过上面的以太网，这样可以避开偶尔发生拥塞的网桥

示例 3-41 在 Roo 的路由表中显示有一条经 Kanga 接口 E1 指向 Milne 的静态主机路由

```
Roo#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route
Gateway of last resort is not set
     172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks
C       172.16.20.0/24 is directly connected, Ethernet0
C       172.16.21.0/24 is directly connected, Ethernet1
S       172.16.20.75/32 [1/0] via 172.16.21.2
Roo#
```

在示例 3-42 中，从 Roo 向 Milne 执行跟踪命令，发现以下症状。Kanga 应该发向 Milne 的数据包被发向 Roo 的接口 E0。Roo 又将数据包发给 Kanga 接口 E1，接着 Kanga 再次将数据包发回给 Roo。这看上去像是发生了路由选择环路，但是为什么呢？

此故障的可疑之处在于 Kanga 不应该对数据包进行路由选择，但事实恰恰相反。

Kanga 应该能够识别出数据包的目标地址属于它的直连网络 172.16.20.0，然后使用数据链路向主机传送数据包。因此疑点发生在数据链路上。路由器是否具有经过某条逻辑路径到达目标网络的正确信息，这一点我们可以通过查看路由表来获知。同样的，我们应该检查 ARP 高速缓冲区，来确定路由器是否具有经过某条物理路径到达某台主机的正确信息。

示例 3-42 从 Roo 到 Milne 进行跟踪，发现 Kanga 将本应发往正确目的地的数据包回发给了 Roo

```
Roo#trace 172.16.20.75
Type escape sequence to abort.
Tracing the route to 172.16.20.75
1 172.16.21.2 0 msec 0 msec 0 msec
2 172.16.20.1 4 msec 0 msec 0 msec
3 172.16.21.2 4 msec 0 msec 0 msec
4 172.16.20.1 0 msec 0 msec 4 msec
5 172.16.21.2 0 msec 0 msec 4 msec
6 172.16.20.1 0 msec 0 msec 4 msec
7 172.16.21.2 0 msec 0 msec 4 msec
8 172.16.20.1 0 msec 0 msec 4 msec
9 172.16.21.2 4 msec 0 msec 4 msec
10 172.16.20.1 4 msec 0 msec 4 msec
11 172.16.21.2 4 msec
Roo#
```

示例 3-43 给出了 Kanga 的 ARP 高速缓冲。在 Kanga 的高速缓冲中，Milne 的 IP 地址与 MAC 标识符 00e0.1e58.dc39 相对应。但是当检查 Milne 的接口时，发现 Milne 的 MAC 标识符为 0002.6779.0f4c，因此断定 Kanga 一定获取了不正确的信息。

示例 3-43 在 Kanga 的 ARP 高速缓冲中，有一个关于 Milne 的表项，其中数据链路标识是错误的

```
Kanga#show arp
Protocol  Address      Age (min)   Hardware Addr   Type   Interface
Internet  172.16.21.1          2   00e0.1e58.dc3c  ARPA   Ethernet1
Internet  172.16.20.2          -   00e0.1e58.dcb1  ARPA   Ethernet0
Internet  172.16.21.2          -   00e0.1e58.dcb4  ARPA   Ethernet1
Internet  172.16.20.75         2   00e0.1e58.dc39  ARPA   Ethernet0
Kanga#
```

再次查看 Kanga 的 ARP 高速缓冲，令人感到疑惑的是与 Milne 相对应的 MAC 标识符类似于 Kanga 自己 Cisco 接口的 MAC 标识符（路由器接口的 MAC 地址没有相应的存活时间）。因为 Milne 不是 Cisco 产品，所以 MAC 标识符的前 3 个字节应该与 Kanga 接口的 MAC 标识符不同。网络上其他的 Cisco 产品惟有 Roo，因此应该检查一下 Roo 的 ARP 高速缓冲（参见示例 3-44）。经查 Roo 接口 E0 的 MAC 标识符即为 00e0.1e58.dc39。

示例 3-44 Roo 的 ARP 高速缓冲显示出 Kanga 获取的 Milne 的 MAC 标识符实际上是 Roo 的接口 E0 的

```
Roo#show arp
Protocol  Address      Age (min)   Hardware Addr   Type   Interface
Internet  172.16.21.1          -   00e0.1e58.dc3c  ARPA   Ethernet0
Internet  172.16.20.1          -   00e0.1e58.dc39  ARPA   Ethernet0
Internet  172.16.20.2          7   00e0.1e58.dcb1  ARPA   Ethernet0
Internet  172.16.21.2          7   00e0.1e58.dcb4  ARPA   Ethernet1
Roo#
```

所以 Kanga 错误地认为 Roo 的接口 E0 就是 Milne 的接口 。Kanga 使用 00e0.1e58.dc39 作为发向 Milne 数据帧的目标标识符，Roo 接收到该帧，在读取封装数据包的目标地址之后，又将数据包路由回 Kanga。

但 Kanga 是如何得到这个错误信息的呢？答案是代理 ARP 。当 Kanga 首次收到发往 Milne 的数据包时，它将发送 ARP 请求，该请求将询问 Milne 的数据链路标识符。Milne 发

回了响应，但是 Roo 也在接口 E0 收到了此 ARP 请求。由于在 Roo 上也有一条通向 Milne 的路由，这条路由所在的网络不是 Roo 收到 ARP 请求的网络，所以 Roo 发送了一个代理 ARP 应答。Kanga 收到 Milne 的 ARP 应答后将相关信息输入到 ARP 高速缓冲内。由于网桥的时延造成了来自 Roo 代理 ARP 的应答随后到达 Kanga，这时 Kanga 用新信息覆盖了 ARP 缓冲内的原始信息。

解决该问题的办法有两种。一种是使用以下命令关闭 Roo E0 接口上的代理 ARP 功能：

```
Roo(config)#interface e0
Roo(config-if)#no ip proxy-arp
```

第二种办法是在 Kanga 上为 Milne 配置静态 ARP 表项：

```
Kanga(config)#arp 172.16.20.75 0002.6779.0f4c arpa
```

该表项不会被任何 ARP 应答所覆盖。示例 3-45 显示出正在输入静态 ARP 表项以及 Kanga 上 ARP 高速缓冲的相应结果。

示例 3-45 静态 ARP 表项纠正了代理 ARP 所造成的错误

```
Kanga(config)#arp 172.16.20.75 0002.6779.0f4c arpa
Kanga(config)#^Z
Kanga#
%SYS-5-CONFIG_I: Configured from console by console
Kanga#show arp
Protocol  Address      Age (min)   Hardware Addr    Type   Interface
Internet  172.16.21.1         10   00e0.1e58.dc3c   ARPA   Ethernet1
Internet  172.16.20.2          -   00e0.1e58.dcb1   ARPA   Ethernet0
Internet  172.16.21.2          -   00e0.1e58.dcb4   ARPA   Ethernet1
Internet  172.16.20.75         -   0002.6779.0f4c   ARPA
Kanga#
```

两种方法哪一种最好取决于网络环境。使用静态 ARP 表项意味着如果 Milne 的网络接口被更换，那么需要修改相应的 ARP 表项来反应新的 MAC 标识符。另一方面，如果没有主机使用代理 ARP 功能，关闭此功能也是一种很好的办法。

3.3.3 案例研究：被取代的路由器

图 3-11 中有两台路由器 Honeypot 和 Honeybee 通过帧中继链路相连，路由器被配置为 IPv4 和 IPv6，并且使用静态路由建立了静态路由表。

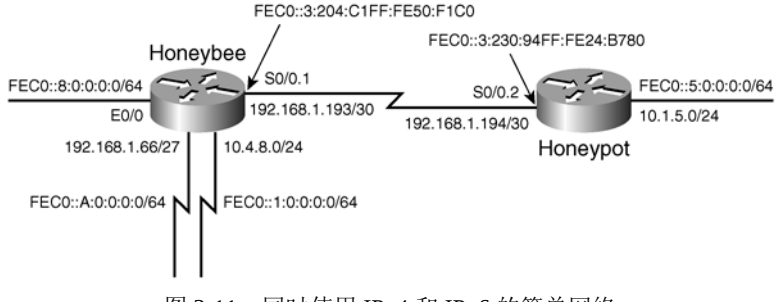


图 3-11 同时使用 IPv4 和 IPv6 的简单网络

示例 3-46 和示例 3-47 分别给出了 Honeypot 和 Honeybee 的路由配置。

示例 3-46 Honeypot 的路由配置

```
ip route 10.4.0.0 255.255.0.0 192.168.1.193
ip route 192.168.1.0 255.255.255.0 192.168.1.193
ipv6 route FEC0::/62 Serial 0/0.2
ipv6 route FEC0:0:0:8::/62 FEC0::3:204:C1FF:FE50:F1C0
```

示例 3-47 Honeybee 的路由配置

```
ip route 10.1.0.0 255.255.0.0 192.168.1.194
ipv6 route FEC0:0:0:5::/64 Serial 0/0.1
```

如示例 3-48 所示，在两台路由器之间进行 ping 和 trace 测试，结果显示网络工作正常。而且两个站点之间的流量也正常。

示例 3-48 Pings 和 traces 的结果显示两台路由器之间可以成功地进行通信

```
Honeypot#ping
Protocol [ip]:
Target IP address: 10.4.8.1
Extended commands [n]: y
Source address or interface: 10.1.5.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.4.8.1, timeout is 2 seconds:
Packet sent with a source address of 10.1.5.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/33 ms
-
Honeypot#ping fec0:0:0:8:204:c1ff:fe50:f1c0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0::8:204:C1FF:FE50:F1C0, timeout is 2 secon
ds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/32/33 ms

Honeypot#trace
Protocol [ip]: ipv6
Target IPv6 address: fec0:0:0:8:204:c1ff:fe50:f1c0
Source address: fec0:0:0:5:230:94ff:fe24:b780
Type escape sequence to abort.
Tracing the route to FEC0::8:204:C1FF:FE50:F1C0
  1 FEC0::3:204:C1FF:FE50:F1C0 24 msec 24 msec 24 msec

Honeypot#trace
Protocol [ip]: ipv6
Target IPv6 address: fec0::a:0:0:0:1
Source address: fec0::5:230:94ff:fe24:b780
Type escape sequence to abort.
Tracing the route to FEC0:0:0:A::1
 1 FEC0::3:204:C1FF:FE50:F1C0 24 msec 24 msec 24 msec
```

Honeypot 可以从自己的以太网地址到达 Honeybee 的以太网地址，IPv4 和 IPv6 都可以。Honeypot 还可以通过 Honeybee 到达 FEC0:0:0:A::1。

可是 Honeybee 被一台新的路由器替换了，当把 Honeybee 的配置导入新路由器时，所有的接口都工作正常。两站点之间的测试结果显示 IPv4 工作正常，但是 IPv6 发生故障（参见示例 3-49）。

示例 3-49 在路由器被替换后，IPv4 工作正常，但是 IPv6 却无法工作

```
Honeypot#ping
Protocol [ip]:
Target IP address: 10.4.8.1
```

（待续）

```
Extended commands [n]: y
Source address or interface: 10.1.5.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.4.8.1, timeout is 2 seconds:
Packet sent with a source address of 10.1.5.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/30/32 ms
-
Honeypot#ping fec0:0:0:8:204:c1ff:fe50:f1c0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FEC0::8:204:C1FF:FE50:F1C0, timeout is 2 seconds:
...H.
Success rate is 0 percent (0/5)
-
Honeypot#trace
Protocol [ip]: ipv6
Target IPv6 address: fec0::8:204:c1ff:fe50:f1c0
Source address: fec0::5:230:94ff:fe24:b780
Type escape sequence to abort.
Tracing the route to FEC0::8:204:C1FF:FE50:F1C0
  1 FEC0::3:2B0:64FF:FE30:1DE0 24 msec 24 msec 24 msec
  2  *  !H  *

Honeypot#trace
Protocol [ip]: ipv6
Target IPv6 address: fec0::a:0:0:0:1
Source address: fec0::5:230:94ff:fe24:b780
Type escape sequence to abort.
Tracing the route to FEC0:0:0:A::1
 1 FEC0::3:2B0:64FF:FE30:1DE0 24 msec 20 msec 20 msec
```

向 Honeybee 以太网的 IPv6 地址 FEC0::8:204:c1FF:FE50:F1C0 ping 和 trace 都告失败，但是向 FEC0::A:0:0:0:1trace 仍然是成功的。

让我们看一下示例 3-50 中 Honeypot 的路由表。IPv6 的路由表中有一条路由通过下一跳地址 FEC0::3:204:C1FF:FE50:F1C0 指向 FEC0:0:0:8::/62。

示例 3-50 IPv6 的路由表没有发生变化，并且安装了静态路由

```
Honeypot#show ipv6 route
IPv6 Routing Table - 8 entries
L   FE80::/10 [0/0]
     via ::, Null0
S   FEC0::/62 [1/0]
     via ::, Serial0/0.2
C   FEC0:0:0:3::/64 [0/0]
     via ::, Serial0/0.2
L   FEC0::3:230:94FF:FE24:B780/128 [0/0]
     via ::, Serial0/0.2
C   FEC0:0:0:5::/64 [0/0]
     via ::, Ethernet0/0
L   FEC0::5:230:94FF:FE24:B780/128 [0/0]
     via ::, Ethernet0/0
S   FEC0:0:0:8::/62 [1/0]
     via FEC0::3:204:C1FF:FE50:F1C0
```

从示例 3-50 可以知道，通过 FEC0::3:204:C1FF:FE50:F1C0 可以到达 FEC0:0:0:8::/62，并且 FEC0:0:0:3::/64 被连接到接口 Serial0/0.2 上；去往 FEC0:0:0:8::/64 和 FEC0:0:0:A::/64 的流量都会从该接口转发给 Honeybee。

替代操作前后的路由是完全一样的。那么问题出在哪里呢？或许 Honeybee 的以太网接口没有正常工作，这可以查看一下示例 3-51 中 Honeybee 的以太网接口状态。

示例 3-51 命令 **show ipv6 interface Ethernet 0/0** 可以显示接口的状态以及有关 IPv6 的一些配置信息

```
Honeybee#show ipv6 interface e 0/0
Ethernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::2B0:64FF:FE30:1DE0
  Global unicast address(es):
    FEC0::8:2B0:64FF:FE30:1DE0, subnet is FEC0:0:0:8::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF30:1DE0
```

从示例 3-51 中可以看到接口状态正常并且配置了 IPv6。根据网络的文档和图表，该接口在替换前可以被 ping 通，但是替换后就不行了。

如果我们仔细查看以太网接口的输出信息，可以发现全球单播地址的后 64 位发生了变化。现在地址变为 FEC0::8:2B0:64FF:FE30:1DE0。因为 IPv6 地址是 EUI-64 格式，所以后 64 位由接口的 MAC 地址确定。当路由器被替换后，MAC 地址也随之改变。路由器上不再有 FEC0::8:204:C1FF:FE50:F1C0 存在。这就是 ping 不通的原因。

但是根据路由表（参见示例 3-50），对于目标网络 FEC0:0:0:8::/62，Honeypot 的静态路由仍然指向 FEC0::3:204:C1FF:FE50:F1C0。如果 MAC 地址发生了变化，那么这个地址就不是同步接口的地址了。示例 3-52 给出了 Honeybee 同步接口的新地址。

示例 3-52 在路由器上，所有配置 EUI-64 地址的同步接口地址后 64 位都是相同的，它来自路由器上的一个 MAC 地址

```
Honeybee#show ipv6 interface serial0/0.1
Serial0/0.1 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::2B0:64FF:FE30:1DE0
  Description: Link to Piglet
  Global unicast address(es):
    FEC0::3:2B0:64FF:FE30:1DE0, subnet is FEC0:0:0:3::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF30:1DE0
```

如果目标地址在 FEC0:0:0:8::/62 的范围内，Honeypot 的 IPv6 静态路由将把相关流量指向下一跳地址 FEC0::3:204:C1FF:FE50:F1C0。虽然下一跳地址是原来路由器的 MAC 地址，然而使用该路由的流量仍然可以被成功路由。

这里虽然指定的下一跳地址不再是合法的，但是它和新的合法地址在相同的子网中（FEC0:0:0:3::/64），正像我们在 Honeypot 的路由表中看到的那样，这个子网被连接到 Honeypot 的接口 Serial0/0.2 上。通过对转发表的递归查询，虽然指定的下一跳地址不存在，但是去往 FEC0:0:0:A::1 的流量依然从 Serial0/0.2 送出。

每当路由器或接口卡被替换时，一定要记得修改参考旧路由器 EUI-64 格式 IPv6 地址的路由表项。

3.3.4 案例研究：追踪 IPv6 故障路由

在图 3-3 中，在 Honeypot 和 Honeytree 之间添加了一条链路，当主链路万一发生中断时，

它可以作为备份链路。改动后的新网络如图 3-12 所示。由于使用该网络的 IPv6 应用对时延不敏感，但是对带宽却要求很高，所以网络管理员决定使用新添加的链路作负载均衡，因此在每台路由器上添加静态路由。

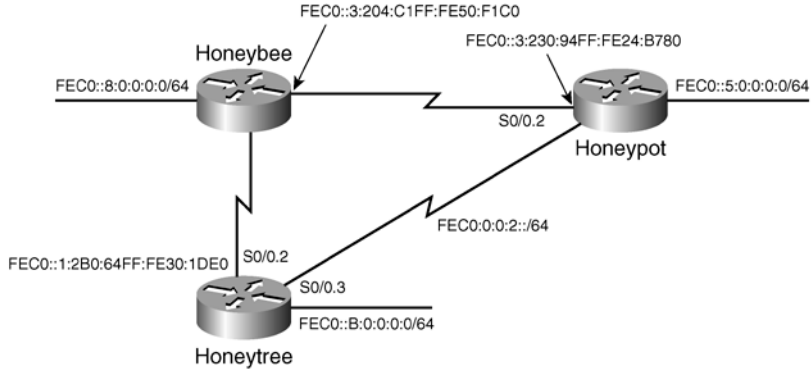


图 3-12 在 IPv6 网络中添加了另一条可供选择的路径，网络变成三角形网络，使得路由器之间可以相互到达

如示例 3-53 所示，对应于 Honeypot 每个已存在的路由，示例中又添加了第二条路由。示例 3-54 和示例 3-55 也分别对 Honeytree 和 Honeybee 进行了类似的操作。

示例 3-53 对于图 3-12 的新网络，Honeypot 的相应配置

```
ipv6 route FEC0::/62 FEC0::2:2B0:64FF:FE30:1DE0
ipv6 route FEC0::/62 FEC0::3:204:C1FF:FE50:F1C0
ipv6 route FEC0:0:0:8::/62 FEC0::2:2B0:64FF:FE30:1DE0
ipv6 route FEC0:0:0:8::/62 FEC0::3:204:C1FF:FE50:F1C0
```

示例 3-54 对于图 3-12 的新网络，Honeytree 的相应配置

```
ipv6 route FEC0:0:0:3::/64 FEC0::2:230:94FF:FE24:B780
ipv6 route FEC0:0:0:3::/64 FEC0::1:204:C1FF:FE50:F1C0
ipv6 route FEC0:0:0:5::/64 FEC0::2:230:94FF:FE24:B780
ipv6 route FEC0:0:0:5::/64 FEC0::1:204:C1FF:FE50:F1C0
ipv6 route FEC0:0:0:8::/64 FEC0::1:204:C1FF:FE50:F1C0
ipv6 route FEC0:0:0:8::/64 FEC0::2:230:94FF:FE24:B780
```

示例 3-55 对于图 3-12 的新网络，Honeybee 的配置

```
ipv6 route FEC0:0:0:2::/64 FEC0::1:2B0:64FF:FE30:1DE0
ipv6 route FEC0:0:0:2::/64 FEC0::3:230:94FF:FE24:B780
ipv6 route FEC0:0:0:5::/64 FEC0::1:2B0:64FF:FE30:1DE0
ipv6 route FEC0:0:0:5::/64 FEC0::3:230:94FF:FE24:B780
ipv6 route FEC0:0:0:B::/64 FEC0::3:230:94FF:FE24:B780
ipv6 route FEC0:0:0:B::/64 FEC0::1:2B0:64FF:FE30:1DE0
```

现在，IPv6 的路由选择间歇性的出现故障，有时工作，有时不工作。可以看出路由表中的静态路由与设计的完全相同。从 Honeypot 向 Honeybee 的以太网接口进行 ping 测试，有时成功，有时出现主机不可达（参见示例 3-56）。

示例 3-56 IPv6 有时可以 ping 通

```
Honeypot#ping fec0::8:204:c1ff:fe50:f1c0

Type escape sequence to abort.
```

（待续）

```
Sending 5, 100-byte ICMP Echos to FEC0::8:204:C1FF:FE50:F1C0, timeout is 2 secon
ds:
!!HHH
Success rate is 40 percent (2/5), round-trip min/avg/max = 60/90/120 ms
Honeypot#
```

在 Honeypot 上调试 IPv6 的 ICMP 数据包，你可以发现能成功地收到一些响应数据包（参见示例 3-57），但是还会收到一些来自 Honeytree 的目标不可达的 ICMP 消息。

示例 3-57　在 Honeypot 上的调试结果显示：有时成功，有时失败

```
ICMPv6: Sending echo request to FEC0::8:204:C1FF:FE50:F1C0
ICMPv6: Received ICMPv6 packet from FEC0::8:204:C1FF:FE50:F1C0, type 129
ICMPv6: Received echo reply from FEC0::8:204:C1FF:FE50:F1C0
ICMPv6: Sending echo request to FEC0::8:204:C1FF:FE50:F1C0
ICMPv6: Received ICMPv6 packet from FEC0::2:2B0:64FF:FE30:1DE0, type 1
ICMPv6: Received ICMP unreachable code 3 from FEC0::2:2B0:64FF:FE30:1DE0
```

在 Honeytree 上调试 ICMPv6 的结果（参见示例 3-58）显示，IPv6 的数据包正在从 S0/0.3 和 S0/0.2 到达。

示例 3-58　调试结果显示数据包从两个不同的接口到达

```
Honeytree#
 IPV6: source FEC0::2:230:94FF:FE24:B780 (Serial0/0.3)
       dest FEC0::8:204:C1FF:FE50:F1C0 (Serial0/0.2)
       traffic class 0, flow 0x0, len 100+4, prot 58, hops 63, forwarding
 IPV6: source FEC0::8:204:C1FF:FE50:F1C0 (Serial0/0.2)
       dest FEC0::2:230:94FF:FE24:B780 (Serial0/0.3)
       traffic class 0, flow 0x0, len 100+4, prot 58, hops 63, forwarding
 IPV6: source FEC0::2:230:94FF:FE24:B780 (Serial0/0.3)
       dest FEC0::8:204:C1FF:FE50:F1C0 (Serial0/0.3)
       traffic class 0, flow 0x0, len 100+4, prot 58, hops 63, destination is not
connected
 IPv6: SAS picked source FEC0::2:2B0:64FF:FE30:1DE0 for FEC0::2:230:94FF:FE24:B780
(Serial0/0.3)
 ICMPv6: Sending ICMP unreachable code 3 to FEC0::2:230:94FF:FE24:B780 about
FEC0::8:204:C1FF:FE50:F1C0
```

源地址为 fec0::2:230:94ff:fe24:b780 的数据包从接口 S0/0.3 到达 Honeytree（来自 Honeypot 的数据包经过 Honeypot 至 Honeytree 的链路到达）。对于目标网络 fec0::8:204:c1ff:fe50:f1c0 来说，出站接口是 S0/0.2 或 S0/0.3。这是因为 Honeytree 执行基于数据包的负载均衡，在 S0/0.2 和 S0/0.3 之间交替转发数据包。当数据包从 S0/0.2 出站并且从 S0/0.3 入站时，数据包将会被转发。但是如果从 S0/0.3 出站，而又从 S0/0.3 入站，则路由器会产生 ICMP 错误信息，表明 ping 失败。

如果数据包从同步接口出站后又从相同的接口入站，那么就表明存在路由环路。由于路由器正在处理交换和基于数据包的负载均衡，每台路由器都会轮流使用两条路由，所以一些数据包会从出站的接口又入站。

3.4　展　望

对于精确地控制网络的路由选择行为来说，静态路由不失为一个强有力的工具。然而，如果经常发生网络拓扑变化，那么手动配置方式导致静态路由的管理工作根本无法进行下去。动态路由选择协议能够使网络迅速并自动地响应网络拓扑的变化。在研究特定 IP 路由选择协议的细节之前，我们首先需要研究一些围绕动态协议的常见问题。第 4 章将介绍动态路由选

择协议。

3.5 总结表：第 3 章命令总结

命令	描述
arp *ip-address hardware-address*	把 IP 类型（别名）地址静态地映射到硬件地址
debug ip packet	显示有关接收、生成、转发 IP 数据包的信息。而关于快速交换的数据包信息将不被显示
debug ipv6 packet	显示有关接收、生成、转发 IPv6 数据包的信息
ip cef	为 IPv4 启用 Cisco 急速转发功能
ip load-sharing{per-packet | per-destination}	在出站接口上配置负载均衡方式
ip proxy-arp	启用代理 ARP 功能
ip route *prefix mask*{*address*|*interface*[*next-hop-address*]}[*distance*] [**permanent**] [**name** *naem*] [**teg** *tag-number*]	向路由表添加静态路由
ip route-cache	在接口上为 IPv4 配置交换高速缓冲的类型
ipv6 cef	为 IPv6 启用 Cisco 急速转发功能。IPv4 的 CEF 功能必须在 IPv6 CEF 功能之前打开
Ipv6 unicast-routing	启动 IPv6 路由选择。缺省情况下 IPv6 路由选择是关闭的
Ipv6 route *prefix/prefix length* {*address* | *interface* [*next-hop-address*] } [*distance*] [**permanent**] [**name** *name*] [**tag** *tag-number*]	静态添加 IPv6 路由
show cdp neighbor detail	显示邻居路由器的 IOS 版本号以及 IPv4 和 IPv6 接口配置等信息
show ip cef	显示 CEF 转发高速缓冲消息，包括 CEF 未启用消息
show ipv6 cef {*interface***} detail**	显示接口上 CEF 功能是否被打开
show ipv6 interface {*interface***}**	显示接口以及 IPv6 特定的接口信息
show ip route	显示 IP 路由表
show ipv6 route	显示 IPv6 路由表
show ipvb cef	

3.6 复 习 题

1．路由表中需要保存哪些信息？
2．当路由表指明对一个地址进行了变长子网划分时，这意味着什么？
3．什么是非连续子网？
4．使用什么 IOS 命令检查 IPv4 路由表？
5．使用什么 IOS 命令检查 IPv6 路由表？
6．在路由表中与非直连路由相关的括号内的两个数字表示什么？
7．当使用出站接口代替静态路由中的下一跳地址时，路由表会有什么不同？
8．什么是汇总路由？在静态路由选择的上下文中，汇总路由怎样起作用？
9．什么是管理距离？
10．什么是浮动静态路由？
11．等价均分负载与非等价均分负载之间有什么不同？
12．接口上的交换模式怎样影响均分负载？

13．什么是递归表查询？

3.7 配置练习

1．如图 3-13 所示的网络，为每台路由器配置静态路由。要求每个子网都有独立的表项。

2．使用最少的路由表项重新配置练习 1 中的静态路由[1]（提示：RTA 仅有两条静态路由）。

3．如图 3-14 所示的网络，为每台路由器配置静态路由。假设所有链路的介质相同。为保证最大利用率和线路冗余，请使用负载均衡和浮动静态路由技术。

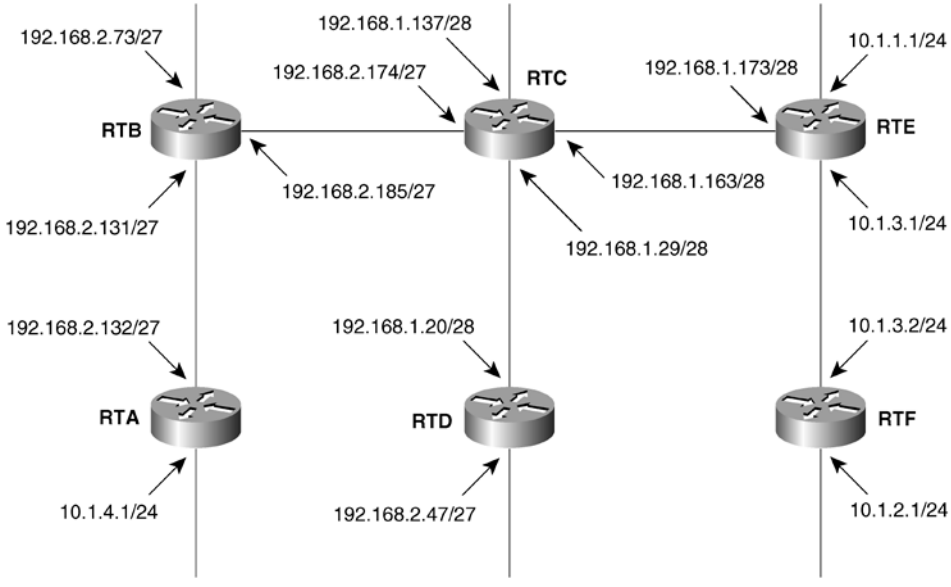


图 3-13 配置练习 1 和练习 2 中用到的网络

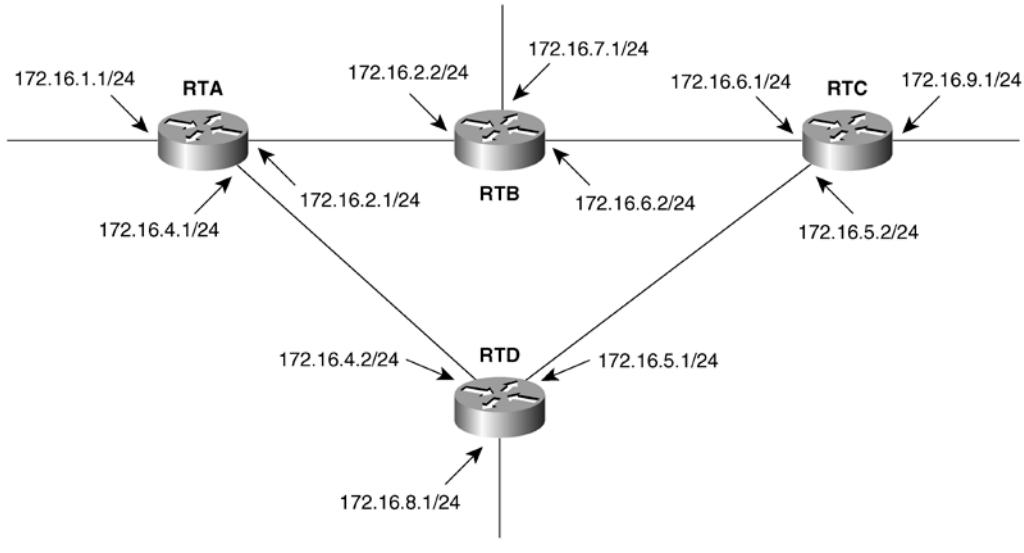


图 3-14 配置练习 3 用到的网络

[1] 如果在实验室做此练习，请确保在所有 6 台路由器上都配置命令 **ip classless**。

3.8 故障诊断练习

1．在图 3-2 所示的网络和相关配置中，把 Piglet 的路由配置由

```
Piglet(config)# ip route 192.168.1.0 255.255.0 192.168.1.193
Piglet(config)# ip route 10.4.0.0 255.255.0.0 192.168.1.193
```

修改为：

```
Piglet(config)# ip route 192.168.1.0 255.255.255.224 192.168.1.193
Piglet(config)# ip route 10.0.0.0 255.255.0.0 192.168.1.193
```

会发生什么情况？

2．示例 3-59 给出了图 3-15 中路由器的静态路由配置。

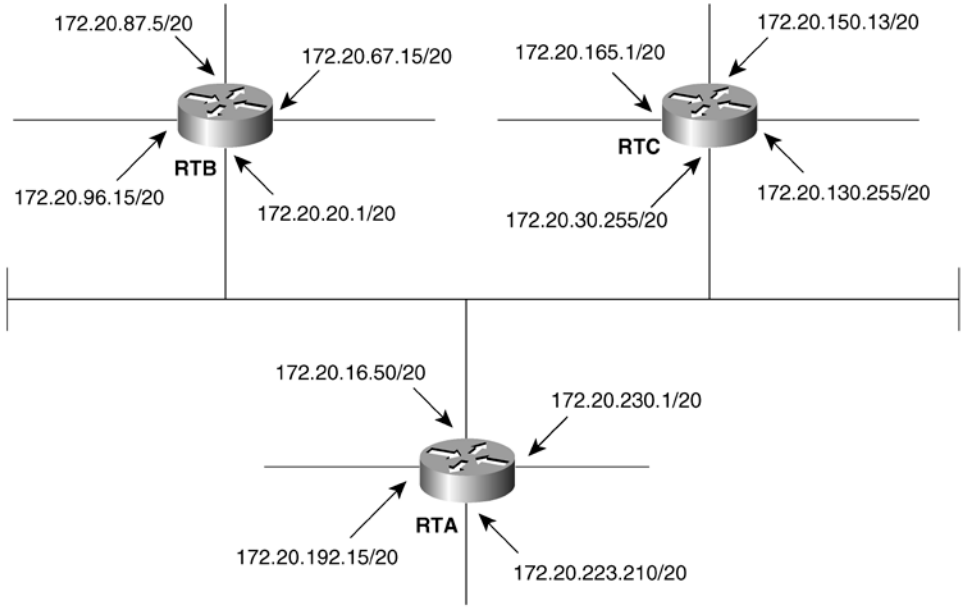


图 3-15 故障诊断练习 2 用到的网络

示例 3-59 图 3-15 中路由器的静态路由配置

```
!RTA
ip route 172.20.96.0 255.255.240.0 172.20.20.1
ip route 172.20.82.0 255.255.240.0 172.20.20.1
ip route 172.20.64.0 255.255.240.0 172.20.20.1
ip route 172.20.160.0 255.255.240.0 172.20.30.255
ip route 172.20.144.0 255.255.240.0 172.20.30.255
ip route 172.20.128.0 255.255.240.0 172.20.30.255
!RTB
ip route 172.20.192.0 255.255.240.0 172.20.16.50
ip route 172.20.224.0 255.255.240.0 172.20.16.50
ip route 172.20.128.0 255.255.240.0 172.20.16.50
ip route 172.20.160.0 255.255.240.0 172.20.30.255
ip route 172.20.144.0 255.255.240.0 172.20.30.255
ip route 172.20.128.0 255.255.240.0 172.20.30.255
```

（待续）

```
!RTC
ip route 172.20.192.0 255.255.240.0 172.20.16.50
ip route 172.20.208.0 255.255.255.0 172.20.16.50
ip route 172.20.224.0 255.255.240.0 172.20.16.50
ip route 172.20.96.0 255.255.240.0 172.20.20.1
ip route 172.20.82.0 255.255.240.0 172.20.20.1
ip route 172.20.64.0 255.255.240.0 172.20.20.1
```

用户抱怨网络中存在一些连通性问题。请在静态路由配置中找出错误。

3．图 3-16 给出了另一个网络，同样有用户抱怨存在连通性问题。示例 3-60 到示例 3-63 分别给出了 4 台路由器的路由表，请找出静态路由配置的错误。

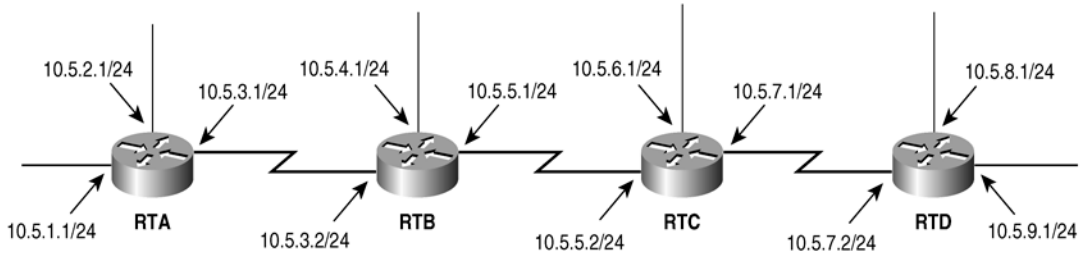


图 3-16 故障诊断练习 3 的网络

示例 3-60 图 3-16 中 RTA 的路由表

```
RTA#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route
Gateway of last resort is not set
     10.0.0.0/8 is subnetted, 9 subnets
S       10.5.9.0 [1/0] via 10.5.3.2
S       10.5.8.0 [1/0] via 10.5.3.2
S       10.5.7.0 [1/0] via 10.5.3.2
S       10.5.6.0 [1/0] via 10.5.3.2
S       10.5.5.0 [1/0] via 10.5.3.2
S       10.5.4.0 [1/0] via 10.5.3.2
C       10.5.3.0 is directly connected, Serial0
C       10.5.2.0 is directly connected, TokenRing1
C       10.5.1.0 is directly connected, TokenRing0
RTA#
```

示例 3-61 图 3-16 中 RTB 的路由表

```
RTB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route
Gateway of last resort is not set
     10.0.0.0/8 is subnetted, 9 subnets
S       10.5.9.0 [1/0] via 10.5.5.2
S       10.5.8.0 [1/0] via 10.5.5.2
S       10.5.7.0 [1/0] via 10.5.5.2
S       10.5.6.0 [1/0] via 10.5.5.2
C       10.5.5.0 is directly connected, Serial1
C       10.5.4.0 is directly connected, TokenRing0
```

（待续）

```
C       10.5.3.0 is directly connected, Serial0
S       10.5.2.0 [1/0] via 10.5.3.1
S       10.5.1.0 [1/0] via 10.5.3.1
RTB#
```

示例 3-62 图 3-16 中 RTC 的路由表

```
RTC#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route, o - ODR
Gateway of last resort is not set
     10.0.0.0/24 is subnetted, 8 subnets
S       10.5.9.0 [1/0] via 10.5.7.2
S       10.5.8.0 [1/0] via 10.5.5.1
C       10.5.7.0 is directly connected, Serial1
C       10.5.6.0 is directly connected, Ethernet0
S       10.1.1.0 [1/0] via 10.5.5.1
C       10.5.5.0 is directly connected, Serial0
S       10.5.3.0 [1/0] via 10.5.5.1
S       10.5.2.0 [1/0] via 10.5.5.1
RTC#
```

示例 3-63 图 3-16 中 RTD 的路由表

```
RTD#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route, o - ODR
Gateway of last resort is not set
     10.0.0.0/24 is subnetted, 9 subnets
C       10.5.9.0 is directly connected, Ethernet1
C       10.5.8.0 is directly connected, Ethernet0
C       10.5.7.0 is directly connected, Serial0
S       10.5.6.0 [1/0] via 10.5.7.1
S       10.5.5.0 [1/0] via 10.5.7.1
S       10.4.5.0 [1/0] via 10.5.7.1
S       10.5.3.0 [1/0] via 10.5.7.1
S       10.5.2.0 [1/0] via 10.5.7.1
S       10.5.1.0 [1/0] via 10.5.7.1
RTD#
```

本章包括以下主题：

- 路由选择协议基础；
- 距离矢量路由选择协议；
- 链路状态路由选择协议；
- 内部和外部网关协议；
- 静态或动态路由选择。

第 4 章

动态路由选择协议

上一章解释了路由器为了正确地交换数据包到达各自的目的地所需要知道的信息，以及怎样手工向路由表输入这些信息。本章将讨论路由器如何发现这些信息，并且通过动态路由选择协议与其他路由器共享这些信息。路由选择协议（routing protocol）作为路由器之间进行相互交流的语言，用于实现可达性信息和网络状态的共享。

动态路由选择协议不仅执行路径决策和路由表更新功能，而且还要在最优路径不可用时决策下一条最优路径。动态路由选择相比静态路由选择而言最大的优势在于，动态路由选择能够缓解拓扑变化带来的影响。

显然，为了正确地通信，通信双方必须使用相同的语言。自从 IP 路由选择出现以来，共有 8 种主要的 IP 路由选择协议可供选择；[1] 如果一台路由器使用 RIP 与另一台使用 OSPF 的路由器进行对话，那么它们将无法实现信息共享，因为它们使用的语言不相同。后继章节将会分析所有目前在用的 IP 路由选择协议，甚至涉及如何使路由器“能说两种语言”，但是首先有必要研究一下所有路由选择协议共同的一些特性和问题——IP 或其他方面。

4.1 路由选择协议基础

所有路由选择协议都是围绕着一种算法而构建的。通常，一种算法是一个逐步解决问题的过程。一种路由算法至少应指明以下内容：

- 向其他路由器传送网络可达性信息的过程。
- 从其他路由器接收可达性信息的过程。

[1] 在这 8 种协议中，BGP 已经取代了 EGP，Cisco Systems 公司的 EIGRP 也取代了它自己的 IGRP，RIPv2 也正在迅速取代 RIPv1。

- 基于现有可达性信息决策最优路由的过程以及在路由表中记录这些信息的过程。
- 响应、修正和通告网络中拓扑变化的过程。

对所有路由选择协议来说，几个共同的问题是路径决策、度量、收敛和负载均衡。

4.1.1 路径决策

在网络内的所有子网都必须连接到一台路由器上，无论什么情况下，只要路由器有接口连接到一个网络上，那么该接口必须具有一个属于该网络的地址[1]。这个地址就是可达性信息的起始点。

图 4-1 给出了一个包含 3 台路由器的网络。路由器 A 知道网络 192.168.1.0、 192.168.2.0 和 192.168.3.0 的存在，因为路由器有接口连接到这些网络上， 并且配置了相应的地址和掩码。同样的，路由器 B 知道网络 192.168.3.0、192.168.4.0、192.168.5.0 和 192.168.6.0 的存在，路由器 C 知道网络 192.168.6.0、 192.168.7.0 和 192.168.1.0 的存在。由于每个接口都实现了所连接网络的数据链路和物理层协议，因此路由器也知道网络的状态（工作正常“up”或发生故障“down”）。

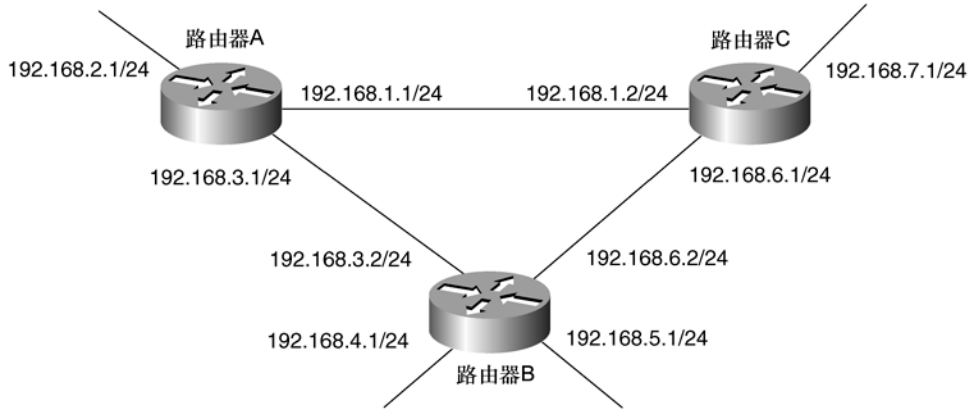


图 4-1 每台路由器从分配给它的接口地址和掩码可以知道它的直连网络

信息共享过程看上去很简单。考虑路由器 A：

步骤 1：路由器 A 检查自己的 IP 地址和相关掩码，然后推导出与自身所连接的网络是 192.168.1.0、192.168.2.0 和 192.168.3.0。

步骤 2：路由器 A 将这些网络连同某种标记一起保存到路由表中，其中标记指明了网络是直连网络。

步骤 3：路由器 A 向数据包中加入以下信息：“我的直连网络是 192.168.1.0、192.168.2.0 和 192.168.3.0。

步骤 4：路由器 A 向路由器 B 和路由器 C 发送这些路由信息数据包的拷贝，或者叫做路由选择更新。

路由器 B 和路由器 C 执行与路由器 A 完全相同的步骤，并且也向路由器 A 发送带有与它们直接相连的网络的更新信息。路由器 A 将接收到的信息连同发送路由器的源地址一起写

[1] 许多点到点链路都被配置为“未编号”的链路——也就是不给点到点链路两端的接口分配地址。但是这又不违背每个接口都必须有地址这个原则，因为这些接口通常使用路由器上的环路地址作为代理地址。

入路由表。现在，路由器 A 知道了所有的网络，而且还知道连接这些网络的路由器的地址。

这个过程看似非常简单。那么为什么路由选择协议比这更复杂呢？让我们重新看一下图 4-1。

- 路由器 A 将来自路由器 B 和路由器 C 的更新信息保存到路由表之后，它应该用这些信息作什么？例如，路由器 A 是否应该将路由器 C 的数据包信息传递给路由器 B，还是将路由器 B 的路由选择信息包传递给路由器 C 呢？
- 如果路由器 A 没有转发这些更新消息，那么就不能完成信息共享。例如，如果路由器 B 和路由器 C 之间的链路不存在，那么这两台路由器就无法知道对方的网络。因此路由器 A 必须转发那些更新信息，但是这样做又产生了新的问题。
- 如果路由器 A 从路由器 B 和路由器 C 那里知道网络 192.168.4.0，那么为了到达该网络应该使用哪一台路由器呢？它们都合法吗？谁是最优路径呢？
- 什么机制可以确保所有路由器能接收到所有的路由选择信息，而且这种机制还可以阻止更新数据包在网络中无休止地循环下去呢？
- 如果路由器共享某个直连网络（192.168.1.0、192.168.3.0 和 192.168.6.0），那么路由器是否仍旧应该通告这些网络呢？

这些问题同开头解释路由选择协议一样显得有点过分单纯，但是它们给读者的感觉却是：正是这些问题造成了协议的复杂性。每种路由选择协议无论如何都需要解决这些问题，这在下面的章节中将会变得更加清楚。

4.1.2 度量

当有多条路径到达相同目标网络时，路由器需要一种机制来计算最优路径。度量（metric）是指派给路由的一种变量，作为一种手段，度量可以按最好到最坏，或按最先选择到最后选择的顺序对路由进行等级划分。考虑下面的例子，了解为什么需要度量。

如图 4-1 所示，假设在网络中信息共享可以正常进行，并且路由器 A 中的路由表如表 4-1 所示。

表 4-1 有关图 4-1 中路由器 A 的一个不完善的路由表

网络	下一跳路由器
192.168.1.0	直接被连接
192.168.2.0	直接被连接
192.168.3.0	直接被连接
192.168.4.0	B，C
192.168.5.0	B，C
192.168.6.0	B，C
192.168.7.0	B，C

路由表说明了前 3 个网络直接连接到路由器，因而从路由器 A 到达它们不需要进行路由选择。根据路由表，后 4 个网络需要经过路由器 B 或路由器 C 才能到达。这些信息同样是正确的。但是，如果通过路由器 B 或路由器 C 都可以到达网络 192.168.7.0，那么优先选择哪一条路径呢？这时就需要度量对这两条路径进行等级划分。

不同的路由选择协议使用不同的度量。例如，RIP 定义含有路由器跳数最少的路径是最优路径；EIGRP 基于路径沿路最小带宽和总时延定义最优路径。下一小节将给出这些度量和其他

常用度量的基本定义。更复杂的内容——例如某些路由选择协议（EIGRP）怎样使用多个参数来计算度量以及如何处理度量值相同的路由——将在本书后面讲述各协议的章节中讨论。

1. 跳数

跳数（Hop Count）度量可以简单地记录路由器跳数。例如，如果数据包从路由器 A 的接口 192.168.3.1 发出，经过路由器 B 到达网络 192.168.5.0，则记为 1 跳；如果从路由器接口 192.168.1.1 发出，经路由器 C 和路由器 B 到达网络 192.168.5.0，记为 2 跳。假设仅使用跳数作为度量，那么最优路径就是跳数最少的路径，在本例中就是 A-B。

但 A-B 是真正的最优路径吗？如果 A-B 是一条 DS-0 链路，并且 A-C 和 C-B 都是 T1 链路，那么跳数为2的路由实际上可能是最优路径，因为带宽对如何有效地使流量通过网络影响很大。

2. 带宽

带宽（Bandwidth）度量将会选择高带宽路径，而不是低带宽链路。然而带宽本身可能不是一个好的度量。如果两条 T1 链路或其中一条被其他流量过多占用，那么与一个 56KB 的空闲链路相比到底谁好呢？或者一条高带宽但时延也很大的链路又如何呢？

3. 负载

负载（Load）度量反应了流量占用沿途链路带宽的数量。最优路径应该是负载最低的路径。

不像跳数和带宽，路径上的负载会发生变化，因而度量也会跟着变化。这里要当心，如果度量变化过于频繁，路由波动——最优路径频繁变化——可能就发生了。路由波动会对路由器的 CPU、数据链路的带宽和全网稳定性产生负面影响。

4. 时延

时延（Delay）是度量数据包经过一条路径所花费的时间。使用时延作度量的路由选择协议将会选择最低时延的路径作为最优路径。有多种方法可以测量时延。时延不仅要考虑链路时延，而且还要考虑路由器的处理时延和队列时延等因素。另一方面，路由的时延可能根本无法测量。因此，时延可能是沿路径各接口所定义的静态延时量的总和，其中每个独立的时延量都是基于连接接口的链路类型估算得到的。

5. 可靠性

可靠性（Reliability）度量是用来测量链路在某种情况下发生故障的可能性，可靠性可以是变化的或固定的。可变可靠性度量的例子是链路发生故障的次数，或特定时间间隔内收到错误的次数。固定可靠性度量是基于管理员确定的一条链路的已知量。可靠性最高的路径将会被最优先选择。

6. 代价

由管理员设置的代价（Cost）度量可以反应更优或更差路由。通过任何策略或链路特性都可以对代价进行定义，同时代价也可以反应出网络管理员对路径的随意判断。因而代价是一个描述无量纲度量的术语。

每当谈论起路由选择的话题时，常常会把代价作为一种通用术语。例如，“RIP 基于跳数选择代价最低的路径”。还有一个通用术语是最短（shortest），如“RIP 基于跳数选择最短路径”。当在这种情况中使用它们时，最小代价（最高代价）或最短（最长）仅仅指的是路由选择协议基于自己特定的度量对路径的一种看法。

4.1.3 收敛

动态路由选择协议必须包含一系列过程，这些过程用于路由器向其他路由器通告本地的直连网络，接收并处理来自其他路由器的同类信息，以及传递从其他路由器接收到的信息。此外，路由选择协议还需要定义已确定的最优路径的度量。

对路由选择协议来说，另一个标准是网络上所有路由器的路由表中的可达性信息必须一致。在图 4-1 中，如果路由器 A 确定了经过路由器 C 到达网络 192.168.5.0 是最优路径，而路由器 C 确定到达相同网络的最优路径是经过路由器 A，那么路由器 A 发向 192.168.5.0 的数据包到达路由器 C 后又被发回给路由器 A，路由器 A 又再次发给路由器 C，如此往复循环。我们称这种在两个或多个目标网络之间流量的持续循环为路由选择环路（routing loop）。

使所有路由表都达到一致状态的过程叫做收敛（convergence）。全网实现信息共享以及所有路由器计算最优路径所花费的时间总和就是收敛时间。

图 4-2 所示的网络已经收敛，但是现在拓扑又发生了变化。最左边的两台路由器之间的链路发生故障，这两台直接相连的路由器都从数据链路协议获知链路故障，转而通知它们的邻居该链路不再可用。邻接路由器立即更新路由表并通知它们的邻居，这个过程一直持续到所有路由器都知道此变化为止。

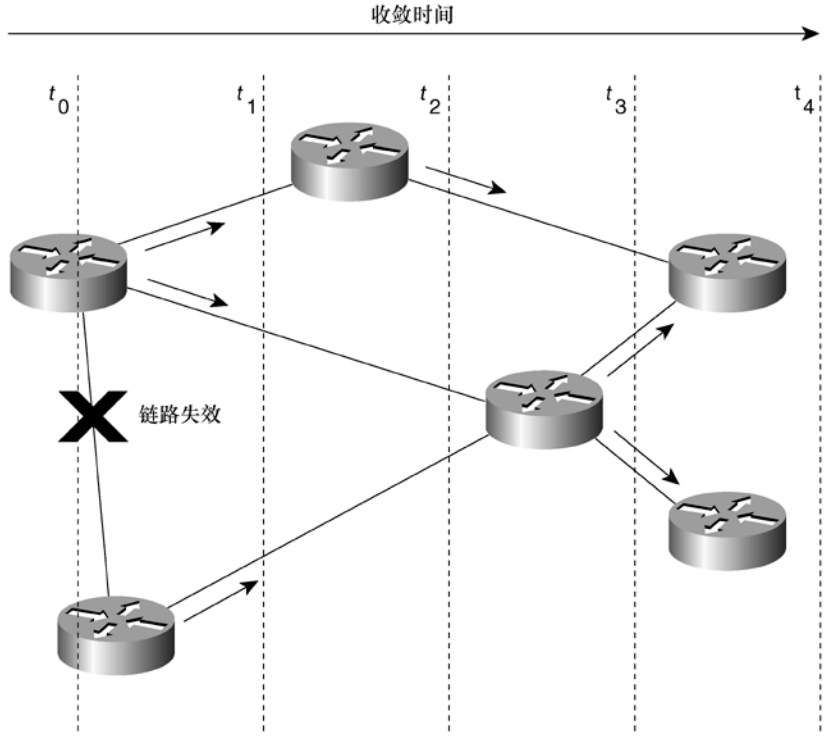


图 4-2 拓扑发生变化后重新收敛需要一定时间。当网络处于未收敛状态时，路由器易受到错误路由选择信息的影响

注意，在 t_2 时刻，最左边的 3 台路由器知道拓扑发生了变化，但最右边的 3 台路由器依然不知道。最右边的 3 台路由器仍旧保存着原来的路由信息并继续交换数据包。这时网络处于未收敛状态，正是在这段时间里可能发生路由选择错误。因此，在任何路由选择协议里收

敛时间是一个重要的因素。在拓扑发生变化之后，一个网络的收敛速度越快越好。

4.1.4 负载均衡

回忆一下第 3 章的内容，为了有效地使用带宽，负载均衡作为一种手段，将流量分配到相同目标网络的多条路径上。如图 4-1 所示，再让我们考虑一下这个讨论负载均衡有效性的例子。图中所有网络都存在两条可达路径。如果网络 192.168.2.0 上的设备向 192.168.6.0 上的设备发送一组数据包流，路由器 A 可以经过路由器 B 或路由器 C 发送这些数据包。在这两种情况下，到目的网络的距离都是 1 跳。然而，在一条路径上发送所有的数据包不能最有效地利用可用带宽；因此应该执行负载均衡交替使用两条路径。正如第 3 章所述，负载均衡可以是等代价或不等代价，基于数据包或基于目标地址的。

4.2 距离矢量路由选择协议

大多数路由选择协议都属于如下两类的其中之一：距离矢量（distance ventor）和链路状态（link state）。这里首先对距离矢量路由选择协议的基础内容进行分析，在下一节中将讨论链路状态路由选择协议。大多数距离矢量算法是以 R.E.Bellman[1]、L.R.Ford 和 D.R.Fulkerson[2]所做的工作为基础的，由于这个原因，所以有时距离矢量算法又称为 *Bellman-Ford* 或 *Ford-Fulkerson* 算法。但值得注意的是 EIGRP 是一个例外情况，它是基于 J.J.Garcia Luna Aceves 开发的算法实现的。

距离矢量名称的由来是因为路由是以矢量（距离，方向）的方式被通告出去的，其中距离是根据度量定义的，方向是根据下一跳路由器定义的。例如，“目标 A 在下一跳路由器 X 的方向，距此 5 跳之远”。这个表述隐含了每台路由器向邻接路由器学习它们所观察到的路由信息，然后再向外通告自己观察到的路由信息。因为每台路由器在信息上都依赖于邻接路由器，而邻接路由器又从它们的邻接路由器哪里学习路由，依次类推，所以距离矢量路由选择有时又被认为是“依照传闻进行路由选择”。

下面都属于距离矢量路由选择协议：

- IP 路由选择信息协议（RIP）；
- Xerox 网络系统的 XNS RIP；
- Novell 的 IPX RIP；
- Cisco Systems 的 Internet 网关路由选择协议（IGRP）和增强型 Internet 网关路由协议（EIGRP）；
- DEC 的 DNA 阶段 4；
- Apple Talk 的路由选择表维护协议（RTMP）。

4.2.1 通用属性

典型的距离矢量路由选择协议通常会使用一个路由选择算法，算法中路由器通过广播整

[1] R.E.bellmain。*Dynamic Programming*。普林斯顿，新泽西：普林斯顿大学出版；1957。
[2] L.R.Ford Jr.和 D.R.Fulerson。*Flows in Networks*。普林斯顿，新泽西：普林斯顿大学出版；1962。

个路由表，定期地向所有邻居发送路由更新信息。[1]

上面这个表述包含了大量信息，下面将更加详细地讨论这些内容。

1. 定期更新（Periodic Updates）

定期更新意味着每经过特定时间周期就要发送更新信息。这个时间周期从 10s（AppleTalk 的 RTMP）到 90s（Cisco 的 IGRP）。这里有争议的是如果更新信息发送过于频繁可能会引起拥塞；但如果更新信息发送不频繁，收敛时间可能长的不能被接收。

2. 邻居（Neighbours）

在路由器上下文中，邻居通常意味着共享相同数据链路的路由器或某种更高层的逻辑邻接关系。距离矢量路由选择协议向邻接路由器 [2]发送更新信息，并依靠邻居再向它的邻居传递更新信息。因此，距离矢量路由选择被说成使用逐跳更新方式。

3. 广播更新（Broadcast Updates）

当路由器首次在网络上被激活时，路由器怎样寻找其他路由器呢？它将如何宣布自己的存在呢？这里有几种方法可以采用。最简单的方法是向广播地址发送（在 IP 网络中，广播地址是 255.255.255.255）更新信息。使用相同路由选择协议的邻居路由器将会收到广播数据包并且采取相应的动作。不关心路由更新信息的主机和其他设备仅仅丢弃该数据包。

4. 全路由选择表更新

大多数距离矢量路由选择协议使用非常简单的方式告诉邻居它所知道的一切，该方式就是广播它的整个路由表，但下面我们会讨论几个特例。邻居在收到这些更新信息之后，它们会收集自己需要的信息，而丢弃其他信息。

4.2.2 依照传闻进行路由选择

在图 4-3 中，正在执行一个距离矢量算法，其中使用跳数作为度量。在 t_0 时刻，路由器 A 到路由器 D 正好可用。让我们沿最上面一行查看路由表，在 t_0 时刻 4 台路由器所具有的惟一信息就是它们的直连网络。路由表标识了这些网络，并且指明它们没有经过下一跳路由器，是直接连接到路由器上的，所以跳数为 0。每台路由器都将在它所有的链路上广播这些信息。

在 t_1 时刻，路由器接收并处理第 1 个更新信息。查看此时路由器 A 的路由表，路由器 B 发给路由器 A 的更新信息发现路由器 B 能够到达网络 10.1.2.0 和 10.1.3.0，而且距离都为 0 跳。如果这些目标网络距离路由器 B 为 0 跳，那么距离路由器 A 则为 1 跳。所以路由器 A 将跳数增加 1，然后检查自己的路由表。路由表中显示网络 10.1.2.0 已知，且距离为 0 跳，小于路由器 B 通告的跳数，所以路由器 A 忽略此信息。

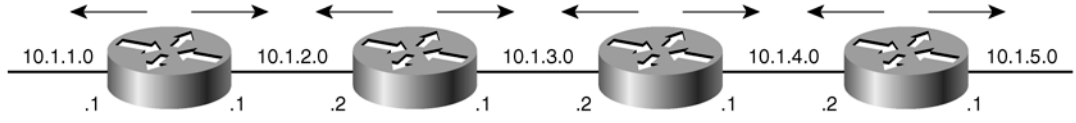
由于网络 10.1.3.0 对路由器 A 来说是新信息，所以路由器 A 将其输入到路由表中。更新数据包的源地址是路由器 B 的接口地址（10.1.2.2），因此该地址连同计算的跳数一起也被保存到路由表中。

---

[1] Cisco 的增强型 IGRP 明显不同于这种协定。EIGRP 虽然是距离矢量协议，但是它既不定期发送更新信息，也不进行广播，而且更新信息也不是整个路由表。第 7 章将会讨论 EIGRP。

[2] 这个表述不完全正确。在某些实现中主机也监听路由更新信息；但是在这个讨论中重要的是路由器如何工作。

注意，在 t_1 时刻其他路由器也执行了类似的操作。例如，路由器 C 忽略了来自路由器 B 关于 10.1.3.0 的信息以及来自路由器 C 关于 10.1.4.0 的信息，但是保存了以下信息：经过路由器 B 的接口地址 10.1.3.1 可以到达网络 10.1.2.0 以及经过路由器 C 的接口地址 10.1.4.2 可以到达网络 10.1.5.0。经计算得知路由器 C 到达这两个网络的距离都为 1 跳。



	路由器A			路由器B			路由器C			路由器D		
t_0	网络	下一跳	跳数	网络	下一跳	跳数	网络	下一跳	跳数	网络	下一跳	跳数
	10.1.1.0	--	0	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0
	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0	10.1.5.0	--	0
t_1	网络	下一跳	跳数	网络	下一跳	跳数	网络	下一跳	跳数	网络	下一跳	跳数
	10.1.1.0	--	0	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0
	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0	10.1.5.0	--	0
	10.1.3.0	10.1.2.2	1	10.1.1.0	10.1.2.1	1	10.1.2.0	10.1.3.1	1	10.1.3.0	10.1.4.1	1
				10.1.4.0	10.1.3.2	1	10.1.5.0	10.1.4.2	1			
t_2	网络	下一跳	跳数	网络	下一跳	跳数	网络	下一跳	跳数	网络	下一跳	跳数
	10.1.1.0	--	0	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0
	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0	10.1.5.0	--	0
	10.1.3.0	10.1.2.2	1	10.1.1.0	10.1.2.1	1	10.1.2.0	10.1.3.1	1	10.1.3.0	10.1.4.1	1
	10.1.4.0	10.1.2.2	2	10.1.4.0	10.1.3.2	1	10.1.5.0	10.1.4.2	1	10.1.2.0	10.1.4.1	2
				10.1.5.0	10.1.3.2	2	10.1.1.0	10.1.3.1	2			
t_3	网络	下一跳	跳数	网络	下一跳	跳数	网络	下一跳	跳数	网络	下一跳	跳数
	10.1.1.0	--	0	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0
	10.1.2.0	--	0	10.1.3.0	--	0	10.1.4.0	--	0	10.1.5.0	--	0
	10.1.3.0	10.1.2.2	1	10.1.1.0	10.1.2.1	1	10.1.2.0	10.1.3.1	1	10.1.3.0	10.1.4.1	1
	10.1.4.0	10.1.2.2	2	10.1.4.0	10.1.3.2	1	10.1.5.0	10.1.4.2	1	10.1.2.0	10.1.4.1	2
	10.1.5.0	10.1.2.2	3	10.1.5.0	10.1.3.2	2	10.1.1.0	10.1.3.1	2	10.1.1.0	10.1.4.1	3

图 4-3　距离矢量协议逐跳收敛

在 t_2 时刻，随着更新周期再次到期，另一组更新消息被广播。路由器 B 发送了最新的路由表；路由器 A 再次将路由器 B 通告的跳数加 1 后与自己的路由表比较。像上次一样，路由器 A 又一次丢弃了关于 10.1.2.0 的信息。由于网络 10.1.3.0 已知且跳数没有发生变化，所以该信息也被丢弃。惟有 10.1.4.0 被作为新的信息输入到路由表中。

在 t_3 时刻，网络已收敛。每台路由器都已经知道了每个网络以及到达每个网络的下一跳路由器的地址和距离跳数。

这里打个比方。你正在新墨西哥洲北部的 Sangre de Cristo 山中漫步，如果你不会迷路的话，这里是一个迷人的漫步场所。但是如果你迷路了且遇到一个叉路口，一个路标指向西面，上面写着“陶斯镇，15 英里”。这时你除了相信这个路标外别无选择。你不知道 15 英里外的地形是什么，你也不知道是否有更好的路，或者这个路标是否正确。如果有人将路标转个方向，那么你不但不能去往安全的地方反而走向森林的更深处！

距离矢量算法提供了指向网络的路标。[1]该算法给出了方向和距离，但是没有给出沿着这条路径行走的细节。就像叉路口的路标一样，它很容易受到意外或故意的误导。下面是距离矢量算法所面临的一些困难及算法的改进。

[1] 一般是拿路标做比喻。你可以在 Radia Perlman 的 *Interconnections* 一书中找到一个好的表达，参见 205 页～210 页。

4.2.3 路由失效计时器

在图 4-3 中，网络已经收敛，那么当部分网络拓扑发生变化时，它怎样处理重新收敛问题呢？如果网络 10.1.5.0 发生故障，答案很简单——在下一个更新周期中，路由器 D 将这个网络标记为不可达并且发送该信息。

但是如果网络 10.1.5.0 没有故障，而是路由器 D 发生故障该怎么办？这时路由器 A、路由器 B 和路由器 C 的路由表中仍然保存着关于网络 10.1.5.0 的信息，虽然该信息不再有用，但是却没有路由器通知它们。它们将不知不觉地向一个不可达的网络转发着数据包——即在网络中打开了一个黑洞。

处理这个问题的办法是为路由表中的每个表项设置路由失效计时器。例如，当路由器 C 首次知道 10.1.5.0 并将其输入到路由表中时，路由器 C 将为该路由设置计时器。每隔一定时间间隔路由器 C 都会收到路由器 D 的更新信息，路由器 C 在丢弃有关 10.1.5.0 的信息的同时复位该路由的计时器。

如果路由器 D 发生故障，路由器 C 将不能接收到关于 10.1.5.0 的更新信息。这时计时器将会超时，路由器 C 将把该路由标记为不可达，并将在下一个更新周期时传递该信息。

路由超时的典型周期范围是 3~6 个更新周期。路由器在丢失单个更新信息之后将不会使路由无效的，因为数据包的损坏、丢失或者某种网络延时都会造成这种事件的发生。但是，如果路由失效周期太长，网络收敛速度将会过慢。

4.2.4 水平分隔

根据到目前为止所描述的距离矢量算法，每台路由器在每个更新周期都要向每个邻居发送它的整个路由表。但是这真的有必要吗？在图 4-3 中，路由器 A 知道的每个距离大于 0 跳的网络都是从路由器 B 学习来的。常识表明，如果路由器 A 将学自路由器 B 的网络再广播给路由器 B，那么这是一种资源浪费。显然，路由器 B 已经知道这些网络。

路由的指向与数据包流动方向相反的路由被称为逆向路由（reverse route）。水平分隔（split horizon）是一种在两台路由器之间阻止逆向路由的技术。

这样做除了不会浪费资源，还有一个很重要的原因是不会把从路由器学习到的可达性信息再返回给这台路由器。动态路由选择协议最重要的功能是监测和补偿拓扑变化——如果到网络的最优路径不可用，协议必须寻找下一个最优路径。

再看一下图 4-3 中已收敛的网络，假设网络 10.1.5.0 发生故障。路由器 D 监测到该故障，将网络标记为不可达并在下一更新周期通知路由器 C。然而在路由器 D 的更新计时器触发更新之前，意想不到的事情发生了。路由器 C 的更新消息到达了路由器 D，声明路由器 C 可以到达网络 10.1.5.0，距离为 1 跳！还记得上面路标的比喻吗？路由器 D 不知道路由器 C 通告的下一条最优路径并不合理，因而路由器 D 将跳数加 1 并在路由表中记录以下信息：通过路由器 C 的接口（10.1.4.1）可以到达网络 10.1.5.0，距离为 2 跳。

此时目标地址为 10.1.5.3 的数据包到达路由器 C，路由器 C 查询路由表并将数据包转发给路由器 D。路由器 D 查询路由表又将数据包转发给路由器 C，路由器 C 再转回给路由器 D，一直无穷尽地进行下去，因而导致路由环路的发生。

执行水平分隔可以阻止路由环路的发生。有两类水平分隔方法：简单水平分隔法和毒性逆转水平分隔法。

简单水平分隔的规则是，从某接口发送的更新消息不能包含从该接口收到的更新所包含的网络。

在图 4-4 中，路由器执行简单的水平分隔。路由器 C 向路由器 D 发送了关于网络 10.1.1.0、10.1.2.0 和 10.1.3.0 的更新信息。其中没有包含网络 10.1.4.0 和 10.1.5.0，因为它们是从路由器 D 获取的。同样的，发送给路由器 B 的更新信息包括了网络 10.1.4.0 和 10.1.5.0，而没有提及 10.1.1.0、10.1.2.0 和 10.1.3.0。

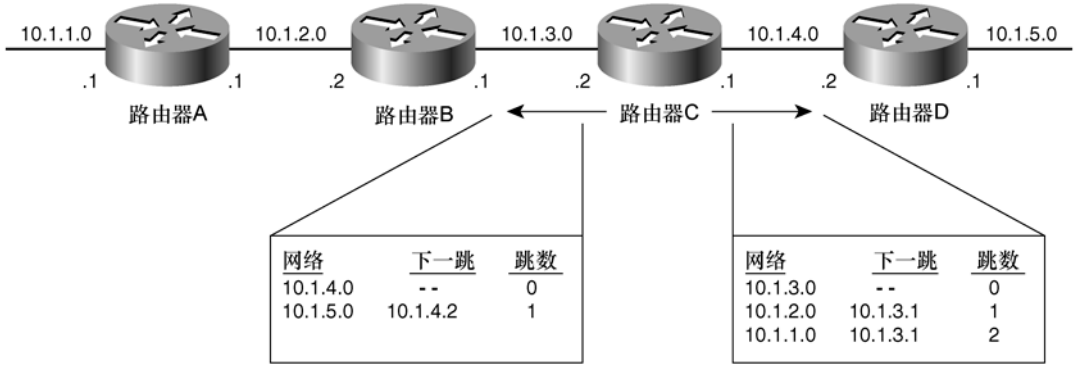


图 4-4　简单水平分隔没有把从邻居那里获取的路由通告给邻居

简单水平分隔采用抑制信息的工作方式。毒性逆转水平分隔法是一种改进方法，它可以提供更积极的信息。

毒性逆转水平分隔法的规则是，当更新信息被发送出某接口时，信息中将指定从该接口收到的更新信息中获取的网络是不可达的。

在图 4-4 中，事实上，路由器 C 向路由器 D 通告了网络 10.1.4.0 和 10.1.5.0，但是这些网络都被标记为不可达。图 4-5 给出了从路由器 C 到路由器 B 和路由器 D 的路由表，看上去很相似。注意，通过设置度量为无穷大可以标记网络为不可达；换言之，网络无穷远。下一小节将讨论路由选择协议中无穷大的概念。

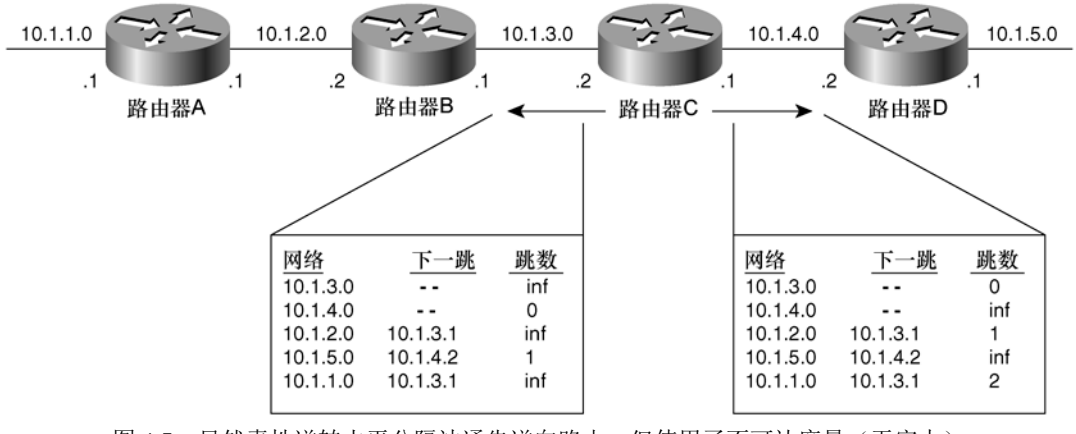


图 4-5　虽然毒性逆转水平分隔法通告逆向路由，但使用了不可达度量（无穷大）

毒性逆转水平分隔法被认为比简单水平分隔法更安全更健壮——一种“坏信息总比没消息好”的方法。例如在图 4-5 中，假设路由器 B 收到错误信息使其相信经过路由器 C 可以到

达子网 10.1.1.0。简单水平分隔法无法纠正这种错误，而路由器 C 的毒性逆转更新信息可以立刻制止这种潜在的环路。正因如此，大部分现代距离矢量算法的实现都使用了毒性逆转水平分隔法。缺点是使路由更新数据包更大了，可能会加剧链路的拥塞问题。

4.2.5 计数到无穷大

水平分隔法切断了邻居路由器之间的环路，但是它不能割断网络中的环路，如图 4-6 所示，这里还是 10.1.5.0 发生故障。路由器 D 向路由器 C（虚箭头）和路由器 B（实箭头）发送了相应的更新信息。于是路由器 B 将经过路由器 D 的路由标记为不可达，而此时路由器 A 正在向外通告到达 10.1.5.0 的次最优路径，距离为 3 跳；因此路由器 B 在路由表中记录下此路由。

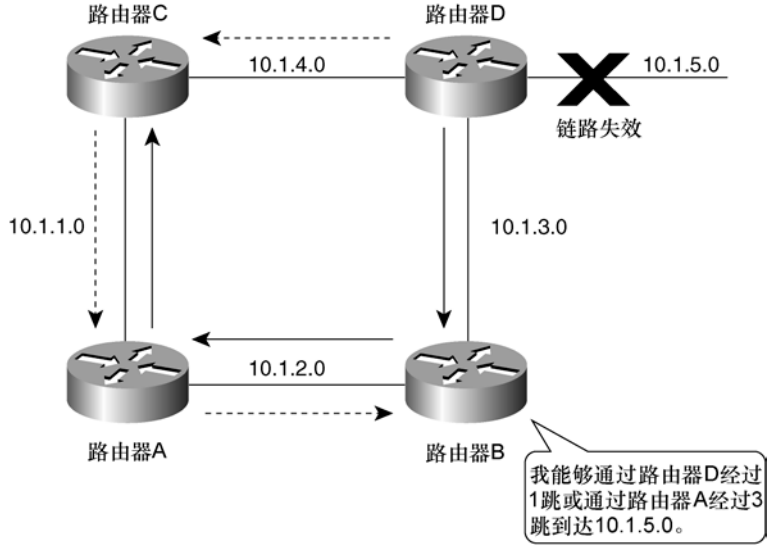


图 4-6 水平分隔法不能阻止这里的路由环路

路由器 B 现在又通知路由器 D 它有另一条路由可以到达 10.1.5.0。于是路由器 D 也记录下这个路由，并通知路由器 C 它有一条距离 10.1.5.0 为 4 跳的路由。路由器 C 又告诉路由器 A 距离 10.1.5.0 有 5 跳远。路由器 A 告诉路由器 B 有 6 跳远。

“啊”，路由器 B 想，“虽然路由器 A 到网络 10.1.5.0 的路径在不断加长。不过它是惟一可用的路径，所以我将使用这条路径!”

路由器 B 又将跳数加到 7，并通知路由器 D，如此循环下去。这种情况就叫计数到无穷大，因为到 10.1.5.0 的跳数将会持续增加到无穷大。虽然所有路由器都执行了水平分隔，但对此无能为力。

减轻计数到无穷大影响的方法是定义无穷大。大多数距离矢量协议定义无穷大为 16 跳。在图 4-6 中，随着更新消息在路由器中转圈，到 10.1.5.0 的跳数最终将增加到 16。那时网络 10.1.5.0 将被认为不可达。

这也是路由器如何通告网络不可达的一种方法。一个网络发生故障，不管它是毒性逆转路由，还是超过最大网络尺寸 15 跳的路由，路由器将把所有跳数为 16 的路由看作不可达。

设置最大跳数 15 有助于解决计数到无穷大的问题，但是收敛速度仍旧非常慢。假设更新周期为 30s，网络可能花 7.5min 达到收敛，在这期间容易受到路由错误的影响。触发更新可以用于减少收敛时间。

4.2.6 触发更新

触发更新（Triggered Update）又叫快速更新，非常简单：如果一个度量变好或变坏，那么路由器将立即发送更新信息，而不等更新计时器超时。这样重新收敛的速度将会比每台路由器必须等待更新周期的方式快，而且可以大大减少计数到无穷大所引发的问题，虽然不能完全消除。定期更新和触发更新可能会一起发生，因而路由器可能会在收到来自触发更新的正确信息之后又收到来自未收敛路由器的错误信息。这种情况表明，当网络正在进行重新收敛时，还会发生混乱和路由错误。但是触发更新将有助于更快地消除这些问题。

对触发更新进一步的改进是更新信息中仅包括实际触发该事件的网络，而不是包括整个路由表。触发更新技术减少了处理时间和对网络带宽的影响。

4.2.7 抑制计时器

触发更新为正在重新进行收敛的网络增加了应变能力。为了降低接受错误路由选择信息的可能性，抑制计时器（Holddown Timer）引入了某种程度的怀疑量。

如果到一个目标的距离增加（例如，跳数由 2 增加到 4），那么路由器将为该路由设置抑制计时器。直到计时器超时，路由器才可以接受有关此路由的更新信息。

显然，这也是一种折衷办法。错误路由选择信息进入路由表的可能性被减小了，但是重新收敛的时间也被耗费了。像其他计时器一样，必须小心设置抑制计时器。如果挂起时间太短，不起作用；如果太长，正常路由则会受到不利的影响。

4.2.8 异步更新

图 4-7 给出了一组连接在以太网骨干上的路由器。路由器将不会同时广播更新信息，如果发生这种情况，更新数据包会发生碰撞。但是当几台路由器共享一个广播网络时可能会发生这种情况。因为在路由器中，更新处理所带来的系统时延将导致更新计时器趋于同步。当几台路由器的计时器同步后，碰撞随之发生，这又进一步影响到系统时延，最终共享广播网络的所有路由器都可能变得同步起来。

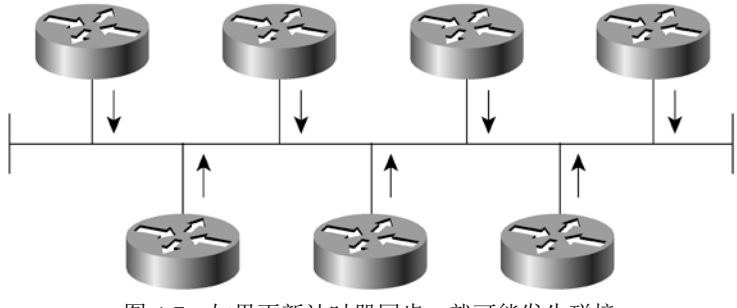


图 4-7 如果更新计时器同步，就可能发生碰撞

可以使用下面两种方法维持异步更新（Asynchronous Update）：

- 每台路由器的更新计时器独立于路由选择进程，因而不会受到路由器处理负载的影响。
- 在每个更新周期中加入一个小的随机时间或定时抖动作为偏移。

如果路由器实现了严格的、独立于系统的计时器方法，那么还必须按照随机的方式激活共享一个广播网络的所有路由器。因为并行启动整组路由器——比如发生大面积断电——可能造成所有计时器同时发送更新信息。

如果随机变量与共享广播网络的路由器数量相比足够大，那么增加更新周期的随机性是有效的。Sally Floyd 和 Van Jacobson[1]曾作过计算，在足够大的网络中，过小的随机化会被路由器所克服，为了保证有效性，更新计时器应该分布在中等更新周期的 50%的范围内。

4.3 链路状态路由选择协议

距离矢量路由器所使用的信息可以比拟为由路标提供的信息。链路状态路由选择协议像是一张公路线路图。链路状态路由器是不容易被欺骗而作出错误的路由决策的，因为它有一张完整的网络图。链路状态不同于距离矢量依照传闻进行路由选择的工作方式，原因是链路状态路由器从对等路由器 [2]那里获取第一手信息。每台路由器会产生一些关于自己、本地直连链路、这些链路的状态（以此而得名）和所有直接相连邻居的信息。这些信息从一台路由器传送到另一台路由器，每台路由器都做一份信息拷贝，但是决不改动信息。最终目的是每台路由器都有一个相同的有关网络的信息，并且每台路由器可以独立地计算各自的最优路径。

链路状态协议，有时叫*最短路径优先协议*或*分布式数据库协议*，是围绕着图论中的一个著名算法——E.W.Dijkstra 的最短路径算法设计的。链路状态协议有以下几种：

- IP 开放式最短路径优先（OSPF）；
- CLNS 或 IP ISO 的中间系统到中间系统（IS-IS）；
- DEC 的 DNA 阶段 5；
- Novell 的 NetWare 链路服务协议（NLSP）。

虽然链路状态协议确实考虑的比距离矢量协议更复杂，但是基本功能却一点也不复杂。

步骤 1：每台路由器与它的邻居之间建立联系，这种联系叫做邻接关系。

步骤 2：每台路由器向每个邻居 [3]发送被称为链路状态通告（LSA）的数据单元。对每台路由器链路都会生成一个 LSA，LSA 用于标识这条链路、链路状态、路由器接口到链路的代价度量值以及链路所连接的所有邻居。每个邻居在收到通告之后将依次向它的邻居转发（泛洪）这些通告。

步骤 3：每台路由器要在数据库中保存一份它所收到的 LSA 的备份。如果所有工作正常，所有路由器的数据库应该相同。

步骤 4：完整的拓扑数据库，也叫链路状态库，Dijkstra 算法使用它对网络图进行计算得出到每台路由器的最短路径。接着链路状态协议对链路状态数据库进行查询找到每台路由器所连接的子网，并把这些信息输入到路由表中。

4.3.1 邻居

邻居发现是建立链路状态环境并运转的第一步。与友邻术语一样，这一步将使用 Hello

[1] S.Floyd 和 V.Jacobson。“周期性路由选择消息的同步。”ACM Sigcomm’93 Symposium， 1993 年 9 月。

[2] 就是所有的路由器使用相同的路由选择协议。

[3] LSA 是一个 OSPF 术语，相应由 IS-IS 建立的数据单元叫链路状态 PDU（LSP）。但对于常规讨论，LSA 更适合我们的需要。

协议（Hello Protocol）。Hello 协议定义了一个 Hello 数据包的格式和交换数据包并处理数据包信息的过程。

Hello 数据包至少应包含一个路由器 ID CRID 和发送数据包的网络地址。路由器 ID 可以将发送该数据包的路由器与其他路由器惟一地区分开。例如，路由器 ID 可以是路由器的一个接口的 IP 地址。数据包的其他字段可以携带子网掩码、Hello 间隔、线路类型描述符和帮助建立邻居关系的标记，其中 Hello 间隔是路由器在宣布邻居死亡之前等待的最大周期。

当两台路由器已经互相发现并将对方视为邻居时，它们要进行数据库同步过程，即交换和确认数据库信息直到数据库相同为止。数据库同步的细节参见第 8 章和第 9 章的内容。为了执行数据库同步，邻居之间必须建立邻接关系，即它们必须就某些特定的协议参数，如计时器和对可选择能力的支持，达成一致意见。通过使用 Hello 数据包建立邻接关系，链路状态协议就可以在受控的方式下交换信息。与距离矢量相比，这种方式仅在配置了路由选择协议的接口上广播更新信息。

除建立邻接关系之外，Hello 数据包还可作为监视邻接关系的握手信号。如果在特定的时间内没有从邻接路由器收到 Hello 数据包，那么认为邻居路由器不可达，随即邻接关系被解除。典型的 Hello 数据包交换间隔为 10s，典型的死亡周期是数据包交换间隔的 4 倍。

4.3.2 链路状态泛洪扩散

在建立邻接关系之后，路由器开始发送 LSA。 从术语泛洪扩散（Flooding）我们可以得知，通告被发送给每个邻居。路由器保存接收到的 LSA，并依次向每个邻居转发，除了发送该 LSA 的邻居之外。这个过程是链路状态优于距离矢量的一个原因。LSA 几乎是立刻被转发，而距离矢量在发送路由更新之前必须运行算法并更新自身的路由表，甚至对触发路由更新也是如此。因此，当网络拓扑改变时，链路状态协议收敛速度远远快于距离矢量协议。

泛洪扩散过程是链路状态协议中最复杂的一部分。有几种方式可以使泛洪扩散更高效和更可靠，如使用单播和组播地址、校验和以及主动确认。在有关具体协议的章节中将讨论这些主题，但是有两个过程对泛洪扩散是极其重要的：排序和老化。

1. 序列号

到目前为止，泛洪扩散的一个难点是当所有路由器收到所有 LSA 时，泛洪扩散必须停止。我们可以希望数据包中的 TTL 值终止过期的数据包，但是之前几乎不可能有效地控制 LSA 在网络中漫游。如图 4-8 所示，路由器 A 连接的子网 172.22.4.0 发生故障，因而路由器 A 向邻居路由器 B 和路由器 D 泛洪扩散一个 LSA，以便通告该链路的新状态。路由器 B 和路由器 D 忠实地向它们的邻居扩散该 SLA，依次类推。

看一下在路由器 C 上会发生什么。在 t_1 时刻，一个来自路由器 B 的 LSA 到达路由器 C，路由器 C 将相关信息输入到拓扑数据库，并且向路由器 F 进行转发。在 t_3 时刻，另一份相同的 LSA 拷贝经 A-D-E-F-C 路径到达路由器 C 。路由器 C 发现数据库中已经存在该 LSA，问题是路由器 C 是否应该向路由器 B 转发这个 LSA？答案是不转发，因为路由器 B 已经收到了这个通告。由于路由器 C 从路由器 F 接收到的 LSA 的序列号与早先从路由器 B 接受的 LSA 的序列号相同，所以路由器 C 也知道这一情况。

当路由器 A 发送 LSA 时，在每个拷贝中的序列号都是一样的。这个序列号和 LSA 的其

他部分一起被保存在路由器的拓扑数据库中，当路由器收到数据库中已存在的 LSA 且序列号相同时，路由器将丢弃这些信息。如果信息相同但是序列号更大，那么接收的信息和新序列号被保存到数据库中，并且泛洪扩散该 LSA。按照这种方式，当所有路由器都收到 LSA 的最新拷贝时，泛洪扩散将停止。

到目前为止，根据所描述的，路由器好像仅对数据库中 LSA 与新收到的 LSA 是否相同进行验证，并据此作出扩散/丢弃决定，并没有使用序列号。但是假设图 4-8 中的网络 172.22.4.0 在发生故障后马上恢复正常。路由器 A 可能会发出通告网络故障的 LSA，序列号为 166，接着再发送通告网络正常的 LSA，序列号为 167。路由器 C 先后接收到沿路径 A-B-C 扩散过来的 LSA，分别是关于网络发生故障和故障恢复的通告，但是接着路由器 C 又收到沿路径 A-D-E-F-C 扩散过来的关于网络故障的 LSA。没有序列号，路由器 C 将不知道是否应该相信最后到达的关于网络故障的 LSA。而使用序列号，路由器 C 的数据库可以显示来自路由器 A 的 LSA 的序列号为 167，而后面到达的 LSA 序列号为 166，因此该 LSA 被认为是过时的信息而被丢弃。

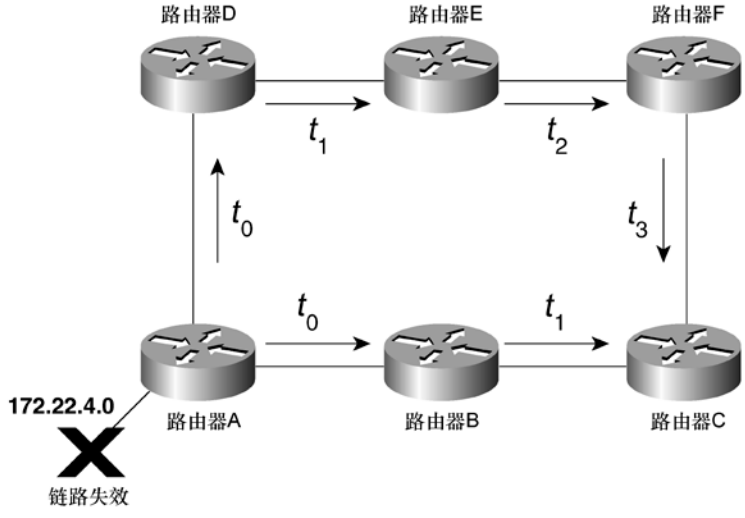


图 4-8 当拓扑发生变化时，通告该变化的 LSA 在整个网络上扩散

因为序列号被携带在 LSA 中的一个固定字段内，所以序列号一定有上限，那么当序列号到达上限时会发生什么呢？

（1）线性序列号空间

一种办法是使用一个非常大的线性序列号空间以至于根本不可能到达上限。例如，如果使用 32 位长字段，那么从 0 开始将有 2^{32}＝4 294 967 296 个可用序列号。即使路由器每 10s 产生一个新的状态数据包，它也需要花 1361 年才能用尽所有序列号。几乎没有路由器被希望持续这样长的时间。

很不幸，在这个不够完美的世界上，故障依然会发生。如果一个链路状态路由选择进程用完了所有序列号，那么它在重新使用最低序列号之前必须停止，并等待它所发出的 LSA 在所有数据库中都不再被使用（参见本章“老化（Aging）”部分的内容）。

在路由器启动期间会出现一种更常见的困难。如果路由器 A 重启后，它无法记得上次使用的序列号，必须重新使用 1。但是如果路由器 A 的邻居仍然在数据库中保留了路由器 A 上次的序列号，那么越小的序列号也就是越旧的序列号，因而会被忽略。而且，路由选择进程必须一直等到网络上所有陈旧的 LSA 都消失为止。假如最大的时间是 1 个小时或更长，那么

这种方法没有任何吸引力。

更好的解决办法是在泛洪扩散行为中添加新的规则：如果一台重新启动的路由器向邻居发送 LSA 的序列号比邻居保存的序列号还要老，那么邻居会发回自己保存的 LSA 和序列号。这台路由器将知道启动前自己使用的序列号并作出相应的调整。

然而需要小心，最近使用过的序列号不能接近上限。否则，重新启动的路由器将不得不再次重新启动。必须设定规则限制路由器“跳跃”地使用序列号。例如，规则指明一次序列号的增加不能超过整个序列号空间的二分之一（实际公式比例子要复杂，因为要考虑年龄的限制）。

IS-IS 使用 32 位线性序列号空间。

（2）循环序列号空间

另一种方法是使用循环序列号空间，数字是循环使用的——在 32 位空间内紧跟在 4 294 967 295 后面的是 0。然而在这里的故障也会让人左右为难，重新启动的路由器可能会遇到同线性序列号一样的问题。

循环序列编号建立了一个不合逻辑的奇特的位。如果 x 是 1 到 4 294 967 295 之间任意的一个数，那么 $0 < x < 0$。在运行正常的网络中通过声明两条规则可以维持这种条件，其中声明的规则用来确定什么时候一个序列号大于或小于另一个序列号。假设序列号空间为 n，有两个序列号 a 和 b，如果满足以下任意一种条件，则认为 a 更新（数量更大）：

- $a > b$ 且（$a - b$）$\leq n/2$
- $a < b$ 且（$b - a$）$> n/2$

为了简单起见，如图 4-9 所示，我们使用一个 6 位序列号空间：

$n = 2^6 = 64$，所以 $n/2 = 32$。

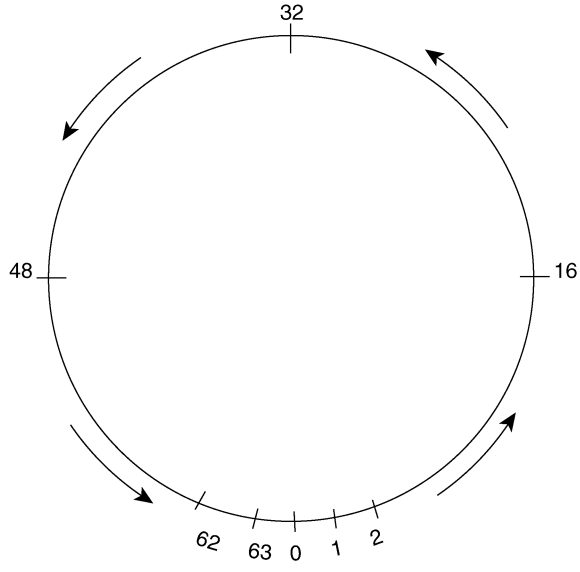


图 4-9 6 位循环地址空间

假设给定两个序列号 48 和 18，由规则 1 可得 48 较新：

$48 > 18$ 且（$48 - 18$）$= 30$，且 $30 < 32$。

假设给定两个序列号 3 和 48，由规则 2 可得 3 较新：

$3 < 48$ 且（$48 - 3$）$= 45$，且 $45 > 32$。

假设给定两个序列号 3 和 18，由规则 1 可得 18 较新：

$18 > 3$ 且（$18-3$）＝15，且 $15 < 32$。

可以看出规则强制序列号循环使用。

但是在运行状况不是很正常的网络上又会如何？设想在一个网络中使用 6 位序列号空间。现在其中的一台路由器决定离线，当它在离线之时，突然发送了 3 个相同且序列号为 44（101100）的 LSA。不幸的是，一个邻居也发生了故障——丢掉了几位数据，丢失的数据位是第 2 个 LSA 和第 3 个 LSA 序列号中的 1 位。随即该邻居路由器向外泛洪扩散了这 3 个 LSA。结果造成 3 个 LSA 序列号各不相同。

44	（101100）
40	（101000）
8	（001000）

应用循环规则可得 44 比 40 更新，40 比 8 更新，8 又比 44 更新！这个结论将使 3 个 LSA 都持续扩散下去，数据库也不断地被最新的 LSA 更新，直到数据库缓冲被塞满，CPU 超载为止，最终整个网络崩溃。

这一连锁事件听上去好像有些牵强，然而它确实是真实的。现代网络的先驱 ARPANET 早期曾经使用过带有 6 位序列号空间的链路状态协议。在 1980 年 10 月 27 号，两台路由器发生了上面所说的故障，从而造成 ARPANET 的停顿。[1]

（3）棒棒糖形序列号空间

这个古怪名称的结构是由 Radia Perlman 博士提出的。[2] 棒棒糖形序列号空间是线性序列号空间和循环序列号空间的综合；如果你考虑一下，你会发现棒棒糖形序列号空间有一个线性组件和一个圆形组件。圆形空间的缺点是不存在一个数小于其他所有的数。线性空间的缺点是不能循环使用序列号，即序列号是有限的。

当路由器 A 重新启动时，它将从小于其他所有数的 a 开始。邻居将会识别出这个数，这时如果邻居数据库中还保留有上次序列号为 b 的 LSA，那么它们会发送 b 给路由器 A，则路由器 A 将跳至该序列号。在知道自己重启前使用的序列号之前，路由器 A 可能会发送不止一个 LSA。因此，在邻居通知路由器 A 上次使用的序列号或上次使用的序列号从所有数据库中消失之前，必须准备充足的启动数以便路由器 A 不会用光所有序列号。

这些线性重启序列号形成了棒棒糖的棒。在这些数用完或邻居提供了使路由器 A 跳转的序列号之后，路由器 A 进入循环数空间，也就是棒棒糖的糖体部分。

在设计棒棒糖地址空间时使用了有符号数，即$-k < 0 < k$。从$-k$ 到 1 的负数形成棒棒，从 0 到 k 的正数形成循环空间。Perlman 的序列号规则如下。假设给出两个数 a、b 及一个序列号空间 n，当且仅当：

① $a < 0$ 且 $a <$ b，或

② $a > 0$，$a < b$，且（$b - a$）$< n/2$ 或

③ $a > 0$，$b > 0$，$a > b$，且（$a - b$）$> n/2$。

认为 b 比 a 更新。

图 4-10 给出了棒棒糖形序列号空间的一个实现。使用 32 位有符号数 N 可以产生 2^{31} 个正数和 2^{31} 个负数。$-N$（-2^{31} 或 0x80000000）和 $N-1$（$2^{31}-1$ 或 0x7FFFFFFF）没有被使

[1] E.C.Rose。“网络控制协议的脆弱性：一个例子。”计算机通信回顾，1981 年 7 月。

[2] R.Perlman。“路由选择信息的容错广播。”计算机网络，第 7 卷，1983 年 12 月，395～405 页。

用。路由器在线时将从−N＋1（0x80000001）开始使用序列号，一直增加到 0，这时将进入循环数空间。当序列号到达 N−2（0x7FFFFFFE）时，序列号将返回到 0（不使用 N−1）。

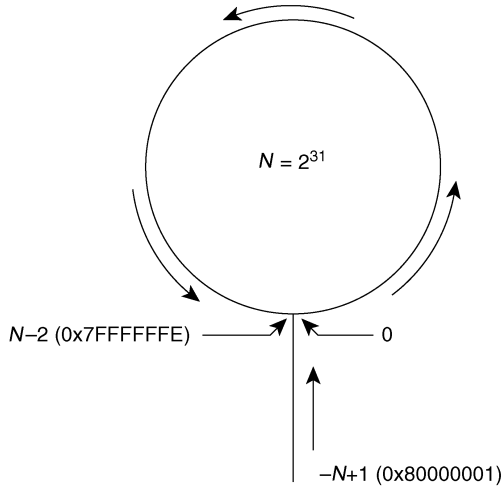


图 4-10 棒棒糖形序列号空间

假设路由器再次重新启动，最后一个被发送的 LSA 序列号是 0x00005de3（循环序列号空间的一部分）。路由器重启后，当它与邻居同步数据库时，路由器发送序列号为 0x80000001（−N＋1）的 LSA。邻居检查它的数据库发现该路由器重启前 LSA 的序列号为 0x00005de3，这时邻居将把这个 LSA 发给重新启动的路由器，实际上是说，“这是你离开时的位置。”接着重启路由器记录下这个序列号为正数的 LSA。如果下一时刻路由器需要发送新的 LSA，它将使用序列号 0x00005de6。

棒棒糖形序列号空间是用在 OSPF 的最初版本 OSPFv1（RFC1131）中的。虽然使用有符号数是对线性数空间的一个改进，但是发现棒棒糖形序列号空间的循环部分与纯粹的循环空间同样容易出现歧义。OSPFv1 的部署一直处于试验阶段。当前使用的 OSPF 版本 OSPFv2（最初见 RFC 1247）采用了线性和棒棒糖形序列号空间的最好的特性。OSPFv2 像棒棒糖形序列号数一样使用了从 0x80000001 开始的有符号数空间。但是当序列号数变为正数时，序列号空间持续保持线性直至到达最大数 0x7FFFFFFF 为止。此刻，在重启之前 OSPF 进程必须从所有链路状态数据库中删除 LSA。

2. 老化（Aging）

LSA 格式中将要包含一个用于通告年龄的字段。当 LSA 被创建时，路由器将该字段设置为 0。随着数据包的扩散，每台路由器都会增加通告中的年龄。[1]

老化过程为泛洪扩散过程增加了另一层可靠性，该协议为网络定义了一个最大年龄差距（MaxAgeDiff）值。路由器可能接收到一个 LSA 的多个拷贝，其中序列号相同，年龄不同。如果年龄的差距小于 MaxAgeDiff，那么认为是由于网络的正常时延造成了年龄的差异，因此数据库原有的 LSA 继续保存，新收到的 LSA（年龄更大）不被扩散。如果年龄差距超过 MaxAgeDiff，那么认为网络发生异常，因为新被发送的 LSA 的序列号值没有增加。在这种情况下，较新的 LSA 被记录下来，并将数据包扩散出去。典型的 MaxAgeDiff 值为 15min（用于 OSPF）。

[1] 当然，另一个选项是从某个最大年龄开始，然后递减。OSPF 是递增；IS-IS 是递减。

若 LSA 驻留在数据库中，则 LSA 的年龄会不断增加。如果链路状态记录的年龄增加到某个最大年龄值（MaxAge）——由特定的路由选择协议定义——那么一个带有 MaxAge 值的 LSA 被泛洪扩散到所有邻居，邻居随即从数据库中删除相关记录。

当 LSA 的年龄到达 MaxAge 时，将被从所有的数据库中删除，这需要有一种机制来定期地确认 LSA 并且在达到最大年龄之前将它的计时器复位。链路状态刷新计时器（LSRefeshTime）就是做此用途的；[1] 一旦计时器超时，路由器将向所有邻居泛洪扩散新的 LSA，收到的邻居会把有关路由器记录的年龄设置为新接收到的年龄。OSPF 定义 MaxAge 为 1 小时，LSRefreshTime 为 30min。

4.3.3 链路状态数据库

除了泛洪扩散 LSA 和发现邻居， 链路状态路由选择协议的第 3 个主要任务是建立链路状态数据库。链路状态或拓扑数据库把 LSA 作为一连串记录保存下来。虽然 LSA 还包括序列号、年龄和其他信息，但这些变量主要用于管理泛洪扩散进程。对于最短路径的决策进程来说，通告路由器的 ID、连接的网络和邻居路由器以及与网络和邻居相关联的代价是非常重要的信息。正如前面句子所隐含的，LSA 还包括两类通用信息：[2]

- **路由器链路信息**——使用三元组（路由器 ID、邻居 ID、代价）通告路由器的邻居路由器，这里的代价是指链路到邻居的代价。
- **末梢网络信息**——使用三元组（路由器 ID、网络 ID、代价）通告路由器直接连接的末梢网络（没有邻居的网络）。

最短路径优先（SPF）算法对路由器链路信息进行一次计算以建立到每台路由器的最短路径，然后使用末梢网络信息向路由器添加网络。图 4-11 给出的网络包括路由器及路由器之间的链路，为简单起见没有给出末梢网络。注意，在几条链路两端的代价各不相同，因为代价与接口的出站方向有关。例如，从 RB 到 RC 的链路代价为 1，但是对相同的链路，从 RC 到 RB 方向的代价却为 5。

对于图 4-11 的网络，表 4-2 给出了一个通用的链路状态数据库，在每台路由器中都保存了一份拷贝。当你阅读这个数据库时，你将会发现数据库完整地描述了网络。现在通过运行 SPF 算法可以计算出一棵到达每台路由器的最短路径树。

表 4-2　　关于图 4-11 中网络的拓扑数据库

路由器 ID	邻居	代价
RA	RB	2
RA	RD	4
RA	RE	4
RB	RA	2
RB	RC	1
RB	RE	10
RC	RB	5
RC	RF	2
RD	RA	4
RD	RE	3
RD	RG	5

[1] LSRefreshTime、MaxAge 和 MaxAgeDiff 是 OSPF 架构中的常量。

[2] 实际上，信息的种类不止两种，而且还包括多种链路状态数据包类型。这些在具体路由选择协议的章节中会提到。

续表

路由器 ID	邻居	代价
RE	RA	5
RE	RB	2
RE	RD	3
RE	RF	2
RE	RG	1
RE	RH	8
RF	RC	2
RF	RE	2
RF	RH	4
RG	RD	5
RG	RE	1
RH	RE	8
RH	RF	6

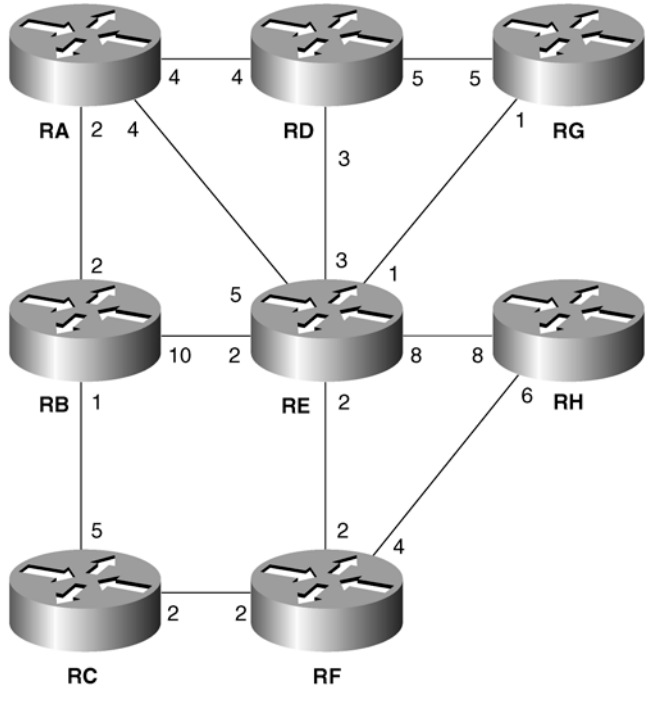


图 4-11 链路代价是按照出站接口的方向计算的，并且在一个网络中所有接口的代价没有必要完全相同

4.3.4 SPF 算法

在路由选择领域里，很不幸的是 Dijkstra 算法常常被认为是最短路径优先算法。毕竟每个路由选择协议的目标都是计算最短路径。另一个不幸的是 Dijkstra 算法常常被描述的比实际复杂得多，因为许多作者都使用集合论符号讨论它。最清晰的描述来自 E.W.Dijkstra 的原稿。这里将使用他的原话，并插入针对链路状态路由选择协议的解释：

构造一棵树[a]，使 *n* 个节点之间的总长最小（树是一个在每两个节点之间仅有一条路径

的图）。

在我们给出的构造过程中，分枝被分成 3 个集合:

Ⅰ．被明确分配给构造中的树的分枝（它们将在子树中）;

Ⅱ．这个分枝的隔壁分枝被添加到集合Ⅰ;

Ⅲ．剩余的分枝（抛弃或不考虑）。

节点被分成 2 个集合:

A. 被集合Ⅰ中的分枝连接的节点;

B. 剩余的节点（集合Ⅱ中有且仅有一个分枝将指向这些节点中的每一个节点）。

下面我们开始构造树，首先选择任意一个节点作为集合 A 中仅有的成员，并将所有拿这个节点做端点的分枝放入集合Ⅱ中。开始集合Ⅰ是空的。然后我们重复执行下面两步。

步骤 1: 集合Ⅱ中最短的分枝被移出并加入集合Ⅰ。结果，一个节点被从集合 B 传送到集合 A。

步骤 2: 考虑从这个节点（刚才被传送到集合 A 中的节点）通向集合 B 中节点的分枝。如果构建中的分枝长于集合Ⅱ中相应的分枝，那么分枝被丢弃; 否则，用它替代集合Ⅱ中相应的分枝，并且丢弃后者。

接着我们回到第 1 步并重复此过程直到集合Ⅱ和集合 B 为空。集合Ⅰ中的分枝形成所要求的树。[1]

配合路由器的算法，第一点注意的是，Dijkstra 描述了 3 个分枝集合：Ⅰ、Ⅱ和Ⅲ。在路由器中，使用 3 个数据库表示 3 个集合：

- **树数据库**——树数据库用来表示集合Ⅰ。通过向数据库中添加分枝实现向最短路径树中添加链路（分枝）。当算法完成时，这个数据库将可以描述最短路径树。
- **候选对象数据库**——候选对象数据库对应集合Ⅱ。按照规定的顺序从链路状态数据库向该列表中复制链路，作为向树中添加的候选对象。
- **链路状态数据库**——按照前面的描述，这里保存所有链路，这个拓扑数据库对应集合Ⅲ。

Dijkstra 还指定了两个节点集合——A 和 B。这里的节点是路由器。这些路由器被明确地用路由器链路三元组（路由器 ID、邻居 ID、代价）中的邻居 ID 表示。集合 A 由树数据库中链路所连接的路由器组成。集合 B 是所有其他的路由器。由于全部要点是发现到每台路由器的最短路径，所以当算法结束时集合 B 应该为空。

下面是一台路由器中采用的 Dijkstra 算法版本：

步骤 1：路由器初始化树数据库，将自己作为树的根。这表明路由器作为它自己的邻居，代价为 0。

步骤 2：在链路状态数据库中，所有描述通向根路由器邻居链路的三元组被添加到候选对象数据库中。

步骤 3：计算从根到每条链路的代价，候选对象数据库中代价最小的链路被移到树数据库中。如果两个或更多的链路离根的最短代价相同，选择其中一条。

步骤 4：检查添加到树数据库中的邻居 ID。除了邻居 ID 已在树数据库中的三元组之外，链路状态数据库中描述路由器邻居的三元组被添加到候选对象数据库中。

步骤 5：如果候选对象数据库中还有剩余的表项，回到第 3 步。如果候选数据库为空，

[1] E.W.Dijkstra。“关于连通图中两个问题的注释。”Numerische Mathematik。第一卷，1959 年，269～271 页。

那么终止算法。在算法终止时，在树数据库中，每个单一的邻居 ID 表项将表示 1 台路由器，到此最短路径树构造完毕。

表 4-3 总结了应用 Dijkstra 算法为图 4-11 中的网络构建最短路径树的过程和结果。路由器 RA 正在运行算法，并使用表 4-2 中的链路状态数据库。图 4-12 给出了通过该算法为路由器 RA 构造的最短路径树。在每台路由器完成自己最短路径树的计算之后，它会检查其他路由器的网络链路信息，并且相当容易地把末梢网络添加到树中。根据这些信息，表项可以被制作为路由表。

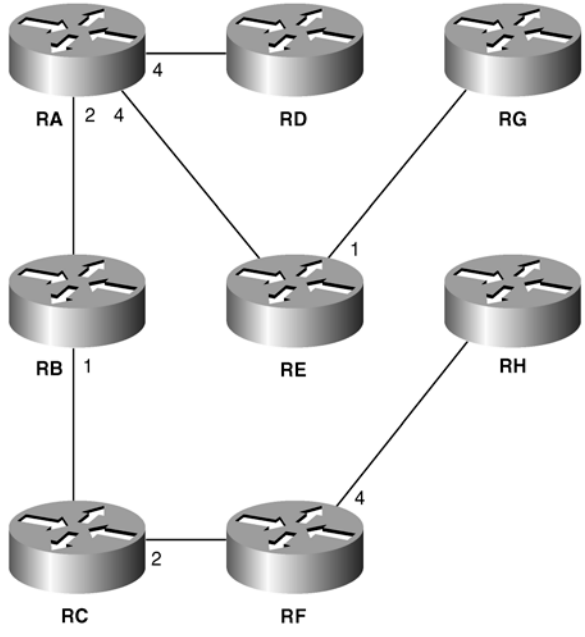


图 4-12 用表 4-3 中的算法得到的最短路径树

表 4-3 对表 4-2 的数据库应用 Dijkstra 算法

候选对象	到根的代价	树	描述
		RA，RA，0	路由器 A 把自己作为树的根
RA，RB，2 RA，RD，4 RA，RE，4	2 4 4	RA，RA，0	到所有 RA 邻居的链路被添加到候选对象列表
RA，RD，4 RA，RE，4 RB，RC，1 ~~RB，RE，10~~	4 4 3 12	RA，RA，0 RA，RB，2	（RA，RB，2）是候选列表中代价最小的链路，所以被添加到树中。所有 RB 的邻居除了已在树中的都被添加到候选列表。（RA，RE，4）到 RE 的代价比（RB，RE，10）小，所以后者被从候选列表中丢弃
RA，RD，4 RA，RE，4 RC，RF，2	4 4 5	RA，RA，0 RA，RB，2 RB，RC，1	（RB，RC，1）是候选列表中代价最小的链路，所以被添加进树中。所以 RC 的邻居除了已在树中的都将变为候选对象
RA，RE，4 RC，RF，2 RD，RE，3 RD，RG，5	4 5 7 9	RA，RA，0 RA，RB，2 RB，RC，1 RA，RD，4	（RA，RD，4）和（RA，RE，4）离 RA 的代价都为 4；（RC，RF，2）代价为 5。（RA，RD，4）被添加到树中，并且它的邻居成为候选对象。在候选列表中有两条路径到 RE，从 RA 出发（RD，RE，3）因代价更高而被丢弃

续表

候选对象	到根的代价	树	描述
RC，RF，2 ~~RD，RG，5~~ RE，RF，2 RE，RG，1 RE，RH，8	5 9 6 5 12	RA，RA，0 RA，RB，2 RB，RC，1 RA，RD，4 RA，RE，4	（RF，RE，1）被添加到树中，所有不在树中的 RE 邻居都被添加进候选列表。到 RG 代价最高的链路被丢弃
RE，RF，2 RE，RG，1 ~~RE，RH，8~~ RF，RH，4	6 5 12 9	RA，RA，0 RA，RB，2 RB，RC，1 RA，RD，4 RA，RE，4 RC，RF，2	（RC，RF，2）被添加进树中并且它的邻居被添加进候选列表。由于从 RA 出发（RE，RG，1）代价相同（5），所以使用（RE，RG，1）替代。到 RH 代价更高的路径被丢弃
RF，RH，4	9	RA，RA，0 RA，RB，2 RB，RC，1 RA，RD，4 RA，RE，4 RC，RF，2 RE，RG，1	（RE，RG，1）被添加到树中。RG 所有邻居都在树中，所以没有对象被添加到候选列表中
		RA，RA，0 RA，RB，2 RB，RC，1 RA，RD，4 RA，RE，4 RC，RF，2 RE，RG，1 RF，RH，4	（RF，RH，4）是候选列表中代价最小的链路，所以被添加到树中，候选列表中不再有候选对象，所以算法终止。最短路径树构造完毕

4.3.5 区域

一个区域是构成一个网络的路由器的一个子集。将网络划分为区域是针对链路状态协议的 3 个不利影响所采取的措施。

- 必要的数据库要求内存的数量比距离矢量协议更多。
- 复杂的算法要求 CPU 时间比距离矢量协议更多。
- 链路状态泛洪扩散数据包对可用带宽带来了不利的影响，特别是不稳定的网络。

目前设计的链路状态协议和使用该协议的路由器都能够减少这些影响，但是却没有消除这些影响。在最后一节我们在含有 8 台路由器的网络中，分析了链路状态数据库看上去是什么样，SPF 算法是如何工作的。但要记住，对于连接到 8 台路由器上的末梢网络，并由此形成的 SPF 树的叶节点，在这里没有考虑。现在我们假设一个有 8000 台路由器的网络，你就能理解对内存、CPU 和带宽所造成影响的利害关系了。

正如图 4-13 所示，通过划分区域可以减小这些影响。当一个网络被划分为多个区域时，在一个区域内的路由器仅需要在本区域扩散 LSA，因而只需要维护本区域的链路状态数据库。数据库越小，意味着需要内存越少，运行 SPF 算法需要的 CPU 周期也越少。如果拓扑改变频繁发生，引起的扩散将被限制在不稳定的区域。

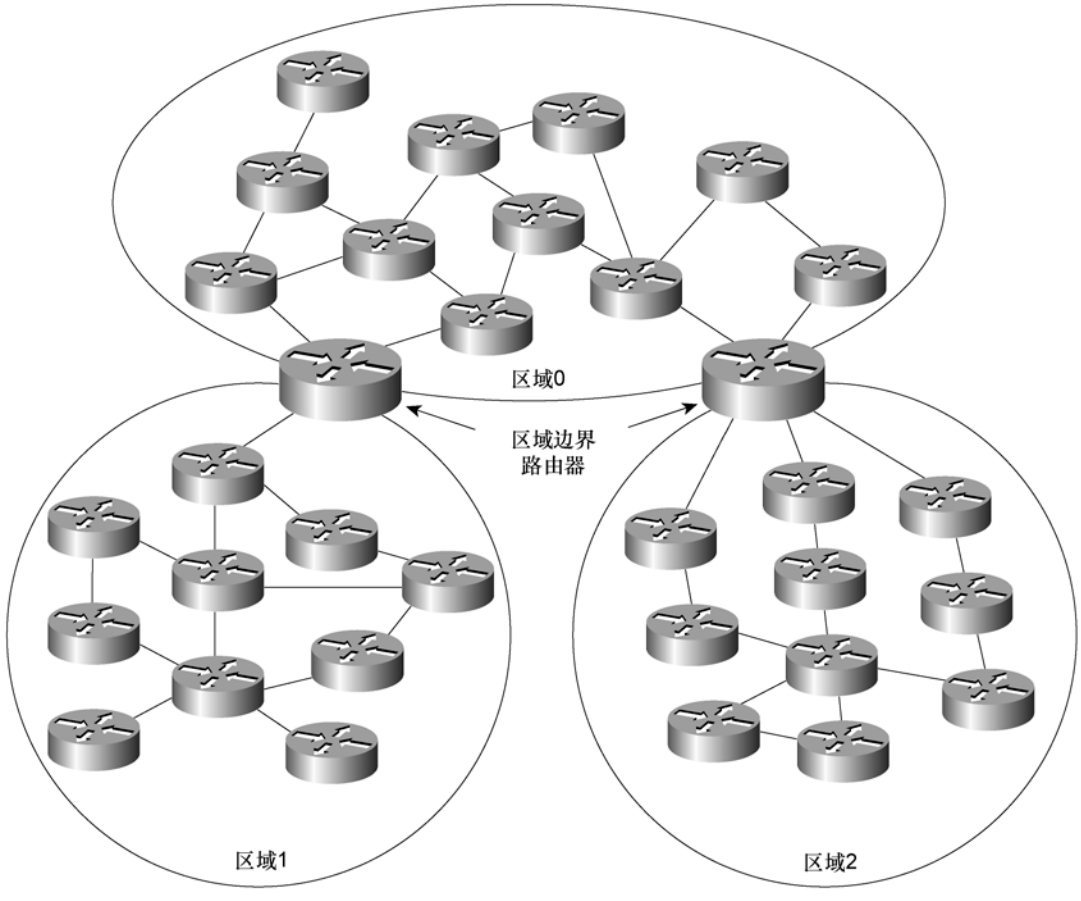


图 4-13 区域的使用减少了链路状态对系统资源的需求

区域边界路由器是连接两个区域的路由器，它属于所连接的两个区域，而且必须为每个区域维护各自的拓扑数据库。正像在网络上发送数据包到其他网络的主机仅需要知道如何找到本地路由器一样，在一个区域内想往其他区域发送数据包的路由器仅需要知道怎样找到本地区域边界路由器。换句话说，区域间路由器/区域内路由器之间的关系就如同主机/路由器之间的关系，除了所在层次更高一些。

距离矢量协议，如 RIP 和 IGRP，不使用区域。假设这些协议除了把一个大型的网络看成一个单一的实体外，没有任何依靠，那么它们就必须计算出到每个网络的路由，并且每 30s 或 90s 就必须广播这个巨大的路由表。很明显，利用区域的链路状态协议可以节省系统资源。

4.4 内部和外部网关协议

区域向网络体系结构中引入了一个层次，在此基础上，将一组区域组成一个更大的区域又向网络体系结构中引入了另一个层次。这些更高层的区域在 IP 领域内叫自主系统，在 ISO 模型中叫路由选择域。

一个自主系统被定义为在共同管理域下的一组运行相同路由选择协议的路由器。假设现

代网络的活动存在变移性，那么该定义的后半部分将不是非常准确。部门、分公司乃至整个公司常常会合并，随着它们的合并原来设计使用不同路由选择协议的网络也合并在一起。结果是，许多网络使用多种不精确的等级将多种路由选择协议结合在一起，并处于共同的管理之下。所以自主系统的当前定义应该是在共同管理下的网络。

在一个自主系统内运行的路由选择协议称为内部网关协议（IGP）。在本章的例子中出现的所有距离矢量和链路状态协议都是 IGP。

在自主系统之间或路由选择域之间的路由选择协议称为外部网关协议（EGP）。IGP 发现网络之间的路径，而 EGP 发现自主系统之间的路径。下面这些都是 EGP：

- 边界网关协议（BGP）。
- IP 外部网关协议（EGP）（是，一个名为 EGP 的 EGP）。
- ISO 的域间路由选择协议（IDRP）。

Novell 也把 EGP 功能合并到 NLSP 中，叫第 3 层路由选择。

在给出这些定义的同时，还必须要提到的是，术语自主系统（autonomous system）的通常用法并不是绝对的。各种标准文档、文献以及人们都趋向于给这个术语多种含义；其结果是，对于理解听到或读到该术语的上下文环境是十分重要的。

本书在两种上下文环境中使用自主系统：

- 如本节开始定义的，自主系统可以指路由选择域。在该上下文环境中，一个自主系统是一个有一个或多个 IGP 的系统，它完全自主于其他 IGP 系统。在这些自主系统之间使用 EGP。
- 自主系统也可以认为是一个过程域，或一个 IGP 进程，它自主于其他 IGP 进程。例如，使用 OSPF 的路由器可以被认为是 OSPF 自主系统。EIGRP 的也可以在这种上下文环境中使用自主系统。在这些自主系统之间使用了路由的重新分配。

在本书中，上下文环境将指明在不同地方所讨论的自主系统是属于哪一种形式的自主系统。

4.5 静态或动态路由选择

在阅读完动态路由选择协议的显赫细节之后，留下的印象一定是动态路由选择协议比静态路由选择协议好。动态路由选择协议的主要任务是自动监测和适应网络拓扑的变化，记住这一点很重要。但是，为实现“自动化”需要在带宽、队列空间、内存和处理时间上付出代价。

使用动态路由选择协议的另一个代价是减少了对路由选择的控制。对于一个指定的目标网络，路由选择协议可以代替我们确定最佳路径。如果需要进行精确的路由选择，特别是当你期望的路径和路由选择协议选择的不一致时，还是使用静态路由选择更好一些。

静态路由选择的最大缺陷是管理困难。这对于存在许多可选路由的中型和大型网络来说是真实的，但对于几乎没有可选路由的小型网络来说就不适用了。

如图 4-14 所示，在较小的网络中很流行星型（hub-and-spoke）拓扑。如果到路由器的一个分支发生中断，是否有另一条路由供动态路由选择协议选择？因此对于这种网络来说，十分适合使用静态路由选择协议。在中心路由器上为每个分支上的路由器配置一条静态路由，而在每台分支路由器上配置一条指向中心路由器的缺省路由，这样网络就可以工作了（缺省路由将在第 12 章中介绍）。

在设计网络时，最简单的解决方法常常是最好的办法。在确定静态路由选择协议不能满足设计要求之后，可以选择动态路由选择协议。

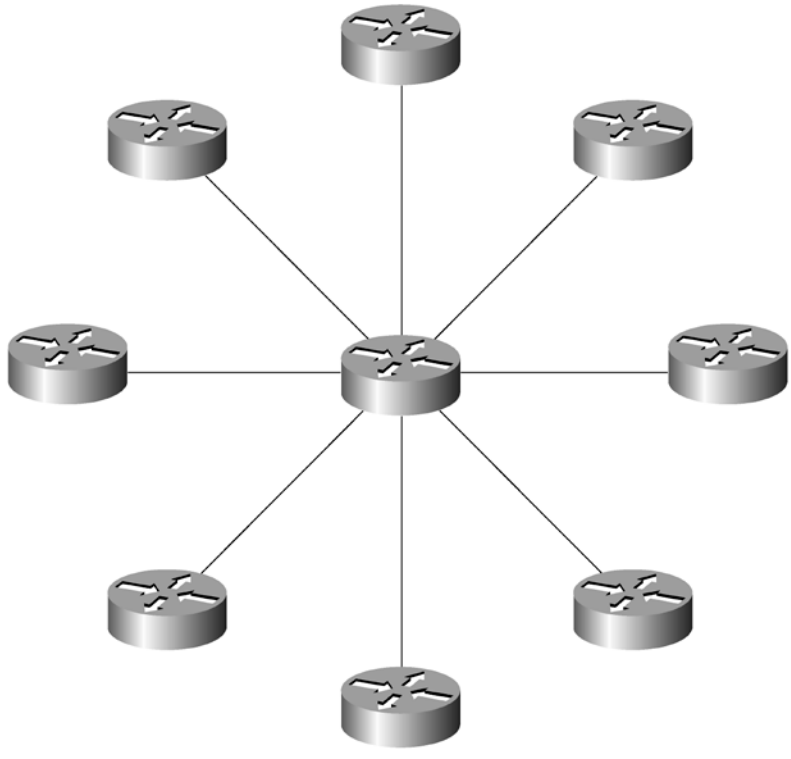


图 4-14 星型网络适合使用静态路由选择

4.6 展 望

既然动态路由选择协议的基础知识已经分析完了，现在该是讨论具体的路由选择协议的时候了。下一章是 RIP，一种最古老最简单的动态路由选择协议。

4.7 推荐读物

Perlman, R. *Interconnections: Bridges and Routers.*Reading, Massachusetts: Addison-Wesley; 1992.

4.8 复 习 题

1．什么是路由选择协议？
2．路由选择算法执行的基本过程是什么？

3．为什么路由选择协议使用度量？
4．什么是收敛时间？
5．什么是负载均衡？给出负载均衡的 4 种不同类型。
6．距离矢量路由选择协议是什么？
7．指出距离矢量协议存在的几个问题。
8．邻居是什么？
9．路由失效计时器的作用是什么？
10．解释简单水平分隔与毒性逆转水平分隔的区别？
11．什么是计数到无穷大问题？如何对其进行控制？
12．什么是抑制计时器？它们如何工作？
13．距离矢量和链路状态路由选择协议的区别是什么？
14．拓扑数据库的作用是什么？
15．解释链路状态网络中收敛所包含的基本步骤。
16．在链路状态协议中序列号为什么重要？
17．在链路状态协议中老化服务的作用是什么？
18．解释 SPF 算法是怎样工作的。
19．区域对链路状态网络的好处是什么？
20．什么是自主系统？
21．IGP 和 EGP 的区别是什么？



第二部分

内部路由选择协议

第 5 章　路由选择信息协议（RIP）

第 6 章　RIPv2、RIPng 和无类别路由选择

第 7 章　增强型内部网关路由选择协议（EIGRP）

第 8 章　开放最短路径优先协议（OSPFv2）

第 9 章　OSPFv3

第 10 章　集成 IS-IS 协议

本章包括以下主题：

- RIP 的基本原理与实现；
- 配置 RIP；
- RIP 故障诊断。

第 5 章

路由选择信息协议（RIP）

RIP 协议作为最早的距离矢量型 IP 路由选择协议依然被广泛地使用着，当前存在着两个版本。本章介绍的是 RIP 协议的第 1 版，第 6 章中包括了 RIP 协议第 2 版的内容，后者对 RIPv1 的功能作了部分增强。版本 1 和版本 2 最主要的区别是，RIPv1 是有类别路由选择协议，而 RIPv2 是无类别路由选择协议。本章介绍有类别路由选择，第 6 章介绍无类别路由选择。同时，第 6 章也介绍了 RIPng 协议，这是 RIPv2 协议为了支持 IPv6 协议而设计的修订版本。

距离矢量协议基于 Bellman[1]、Ford 和 Fulkerson[2]开发的路由选择算法，一些早期网络在 1969 年就已经实现了这个协议，例如 ARPANET 和 CYCLADES 等网络。在 20 世纪 70 年代中期，Xerox 公司开发了一种称为，PARC[3]Universal Protocol 的协议也简称为 PUP 协议，运行在它的可以说是现代以太网前身的 3Mbit/s 带宽的试验网络上。PUP 使用网关信息协议（GWINFO）来路由，并发展成为 Xerox 网络系统（XNS）协议簇。同时，GWINFO 发展成为 XNS 路由选择信息协议（XNS RIP）。XNS RIP 也随之成为一些常用的路由选择协议的前身，如 Novell 的 IPX RIP、AppleTalk 的 RTMP（路由选择表维护协议），以及 IP RIP。

1982 年，伯克利发布的 UNIX 4.2BSD 版中，通过一个称为“routed”的守护进程实现了 RIP 协议；很多后来的 UNIX 版本都是基于流行的 UNIX 4.2BSD 版本的，并且也都通过一个称为“routed”或“gated”[4]的进程支持 RIP 协议。有意思

1 R. E. Bellman. *Dynamic Programming.* Princeton, New Jersey: Princeton University Press; 1957.
2 L. R. Ford Jr. and D. R. Fulkerson. *Flows in Networks*. Princeton, New Jersey: Princeton University Press; 1962.
3 Palo Alto Research Center.
4 读作“route-dee”和“gate-dee”。

的是，直到 1988 年才发布了一个 RIP 协议的标准，那时 RIP 已经被广泛应用。这个标准就是 RFC 1058，由 Charles Hedrick 撰写，是 RIP 协议第 1 版的比较正式的标准。

由于读者阅读的文献不同，从而对 RIP 协议褒贬不一。RIP 协议虽然没有后来一些路由选择协议功能强大，但它简单易用，已被广泛的应用，这意味着 RIP 协议在实际网络的实施中碰到的兼容性问题会比较少。在小型网络数据链路中，RIP 协议设计的相当单一。在这些限定的条件下，尤其是许多 UNIX 环境下，RIP 协议依然是一个受欢迎的路由选择协议。

5.1 RIP 的基本原理与实现

RIP 协议的处理是通过 UDP 520 端口来操作的。所有的 RIP 消息都被封装在 UDP 用户数据报协议中，源和目的端口字段的值被设置为 520。RIP 定义了两种消息类型：请求消息（request messages）和响应消息（response messages）。请求消息用来向邻居路由器发送一个更新（update），响应消息用来传送路由更新。RIP 的度量是基于“跳”数（hop count）的，1 跳表示是与发出通告的路由器相直连的网络，16 跳表示网络不可到达。

开始时，RIP 从每个启用 RIP 协议的接口广播出带有请求消息的数据包。接着，RIP 程序进入一个循环状态，不断地侦听来自其他路由器的 RIP 请求或响应消息，而接收请求的邻居路由器则回送包含它们的路由表的响应消息。

当发出请求的路由器收到响应消息时，它将开始处理附加在响应消息中的路由更新信息。如果路由更新中的路由条目是新的，路由器则将新的路由连同通告路由器的地址一起加入到自己的路由表中，这里通告路由器的地址可以从更新数据包的源地址字段读取。如果网络的 RIP 路由已经在路由表中，那么只有在新的路由拥有更小的跳数时才能替换原来存在的路由条目。如果路由更新通告的跳数大于路由表已记录的跳数，并且更新来自于已记录条目的下一跳路由器，那么该路由将在一个指定的抑制时间段（holddown period）内被标记为不可到达。如果在抑制时间超时后，同一台邻居路由器仍然通告这个有较大跳数的路由，路由器则接受该路由新的度量值。[1]

5.1.1 RIP 的计时器和稳定性

路由器启动后，平均每隔 30s 从每个启动 RIP 协议的接口不断地发送响应消息。除了被水平分隔法则抑制的路由条目之外，响应消息（或称为更新消息）包含了路由器的整个路由表。这个周期性的更新由更新计时器（update timer）进行初始化，并且包含一个随机变量用来防止表的同步。[2] 结果，一个典型的 RIP 处理单个更新的时间大约是 25~35s。RIP_JITTER 是 Cisco IOS 中专有的一个随机变量，它缩短到一般更新时间的 15%（即 4.5s）。因此，Cisco 路由器的更新时间在 25.5~30s 之间变化（如图 5-1 所示）。路由更新的目的地址是到所有主机的广播地址 255.255.255.255。[3]

---

[1] Holddowns 用于 Cisco IOS, 但它不是 RFC 1058 指定的稳定性特性之一。

[2] 路由表的同步在第 4 章中讨论。

[3] RIP 的一些实现方式也可以只在广播型介质网络上广播，而在点到点的链路上直接发送给对端直连的邻居路由器。Cisco 路由器中，如果不改变配置成其他方式的话，RIP 更新将在任何类型的链路上广播。

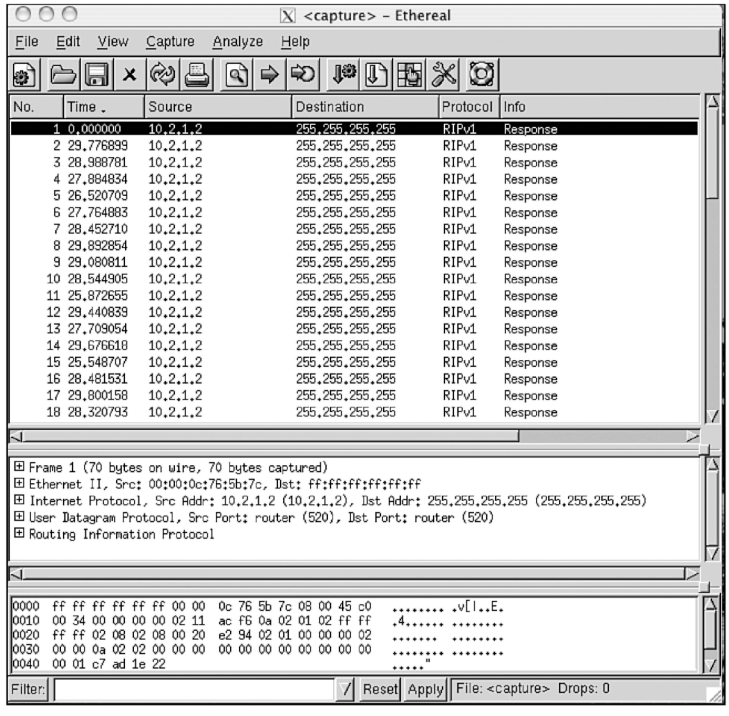


图 5-1 RIP 协议在每次重置计时器时，会增加一个小随机变量到更新计时器以避免路由表的同步。在 Cisco 路由器中，RIP 的更新时间在 25.5~30s 之间变化，从图中可以看出这些更新的增量时间

RIP 也使用一些其他的计时器。回忆一下第 4 章，距离矢量协议用到的无效计时器（invalidation timer）用来限制停留在路由表中的路由未被更新的时间。RIP 称这个计时器为限时计时器（expiration timer）或超时计时器（timeout timer）；在 Cisco IOS 中，称为无效计时器（invalid timer）。无论什么时候，当有一条新的路由被建立，超时计时器就会被初始化为 180s，而每当接收到这条路由的更新消息时，超时计时器又将被重置成计时器的初始化值，即 180s。如果一条路由的更新在 180s（6 个更新周期）内还没有收到，那么这条路由的跳数将变成 16，也就是标记为不可到达的路由。

另一种计时器，称为垃圾收集（garbage collection）或刷新计时器（flush timer），它们所设置的时间长度一般比限时计时器的时间长 240~60s。[1] 如果垃圾收集计时器也超时了，则该路由将被通告为一条度量值为不可到达的路由，同时从路由表中删除该路由项。示例 5-1 显示了路由表中有一条被标记为不可达的路由，但还没有被刷新清除。

示例 5-1 这台路由器经过 6 个更新周期的时间还没有收到关于子网 10.3.0.0 的更新。从而将这条路由标记为不可到达，但还没有被从路由表中刷新清除

```
Mayberry#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
     10.0.0.0 255.255.0.0 is subnetted, 4 subnets
```

（待续）

[1] Cisco 路由器使用 60s 的垃圾收集计时器，虽然 RFC 1058 规定为 120s。

```
C     10.2.0.0 is directly connected, Serial0
R     10.3.0.0 255.255.0.0 is possibly down,
        routing via 10.1.1.1, Ethernet0
C     10.1.0.0 is directly connected, Ethernet0
R     10.4.0.0 [120/1] via 10.2.2.2, 00:00:00, Serial0
Mayberry#
```

第 3 个计时器是抑制计时器（holddown timer）。虽然 RFC 1058 没有关于 Holddowns 的介绍，但在 Cisco 路由器中运行的 RIP 协议使用了它们。如果一条路由更新的跳数大于路由表已记录的该路由的跳数，那么将会引起该路由进入长达 180s（即 6 个路由更新周期）的抑制状态阶段。

这 4 个计时器可以通过下面的命令来操作：

```
timers basic update invalid holddown flush
```

该命令适用于 RIP 协议整个进程的运行处理。如果一台路由器的计时被改变了，那么这个 RIP 域中的所有路由器的计时都必须改变。因此，如果没有特别的原因，不应该改变这些计时器的缺省值。

RIP 使用带毒性逆转（poison reverse）的水平分隔（split horizon）和触发更新（triggered updates）。不像普通的定期更新，触发更新只要在有路由的度量值发生改变时就会产生，而且触发更新不会引起接收路由器重置它们的更新计时器；因为如果这么做的话，网络拓扑的改变会导致很多路由器在同一时间重置，从而引起定期的路由更新变得同步。为了避免拓扑改变后造成触发更新“风暴”，还需要使用另外一个计时器。当一个触发更新传播时，这个计时器被随机的设置为 1～5s 之间的数值；在这个计时器计时超时前不能发送并发的触发更新。

一些主机可以在“静”模式下使用 RIP。这些所谓的“静”主机不产生 RIP 的更新消息，而只侦听 RIP 的更新消息，从而更新它们自己的路由表。举一个例子，在一台 UNIX 主机上可以使用带“*-q*”选项的“routed”启动“静”模式下的 RIP。

5.1.2 RIP 消息格式（RIP Message Format）

RIP 的消息格式如图 5-2 所示。每条消息包含一条命令（Command）、一个版本号和路由条目（最大 25 条）。每个路由条目包括地址族标识（address family identifier）、路由可达的 IP 地址和路由的跳数。如果某台路由器必须发送大于 25 条路由的更新消息，那么必须产生多条 RIP 消息。注意，RIP 消息的开始部分（头部）占用 4 个八位组字节（octets），而每个路由条目占用 20 个八位组字节。因此，RIP 消息的大小最大为 4+（25×20）=504 个八位组字节，再加上 8 个字节的 UDP 头部，RIP 数据报的大小（不含 IP 包的头部）最大可达 512 个八位组字节。

- **命令（Command）**——取值 1 或 2，1 表示该消息是请求消息，2 表示该消息是响应消息。其他的取值都不被使用或保留用作私有用途。
- **版本号（Version）**——对于 RIPv1，该字段的值设置为 1。
- **地址族标识（Address Family Identifier，AFI）**——对于 IP 该项设置为 2。只有一个例外情况，该消息是路由器（或主机）整个路由表的请求，这将在后面的章节讨论。
- **IP 地址（IP Address）**——路由的目的地址。这一项可以是主网络地址、子网地址或主机路由地址。在 5.1.4 小节，将说明如何区分这 3 种类型的路由。
- **度量（Metric）**——正如前面章节所说，在 RIP 中指跳数，该字段的取值范围在 1～16 之间。

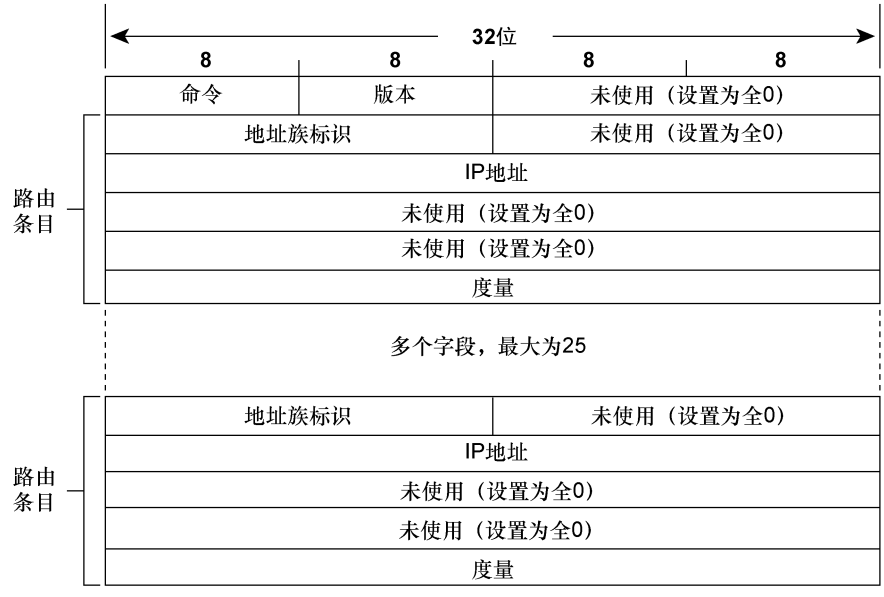


图 5-2 RIP 的消息格式

图 5-3 显示了用协议分析仪观察到的一个 RIP 消息的解码。

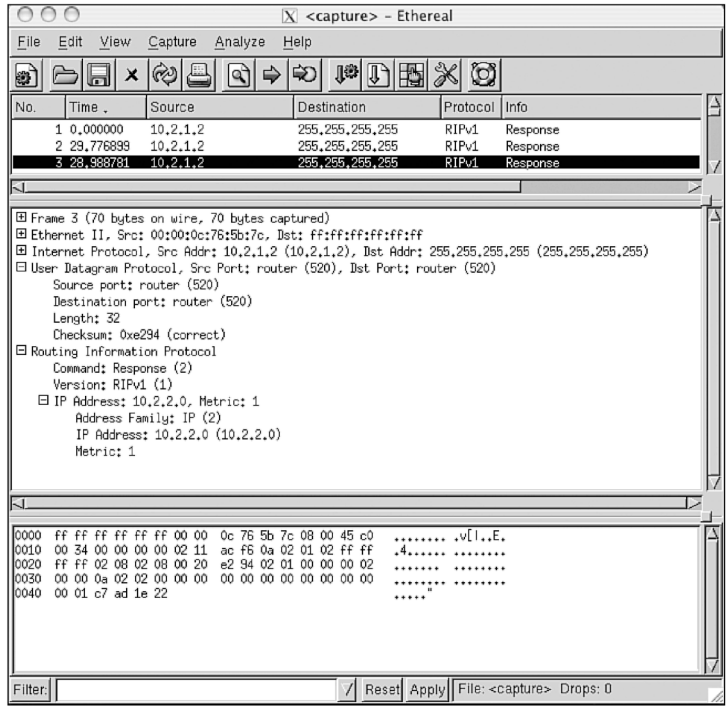


图 5-3 从协议分析仪可以看出，RIPv1 不使用子网掩码和下一跳字段。这些字段在 RIPv2 中使用，将在第 7 章中讲述

由于一些历史的影响造成 RIP 消息的格式不尽合理，消息格式中没有使用的比特空间远远大于所使用的。这些影响从 RIP 最初发展到后来就都存在，RIP 最初来自于 XNS 协议，开发人员试图使它适合更广泛的地址族；在 BSD 的影响下，它所使用的套接字（socket）地址

需要 RIP 的字段增大到 32 位字的边界长度。无论初衷如何，读者在第 6 章中将会看到这些未用的字段后来具有很大的用处。

5.1.3 请求消息类型（Request Message Type）

RIP 请求消息可以请求整个路由表信息，也可以仅请求某些具体路由的信息。就前一种情况而言，请求消息含有一个地址族标识字段为 0（地址为 0.0.0.0），度量值为 16 的单条路由，接收到这个请求的设备将通过单播方式向发出请求的地址回送它的整个路由表，并遵循一些规则如水平分隔（split horizon）和边界汇总（boundary summarization，在 5.1.4 小节中讲述）。

一些诊断测试过程可能需要知道某个或某些具体路由的信息。这种情况下，请求消息可以与特定地址的路由条目一起发送。接收到该请求的设备将根据请求消息逐个处理这些条目，构成一个响应消息。如果该设备的路由表中已有请求消息中地址相对应的路由条目，则将其路由条目的度量值填入 metric 字段。如果没有，metric 字段就被设置为 16。在不考虑水平分隔或边界汇总的情况下，响应消息将正确地告诉这台路由器了解的信息。

前面已经提到，主机可以在“静”模式下运行 RIP。这种方法允许网络中的主机不需要发送无用的 RIP 响应消息，也可以通过侦听来自路由器的 RIP 更新来确保自己的路由表保持最新。但是，网络诊断进程可能需要检查这些“静”主机的路由表，因此，RFC 1508 指出：如果一台“静”主机接收到一个来自 UDP 端口的请求消息，而不是来自标准的 RIP 520 端口，那么该主机就必须发送一个响应消息。

5.1.4 有类别路由选择（Classful Routing）

示例 5-2 中的路由表包含了由 RIP 得到的路由信息，这可以从每个路由条目左边的关键字识别出来。这些路由条目的权值由方括号中的元组来表示，与第 3 章中讨论的一样，第一个数字表示管理距离（administrative distance），第二个数表示度量值（metric）。从中很容易看出，RIP 的管理距离为 120，如前所述，RIP 的度量是基于跳数的。因此，网络 10.8.0.0 通过 E0 或 S1 需要 2 跳可到达。如果到达同一个目的网络有多条跳数相等的路由，那么 RIP 进行等价路径的负载均衡。示例 5-2 中的路由表包含了多条等价路径的路由条目。

示例 5-2 这个路由表包含了主网络 10.0.0.0 和 172.25.0.0 的子网，所有这些非直连网络都是从 RIP 协议学习得到的

```
MtPilate#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
    10.0.0.0 255.255.0.0 is subnetted, 9 subnets
R    10.10.0.0 [120/3] via 10.5.5.1, 00:00:20, Serial1
               [120/3] via 10.1.1.1, 00:00:21, Ethernet0
```

（待续）

```
R    10.11.0.0 [120/3] via 10.5.5.1, 00:00:21, Serial1
               [120/3] via 10.1.1.1, 00:00:21, Ethernet0
R    10.8.0.0 [120/2] via 10.1.1.1, 00:00:21, Ethernet0
              [120/2] via 10.5.5.1, 00:00:21, Serial1
R    10.9.0.0 [120/2] via 10.5.5.1, 00:00:21, Serial1
              [120/2] via 10.1.1.1, 00:00:21, Ethernet0
R    10.3.0.0 [120/1] via 10.1.1.1, 00:00:21, Ethernet0
              [120/1] via 10.5.5.1, 00:00:21, Serial1
C    10.1.0.0 is directly connected, Ethernet0
R    10.6.0.0 [120/1] via 10.1.1.1, 00:00:21, Ethernet0
              [120/1] via 10.5.5.1, 00:00:22, Serial1
R    10.7.0.0 [120/2] via 10.1.1.1, 00:00:22, Ethernet0
              [120/2] via 10.5.5.1, 00:00:22, Serial1
C    10.5.0.0 is directly connected, Serial1
   172.25.0.0 255.255.255.0 is subnetted, 3 subnets
R    172.25.153.0 [120/1] via 172.25.15.2, 00:00:03, Serial0
R    172.25.131.0 [120/1] via 172.25.15.2, 00:00:03, Serial0
C    172.25.15.0 is directly connected, Serial0
```

当一个数据包进入宣告 RIP 的路由器后，路由器将执行路由表的查询，逐步排除数据包的路由选择范围，直到只剩下一条惟一的路径。路由器首先读出目的地址的网络部分（基于有类别路由选择协议的主网络号），查看这个网络部分在路由表中是否有其匹配的条目。根据有类别路由表查询规则的第一步，读出基于 A 类、B 类或 C 类主网分类的网络号。如果没有匹配的主网络，这个数据包就被丢弃，同时发出一个 ICMP 目的不可达的消息给发出该数据包的源。如果存在匹配该数据包网络部分的主网络，那么路由表中会列出匹配这个主网络的子网，并进一步在这些子网中进行查询；此时，如果能找到一个匹配的子网条目，那么该数据包将可以被路由器转发，否则，该数据包将被丢弃并发出一个 ICMP 目的不可达的消息。

1. 有类别路由选择：直连的子网络

有类别路由查询可以用下面 3 个例子来表述（参见示例 5-2）：

（1）假设有一个目的地址为 192.168.35.3 的数据包进入路由器，由于该路由器在路由表中没有发现和网络 192.168.35.0 匹配的条目，因此该数据包将被丢弃。

（2）假设有一个目的地址为 172.25.33.89 数据包进入路由器，在路由表中有一个和 B 类网络 172.25.0.0/24 匹配的条目，那么进一步检查路由表中列出的网络 172.25.0.0/24 的子网条目；显然没有和网络 172.25.33.0 匹配的子网条目，因此该数据包被丢弃。

（3）最后一个例子，假设要到达地址 172.25.153.220 的数据包进入路由器，这时，路由表中有和网络 172.25.0.0/24 匹配的条目，进一步检查到有和子网 172.25.153.0 匹配的条目，因此，该数据包将被转发到下一跳地址 172.25.15.2。

另外，请注意观察图 5-2 中所显示的一个情况，RIP 协议的消息中并没有随同路由条目一起通告子网掩码，从而路由器中也没有和单独的子网相关联的掩码。因此，一台路由器的转发数据库如同示例 5-2 中所显示的那样，当它收到了一个目的地址为 172.25.131.23 的数据包，即使这个地址被完全子网化，路由器也没有确切的方法来识别子网位的结束位置和主机位的开始位置。

路由器惟一可以借助的就是，假定在整个网络中，对于同一个主网络地址使用相同的掩码，这个掩码就是配置在与网络 172.25.0.0 相连的某一台路由器接口之上的。对于从目的地

址的子网部分得出的主网络 172.25.0.0，它将使用自己的掩码。正如本章所有图示中所显示的路由表那样，如果一个网络是和路由器直连的，那么路由器将在路由表中作为一个标题条目列出该网络和该网络所连接的接口的子网掩码，然后列出它所知道的关于这个网络的所有子网。如果一个网络和路由器不是直接相连的，那么路由表将仅仅列出这个网络的主网络而不列出与它相关联的掩码。

因为在有类别路由选择协议进行路由选择的情况下，数据包的目的地址是通过在路由器接口本地配置的子网掩码来识别的，所以在同一个主网络范围中的所有子网掩码应该是一致的。

2. 有类别路由选择：在边界路由器上的路由汇总

在前面的讨论中产生了这样一个问题，就是当一个网络没有和路由器的任何接口相连接时，RIP 协议应该如何识别一个主网络的子网呢？如果没有一个接口和该目的地址对应的 A 类、B 类或 C 类主网络相关联，那么路由器将没有办法正确地识别出所使用的子网掩码，也没有办法正确地标识该子网。

解决方法很简单：如果路由器没有和某个目的网络直接连接，那么该路由器仅仅需要一条简单的路由指向一个直接相连的路由器。

图 5-4 显示了一台处于两个主网络边界上的路由器，这两个主网络是 A 类网络 10.0.0.0 和 C 类网络 192.168.115.0。这台“边界”路由器不会把其中一个主网络的子网的具体信息发送给另一个主网。正如所显示的那样，该路由器执行了自动路由汇总，或称为子网屏蔽（subnet hiding），仅把地址 10.0.0.0 通告给网络 192.168.115.0，同样的会把地址 192.168.115.0 通告给网络 10.0.0.0。

根据这种方式，在网络 192.168.115.0 内的路由器的路由表中，只包含一个单独的路由条目用来引导将到达目的网络 10.0.0.0 的数据包转发到边界路由器，而边界路由器则具有和网络 10.0.0.0 直连的接口，因此它具有一个带子网掩码的子网来为在网络“云”内的数据包选路。示例 5-3 显示了一台在网络 192.168.115.0 中的路由器的路由表，在路由表中可以看到有一条网络 10.0.0.0 的路由条目，它看起来像是一条单独的、没有子网掩码的路由条目。

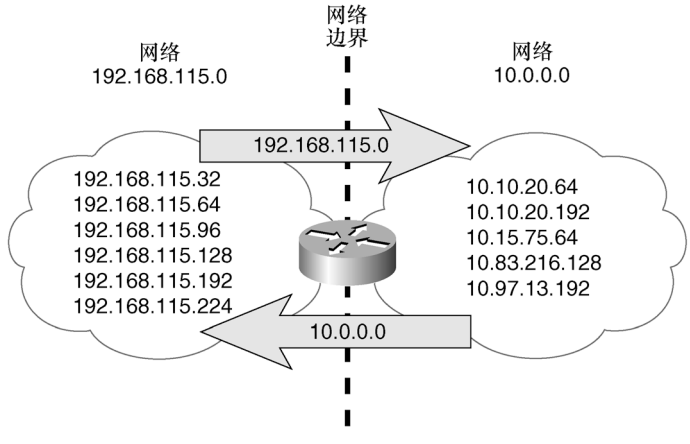


图 5-4　处在两个主网络边界上的路由器不会把其中一个主网络的子网通告给另一个主网

示例 5-3 这台路由器有一个单独的路由条目指向网络 10.0.0.0，该网络可以经过 1 跳到达，所以它的下一跳地址就是边界路由器

```
Raleigh#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
R    10.0.0.0 [120/1] via 192.168.115.40, 00:00:10, Ethernet1
     192.168.115.0 255.255.255.240 is subnetted, 6 subnets
C       192.168.115.32 is directly connected, Ethernet1
R       192.168.115.64 [120/1] via 192.168.115.99, 00:00:13, Ethernet0
C       192.168.115.96 is directly connected, Ethernet0
C       192.168.115.128 is directly connected, Serial0
R       192.168.115.192 [120/1] via 192.168.115.99, 00:00:13, Ethernet0
R       192.168.115.224 [120/1] via 192.168.115.130, 00:00:25, Serial0
Raleigh#
```

第 3 章简要讨论了一个主网络被另一个不同的主网络分割成不连续的子网的情况。这里要注意，这种情况在像 RIP、IGRP 等有类别路由协议中会带来一些问题，不连续的子网在网络边界会被自动汇总。5.2 节中对这类问题和及其解决办法进行了描述。

3. 有类别路由选择：小结

有类别路由选择协议的一个基本特征是，在通告目的地址时不能随之一起通告它的地址掩码。因此，有类别路由选择协议首先必须匹配一个与该目的地址对应于 A 类、B 类或 C 类的主网络号。对于每一个通过这台路由器的数据包：

（1）如果目的地址是一个和路由器直接相连的主网络的成员，那么该网络的路由器接口上配置的子网掩码将被用来确定目的地址的子网。因此，在该主网络中必须自始至终地统一使用这个相同的子网掩码。

（2）如果目的地址不是一个和路由器直接相连的主网络的成员，那么路由器将尝试去匹配该目的地址对应于 A 类、B 类或 C 类的主网络号。

5.2 配置 RIP

与 RIP 协议简易的特点相对应的是，RIP 协议的配置工作也是一项比较简单的事情。首先，用一条命令来启动 RIP 进程，另外还有一条命令用来指定运行 RIP 协议的每一个网络。在此之后，就只有很少的几个配置选项了。

5.2.1 案例研究：一种基本的 RIP 配置

配置一个 RIP 协议，只需两步：

步骤 1：使用 **router rip** 命令启动 RIP 进程。

步骤 2：使用 **network** 命令指定每一个需要运行 RIP 协议的主网络。

图 5-5 显示了一个含有 4 台路由器的网络，它包含 4 个主网络号。路由器 Goober 和网络 172.17.0.0 的两个子网相连。

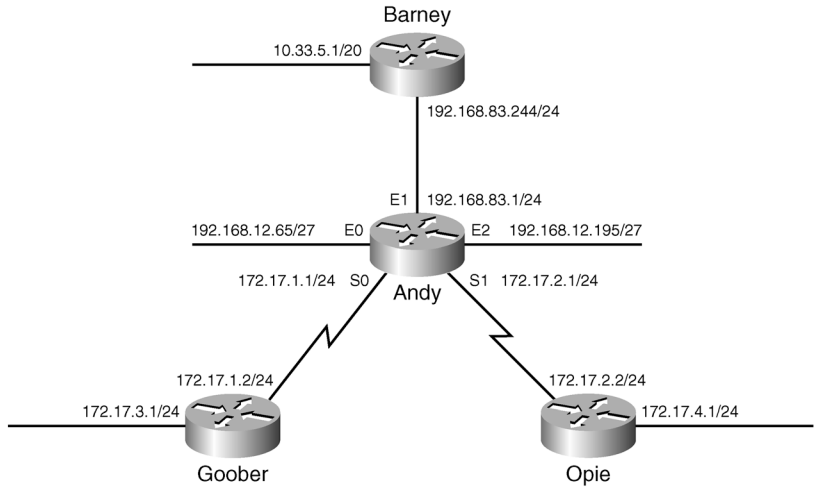


图 5-5 在按照主网络分类的层面上，路由器 Andy 和 Barney 都是网络之间的边界路由器

在示例 5-4 中显示了启动 RIP 协议所必需的命令。

示例 5-4 路由器 Goober 的 RIP 配置

```
router rip
 network 172.17.0.0
```

类似的，路由器 Opie 和同一个网络 172.17.0.0 的两个子网相连，相应的配置命令参见示例 5-5。

示例 5-5 路由器 Opie 的 RIP 配置

```
router rip
 network 172.17.0.0
```

使用任何一个 **router** 命令都需要让路由器进入到 **config-router** 配置模式，这可以通过提示符辨别出来。由于 RIP 协议具有有类别路由选择的特性，从而在网络边界上会出现子网屏蔽的情形，这意味着 **network** 命令中不需要指定子网，而仅仅需要指定相对应的 A 类、B 类或 C 类的主网络地址。任何一个接口，只要它的配置地址属于 **network** 命令指定的网络，都将会运行 RIP。

路由器 Barney 和两个主网络——10.0.0.0 和 192.168.83.0 相连。因此，这两个主网络都需要被指定，参见示例 5-6。

示例 5-6 路由器 Barney 的 RIP 配置

```
router rip
 network 10.0.0.0
 network 192.168.83.0
```

路由器 Andy 和网络 192.168.83.0 的一个子网相连，和网络 192.168.12.0 的两个子网相连，和网络 172.17.0.0 的两个子网相连。示例 5-7 显示了它们的配置。

示例 5-7 路由器 Andy 的 RIP 配置

```
router rip
 network 172.17.0.0
 network 192.168.12.0
 network 192.168.83.0
```

在示例 5-8 中，在路由器 Andy 上打开了 RIP 协议的调试命令 **debug ip rip**。这里要特别注意的是，路由器 Andy 执行了子网屏蔽。由于 E0 和 E2 接口都是和网络 192.168.12.0 相连的，因而子网 192.168.12.64 和 192.168.12.192 可以在这两个接口上进行通告；但是在和其他不同的网络相连的 E1、S0 和 S1 接口上，这两个子网则被进行路由汇总后才通告出去。同样的，网络 192.168.83.0 和网络 172.17.0.0 在通过有类别网络边界时也会被路由汇总后通告出去。注意，路由器 Andy 正在接收一条来自路由器 Barnney 关于网络 10.0.0.0 的汇总路由。最后，从示例中也可以观察到水平分隔的情况。例如，从 E1 接口通告给路由器 Barney 的路由中不再包含网络 10.0.0.0 或 192.168.83.0 的路由条目。

示例 5-8　这些 debug 消息显示了路由器 Andy 上接收和发送的 RIP 更新消息，并可以观察到网络路由的汇总和水平分隔的效果

```
Andy#debug ip rip
RIP protocol debugging is on
Andy#
RIP: sending v1 update to 255.255.255.255 via Ethernet0 (192.168.12.65)
RIP: build update entries
     subnet 192.168.12.192, metric 1
     network 10.0.0.0, metric 2
     network 192.168.83.0, metric 1
     network 172.17.0.0, metric 1
RIP: sending v1 update to 255.255.255.255 via Ethernet1 (192.168.83.1)
RIP: build update entries
     network 192.168.12.0, metric 1
     network 172.17.0.0, metric 1
RIP: sending v1 update to 255.255.255.255 via Ethernet2 (192.168.12.195)
RIP: build update entries
     subnet 192.168.12.64, metric 1
     network 10.0.0.0, metric 2
     network 192.168.83.0, metric 1
     network 172.17.0.0, metric 1
RIP: sending v1 update to 255.255.255.255 via Serial0 (172.17.1.1)
RIP: build update entries
     subnet 172.17.4.0, metric 2
     subnet 172.17.2.0, metric 1
     network 10.0.0.0, metric 2
     network 192.168.83.0, metric 1
     network 192.168.12.0, metric 1
RIP: sending v1 update to 255.255.255.255 via Serial1 (172.17.2.1)
RIP: build update entries
     subnet 172.17.1.0, metric 1
     subnet 172.17.3.0, metric 2
     network 10.0.0.0, metric 2
     network 192.168.83.0, metric 1
     network 192.168.12.0, metric 1
RIP: received v1 update from 172.17.1.2 on Serial0
     172.17.3.0 in 1 hops
RIP: received v1 update from 192.168.83.244 on Ethernet1
     10.0.0.0 in 1 hops
RIP: received v1 update from 172.17.2.2 on Serial1
     172.17.4.0 in 1 hops
```

5.2.2　案例研究：被动接口（Passive Interface）

在图 5-6 所示的网络中增加路由器 Floyd，但不希望在路由器 Floyd 和 Andy 之间交换 RIP 协议的通告消息，这在路由器 Floyd 上可以很容易地实现，参见示例 5-9。

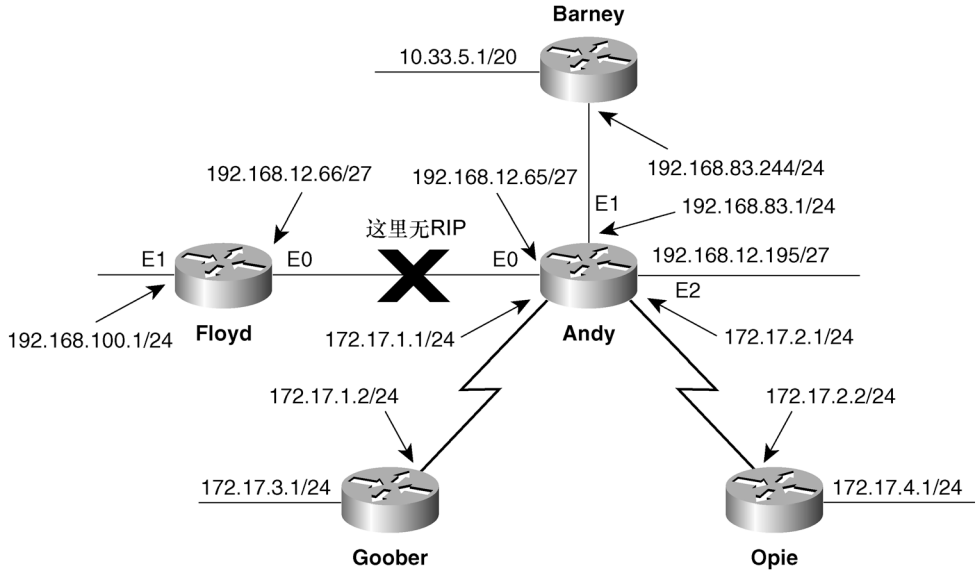


图 5-6 网络的路由策略要求在路由器 Andy 和 Floyd 之间没有 RIP 消息的交换

示例 5-9 路由器 Floyd 的 RIP 配置

```
router rip
 network 192.168.100.0
```

由于没有包含网络 192.168.12.0 的 **network** 命令语句，路由器 Floyd 将不在接口 192.168.12.66 上通告 192.168.12.0。但是，路由器 Andy 有两个和网络 192.168.12.0 相连的接口，因此网络 192.168.12.0 必须包含在 RIP 中。如果一台路由器接口处于一个启动 RIP 协议的网络的子网中，那么路由器会在该接口上发出 RIP 广播，为了阻塞这样的 RIP 广播，在 RIP 的处理中就需要增加一条 **passive-interface** 命令。路由器 Andy 的 RIP 配置参见示例 5-10。

示例 5-10 在路由器 Andy 上，具有被动接口的 RIP 配置

```
router rip
 passive-interface Ethernet0
 network 172.17.0.0
 network 192.168.12.0
 network 192.168.83.0
```

命令 **Passive-interface** 不是 RIP 协议专有的命令，它可以在所有的 IP 路由选择协议中配置使用。使用命令 **passive-interface** 实际上可以说是在一条特定的数据链路上，将路由器作为一台“静”主机来看待。像其他的“静”主机一样，它只是在该特定的链路上侦听 RIP 的广播，从而更新自己的路由表。如果希望避免路由器从一条链路上学到路由信息，就必须使用更复杂的路由更新控制才能实现，这种路由更新控制称为出站更新过滤（filtering out updates）（路由过滤的内容将在第 13 章中讨论）。和“静”主机不同的是，路由器并不在被动接口上响应收到的请求消息。

5.2.3 案例研究：配置单播更新（Unicast Update）

接下来，增加一台新的路由器 Bea，并连接到路由器 Andy 和 Floyd 之间的以太网共

享链路上（如图 5-7 所示）。在路由器 Andy 和 Floyd 之间的链路上依然保留不启动 RIP 协议的路由策略，但在路由器 Bea 和 Andy 之间，Bea 和 Floyd 之间现在都必须交换 RIP 通告消息。

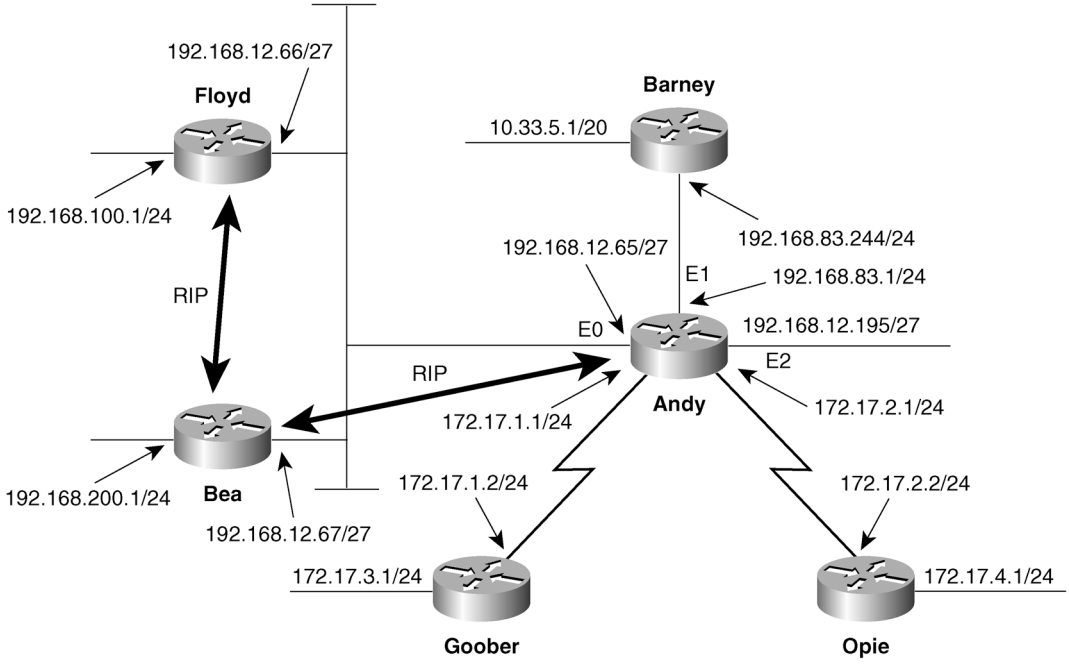


图 5-7 路由器 Andy 和 Floyd 不进行 RIP 更新的交换，但是路由器 Andy 和 Floyd 都与路由器 Bea 交换 RIP 更新

路由器 Bea 的配置很简单，参见示例 5-11。

示例 5-11 路由器 Bea 的 RIP 配置

```
router rip
 network 192.168.12.0
 network 192.168.200.0
```

在路由器 Andy 的 RIP 处理中增加一条额外的命令 **neighbor**，使 RIP 协议能够以单播方式向路由器 Bea 的接口发送通告，而这时，路由器 Andy 上的 **passive-interface** 命令仍继续防止在该链路上广播更新。[1]

路由器 Andy 的配置参见示例 5-12。

示例 5-12 路由器 Andy 的 RIP 配置，其中包括带有启动单播更新的 neighbor 语句的被动接口

```
router rip
 passive-interface Ethernet0
 network 172.17.0.0
 network 192.168.12.0
 network 192.168.83.0
 neighbor 192.168.12.67
```

因为路由器 Floyd 现在必须发送 RIP 通告给路由器 Bea，因此也必须增加一条通告

[1] **neighbor** 的另外应用是，在像帧中继这样的非广播介质型网络上发送单播更新。

192.168.12.0 的 **network** 命令。为了防止广播更新，也要增加一条 **Passive-interface** 命令，并且要增加 **neighbor** 命令以单播方式通告 RIP 更新给路由器 Bea，参见示例 5-13。

示例 5-13 路由器 Floyd 使用 neighbor 192.168.12.67 配置为单播更新

```
router rip
passive-interface Ethernet0
network 192.168.12.0
network 192.168.100.0
neighbor 192.168.12.67
```

在路由器 Andy 上启动调试命令 **debug ip rip events**，可以用来验证更改后的新配置的效果（参见示例 5-14）。路由器 Andy 可以从路由器 Bea 收到路由更新，但是无法从 Floyd 上收到，并且正在以单播方式直接给路由器 Bea 的接口发送路由更新，但是却不在它的 E0 接口上进行广播。

示例 5-14 路由器 Andy 在 E0 接口发送出的惟一路由更新是到路由器 Bea 的单播更新。路由器 Andy 可以收到来自路由器 Bea 的更新，而不是路由器 Floyd 的更新

```
Andy#debug ip rip events
RIP event debugging is on
Andy#
RIP: received v1 update from 192.168.12.67 on Ethernet0
RIP: Update contains 1 routes
RIP: sending v1 update to 255.255.255.255 via Ethernet1 (192.168.83.1)
RIP: Update contains 4 routes
RIP: Update queued
RIP: Update sent via Ethernet1
RIP: sending v1 update to 255.255.255.255 via Ethernet2 (192.168.12.195)
RIP: Update contains 6 routes
RIP: Update queued
RIP: Update sent via Ethernet2
RIP: sending v1 update to 255.255.255.255 via Serial0 (172.17.1.1)
RIP: Update contains 7 routes
RIP: Update queued
RIP: Update sent via Serial0
RIP: sending v1 update to 255.255.255.255 via Serial1 (172.17.2.1)
RIP: Update contains 7 routes
RIP: Update queued
RIP: Update sent via Serial1
RIP: sending v1 update to 192.168.12.67 via Ethernet0 (192.168.12.65)
RIP: Update contains 4 routes
RIP: Update queued
RIP: Update sent via Ethernet0
RIP: received v1 update from 172.17.1.2 on Serial0
RIP: Update contains 1 routes
RIP: received v1 update from 172.17.2.2 on Serial1
RIP: Update contains 1 routes
RIP: received v1 update from 192.168.12.67 on Ethernet0
RIP: Update contains 1 routes
```

虽然路由器 Bea 可以从路由器 Andy 和路由器 Floyd 学到路由，而且在共享的以太网上广播更新，但由于水平分隔法则依然适用，因此可以防止路由器 Bea 将从那两台路由器学到的路由重新通告到它们之间的以太网上。

5.2.4 案例研究：不连续的子网

在图 5-8 中，另一台路由器被添加到原来的网络上，它通过一个 E1 接口和子网 10.33.32.0/20

相连。现在问题出现了，网络 10.0.0.0 的另一个子网——子网 10.33.0.0/20 是和路由器 Barney 相连的，它和子网 10.33.32.0/20 之间只有一条惟一的经过网络 192.168.83.0 和 192.168.12.0 的路由路径，而这是它们完全不同的两个网络。结果，网络 10.0.0.0 将变成为不连续的。

路由器 Barney 会认为自己是网络 10.0.0.0 和网络 192.168.83.0 之间的边界路由器；同样的，路由器 Ernest_T 也会认为自己是网络 10.0.0.0 和网络 192.168.12.0 之间的边界路由器。它们都将通告一条网络 10.0.0.0 的汇总路由，结果路由器 Andy 将会“傻乎乎”地认为它有两条等价的路径可以到达同一个网络。在这种情况下，路由器 Andy 将在与路由器 Barney 和 Ernest_T 相连的链路上进行均分负载，因而，要到达网络 10.0.0.0 的数据包现在只有 50%的机会可以转发到正确的子网上。

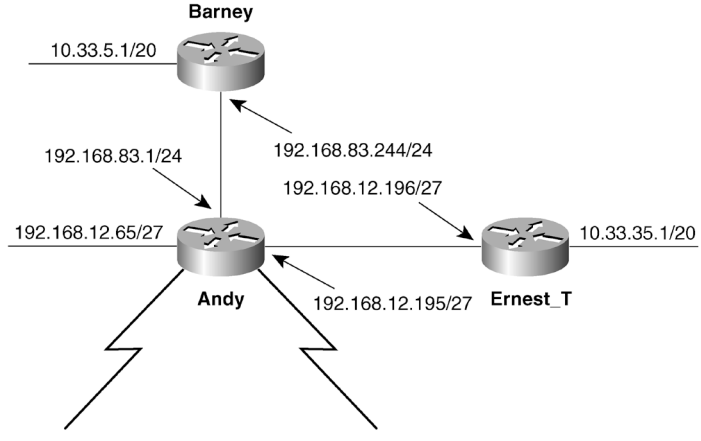


图 5-8 像 RIP 和 IGRP 等有类别路由选择协议不能对图中所示类型的网络拓扑进行路由选择，因为其中的网络 10.0.0.0 的子网被不同的主网络分开了

解决方法是在网络 192.168.83.0/24 和 192.168.12.192/27 所在的同一条链路上配置网络 10.0.0.0 的子网，这可以通过在路由器接口上配置辅助 IP 地址（secondary IP address）实现，所有配置参见示例 5-15、示例 5-16 和示例 5-17。

示例 5-15 路由器 Barney 配置了辅助 IP 地址

```
interface e0
 ip address 10.33.55.1 255.255.240.0 secondary
```

示例 5-16 路由器 Andy 配置了辅助 IP 地址，并在 RIP 中增加了一个新的网络

```
interface e1
 ip address 10.33.55.2 255.255.240.0 secondary
interface e2
 ip address 10.33.75.1 255.255.240.0 secondary
router rip
 network 10.0.0.0
```

示例 5-17 路由器 Ernest_T 配置了辅助 IP 地址

```
interface e0
 ip address 10.33.75.2 255.255.240.0 secondary
```

因为路由器 Andy 在前面的配置中没有和网络 10.0.0.0 相连的接口，所以在 RIP 配置中增加了一条网络声明（**network** 10.0.0.0）。配置的效果可以从图 5-9 中看到，原有的逻辑网络结构依然保留，只是在其网络结构上“叠加（overlaid）”了一个连续的网络 10.0.0.0。

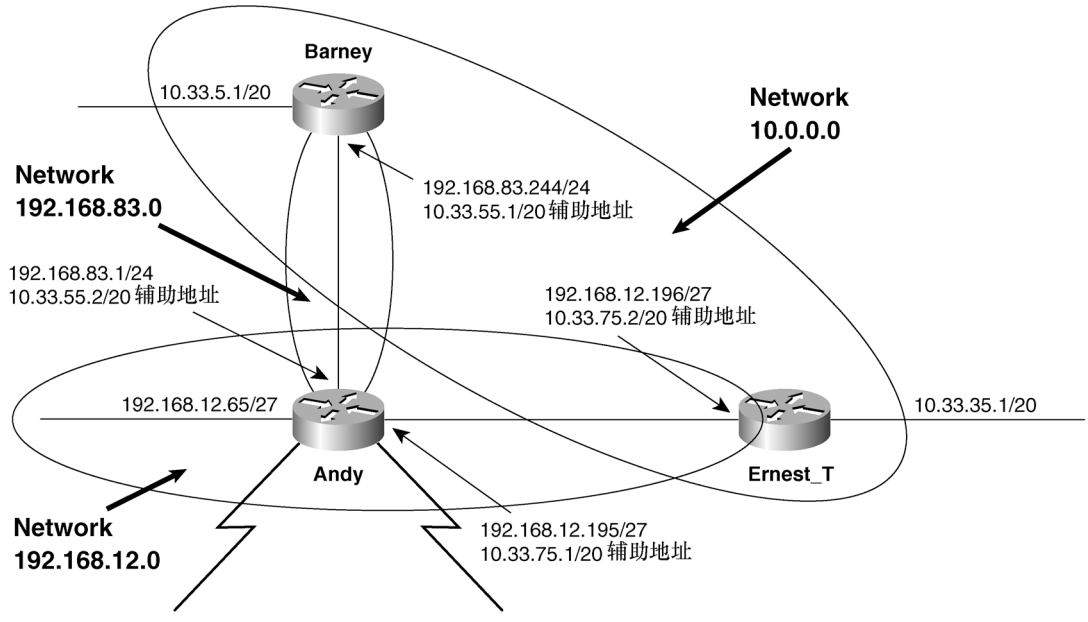


图 5-9 辅助地址被用来在已有其他网络地址的同一条链路上连接网络 10.0.0.0 的子网

示例 5-18 显示了路由器 Ernest_T 的路由表。这里值得注意的是，两条等价路由分别和下一跳地址 10.33.75.1 及 192.168.12.195 相关联。

由于路由选择进程会将辅助地址看作是单独的数据链路，所以在 RIP 协议或 IGRP 协议网络的设计中要很小心地使用。各自的 RIP 更新会在每一个子网里进行广播，如果路由更新比较多而且物理链路的带宽有限（例如串行链路），那么大量的路由更新会造成网络的拥塞。在本章后面的示例 5-21 中，可以看到在这种配置了辅助地址的链路上会产生大量的路由更新。

在插入辅助地址时一定要格外小心，如果不小心忽略了关键字 **secondary**，路由器将会认为该接口的主地址被一个新的地址代替了。在一个正在提供服务的网络接口上犯这种错误将会带来严重的后果。

示例 5-18 在路由器的路由选择进程中可以看到，子网 192.168.12.192/27 和 10.33.64.0/20 虽然属于同一个物理接口，但是它们却有各自的链路

```
Ernest_T#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
    10.0.0.0 255.255.240.0 is subnetted, 4 subnets
C      10.33.32.0 is directly connected, Ethernet1
R      10.33.48.0 [120/1] via 10.33.75.1, 00:00:05, Ethernet0
R      10.33.0.0 [120/2] via 10.33.75.1, 00:00:05, Ethernet0
C      10.33.64.0 is directly connected, Ethernet0
R      192.168.83.0 [120/1] via 192.168.12.195, 00:00:05, Ethernet0
                    [120/1] via 10.33.75.1, 00:00:05, Ethernet0
       192.168.12.0 255.255.255.224 is subnetted, 2 subnets
R      192.168.12.64 [120/1] via 192.168.12.195, 00:00:05, Ethernet0
C      192.168.12.192 is directly connected, Ethernet0
R      192.168.200.0 [120/2] via 192.168.12.195, 00:00:05, Ethernet0
                     [120/2] via 10.33.75.1, 00:00:05, Ethernet0
```

（待续）

```
R     172.17.0.0 [120/1] via 192.168.12.195, 00:00:06, Ethernet0
                 [120/1] via 10.33.75.1, 00:00:06, Ethernet0
Ernest_T#
```

5.2.5 案例研究：控制 RIP 的度量

如图 5-10 所示，在路由器 Ernest_T 和 Barney 之间增加一条串行链路用作备份链路，只有当路由经过路由器 Andy 失败时这条链路才被使用。现在的问题在于，路由器 Barney 的子网 10.33.0.0 和路由器 Ernest_T 的子网 10.33.32.0 之间经过这条串行链路的路径是 1 跳，而经优选的以太链路的路径却是 2 跳。在正常的情况下，RIP 会首先选择串行链路。

可以通过命令 **offset-list** 来改变路由的度量值，该命令指定一个数值来加大路由的度量值，并且参照一个访问列表（access list）[1]来决定哪些路由条目需要修改。命令语法如下所示：

```
offset-list {access-list-number | name} {in| out} offset [type number]
```

路由器 Ernest_T 的配置参见示例 5-19。

示例 5-19 路由器 Ernest_T 在 RIP 配置中配置了入站偏移列表

```
access-list 1 permit 10.33.0.0 0.0.0.0
router rip
 network 192.168.12.0
 network 10.0.0.0
 offset-list 1 in 2 Serial0
```

访问列表的配置确定了关于子网 10.33.0.0 的路由，偏移列表（offset list）的语法含义是："先检查从 S0 接口接收进来的 RIP 通告，如果存在和访问列表 1 指定的地址相匹配的路由条目，那么就把该路由条目的度量值加大 2 跳。"

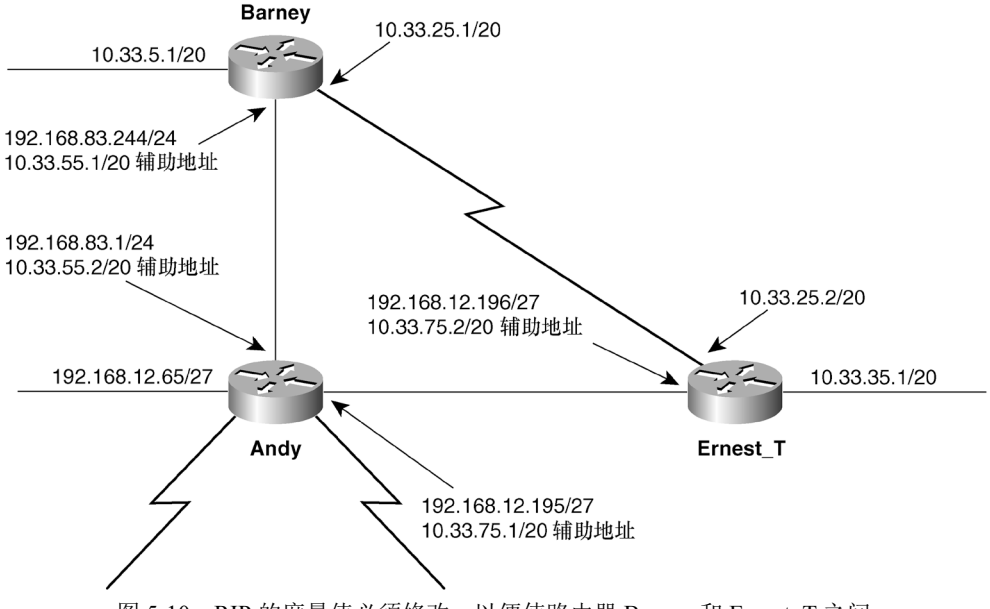


图 5-10 RIP 的度量值必须修改，以便使路由器 Barney 和 Ernest_T 之间 2 跳的以太链路路由优先于 1 跳的串行链路路由

路由器 Barney 更新配置后，在它的配置文件中包含了示例 5-20 中显示的语句。

[1] 参照附录 B 中关于访问列表的讲述。

示例 5-20 路由器 Barney 包含入站偏移列表的 RIP 配置

```
router rip
 offset-list 5 in 2 Serial0
 network 10.0.0.0
 network 192.168.83.0
!
access-list 5 permit 10.33.32.0 0.0.0.0
```

在示例 5-21 中显示了路由器 Ernest_T 的配置所产生的结果。

示例 5-21 偏移列表指定额外增加的跳数，把子网 10.33.0.0/20 经过 S0 接口的路由度量由 1 跳变成了 3 跳。但经过 E0 接口的路由的跳数没有改变，还是 2 跳

```
Ernest_T#debug ip rip
RIP protocol debugging is on
Ernest_T#
RIP: received v1 update from 192.168.12.195 on Ethernet0
     192.168.12.64 in 1 hops
     10.0.0.0 in 1 hops
     192.168.83.0 in 1 hops
     192.168.200.0 in 2 hops
     172.17.0.0 in 1 hops
RIP: received v1 update from 10.33.75.1 on Ethernet0
     10.33.48.0 in 1 hops
     10.33.0.0 in 2 hops
     192.168.83.0 in 1 hops
     192.168.12.0 in 1 hops
     192.168.200.0 in 2 hops
     172.17.0.0 in 1 hops
RIP: received v1 update from 10.33.25.1 on Serial0
     10.33.32.0 in 3 hops
     10.33.48.0 in 1 hops
     10.33.0.0 in 3 hops
     192.168.83.0 in 1 hops
     192.168.200.0 in 3 hops
     172.17.0.0 in 2 hops
RIP: sending v1 update to 255.255.255.255 via Ethernet0 (192.168.12.196)
RIP: build update entries
     network 10.0.0.0, metric 1
RIP: sending v1 update to 255.255.255.255 via Ethernet0 (10.33.75.2)
RIP: build update entries
     subnet 10.33.32.0, metric 1
     subnet 10.33.16.0, metric 1
RIP: sending v1 update to 255.255.255.255 via Ethernet1 (10.33.35.1)
RIP: build update entries
     subnet 10.33.48.0, metric 2
     subnet 10.33.0.0, metric 3
     subnet 10.33.16.0, metric 1
     subnet 10.33.64.0, metric 1
     network 192.168.83.0, metric 2
     network 192.168.12.0, metric 1
     network 192.168.200.0, metric 3
     network 172.17.0.0, metric 2
RIP: sending v1 update to 255.255.255.255 via Serial0 (10.33.25.2)
RIP: build update entries
     subnet 10.33.32.0, metric 3
     subnet 10.33.0.0, metric 3
     subnet 10.33.64.0, metric 1
     network 192.168.12.0, metric 1
     network 192.168.200.0, metric 3
     network 172.17.0.0, metric 2
```

作为另一种选择，路由器也可以通过配置去修改向外通告的出站（outgoing）路由更新

的度量，替代上述两台路由器对从链路上接收的入站（incoming）路由更新的度量的修改。示例 5-22 和示例 5-23 的配置可以和前面的配置达到同样的效果。

示例 5-22 路由器 Ernest_T 使用出站偏移列表的配置

```
router rip
 offset-list 3 out 2 Serial0
 network 192.168.12.0
 network 10.0.0.0
!
access-list 3 permit 10.33.32.0 0.0.0.0
```

示例 5-23 路由器 Barney 使用出站偏移列表的配置

```
router rip
 offset-list 7 out 2 Serial0
 network 10.0.0.0
 network 192.168.83.0
!
access-list 7 permit 10.33.0.0 0.0.0.0
```

偏移列表的其他几个选项在配置时也是有用的。如果不指定使用偏移列表的接口，那么偏移列表将在所有与访问列表匹配的接口上修改所有的入站更新或出站更新。如果不调用访问列表（使用 0 作为访问列表的序列号）来进行匹配，偏移列表将修改所有的入站更新或出站更新。

当在入站或出站更新的通告上选择是否使用偏移列表时，有些需要注意的地方。例如，在一个多于两台路由器的广播网络上，到底是需要某台单独的路由器向它所有的邻居路由器广播偏移修改后的通告，还是需要某台单独的路由器接收偏移修改后的通告，这一点必须要考虑清楚。

在运行的路由上实施偏移列表时也需要特别注意。当一个偏移列表引起下一跳路由器通告的度量值比它正在通告的路由更新的度量值更高时，直到抑制计时器（holddown timer）超时前，这条路由都会被标记为不可到达。

5.2.6 案例研究：最小化更新信息的影响

如果读者希望将由路由选择更新引起的网络流量减少到最小，那么现在可以做到。在缺省情况下，普通的 RIP 更新消息是每 30s 产生一次，包括完整的路由表，除非是在路由条目的度量发生变化时传送的瞬时更新（flash update）。在具有很多子网的网络中，特别是对于一些较低带宽的链路，路由选择更新会对网络流量产生很大影响。对于正在使用流量计费方式付费的链路来说，读者也可能希望将其路由选择更新流量降低到最小。

如图 5-10 所示，假定从路由器 Barney 到 Ernest_T 的新的串行链路利用率很高。我们可以通过两种方式使路由选择协议产生的流量最小化，从而降低 RIP 的路由选择流量：第一种方法是调整路由选择协议的计时器以便降低更新的频率，但是这在主要链路发生故障时会引起较长的收敛时间；另外一种方法是配置触发扩展特性来消除周期性的 RIP 更新。

使用接口模式下的命令 **ip rip triggered** 可以启动 RIP 协议的触发扩展特性。[1] 在一条串

[1] 触发扩展特性是在 RFC 2091 中定义的，并在 12.0（1）T 版本首先引入 IOS 软件系统。

行链路上的两台路由器都必须确定配置了具有触发扩展特性的 RIP 协议后，路由表的更新将会变得最少，仅仅包括路由表最初的交换信息和路由表发生变化时的更新信息。这条命令仅仅在串行链路上有效，并且必须在链路的两端同时配置才会产生效果。

在路由器 Barney 到 Ernest_T 的串行链路接口上配置触发扩展特性，具体配置参见示例 5-24。

示例 5-24 路由器 Barney 配置了触发更新，而不是周期性的更新

```
interface serial 0
 ip rip triggered
```

在路由器Barney的调试信息中，显示Barney试图与链路另一端的路由器建立一个触发关系。路由器 Barney 发送轮询（Poll）并等待确认。当它没有收到任何确认时，路由器 Barney 开始发送正常的 RIPv1 更新。调试命令 **debug ip rip** 和 **debug ip rip trigger** 的输出参见示例 5-25 所示。

示例 5-25 当一台路由器一开始就配置了触发的 RIP 后，它会发送轮询信息来确定邻居是否也配置了触发的 RIP

```
Barney#debug ip rip
RIP protocol debugging is
Barney#debug ip rip trigger
RIP trigger debugging is on
Barney(config)#interface serial 0
Barney(config-if)#ip rip triggered
* 13:22:10.223: RIP: sending triggered request on Serial0 to 255.255.255.255
* 13:22:10.223: RIP: Start poll timer from 10.33.25.1 on Serial0
* 13:22:15.223: RIP-TIMER: polling timer on Serial0(10.33.25.1) expired
* 13:22:15.223: RIP: sending triggered request on Serial0 to 255.255.255.255
* 13:22:15.223: RIP: Start poll timer from source 10.33.25.1 on Serial0
* 13:22:16.733: RIP: received v1 update from 10.33.25.2 on Serial0
* 13:22:16.733:      172.17.8.0 in 1 hops
* 13:22:16.733:      172.17.9.0 in 1 hops
* 13:22:16.737:      172.17.10.0 in 1 hops
* 13:22:16.737:      172.17.11.0 in 1 hops
* 13:22:19.466: RIP-TIMER: sending timer on Serial0 expired
* 13:22:20.223: RIP-TIMER: polling timer on Serial0(10.33.25.1) expired
* 13:22:20.223: RIP: sending triggered request on Serial0 to 255.255.255.255
* 13:22:20.223: RIP: Start poll timer from source 10.33.25.1 on Serial0
* 13:22:25.223: RIP-TIMER: polling timer on Serial0(10.33.25.1) expired
* 13:22:25.223: RIP: sending triggered request on Serial0 to 255.255.255.255
* 13:22:25.223: RIP: Start poll timer from source 10.33.25.1 on Serial0
* 13:22:30.224: RIP-TIMER: polling timer on Serial0(10.33.25.1) expired
* 13:22:30.224: RIP: sending triggered request on Serial0 to 255.255.255.255
* 13:22:30.224: RIP: Start poll timer from source 10.33.25.1 on Serial0
* 13:22:35.224: RIP-TIMER: polling timer on Serial0(10.33.25.1) expired
* 13:22:35.224: RIP: sending triggered request on Serial0 to 255.255.255.255
* 13:22:35.224: RIP: Start poll timer from source 10.33.25.1 on Serial0
* 13:22:40.224: RIP-TIMER: polling timer on Serial0(10.33.25.1) expired
* 13:22:40.224: RIP: Poll 6 times on Serial0 thru 10.33.25.1 without any ack
* 13:22:46.126: RIP: received v1 update from 10.33.25.2 on Serial0
* 13:22:46.126:      172.17.8.0 in 1 hops
* 13:22:46.126:      172.17.9.0 in 1 hops
* 13:22:46.130:      172.17.10.0 in 1 hops
* 13:22:46.130:      172.17.11.0 in 1 hops
* 13:22:49.054: RIP-TIMER: sending timer on Serial0 expired
* 13:22:49.054: RIP: sending v1 update to 255.255.255.255 via Serial0(10.33.25.1)
* 13:22:49.054: RIP: build update entries
```

（待续）

```
* 13:22:49.054:   network 10.0.0.0 metric 2
* 13:22:49.054:   subnet 172.17.0.0 metric 2
* 13:22:49.054:   subnet 172.17.2.0 metric 1
* 13:22:49.058:   subnet 172.17.4.0 metric 2
* 13:22:49.058:   network 192.168.12.0 metric 1
* 13:22:49.058:   network 192.168.83.0 metric 1
* 13:23:13.070: RIP: received v1 update from 10.33.25.2 on Serial0
* 13:23:13.070:      172.17.8.0 in 1 hops
* 13:23:13.074:      172.17.9.0 in 1 hops
* 13:23:13.074:      172.17.10.0 in 1 hops
* 13:23:13.074:      172.17.11.0 in 1 hops
* 13:23:17.385: RIP-TIMER: sending timer on Serial0 expired
* 13:23:17.385: RIP: sending v1 update to 255.255.255.255 via Serial0(10.33.25.1)
* 13:23:17.385: RIP: build update entries
* 13:23:17.385:   network 10.0.0.0 metric 2
* 13:23:17.385:   subnet 172.17.0.0 metric 2
* 13:23:17.385:   subnet 172.17.2.0 metric 1
* 13:23:17.389:   subnet 172.17.4.0 metric 2
* 13:23:17.389:   network 192.168.12.0 metric 1
* 13:23:17.389:   network 192.168.83.0 metric 1
```

调试输出显示了所配置的路由器发送了 6 个触发请求。对于每一个请求，路由器都会设置一个轮询计时器，每个周期为 5s。如果在 5s 内还没有收到确认消息，就会发送另一个触发请求。如果在发送完 6 个触发请求后还没有收到确认消息，那么这个轮询就认为超时了，路由器将等待下一个普通的更新时间，并广播一个 RIP 更新。在路由器 Barney 发送触发请求的时候，路由器 Ernest_T 继续广播它自己的 RIP 更新。

现在路由器 Ernest_T 到路由器 Barney 的串行链路配置了触发的 RIP。通过在路由器 Ernest_T 上的调试，示例 5-26 中显示了路由器 Barney 和 Ernest_T 的初始化过程。

示例 5-26 调试输出显示了两台路由器正在建立一个触发的 RIP 关联关系

```
* 13:35:04.612: RIP: received v1 triggered request from 172.17.1.2 on Serial0
* 13:35:04.616: RIP: Trigger rip running on network 172.17.0.0 thru Serial0
* 13:35:04.616: RIP: 172.17.1.2 change state from DOWN to INIT
* 13:35:04.616: RIP: 172.17.1.2 change state from INIT to LOADING
* 13:35:04.616: RIP: send v1 triggered flush update to 172.17.1.2 on Serial0
* 13:35:04.616: RIP: assigned sequence number 25 on Serial0
* 13:35:04.616: RIP: build update entries
* 13:35:04.620:   route 202: network 192.168.12.0 metric 1
* 13:35:04.620:   route 206: subnet 172.17.2.0 metric 1
* 13:35:04.620:   route 208: network 192.168.83.0 metric 1
* 13:35:04.620:   route 210: network 10.0.0.0 metric 2
* 13:35:04.620:   route 215: subnet 172.17.4.0 metric 2
* 13:35:04.620:   route 217: subnet 172.17.0.0 metric 2
* 13:35:04.624: RIP: Update contains 6 routes, start 202, end 226
* 13:35:04.624: RIP: start retransmit timer of 172.17.1.2
* 13:35:04.680: RIP: received v1 triggered ack from 172.17.1.2 on Serial0 flush seq# 25
* 13:35:04.684: RIP: 172.17.1.2 change state from LOADING to FULL
* 13:35:04.688: RIP: received v1 triggered update from 172.17.1.2 on Serial0
* 13:35:04.688: RIP: sending v1 ack to 172.17.1.2 via Serial0 (172.17.1.1), flush,
seq# 14
* 13:35:04.736: RIP: received v1 triggered update from 172.17.1.2 on Serial0
* 13:35:04.740: RIP: sending v1 ack to 172.17.1.2 via Serial0 (172.17.1.1), seq# 15
* 13:35:04.740:      172.17.9.0 in 1 hops
* 13:35:04.740:      172.17.8.0 in 1 hops
* 13:35:04.740:      172.17.11.0 in 1 hops
* 13:35:04.744:      172.17.10.0 in 1 hops
* 13:35:52.879: RIP-TIMER: sending timer on Serial0 expired
* 13:36:21.907: RIP-TIMER: sending timer on Serial0 expired
* 13:36:47.421: RIP-TIMER: sending timer on Serial0 expired
```

触发状态从 DOWN 状态开始，经过 INIT 和 LOADING 状态，最后为 FULL 状态。而后进行路由信息的交换和更新的确认。在输出信息的结尾部分，读者可以看到 RIP 更新计时器正在超时，但是没有新的更新发送，也没有收到更新。

5.3 RIP 故障诊断

RIP 协议的故障诊断相对来说是比较简单的。对于 RIP 这样的有类别路由选择协议来说，最困难的排错就是出现子网掩码配置错误或者子网不连续的情形。如果路由表包含了不准确的或被丢失的路由，那么就应该检查邻近的所有子网和所有子网掩码的一致性。

最后，有一条命令在一台高速路由器向一台低速路由器发送大量 RIP 消息时可能比较有用。在这种情况下，低速路由器并不能像接收一样快地处理这些路由更新，因此可能会丢失路由信息。这种情况可以通过在 RIP 的处理中使用 **Output-delay** 命令来设置一个 8～50ms 的发包之间的延迟间隙（缺省为 0ms）来解决。

5.4 展　　望

RIP 协议的简单、成熟和使用的广泛性确保了它还将会使用许多年。但是，读者在本章也可以看到，RIP 协议有类别特性的限制。下一章将会继续介绍 RIP 协议，读者可以了解该协议是怎样扩展来支持无类别路由选择的，也可以了解到为了支持 IPv6 协议所做的扩展。

5.5 总结表：第 5 章命令总结

命令	描述
debug ip rip [**events**]	简要地显示路由器收发的 RIP 信息
ip address *ip-address mask* **secondary**	在接口上指定一个 IP 地址作为辅助地址
ip rip triggered	在某个接口上配置 RIP 的触发扩展特性
neighbor *ip-address*	通过指定接口邻居的 IP 地址来建立邻接关系
network *network-number*	指定一个需要运行 RIP 的网络
offset-list {*access-list-number* | *name*} {**in** | **out**} *offset* [*type number*]	指定路由表中一个与指定的访问列表匹配的路由条目，将自己的度量值增加一个指定的偏移量
output-delay *delay*	设定一个指定延迟长度的延迟间隙，以便协调高速路由器和低速路由器之间的延迟问题
passive-interface *type number*	在指定类型和序列号的接口上阻止 RIP 广播
router rip	启动 RIP 进程
timers basic *update invalid holddown flush*	修改指定的计时器的值

5.6 推荐读物

Hedrick，C.“Routing Information Protocol”，RFC 1058，1988 年 6 月。
Meyer，G.和 Sherry，S。“Triggered Extensions to RIP to Support Demand Circuits”。RFC 2091，1997 年 1 月。

5.7 复 习 题

1．RIP 协议使用什么端口？
2．RIP 协议使用什么度量？怎样用度量来表示一个不可达的网络？
3．RIP 协议的更新周期是多少？
4．在一条路由被标记成不可到达之前，必须忽略多少更新？
5．垃圾收集计时器的用途是什么？
6．为什么触发更新要使用一个随机计时器？这个计时器的大小范围是什么？
7．RIP 协议的请求消息和响应消息之间有哪些不同之处？
8．RIP 协议使用哪两种类型的请求消息？
9．在什么情况下会发出一个 RIP 协议的响应消息？
10．为什么 RIP 协议在主网络的边界处会屏蔽子网？

5.8 配置练习

1．写出图 5-11 中所示的 6 台路由器的相关配置，使它们可以利用 RIP 协议来为所有的子网进行路由选择。

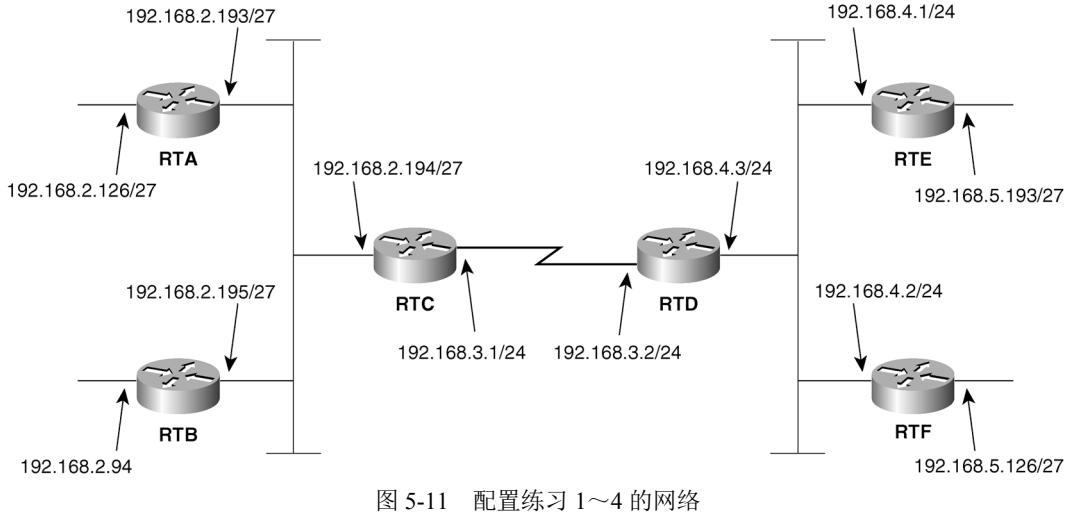


图 5-11 配置练习 1～4 的网络

2．更改配置练习 1 中的配置，使路由器 RTC 和 RTD 之间 RIP 更新的通告由广播方式改为单播方式。

3．在图 5-11 中，路由器 RTC 和 RTD 之间的串行链路的带宽十分有限，请进一步调整 RIP 协议的配置，使得经过这条链路的 RIP 更新能够每两分钟发送一次。这里要仔细考虑的是，必须要更改哪些计时器？以及必须要更改哪些路由器上的计时器？

4．制定一个路由策略，使网络 192.168.4.0 在路由器 RTA 看来是不可到达的，而网络 192.168.5.0 在路由器 RTB 看来是不可到达的，可以利用偏移列表来实现该策略。

5．依照“有类别路由选择：直连的子网络”一节所述，在一个主类别分类网络中的所有子网掩码必须是一致的。但那一节中却没有强调一个主类别分类网络内的子网掩码必须是相同的。图 5-12 中的那两台路由器的 RIP 配置如下：

```
router rip
network 192.168.20.0
```

在这个小型的网络中，数据包可以被正确地路由转发吗？解释一下为什么可以或者为什么不可以。

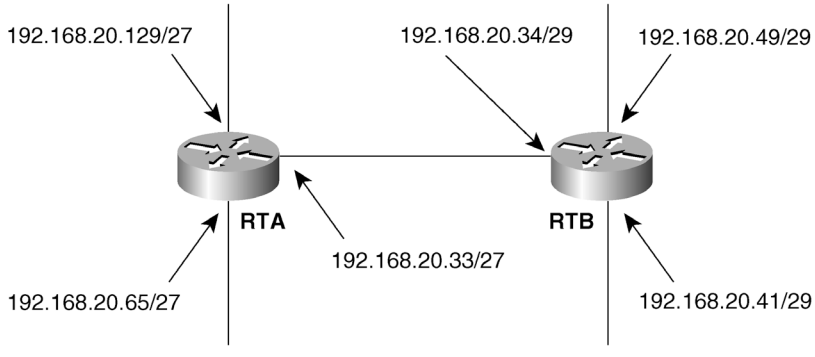


图 5-12 配置练习 5 使用的网络

5.9 故障诊断练习

1．在第一个偏移列表的例子中，路由器 Barney 上的访问列表由

```
access-list 5 permit 10.33.32.0 0.0.0.0
```

更改成

```
access-list 5 deny 10.33.32.0 0.0.0.0
access-list 5 permit any
```

会出现什么结果？

2．在图 5-13 中显示了一个网络，其中有一台路由器的 IP 地址掩码配置错误。在示例 5-27～示例 5-29 中分别显示了路由器 RTA、RTB 和 RTC 的路由表。请读者根据前面所了解的关于 RIP 协议通告和接收路由更新的知识，解释一下路由器 RTB 的路由表中的每一个路由表项。并请解释一下路由器 RTB 的路由表中子网 172.16.26.0 的掩码为什么是 32 位的？在所

有的路由表中，如果存在被丢失的路由条目，请解释为什么？

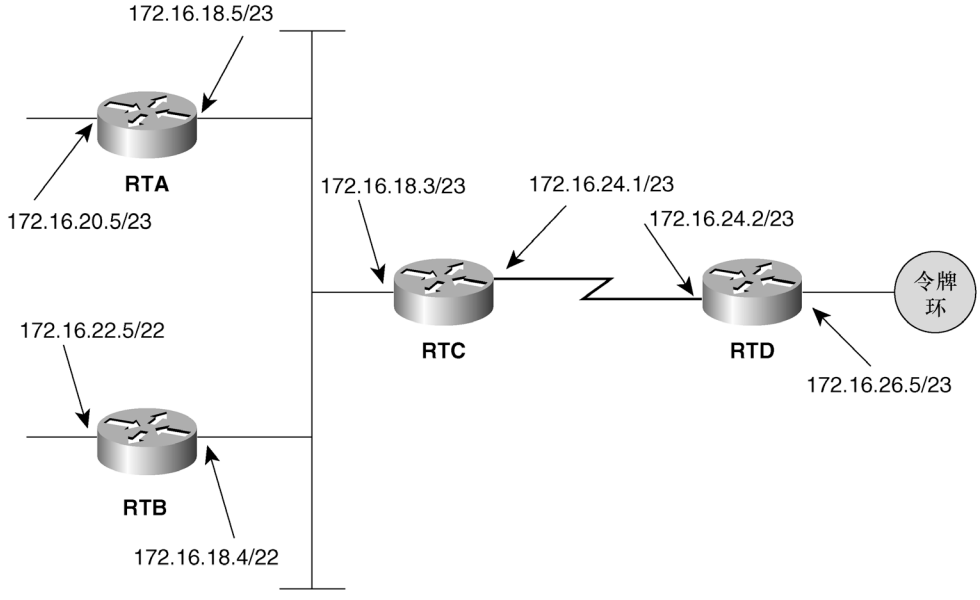


图 5-13 故障诊断练习 2 和 3 的网络

示例 5-27 图 5-13 中路由器 RTA 的路由表

```
RTA#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route
Gateway of last resort is not set
     172.16.0.0/16 is subnetted, 4 subnets
R       172.16.24.0 [120/1] via 172.16.18.3, 00:00:01, Ethernet0
R       172.16.26.0 [120/2] via 172.16.18.3, 00:00:01, Ethernet0
C       172.16.20.0 is directly connected, Ethernet1
C       172.16.18.0 is directly connected, Ethernet0
RTA#
```

示例 5-28 图 5-13 中路由器 RTB 的路由表

```
RTB#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route, o - ODR
Gateway of last resort is not set
     172.16.0.0/16 is variably subnetted, 4 subnets, 2 masks
R       172.16.24.0/22 [120/1] via 172.16.18.3, 00:00:20, Ethernet0
R       172.16.26.0/32 [120/2] via 172.16.18.3, 00:00:20, Ethernet0
C       172.16.20.0/22 is directly connected, Ethernet1
C       172.16.16.0/22 is directly connected, Ethernet0
RTB#
```

示例 5-29 图 5-13 中路由器 RTC 的路由表

```
RTC#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route, o - ODR
Gateway of last resort is not set
     172.16.0.0/23 is subnetted, 4 subnets
C       172.16.24.0 is directly connected, Serial0
R       172.16.26.0 [120/1] via 172.16.24.2, 00:00:09, Serial0
R       172.16.20.0 [120/1] via 172.16.18.5, 00:00:25, Ethernet0
C       172.16.18.0 is directly connected, Ethernet0
RTC#
```

3．在图 5-13 中，子网 172.16.18.0/23 上的用户一直抱怨和子网 172.16.26.0/23 的连接总是时断时续的——有时通，有时不通（路由器 RTB 上配置错误的掩码已经改为正确的了）。起初检查路由器 RTC 和 RTD 的路由表（参见示例 5-30）也没有什么问题，所有的子网都在路由表中。然而经过 1min 或更长一点的时间，发现路由器 RTC 显示的子网 172.16.26.0/23 变得不可到达了（参见示例 5-31），而路由器 RTD 依然显示出所有的子网。再经过几分钟后，路由器 RTC 的路由表中又看到了那个子网（参见示例 5-32）。在显示这 3 幅图示的每一个的时候，路由器 RTD 的路由表都没有变化。请仔细检查示例 5-30～示例 5-32 中的路由表所隐含的问题，看看到底是什么问题。

示例 5-30 图 5-13 中路由器 RTC 和 RTD 的路由表

```
RTC#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route, o - ODR
Gateway of last resort is not set
     172.16.0.0/23 is subnetted, 5 subnets
C       172.16.24.0 is directly connected, Serial0
R       172.16.26.0 [120/1] via 172.16.24.2, 00:02:42, Serial0
R       172.16.20.0 [120/1] via 172.16.18.5, 00:00:22, Ethernet0
R       172.16.22.0 [120/1] via 172.16.18.4, 00:00:05, Ethernet0
C       172.16.18.0 is directly connected, Ethernet0
```

```
RTD#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route
Gateway of last resort is not set
     172.16.0.0/16 is subnetted, 5 subnets
C       172.16.24.0 is directly connected, Serial0
C       172.16.26.0 is directly connected, TokenRing0
R       172.16.20.0 [120/2] via 172.16.24.1, 00:00:00, Serial0
R       172.16.22.0 [120/2] via 172.16.24.1, 00:00:00, Serial0
R       172.16.18.0 [120/1] via 172.16.24.1, 00:00:00, Serial0
```

示例 5-31 在示例 5-30 显示大约 60s 后检查得到的路由器 RTC 和 RTD 的路由表

```
RTC#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route, o - ODR
Gateway of last resort is not set
     172.16.0.0/23 is subnetted, 5 subnets
C       172.16.24.0 is directly connected, Serial0
R       172.16.26.0/23 is possibly down,
           routing via 172.16.24.2, Serial0
R       172.16.20.0 [120/1] via 172.16.18.5, 00:00:19, Ethernet0
R       172.16.22.0 [120/1] via 172.16.18.4, 00:00:24, Ethernet0
C       172.16.18.0 is directly connected, Ethernet0
```

```
RTD#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route
Gateway of last resort is not set
     172.16.0.0/16 is subnetted, 5 subnets
C       172.16.24.0 is directly connected, Serial0
C       172.16.26.0 is directly connected, TokenRing0
R       172.16.20.0 [120/2] via 172.16.24.1, 00:00:15, Serial0
R       172.16.22.0 [120/2] via 172.16.24.1, 00:00:15, Serial0
R       172.16.18.0 [120/1] via 172.16.24.1, 00:00:15, Serial0
```

示例 5-32 在示例 5-31 显示大约 120s 后检查得到的路由器 RTC 和 RTD 的路由表

```
RTC#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route, o - ODR
Gateway of last resort is not set
     172.16.0.0/23 is subnetted, 5 subnets
C       172.16.24.0 is directly connected, Serial0
R       172.16.26.0 [120/1] via 172.16.24.2, 00:00:09, Serial0
R       172.16.20.0 [120/1] via 172.16.18.5, 00:00:11, Ethernet0
R       172.16.22.0 [120/1] via 172.16.18.4, 00:00:18, Ethernet0
C       172.16.18.0 is directly connected, Ethernet0
```

```
RTD#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route
Gateway of last resort is not set
     172.16.0.0/16 is subnetted, 5 subnets
C       172.16.24.0 is directly connected, Serial0
C       172.16.26.0 is directly connected, TokenRing0
R       172.16.20.0 [120/2] via 172.16.24.1, 00:00:19, Serial0
R       172.16.22.0 [120/2] via 172.16.24.1, 00:00:19, Serial0
R       172.16.18.0 [120/1] via 172.16.24.1, 00:00:19, Serial0
```

本章包括以下主题：

- RIPv2 的基本原理与实现；
- RIPng 的基本原理与实现；
- RIPv2 的配置；
- RIPng 的配置；
- RIPv2 和 RIPng 的故障诊断。

第 6 章

RIPv2、RIPng 和无类别路由选择

RIPv2 协议在 RFC17231中进行了定义，[1] 并在 Cisco IOS 11.1 版及后续的版本中得到支持。确切地说，RIPv2 协议不是一个新的协议；它只是在 RIPv1 协议的基础上增加了一些扩展特性，以适用于现代网络的路由选择环境。这些扩展特性有：

- 每个路由条目都携带自己的子网掩码；
- 路由选择更新具有认证功能；
- 每个路由条目都携带下一跳地址；
- 外部路由标志；
- 组播路由更新。

在这些扩展特性中，最重要的就是路由选择更新条目增加了子网掩码的字段，因而 RIPv2 协议可以使用可变长的子网掩码，使其成为一个支持无类别路由选择的协议。

RIPv2 协议是本书中讲述的第一个无类别路由选择协议。因此，本章将同时介绍无类别路由选择和 RIPv2 协议。

RIP 下一代（RIP next generation，RIPng）是 RIPv2 对 IPv6 路由选择的修改，也将在本章中讲述。与 IPv4 不同，IPv6 天生就是无类别的；它没有 A 类、B 类和 C 类地址，或类似的地址分组。因此，RIPng 也是一个无类别路由选择协议。

6.1 RIPv2 的基本原理与实现

所有在 RIPv1 中运用的实现方法、计时器和稳定特性都同样可以在版本 2 中使用，其中只有一个例外，就是路由更新的

[1] 对这个 RFC 的补充还有 RFC 1721——“RIP Version 2 Protocol Analysis”和 RFC 1722——“RIP Version 2 Protocol Applicability Statement”。

广播。RIPv2 协议使用组播的方式向其他宣告 RIPv2 的路由器发出更新消息，它所使用的组播地址是保留的 D 类地址 224.0.0.9。使用组播方式的好处在于，本地网络上相连的和 RIP 路由选择无关的设备不再需要花费时间对路由器广播的更新消息进行解析。组播更新将在 6.1.2 小节进一步阐述。

先来看一下 RIP 协议的消息格式中提供了哪些版本 2 的扩展特性，这一节将主要关注 RIPv2 的操作和这些新增的扩展特性所带来的好处。

6.1.1　RIPv2 的消息格式

RIPv2 的消息格式如图 6-1 所示，它的基本结构和 RIPv1 相同。所有相对于原来协议的扩展特性都是由未使用的字段提供。和版本 1 一样，RIPv2 的更新消息最大可以包含 25 个路由条目。同样的，与版本 1 相同的是，RIPv2 的操作使用 UDP 端口 520，并且数据报（datagram）的大小（包括一个 8 字节的 UDP 头部）最大为 512 个八位组（octet）。

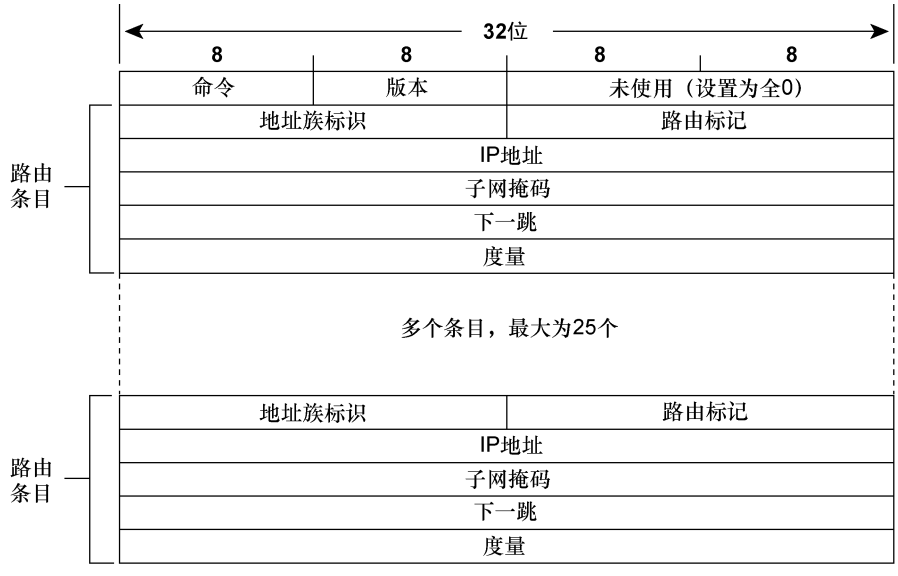


图 6-1　RIPv2 利用了版本 1 的消息格式中的未使用字段，因而这些扩展没有改变版本 1 的基本消息格式

- **命令**（**Command**）——取值 1 和 2，1 表示为请求消息，2 表示响应消息。
- **版本号**（**Version**）——对于 RIPv2，该字段的值设置为 2。如果设置为 0 或者虽设置为 1 但消息是无效的 RIPvl 格式，那么该消息将被丢弃。RIPv2 处理有效的 RIPv1 消息。
- **地址族标识**（**Address Family Identifier，AFI**）——对于 IPv4 协议，该项总是设置为 2。只有一种例外情况，当该消息对路由器（或主机）的整个路由表进行请求时，这个字段将被设置为 0。
- **路由标记**（**Route Tag**）——提供这个字段用来标记外部路由或重新分配到 RIPv2 协议中的路由。默认的情况是使用这个 16 位的字段来携带从外部路由选择协议注入到 RIP 中的路由的自主系统号。虽然 RIP 协议自己并不使用这个字段，但是在多个地点和某个 RIP 域相连的外部路由，可能需要使用这个路由标记字段通过 RIP 域来

交换路由信息。这个字段也可以用来把外部路由编成“组”，以便在 RIP 域中更容易地控制这些路由。关于路由标记的使用将在第 14 章中进一步论述。

- **IP 地址**（**IP Address**）——路由条目的 IPv4 目的地址，它可以是主网络地址、子网地址或主机路由。
- **子网掩码**（**Subnet Mask**）——是一个 32 位的掩码，用来标识 IPv4 地址的网络和子网部分。这个字段的意义将在 6.1.5 小节中论述。
- **下一跳**（**Next Hop**）——如果存在的话，它标识一个比通告路由器的地址更好的下一跳地址。换句话说，它指出的下一跳地址，其度量值比在同一个子网上的通告路由器更靠近目的地。如果这个字段设置为全 0（0.0.0.0），说明通告路由器的地址是最优的下一跳地址。在本节的最后将用一个例子说明这个字段的用处。
- **度量**（**Metric**）——是一个在 1~16 之间的跳数。

图 6-2 显示了 4 台连接在同一条以太网链路上的路由器。[1]路由器 Jicarilla、Mescalero 和 Chiricahua 都属于自主系统 65501，并相互宣告 RIPv2。路由器 Chiricahua 是自主系统 65501 和自主系统 65502 之间的边界路由器，在第二个自主系统中，它把 BGP 路由通告给路由器 Lipan。

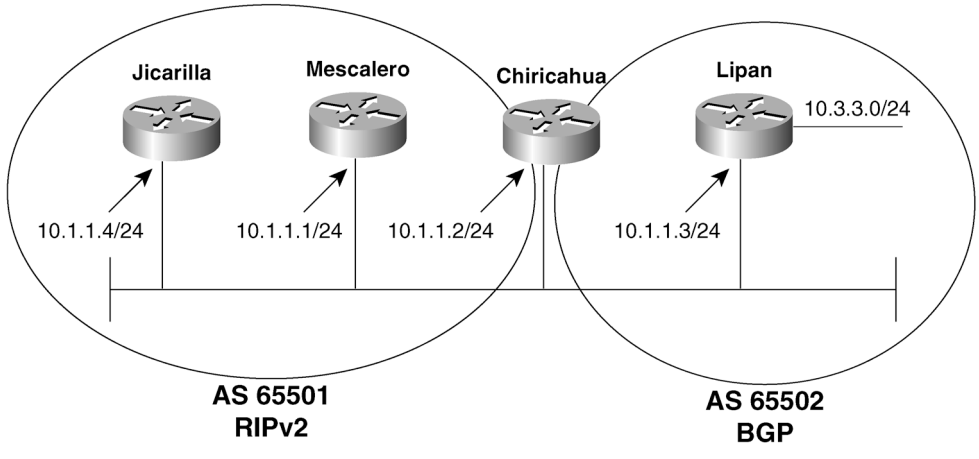


图 6-2 虽然它们共享一条公共的数据链路，但路由器 Jicarilla 和 Mescalero 只宣告 RIPv2，而路由器 Lipan 只宣告 BGP。路由器 Chiricahua 负责把从后者学习到的路由通告给前面两台路由器

在这里，路由器 Chiricahua 正在把它从 BGP 学习到的路由通告给运行 RIP 的路由器（如图 6-3 所示）。[2]在路由器 Chiricahua 的 RIPv2 通告里，它将使用路由标记字段来指出子网 10.3.3.0（掩码是 255.255.255.0）是位于自主系统 65502（0xFFDE）之中的。路由器 Chiricahua 也将使用下一跳字段告诉路由器 Jicarilla 和 Mescalero，到达子网 10.3.3.0 最优的下一跳地址是路由器 Lipan 的接口 10.1.1.3，而不是它自己的接口。注意，由于路由器 Lipan 不运行 RIP 协议，而路由器 Jicarilla 和 Mescalero 不运行 BGP 协议，这样即使在同一个子网中路由器 Lipan 是可达的，路由器 Jicarilla 和 Mescalero 也没有办法直接知道路由器 Lipan 是最优的下一跳路由器。

[1] 该图是根据 RFC 1722 中 Gary Malkin 演示的一个例子改编的。

[2] 重新分配是指把从一个路由选择协议学习到的路由通告给另一个路由选择协议的情况，这将在第 11 章中详细讨论。

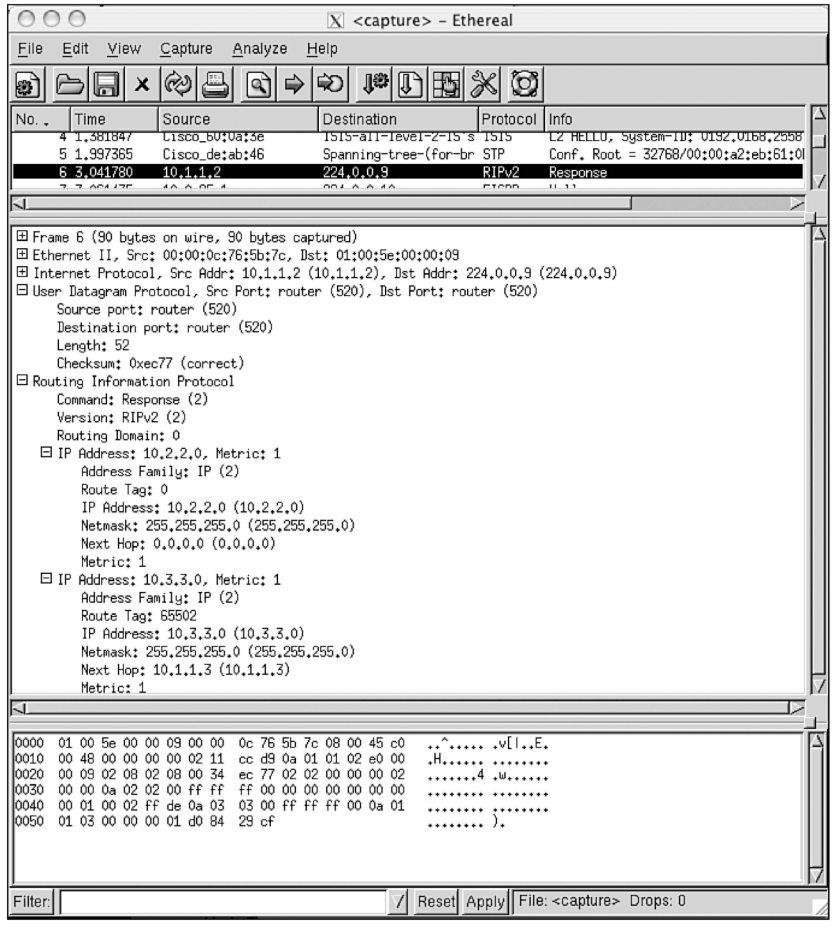


图 6-3 从路由器 Chiricahua 上捕获的 RIPv2 更新信息显示了通告子网 10.3.3.0 时所使用的路由标记、子网掩码和下一跳字段

6.1.2 与 RIPv1 的兼容性

RIPv1 使用灵活的方式来进行路由更新。如果更新消息的版本字段指出 RIP 是版本 1，但所有未使用字段（unused field）的所有位都被设置为 1，那么这个更新消息将被丢弃；如果版本字段设置大于 1，在版本 1 中定义为未使用的字段将被忽略，但会处理这个消息。结果是，像 RIPv2 这种新版本协议就可以向后兼容 RIPv1。

RFC1723 用 4 个设置定义了一个“兼容性开关”，用来允许版本 1 和版本 2 之间的互操作：

- **RIP-1**——只有 RIPv1 的消息被传送。
- **RIP-1 兼容性**——RIPv2 使用广播方式代替多播方式来通告消息，以便 RIPv1 可以接收它们。
- **RIP-2**——RIPv2 使用多播方式将消息通告到目的地址 224.0.0.9。
- **None**——不发送更新。

RFC 建议这些开关基于接口模式上进行配置。在 6.3 节中指出采用 Cisco 的命令可以将

它们的值设置为 1~3，将值设置为 4 已由 **passive-interface** 命令完成。

另外，RFC1723 定义了一个“接收控制开关”来控制更新的接收。对于这个开关，4 种建议的设置是：

- **RIP-1 Only；**
- **RIP-2 Only；**
- **Both；**
- **None。**

此开关也是基于每一个接口上配置的。在 6.3 节中指出了采用 Cisco 的命令可以将它们的值设置为 1～3，将值设置为 4 可以通过使用访问列表来过滤 UDP 源端口号 520，或者在该接口上不包含 **network** 语句，[1]或者配置一个路由过滤列表来完成，后者将在第 13 章中讨论。

6.1.3 无类别路由查找

第5章阐述了有类别路由的查找方法——首先将目的地址与路由选择表中的主网地址匹配，然后匹配主网的子网。如果经过这些步骤找不到匹配项，这个数据包就会被丢弃。

在 IOS11.3 版本之前，IOS 的缺省方式都是有类别路由查找，但缺省路由的查找改成了无类别查找。对于早期的 IOS 版本，即使像 RIPv1 和 IGRP 这样的有类别路由选择协议，读者也可以通过全局命令 **ip classless** 来启用无类别路由查找。当路由器执行无类别路由查找时，它不会注意目的地址的类别，替代方式是，它会在目的地址和所有已知的路由之间逐位（bit-by-bit）执行最佳匹配。当和缺省路由一起工作时，这个特性将变得非常有用，这将在第 12 章中讨论。连同其他一些无类别路由选择协议的特性，无类别路由查找的功能将显得更加强大。

6.1.4 无类别路由选择协议

无类别路由选择协议最根本的特点是，它可以在路由通告中具有携带子网掩码的能力。每条路由拥有子网掩码的一个好处就是，全 0 和全 1 的子网现在可以利用了。在第 1 章中说明了有类别路由选择协议不能区分全 0 子网（例如 172.16.0.0）和主网络号（172.16.0.0）；同样地，它们也不能区分全 1 子网的广播（172.16.255.255）和全部子网的广播（172.16.255.255）。

如果包含了子网掩码，这个困难就不存在了。我们可以容易地看出网络 172.16.0.0/16 是一个主网络号，而 172.16.0.0/24 则是一个全 0 的子网。172.16.255.255/16 和 172.16.255.255/24 也可以同样地区分开来。

在缺省的条件下，即使运行的是无类别路由选择协议，Cisco IOS 软件也将拒绝试图把一个全 0 的子网配置为有效的地址/掩码组合。为了使这个缺省的行为无效，可以使用全局命令 **ip subnet-zero** 进行更改。

---

[1] 这种方法只适用于下面的情形，即路由器上没有其他运行 RIP 协议的接口和相同的主网相连。

每条路由拥有子网掩码的另一个很大的好处就是，可以使用可变长子网掩码和利用一条单一的聚合地址来汇总一组主网络地址。可变长子网掩码将在下一小节讲述，地址聚合（或超网）将在第 7 章中介绍。

6.1.5 可变长子网掩码（VLSM）

如果每一个目的地址都可以单独地携带相关联的子网掩码通告到整个网络中，那么就没有什么理由要求所有的掩码必须是等长的了。这个事实就是可变长子网掩码（VLSM）的基础。

图 6-4 显示了一个可变长子网掩码的简单应用。图中网络的每一条数据链路都必须有一个惟一标识的子网掩码，同时，每一个子网地址段必须包含足够的主机地址数来提供给这条数据链路上相连的设备使用。

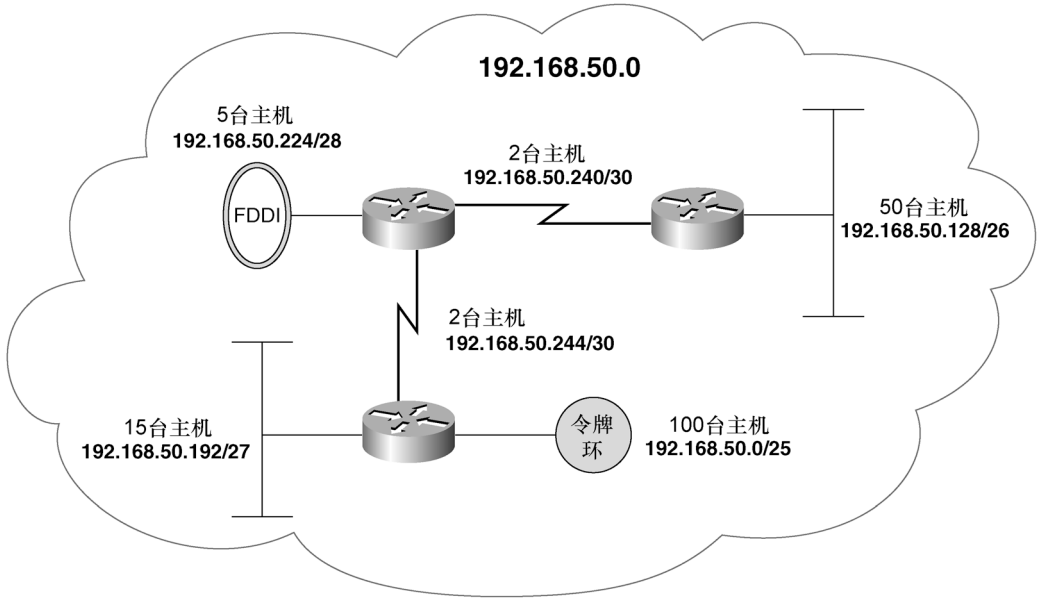


图 6-4 使用 VLSM，C 类地址可以被分成子网，以便提供给这个网络和它的每一条数据链路上的主机使用

考虑分配给这个网络的 C 类网络地址，没有 VLSM 技术是不能完成子网划分的。这里令牌环的那个子网需要 100 个主机地址，也就是需要一个 25 位的掩码（有 1 位被子网化）；任何更长一点的掩码都将无法保留有足够的主机位。但是如果所有的子网掩码都必须是等长的话，那么从这个 C 类地址中只能再分出一个子网。[1]这样将没有办法提供足够的子网数以满足需要。

使用 VLSM，在图 6-4 的网络中不相的主机地址只需要 1 个 C 类的网络地址就可以满足了。表 6-1 显示了这些子网和每个子网内可用的地址范围。

[1] 这个说法假定全 0 和全 1 的子网是可以被路由的，因为对于只有 1 位被子网化时，全 0 和全 1 子网是惟一可用的子网。

表 6-1 图 6-4 的子网

子网/掩码	地址范围	广播地址
192.168.50.0/25	192.168.50.1–192.168.50.126	192.168.50.127
192.168.50.128/26	192.168.50.129–192.168.50.190	192.168.50.191
192.168.50.192/27	192.168.50.193–192.168.50.222	192.168.50.223
192.168.50.224/28	192.168.50.225–192.168.50.238	192.168.50.239
192.168.50.240/30	192.168.50.241–192.168.50.242	192.168.50.243
192.168.50.244/30	192.168.50.245–192.168.50.246	192.168.50.247

很多人，包括许多使用 VLSM 的人利用这项技术解决的问题比上述例子复杂得多。使用 VLSM 技术的关键之处就是，当一个网络地址依据某种标准方式被划分成子网以后，那些子网本身还能够进一步被子网化。事实上，有时候我们会偶尔听到 VLSM 关于“子网的子网化”的说法。

仔细察看表 6-1 的地址（通常是二进制的），就可以发现 VLSM 是怎样工作的。[1]首先，使用一个 25 位的掩码把网络地址划分成两个子网：192.168.50.0/25 和 192.168.50.128/25。第一个子网可以提供 126 个主机地址满足图 6-4 中令牌环网段的需要。

根据第 1 章所讲述的，我们了解到划分子网可以包括扩展的缺省网络掩码，因此一些主机位可以被视为网络位。这种做法同样可以应用于剩下的子网 192.168.50.128/25。其中一个以太网段需要 50 个主机地址，因此余下的子网掩码应该扩展到 26 位，这一步提供了两个子网的子网（sub-subnets）——192.168.50.128/26 和 192.168.50.192/26，每一个小子网都可以提供 62 个可用的主机地址。第一个小子网可以给前面那个大的以太网段使用，剩下的第二个小子网可以进一步子网化提供给其他数据链路使用。

这个过程再重复两次，就可以提供必需的满足大小的子网给小的以太网段和 FDDI 环网段使用。剩余的子网 192.168.50.240/28 可以用作两个串行链路所需要的子网。对于任何点到点的链路，最普遍的情况只需要两个主机地址——每端一个地址。使用 30 位的掩码可以创建这两个串行链路的子网，每个子网正好包括两个可用的主机地址。

点到点的链路需要子网地址，但每个子网只需要两个主机地址，这就是使用 VLSM 的一个理由。例如，图 6-5 显示了一个典型的广域网络的拓扑结构，它将每一个远程路由器都通过帧中继 PVC 与中心路由器（hub router）相连。现代网络设计的经验通常建议在点到点的子接口 [2]上配置每个 PVC 电路。如果没有 VLSM 技术，将必须有相同大小的子网，而这个子网的大小却是主机设备数量最多的子网所需要的地址数。

假设图 6-5 中的网络使用了一个 B 类地址，并且每一台路由器都和几个局域网相连，而这些局域网连接的设备数量最大为 175 台。包含每个 PVC 电路的子网都需要一个 24 位的掩码，这样对网络中的每一个 PVC 链路来说，就有 252 个地址浪费了。使用 VLSM 技术，选用单个子网并使用 30 位的掩码对子网进行子网化就可以满足需要了，可以划分足够的子网最多可以满足 64 个点到点的链路（如图 6-6 所示）。

VLSM 地址设计的例子在本章和后续的章节中都有描述。第 7 章介绍了使用 VLSM 技术的另一个主要用途，即层次化的编址和地址聚合。

---

1 强烈建议读者把这个例子的所有地址转换成二进制的。

2 子接口超出了本书的讲解范围，还不熟悉这些有用工具的读者可以参考 Cisco 配置手册。

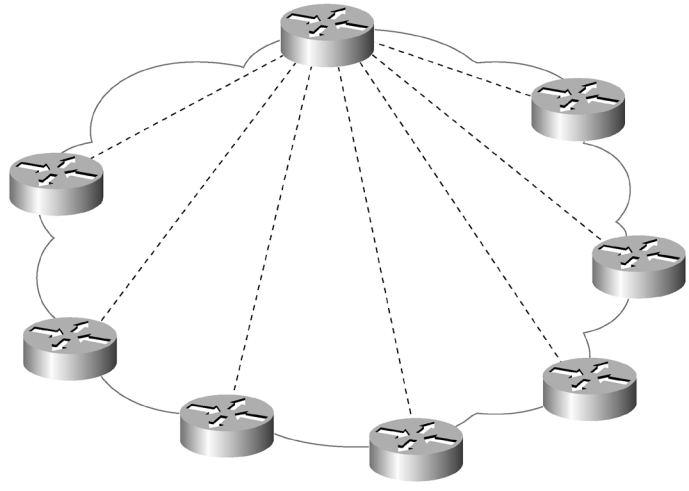


图 6-5 VLSM 允许这些 PVC 虚电路中的每一个电路都配置一个单独的子网，而不浪费主机地址

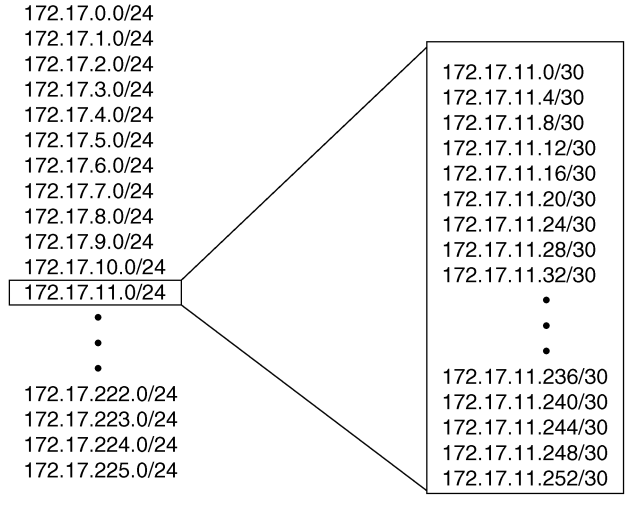


图 6-6 这个 B 类地址使用一个 24 位掩码来划分子网，对子网 172.17.11.0 使用 30 位的掩码进一步地划分更小的子网，结果可以为相关的点到点链路提供 64 个子网

6.1.6 认证

涉及到所有路由选择协议的安全问题，是指一台路由器接受非法路由选择更新消息的可能性。非法的更新消息可能来自试图破坏网络的攻击者，或试图通过欺骗路由器发送数据包到一个错误的目的地的方法来捕获数据包。更普遍的有害更新消息来自出现故障的路由器。RIPv2 协议能够通过更新消息所包含的口令来验证某个路由选择更新消息源的合法性。

RIPv2 是通过更改 RIP 消息中正常情况下应该是第一个路由条目的字段来支持认证的，如图 6-7 所示。在含有认证的单个更新消息中，最大可以携带的路由条目被减少到了 24 个。认证是通过设置地址族标识字段为全 1（0xFFFF）来标识的。对于简单的口令认证，认证的类型（authentication type）是 2（0x0002），剩余的 16 个八位组字节字段携带一个最多有 16 个字符的口令，可以由数字和字母混合组成。口令在字段中按照左对齐的方式，如果一个口令小于 16 个八位组字节的长度，那么字段中没有使用的位被设置成 0 来填充。

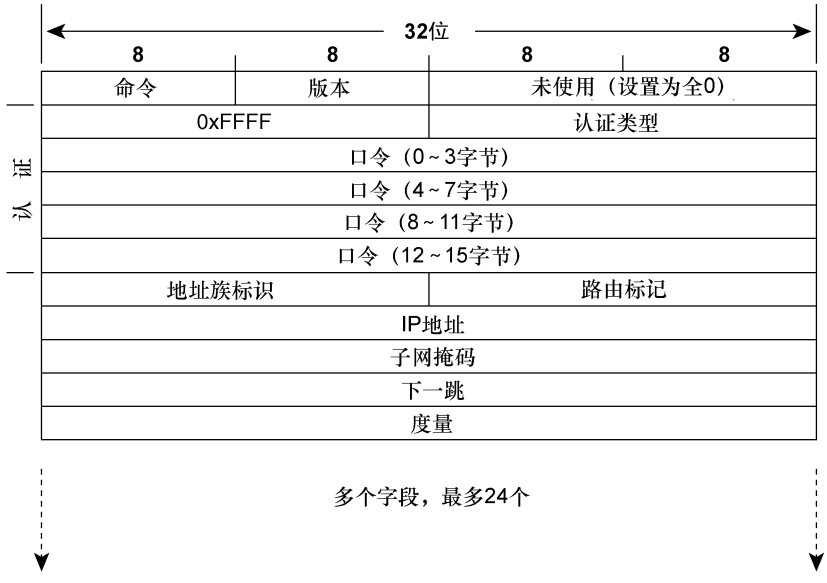


图 6-7 RIPv2 的认证信息，配置是加载在第一个路由条目的字段空间上的

图 6-8 显示了协议分析仪捕获到的一个包含认证的 RIPv2 消息。该图也显示了使用缺省的 RIP 认证的一个安全隐患：口令是以明文方式传输的。任何人捕获到包含 RIPv2 更新消息的数据包，都可以读出它的认证口令。

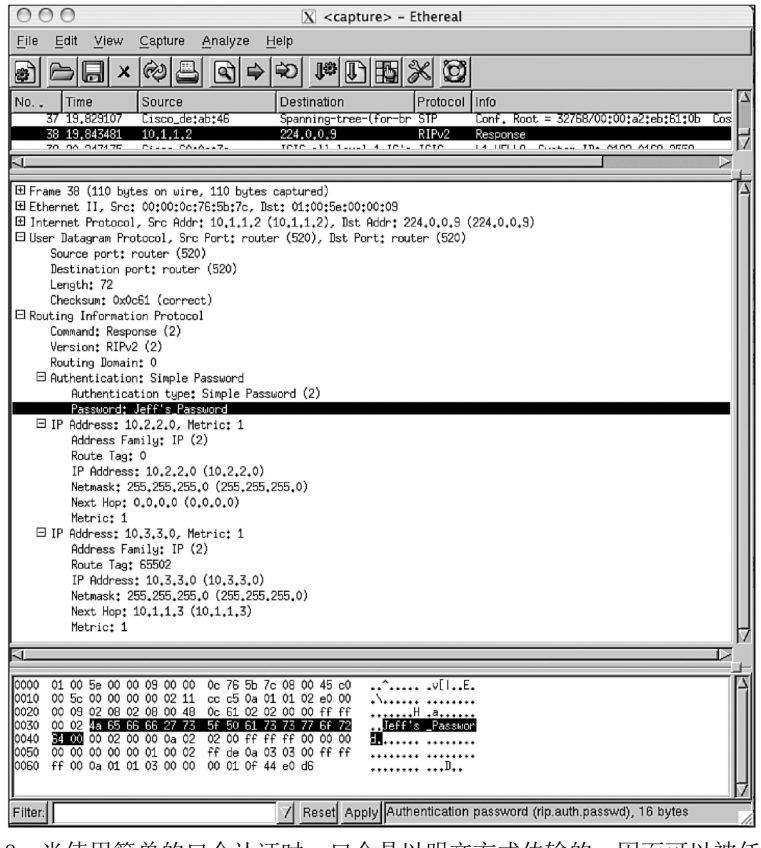


图 6-8 当使用简单的口令认证时，口令是以明文方式传输的，因而可以被任何人通过侦探包含更新的数据包读出口令

虽然 RFC1723 只描述了简单口令认证，但却很有远见地提供了一个认证类型字段。Cisco IOS 软件就是利用这个特定字段，提供了用 MD5 认证来替代简单口令认证的选项。[1]

Cisco 使用了第一个和最后一个路由条目的字段空间，从而达到了 MD5 认证的目的。

MD5 是一个单向的消息摘要（message digest）算法或安全散列函数（secure hash function），由 RSA Data Security,Inc 提出。有时候 MD5 也被作为一个加密校验和（cryptographic checksum），因为它的工作方式和算术校验和（arithmetic checksum）有点相似。MD5 算法是通过一个随意长度的明文消息（例如，一个 RIPv2 的更新消息）和口令计算出一个 128 位的 hash 值的。这个“指纹”随同消息一起传送。拥有相同口令的接收者会计算它自己的 hash 值，如果消息的内容没有被更改，接收者的 hash 值应该和消息中发送者的 hash 值相匹配。

图 6-9 显示了图 6-8 中的同一台路由器的更新消息，但是含有了 MD5 认证。这里的认证类型是 3，并且看不到口令。

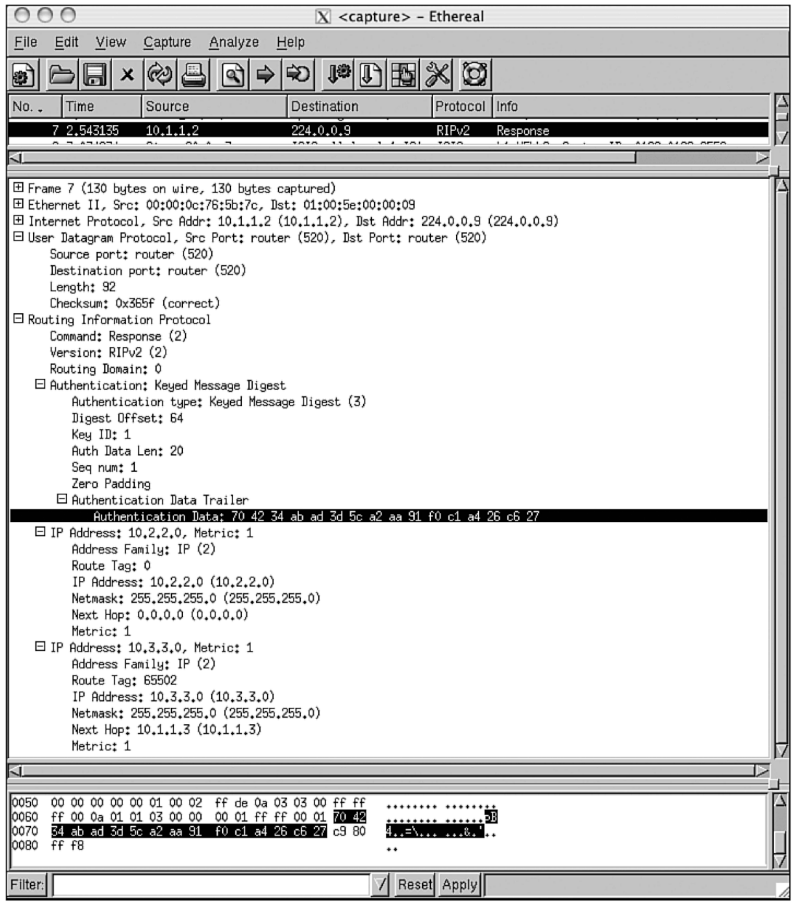


图 6-9 这个更新消息和图 6-8 中的更新来自于同一台路由器，但是它使用了 MD5 认证

[1] MD5 是在 RFC1321 中描述的。欲更好地了解 MD5，请参阅下面这本书：*Network Security: Private Communication in a Public World* 的第 120～122 页，由 Charlie Kaufman、Radia Perlman 和 Mike Spencer 撰写，1995 年 Prentice Hall 出版。

6.2 RIPng 的基本原理与实现

支持 IPv6 的 RIPng 协议虽然是基于 RIPv2 协议的，但它并不是 RIPv2 的简单扩展，它实际上是一个完全独立的协议。RIPng 协议不支持 IPv4，因此读者如果同时在 IPv4 和 IPv6 环境里使用 RIP 作为路由选择协议，就必须运行支持 IPv4 的 RIPv1 或 RIPv2，以及支持 IPv6 的 RIPng。

RIPng 使用与 RIPv2 相同的计时器、过程处理和消息类型。例如，RIPng 像 RIPv2 一样，使用 30s 的更新计时器抖动来避免消息同步，还有 180s 的超时周期、120s 的垃圾收集计时器和 180s 的抑制计时器。它也使用相同的跳数度量，16 跳表示不可到达。RIPng 也用与 RIPv2 相同的方式使用请求和响应消息。另外，除了类似于 RIPv1 和 RIPv2 一样用到少数单播方式外，像 RIPv2 一样，RIPng 大多是以多播方式收发请求和响应消息。RIPng 使用的 IPv6 多播地址是 FF02::9。

除了上述这些类似的功能外，一个例外之处是认证功能。RIPng 本身并没有认证机制，但是承担认证功能的特性已经集成到 IPv6 中了。

当然，RIPng 也不需要像 RIPv2 那样要求具有对 RIPv1 的兼容性开关，因为它本来就不向后支持 IPv4 协议。

如图 6-10 所示，图中显示了 RIPng 的消息格式。和 RIPv1 与 RIPv2 运行在 UDP 端 520 上不同，RIPng 发送和接收消息都是运行在 UDP 端口 521 上的。与 RIPv1 和 RIPv2 的另一个不同之处是它没有设定消息的大小。在这里，消息的大小仅仅依赖于发送它的链路的 MTU 值。



图 6-10 RIPng 的消息格式

- **命令**——总是设置为 1 或 2，1 表示本消息是请求消息，2 表示本消息是响应消息。
- **版本号**——总是设置为 1，RIPng 当前的版本是 RIPngv1。
- **IPv6 前缀**——是指路由条目的 128 位的目的 IPv6 前缀。
- **路由标记**——与 RIPv2 中 16 位的路由标记字段的用法相同：用来标记经过 RIP 路由域传送的外部路由属性。
- **前缀长度**——是一个 8 位的字段，用来指出 IPv6 前缀字段中的地址的有效位数。例如，假如通告的前缀是 3ffe:2100:1201::/48，那么它的前缀长度字段的值是 48（0x30）。如果所通告的是缺省路由，它的 IPv6 前缀是 0:0:0:0:0:0:0:0，前缀长度是 0。
- **度量**——与 RIPv1 和 RIPv2 中一样，是指跳数度量。但是当最大可能值为 16 时，这个字段会把 RIPv1 和 RIPv2 里不必要的 16 位大小的位数减小为 8 位。

RIPng 使用和 RIPv2 相同的方式指定下一跳地址。换句话说，如果是一个有效的非零下一跳地址，表示下一跳路由器不同于发起响应消息的路由器；如果下一跳地址是全 0（0:0:0:0:0:0:0:0），则表示下一跳路由器就是发起响应消息的路由器本身。但是，与 RIPv2 中每一个路由条目都跟随一个下一跳地址不同，RIPng 中只在一个专门的路由条目里指定下一跳地址，并把所有使用这个下一跳地址的路由条目编成组，跟在这个专门的路由条目后面。也就是说，下一跳路由条目指定的下一跳地址应用于跟随在它后面的所有路由条目，要么到这个响应消息的末尾，要么直到发现另一个专门的下一跳路由条目。

如图 6-11 所示，图中显示了下一跳路由条目的格式。它的 128 位地址要么是另一台路由器的 IPv6 地址，要么是::——表示发起该消息的路由器的地址。路由标记和前缀长度字段都设置为全 0。由于接收路由器的度量字段设置为全 1（0xFF），因此它可以识别这是一个下一跳路由条目。

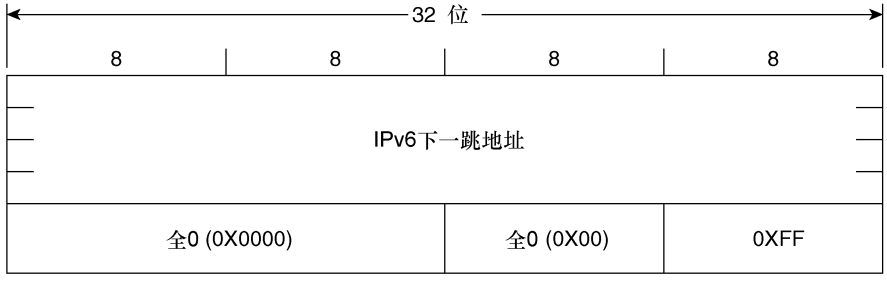


图 6-11 设置为全 1 的度量值表示该路由条目是一个下一跳路由条目。所有跟在这个条目后的路由条目都使用这个下一跳地址，除非是在该消息的末尾或者出现另一个下一跳路由条目

6.3 RIPv2 的配置

由于 RIPv2 是 RIPv1 的增强版，而不是一个单独的协议，因此，在第 5 章中介绍的某些命令都可以以同样的方式在 RIPv2 中正确使用，例如计时器和度量的操作、单播更新或根本不发出更新的配置等。当浏览 RIPv2 协议的配置后，本节余下来的部分将集中讲述一些新的扩展特性的配置。

6.3.1 案例研究：RIPv2 的基本配置

缺省时，在 Cisco 路由器上配置一个 RIP 进程将只发送 RIPv1 的消息，但是同时接收 RIPv1 和 RIPv2 的消息。这个缺省的配置可以通过命令 **version** 来更改，参见示例 6-1。

示例 6-1 命令 version 2 使 RIP 只能接收和发送 RIPv2 的消息

```
router rip
 version 2
 network 172.25.0.0
 network 192.168.50.0
```

在这种配置方式下，路由器只发送和接收 RIPv2 的消息。同样地，路由器也可以配置成只发送和接收 RIPv1 消息的方式，参见示例 6-2。

示例 6-2 命令 version 1 使 RIP 只能接收和发送 RIPv1 的消息

```
router rip
 version 1
 network 172.25.0.0
 network 192.168.50.0
```

可以在路由器配置模式（config-router mode）下输入命令 **no version** 恢复到原来的缺省方式。

6.3.2 案例研究：与 RIPv1 的兼容性

RFC 2453 中建议的基于接口模式下的“兼容性开关”，在 Cisco IOS 软件中可以通过命令 **ip rip send version** 和 **ip rip receive version** 来实现。

在图 6-12 所示的网络中包含了同时宣告 RIPv1 和 RIPv2 的路由器。另外，主机 Pojoaque 是一台运行“routed”进程的 Linux 主机，它只能理解和处理 RIPv1。路由器 Taos 的配置参见示例 6-3。

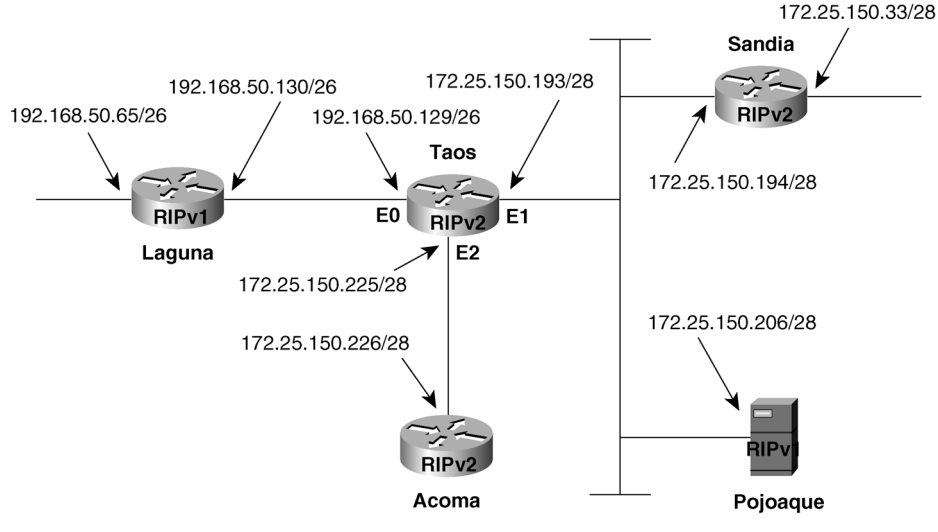


图 6-12 路由器 Taos 正在运行 RIPv2，但是必须宣告版本 1 给其他一些设备

示例 6-3 路由器 Taos 的配置

```
interface Ethernet0
 ip address 192.168.50.129 255.255.255.192
 ip rip send version 1
 ip rip receive version 1
!
interface Ethernet1
 ip address 172.25.150.193 255.255.255.240
 ip rip send version 1 2
!
interface Ethernet2
 ip address 172.25.150.225 255.255.255.240
!
router rip
 version 2
 network 172.25.0.0
 network 192.168.50.0
```

因为路由器 Laguna 是 RIPv1 的一个宣告者，所以路由器 Taos 的 E0 接口要配置成接收和发送 RIPv1 更新的方式，而 E1 接口则配置成同时支持版本 1 和版本 2 更新的方式，以便发送给运行 RIPv1 的路由器 Pojoaque 和运行 RIPv2 的路由器 Sandia。E2 接口没有什么特别的配置，它将按照缺省方式发送和接收版本 2 的更新。

在示例 6-4 中，使用 **debug ip rip** 命令观察路由器 Taos 发送和接收的消息。这里有几个有趣的地方。首先，注意观察 RIPv1 和 RIPv2 消息携带的内容不同。RIPv2 的更新中可以看到地址掩码、下一跳和路由标记字段（在这个实例中，都被设置成全 0）。其次，可以观察到，E1 接口正在以广播方式发送 RIPv1 更新，而以多播方式发送 RIPv2 更新。第三，由于路由器 Taos 没有配置接收 RIPv1，从而来自路由器 Pojoaque（172.25.150.206）的更新将被忽略（路由器 Pojoaque 被错误地配置，并且正在广播它的路由表）。[1]

示例 6-4 使用调试功能，可以从路由器 Taos 上看到 RIP 版本的发送和接收

```
Taos#debug ip rip
RIP protocol debugging is on
Taos#
RIP: received v2 update from 172.25.150.194 on Ethernet1
     172.25.150.32/28 - 0.0.0.0 in 1 hops
RIP: ignored v1 packet from 172.25.150.206 (illegal version)
RIP: sending v1 update to 255.255.255.255 via Ethernet0 (192.168.50.129)
     network 172.25.0.0, metric 1
RIP: sending v1 update to 255.255.255.255 via Ethernet1 (172.25.150.193)
     subnet 172.25.150.224, metric 1
     network 192.168.50.0, metric 1
RIP: sending v2 update to 224.0.0.9 via Ethernet1 (172.25.150.193)
     172.25.150.224/28 - 0.0.0.0, metric 1, tag 0
     192.168.50.0/24 - 0.0.0.0, metric 1, tag 0
RIP: sending v2 update to 224.0.0.9 via Ethernet2 (172.25.150.225)
     172.25.150.32/28 - 0.0.0.0, metric 2, tag 0
     172.25.150.192/28 - 0.0.0.0, metric 1, tag 0
     192.168.50.0/24 - 0.0.0.0, metric 1, tag 0
RIP: received v1 update from 192.168.50.130 on Ethernet0
     192.168.50.64 in 1 hops
RIP: received v2 update from 172.25.150.194 on Ethernet1
     172.25.150.32/28 - 0.0.0.0 in 1 hops
```

在示例 6-4 中，可能最需要关注的就是广播到主机 Pojoaque 的更新了，它没有包含子网

[1] 实际上，这个例子中使用“**routed –s**“选项是故意配置错误的。

172.25.150.32。路由器 Taos 是通过多播方式的 RIPv2 更新从路由器 Sandia 学习这个子网的。但是主机 Pojoaque 由于只宣告 RIPv1 而不能接收这些多播。此外，虽然路由器 Taos 得知这个子网，但是水平分隔法则禁止路由器 Taos 把从这个接口学到的路由再从相同的接口通告出去。

因此，主机 Pojoaque 无法得知子网 172.25.150.32。这里有可供使用的两种修正方法：第一，把路由器 Sandia 配置成可以同时发送 RIP 协议的两个版本；第二，可以通过示例 6-5 的配置在路由器 Taos 的 E1 接口上关闭水平分隔。

示例 6-5 在路由器 Taos 的配置中关闭了水平分割

```
interface Ethernet1
 ip address 172.25.150.193 255.255.255.240
 ip rip send version 1 2
 no ip split-horizon
```

示例 6-6 中显示了更改配置后的结果。现在路由器 Taos 在它的更新里包含了子网 172.25.150.32。可以预见到关闭水平分隔后的一些可能的结果：路由器 Taos 现在不仅通告子网 172.25.150.32 给主机 Pojoaque，而且把这个子网通告回路由器 Sandia。

示例 6-6 在 E1 接口上关闭水平分隔后，路由器 Taos 在通告给主机 Pojoaque 的更新中包含了子网 172.25.150.32

```
Taos#debug ip rip
RIP protocol debugging is on
Taos#
RIP: ignored v1 packet from 172.25.150.206 (illegal version)
RIP: received v2 update from 172.25.150.194 on Ethernet1
     172.25.150.32/28 -> 0.0.0.0 in 1 hops
RIP: sending v1 update to 255.255.255.255 via Ethernet0 (192.168.50.129)
     network 172.25.0.0, metric 1
RIP: sending v1 update to 255.255.255.255 via Ethernet1 (172.25.150.193)
     subnet 172.25.150.32, metric 2
     subnet 172.25.150.224, metric 1
     subnet 172.25.150.192, metric 1
     network 192.168.50.0, metric 1
RIP: sending v2 update to 224.0.0.9 via Ethernet1 (172.25.150.193)
     172.25.150.32/28 -> 172.25.150.194, metric 2, tag 0
     172.25.150.224/28 -> 0.0.0.0, metric 1, tag 0
     172.25.150.192/28 -> 0.0.0.0, metric 1, tag 0
     192.168.50.0/24 -> 0.0.0.0, metric 1, tag 0
```

6.3.3 案例研究：使用 VLSM

参见图 6-12，子网 172.25.150.0/24 已经分配给图中的网络，这个子网通过扩展到 28 位的掩码进一步子网化以满足不同的数据链路。表 6-2 中显示了以二进制和点分十进制表示的可用子网。根据公式 2^n-2，每一个子网[1]可以含有 14 个主机地址。在这些子网中，172.25.150.32、172.25.150.192 和 172.25.150.224 已经被使用。

表 6-2 VLSM 应用于子网 172.25.150.0/24

二进制表示	点分十进制表示
11111111111111111111111111110000	255.255.255.240

[1] 现在，子网的概念应该比较熟悉了，从这里开始，子网作为一个单纯的术语将用来表示一个子网、一个子网的子网、一个子网的子网的子网，等等。

续表

二进制表示	点分十进制表示
10101100000110011001011000000000	172.25.150.0/28
10101100000110011001011000010000	172.25.150.16/28
10101100000110011001011000100000	172.25.150.32/28
10101100000110011001011000110000	172.25.150.48/28
10101100000110011001011001000000	172.25.150.64/28
10101100000110011001011001010000	172.25.150.80/28
10101100000110011001011001100000	172.25.150.96/28
10101100000110011001011001110000	172.25.150.112/28
10101100000110011001011010000000	172.25.150.128/28
10101100000110011001011010010000	172.25.150.144/28
10101100000110011001011010100000	172.25.150.160/28
10101100000110011001011010110000	172.25.150.176/28
10101100000110011001011011000000	172.25.150.192/28
10101100000110011001011011010000	172.25.150.208/28
10101100000110011001011011100000	172.25.150.224/28
10101100000110011001011011110000	172.25.150.240/28

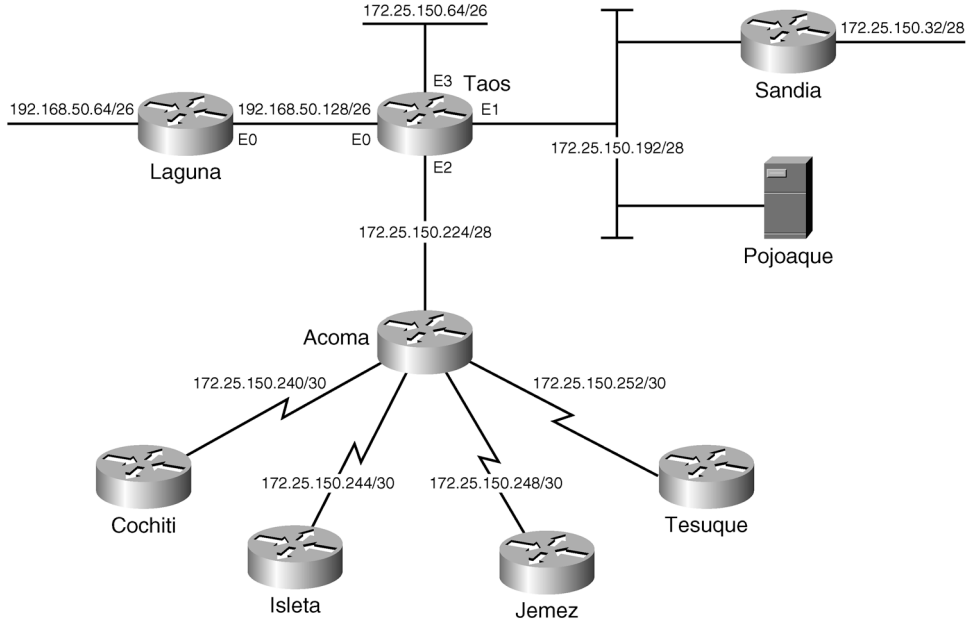


图 6-13　VLSM 技术可以用来满足单个数据链路的地址需求

在图 6-13 中，路由器 Taos 增加了一个新的以太网段 Ethernet3，它包含了 60 台主机。为了给这个数据链路分配地址，需要至少有 6 位主机位的子网。有类别路由选择协议如果使用次一级的地址，应该需要分配 5 个表 6-2 中划分的子网给以太网段 Ethernet3[5×（2^4－2）=70]。利用无类别路由选择协议和 VLSM 技术，表 6-2 中的 4 个子网可以汇总成一个单一的带有 26

位掩码的子网。这一个步骤可以提供 6 个主机位（62 个主机地址），并且没有必要使用次一级的编址。这里的 4 个子网 172.25.150.64/28~172.25.150.112/28 可以合并成单个 26 位掩码的子网 172.25.150.64/26。注意，这里的 4 个子网并不是随意选择的，在上面的 16 个子网组合中，它们开始的 26 位掩码必须是相同和惟一的。[1]

参见图 6-13，图中网络增加了 4 台路由器和 4 条串行链路。如果没有使用 VLSM 技术，这 4 条串行链路就需要使用表 6-2 中的 4 个子网；而使用 VLSM 技术，只要使用表 6-2 中的 1 个子网就可以满足所有 4 条串行链路的需求了。选用子网 172.25.150.240，利用 30 位的掩码创建表 6-3 中的子网，这 4 个更小的子网的任何一个都包含两个主机地址。

表 6-3 应用 30 位的掩码对子网 172.25.150.240 再进行子网划分

二进制表示	点分十进制表示
11111111111111111111111111111100	255.255.255.252
10101100000110011001011011110000	172.25.150.240/30
10101100000110011001011011110100	172.25.150.244/30
10101100000110011001011011111000	172.25.150.248/30
10101100000110011001011011111100	172.25.150.252/30

划分子网的基本目的总是相同的：路由器必须能够使用惟一的地址来标识每一条数据链路，以区别于网络中的其他地址，这也是前面两个例子的共同目的。在第一个例子中，通过减小掩码的大小，将多个地址合并成一个地址，该操作一直进行到所有地址都具有相同的位。注意，这种情况在子网被汇总成主网络地址时也会发生。在第二个例子中，通过扩展子网掩码将单个子网划分为多个更小的子网。

6.3.4 案例研究：不连续的子网和无类别路由选择

图 6-14 显示出新添了 4 台路由器，并且每一台路由器都连接两个以太网段。其中每一处都包含一个属于子网 172.25.150.0/24 的以太网段，并且都不超过 12 台主机。这个需求很容易满足，从表 6-2 中选用 4 个未用的子网进行分配即可。

每台新添路由器下挂的另一个以太网属于子网 192.168.50.0，并且都不超过 25 台主机。子网 192.168.50.64/26 和 192.168.50.128/26 正在被其他链路使用，只剩下子网 192.168.50.0/26 和 192.168.50.192/26。通过增加掩码位到 27 位，这两个子网就被划分成了 4 个子网，每个子网有 5 个主机位——每个子网可以提供 30 个主机地址。表 6-4 显示了这 4 个子网的二进制表示。

表 6-4 使用 27 位掩码对子网 192.169.50.0/26 进一步划分

二进制表示	点分十进制表示
11111111111111111111111111100000	255.255.255.224
11000000101010001100100000000000	192.169.50.0/27
11000000101010001100100000100000	192.169.50.32/27
11000000101010001100100011000000	192.169.50.192/27
11000000101010001100100011100000	192.169.50.224/27

[1] 将几个地址合并成一个地址的技巧将在第 7 章的地址聚合中介绍。

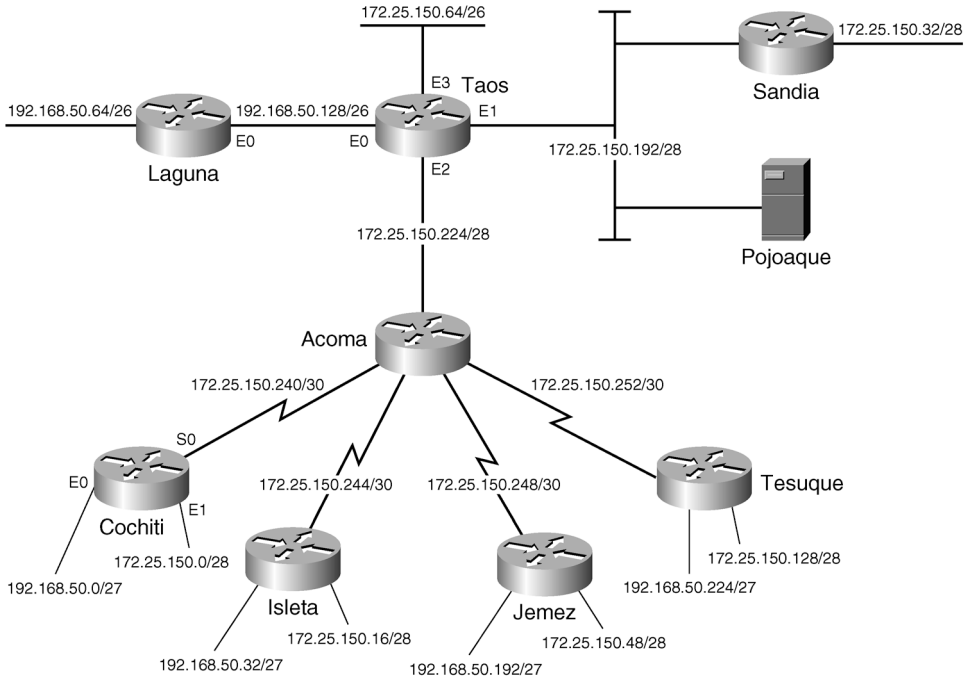


图 6-14 路由器 Cochiti、Isleta、Jemez 和 Tesuque 中的每一台连接了两个以太网段。每台路由器上都包含一个属于子网 172.25.150.0/24 的以太网，另一个则属于子网 192.168.50.0/24

分配完所有的子网地址后，下一个需关注的就是这样一个事实：192.168.50.0 的子网是不连续的。第 5 章展示了一个不连续子网的实例，并演示了怎样使用辅助接口连接它们的。而无类别路由选择协议没有关于不连续子网的这些困难。因为每一条路由更新都包含一个子网掩码，因而一个主网络的子网能够通告给其他的主网络。

但是，RIPv2 协议在缺省情况下会在主网络边界上进行路由汇总，这一点和 RIPv1 相同。为了关闭路由汇总功能以允许被通告的子网通过主网络边界，可以在 RIP 的处理中使用 **no auto-summary** 命令。路由器 Cochiti 的配置参见示例 6-7。

示例 6-7 路由器 Cochiti 的配置，不再支持路由自动汇总

```
interface Ethernet0
 ip address 192.168.50.1 255.255.255.224
!
interface Ethernet1
 ip address 172.25.150.1 255.255.255.240
!
interface Serial0
 ip address 172.25.150.242 255.255.255.252
!
router rip
 version 2
 network 172.25.0.0
 network 192.168.50.0
 no auto-summary
```

路由器 Isleta、Jemez 和 Tesuque 也可以进行类似的配置。在路由器 Taos 和 Acoma 上也必须要关闭路由汇总。回忆图 6-12 中，路由器 Laguna 运行的版本是 RIPv1，为了使这个配置能够正常工作，必须将 RIP 的版本更改为版本 2。

读者仔细考虑一下，可变长子网掩码对仍旧运行 RIPv1 的主机 Pojoaque 产生了什么影响。示例 6-8 中的调试信息显示了路由器 Taos 发送到子网 172.25.150.192/28 上的版本 1 和版本 2 的更新消息。版本 1 的更新消息仅仅包含那些 28 位掩码的子网，这与正在广播更新的子网的掩码是一样的。虽然主机 Pojoaque 不接收子网 172.25.150.64/26 或所有串行链路的子网的通告，但是在这个实例中，关于那些子网地址的分析显示，主机 Pojoaque 依然可以正确地识别出这些子网与它本身的子网地址不同。到达这些子网的数据包也会送到路由器 Taos 上进行路由选择。

示例 6-8　虽然来自路由器 Taos 的 RIPv2 更新包含了图中网络的所有子网，但是 RIPv1 的更新消息却仅仅包含到达网络 192.168.50.0 的汇总路由和网络 172.25.150.0 的一些子网（这些子网的掩码和发送这些更新的接口掩码相同）

```
Taos#debug ip rip
RIP protocol debugging is on
RIP: sending v1 update to 255.255.255.255 via Ethernet0 (172.25.150.193)
     subnet 172.25.150.0, metric 3
     subnet 172.25.150.16, metric 3
     subnet 172.25.150.32, metric 2
     subnet 172.25.150.48, metric 3
     subnet 172.25.150.128, metric 3
     subnet 172.25.150.192, metric 1
     subnet 172.25.150.224, metric 1
     network 192.168.50.0, metric 1
RIP: sending v2 update to 224.0.0.9 via Ethernet0 (172.25.150.193)
     172.25.150.0/28 -> 0.0.0.0, metric 3, tag 0
     172.25.150.16/28 -> 0.0.0.0, metric 3, tag 0
     172.25.150.32/28 -> 0.0.0.0, metric 2, tag 0
     172.25.150.48/28 -> 0.0.0.0, metric 3, tag 0
     172.25.150.64/26 -> 0.0.0.0, metric 1, tag 0
     172.25.150.128/28 -> 0.0.0.0, metric 3, tag 0
     172.25.150.192/28 -> 0.0.0.0, metric 1, tag 0
     172.25.150.224/28 -> 0.0.0.0, metric 1, tag 0
     172.25.150.240/30 -> 0.0.0.0, metric 2, tag 0
     172.25.150.244/30 -> 0.0.0.0, metric 2, tag 0
     172.25.150.248/30 -> 0.0.0.0, metric 2, tag 0
     172.25.150.252/30 -> 0.0.0.0, metric 2, tag 0
     192.168.50.0/27 -> 0.0.0.0, metric 3, tag 0
     192.168.50.32/27 -> 0.0.0.0, metric 3, tag 0
     192.168.50.64/26 -> 0.0.0.0, metric 2, tag 0
     192.168.50.128/26 -> 0.0.0.0, metric 1, tag 0
     192.168.50.192/27 -> 0.0.0.0, metric 3, tag 0
     192.168.50.224/27 -> 0.0.0.0, metric 3, tag 0
```

6.3.5 案例研究：认证

Cisco 实现 RIPv2 的消息认证包含了两种选择——简单的口令或 MD5 认证。另外，也包含了在一个“钥匙链”上定义多个钥匙或口令的选项。这样路由器就可以在不同的时候配置不同的钥匙。

设置 RIPv2 认证的步骤如下：

步骤 1：定义一个带名字的钥匙链。

步骤 2：定义在钥匙链上的钥匙。

步骤 3：在接口上启动认证并指定使用的钥匙链。

步骤 4：指定这个接口使用明文认证还是 MD5 认证。

步骤 5：可选地配置钥匙的管理。

在示例 6-9 中，路由器 Taos 上配置了一个名为“Tewa”的钥匙链，这个钥匙链上惟一的一个钥匙是“Key 1”，它含有一个口令“Kachina”。E0 接口将使用 MD5 认证的这个钥匙去验证来自路由器 Laguna 的更新消息。

示例 6-9 路由器 Taos 有关认证的配置

```
key chain Tewa
 key 1
 key-string Kachina
interface Ethernet 0
 ip rip authentication key-chain Tewa
 ip rip authentication mode md5
```

即使只有一个钥匙，也必须配置钥匙链。虽然进行带认证的更新消息交换的所有路由器必须拥有相同的口令，但是钥匙链的名字却只在本地路由器上有意义。例如，路由器 Laguna 可以有一个名为 Keres 的钥匙链，但是宣告给路由器 Taos 的钥匙的字符串必须是 Kachina。

如果没有添加命令 **ip rip authentication mode md5**，接口将使用缺省的明文认证。虽然在与一些 RIPv2 的设备通信时明文认证可能是必要的，但是只要有可能，几乎总是明智地使用安全性能好得多的 MD5 认证。

钥匙管理（key management）用来进行从一个认证钥匙到另一个认证钥匙的迁移工作的。在示例 6-10 中，路由器 Laguna 的配置是：在 2004 年 7 月 1 日下午 4:30 开始使用第一个钥匙，使用的时长是 12h（43200s）；第二个钥匙从 2004 年 7 月 2 日凌晨 4:00 开始生效，并一直使用到 2004 年 12 月 31 日下午 1:00；第三个钥匙从 2004 年 12 月 31 日下午 12:30 开始生效，并在这个时间以后永久有效。

示例 6-10 路由器 Laguna 有关认证的配置

```
key chain Keres
 key 1
  key-string Kachina
  accept-lifetime 16:30:00 Jul 1 2004 duration 43200
  send-lifetime 16:30:00 Jul 1 2004 duration 43200
 key 2
  key-string Kiva
  accept-lifetime 04:00:00 Jul 2 2004 13:00:00 Dec 31 2004
  send-lifetime 04:00:00 Jul 2 2004 13:00:00 Dec 31 2004
 key 3
  key-string Koshare
  accept-lifetime 12:30:00 Dec 31 2004 infinite
  send-lifetime 12:30:00 Dec 31 2004 infinite
!
interface Ethernet0
 ip address 198.168.50.130 255.255.255.192
 ip rip authentication key-chain Keres
 ip rip authentication mode md5
```

正如配置所显示的，从其他路由器接受的口令和发送消息所使用的口令在管理上是分离的。因此，使用命令 **accept-lifetime** 和 **send-lifetime** 都应该含有一个指定的开始时间和一个指定的持续时间或结束时间，或者指定关键字 ***infinite***。钥匙的号码按照从最低到最高的顺序检查，使用第一个有效的钥匙。

虽然这个配置可以使用 30min 的时间重叠来在不同的系统时钟之间进行校正，但是，这里强烈建议在对钥匙的管理时，使用像网络时钟协议（Network Time Protocol，NTP）这样的时钟同步协议（Time Synchronization Protocol）。[1]

6.4 RIPng 的配置

RIPng 的配置步骤和 RIPv1 与 RIPv2 是相同的：首先创建一个路由选择进程，接着在接口上启动该路由选择协议，然后执行任何所希望的用户配置。但是，RIPng 中所使用的命令却是非常不同。事实上，路由选择进程的创建和第一个接口上路由选择协议的启动都是使用一个接口命令一步完成的。RIPng 的案例研究与 RIPv1 和 RIPv2 中演示的那样，处理过程是相似的，但配置是不同的。

6.4.1 案例研究：RIPng 的基本配置

在路由器上启动 IPv6 的单播路由选择协议后，只需要一条命令就可以启动 IPv6 路由选择的 RIPng 协议。所需命令是一个接口模式下的命令：**ipv6 rip** *process-name* **enable**，它可以创建一个名字为 *process-name* 的 RIPng 进程，同时在接口上启动 RIPng 协议。该命令需要在每一个希望运行 RIPng 协议的接口上启动。

如图 6-15 所示，图中显示了 IPv6、RIPng 和在图 6-14 的网络基础上增加的一些新的以太网接口。

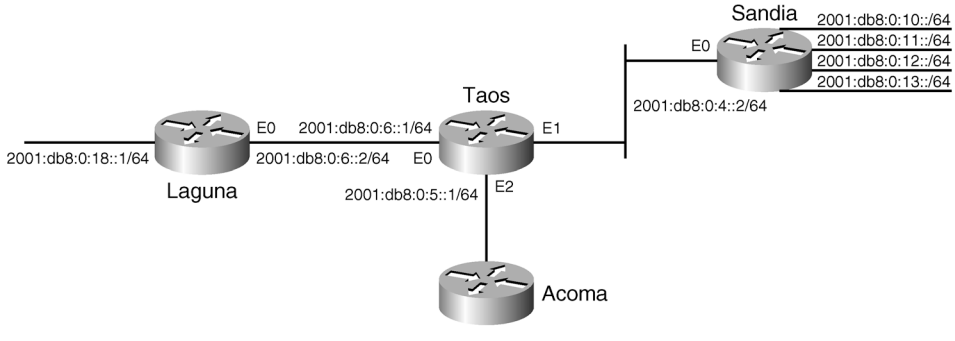


图 6-15 在路由器 Laguna、Taos 和 Sandia 的接口上配置的 IPv6 地址运行 RIPng 协议

示例 6-11 中显示了路由器 Taos 的配置。

示例 6-11 路由器 Taos 的 IPv6 和 RIPng 配置

```
ipv6 unicast-routing
interface Ethernet0
 ipv6 address 2001:db8:0:6::1/64
 ipv6 rip bigMountain enable
```

（待续）

[1] NTP 协议超出了本书的讲述范围，请参考 Cisco 配置手册以便获取更多的信息。

```
interface Ethernet1
 ipv6 address 2001:db8:0:4::1/64
 ipv6 rip bigMountain enable

interface Ethernet2
 ipv6 address 2001:db8:0:5::1/64
 ipv6 rip bigMountain enable
```

RIPng 的创建和在接口上激活 RIPng 都是通过一个接口配置命令实现的。这是 RIPng 的配置与 RIPv1 和 RIPv2 的配置之间的不同之处。回忆一下，在 RIPv1 和 RIPv2 的配置中需要一个全局命令（**router rip**）来创建进程，还需要 **network<***address***>**语句指定运行 RIP 进程的接口。在 RIPng 中，只需要在第一个运行 RIPng 的接口上配置接口命令 **ipv6 rip *bigMountain* enable**，一个命名为 bigMountain 的 RIPng 进程就自动地创建了。另外，IOS 可以自动地创建一个路由选择进程的配置条目（**ipv6 router rip bigMountain**）。

路由器 Taos 上 RIPng 进程的名字指定为 bigMountain。这里请注意，进程的名字是在每一个接口上指定的。这个名字只是和本地路由器相关联，用来区分可能运行在同一台路由器的多个 RIPng 进程。每一个不同接口都可以运行一个惟一的 RIPng 进程，而在这些接口之间不进行信息交换，或者在单个接口上运行多个进程。另一方面，在同一台路由器上只能运行单个 RIPv1 或 RIPv2 进程，一个接口要么属于这个进程，要么不运行 RIP 协议。[1]

示例 6-12 路由器 Taos 的 IPv6 路由表

```
Sandia#show ipv6 route
IPv6 Routing Table - 16 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
C   2001:DB8:0:4::/64 [0/0]
     via ::, Ethernet0
L   2001:DB8:0:4::2/128 [0/0]
     via ::, Ethernet0
R   2001:DB8:0:5::/64 [120/2]
     via FE80::205:5EFF:FE6B:50A1, Ethernet0
R   2001:DB8:0:6::/64 [120/2]
     via FE80::205:5EFF:FE6B:50A1, Ethernet0
C   2001:DB8:0:10::/64 [0/0]
     via ::, Ethernet1
L   2001:DB8:0:10:2B0:64FF:FE30:1DE0/128 [0/0]
     via ::, Ethernet1
C   2001:DB8:0:11::/64 [0/0]
     via ::, Ethernet2
L   2001:DB8:0:11:2B0:64FF:FE30:1DE0/128 [0/0]
     via ::, Ethernet2
C   2001:DB8:0:12::/64 [0/0]
     via ::, Ethernet3
L   2001:DB8:0:12:2B0:64FF:FE30:1DE0/128 [0/0]
     via ::, Ethernet3
C   2001:DB8:0:13::/64 [0/0]
     via ::, Ethernet4
L   2001:DB8:0:13:2B0:64FF:FE30:1DE0/128 [0/0]
     via ::, Ethernet4
```

（待续）

[1] MPLS VPN 允许多个 RIP 进程运行在单个提供商的边缘路由器（Provider Edge Router，PE）上。每一个给定的 VPN 配置一个单一的 RIP 进程。MPLS 和 VPN 的内容已经超出了本书的讲述范围。

```
R   2001:DB8:0:18::/64 [120/3]
     via FE80::205:5EFF:FE6B:50A1, Ethernet0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

如图 6-16 所示，在路由器 Acoma 上增加了两个新的以太网接口。新的网络策略是，在图 6-16 的网络上，IPv6 通信流量应该根据以下策略进行分段：与路由器 Sandia 相连的 LAN 上的主机之间可以互相访问，同时它们也可以访问路由器 Acoma 连接的 Ethernet1 上的设备。与路由器 Laguna 相连的主机可以访问路由器 Acoma 连接的 Ethernet2 上的设备。路由器 Sandia 与 Laguna 上的主机不需要互相访问。路由器 Acoma 连接的 Ethernet1 与 Ethernet2 之间不需要相互访问。

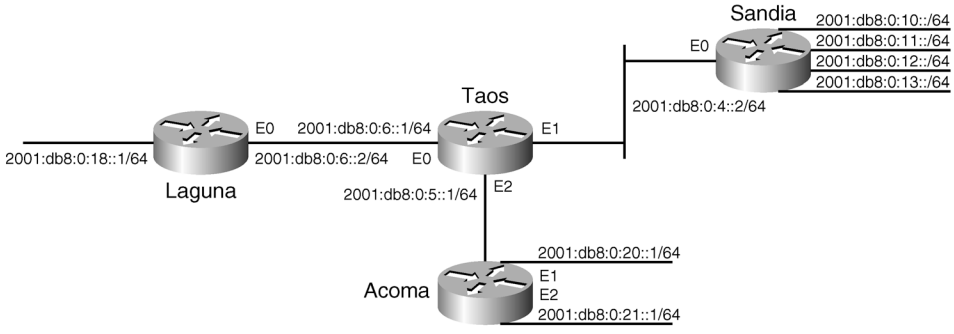


图 6-16 路由器 Acoma 增加到上述网络中，它带有两个运行 IPv6 的以太网客户 LAN 网段

示例 6-13 中的配置被加入到路由器 Taos 中。

示例 6-13 路由器 Taos 的配置通过创建多个 RIPng 路由选择进程来支持对 IPv6 网络流量进行分段的网络策略

```
Interface Ethernet0
 ipv6 rip bigMountain enable
Interface Ethernet1
 no ipv6 rip bigMountain enable
 ipv6 rip smallMountain enable
interface Ethernet2
 ipv6 rip bigMountain enable
 ipv6 rip smallMountain enable

ipv6 router rip smallMountain
 port 527 multicast ff02::9
```

连接到路由器 Laguna 的接口 Ethernet0 仍然属于名字为 bigMountain 的 RIPng 进程。在连接到路由器 Sandia 的接口 Ethernet1 上启动了一个新的名字为 smallMountain 的进程，并在这个接口上关闭了 bigMountain 进程。最后在连接到路由器 Acoma 的接口 Ethernet2 上同时启动了这两个进程。如果在同一个接口上同时运行两个进程，那么这两个 RIPng 路由选择进程不能使用相同的 UDP 端口号。通过改变 smallMountain 进程的端口号，路由器 Taos 发送 smallMountain 进程的更新到 UDP 端口 527 上，而 bigMountain 进程则仍然使用缺省的 RIPng 端口号（即 UDP 端口 521）发送更新消息。接收路由器可以使用端口号来区分所收到的更新消息属于哪一个路由选择进程。一台运行单个路由选择进程的 RIPng 路由器为了处理每一个更新消息，它将使用相同的 UDP 端口号接收来自一台路由器上运行的多个 RIPng 进程发送的更新消息。如果这两个更新消息有关同一个地址的信息存在冲突，例如度量值不同，那么

接收路由表可能会根据每一个更新消息改变。如果一台接收路由器上正在运行多个 RIPng 进程，并且使用相同的 UDP 端口号，那么这台接收路由器将不能区分哪一个 RIPng 进程应该接收这个更新消息。改变第二个和随后的 RIPng 进程的 UDP 端口号可以排除这些故障。所有运行更改 UDP 端口号的 RIPng 进程的路由器和主机都必须使用相同的 UDP 端口号。由于路由器和其他 TCP/IP 设备使用端口号识别数据包需要哪一个 UDP 进程进行转发，而 RIPng 更新消息以多播方式向所有的 RIPng 路由器发送，因此新的 UDP 端口号一定不能与任何运行 RIPng 的路由器上任何其他进程的端口号相同。在这个例子中为 smallMountain 进程配置的端口号是 527。和 smallMountain 命名的 RIPng 进程交换 RIPng 更新消息的路由器必须更改它的端口号。路由器 Sandia 和 Acoma 的配置更改分别参见下面的示例 6-14 与示例 6-15。

示例 6-14 在路由器 Sandia 上更改 UDP 端口号的 RIPng 配置

```
ipv6 router rip smallMountain
  port 527 multicast-group FF02::9
```

示例 6-15 在路由器 Acoma 上更改 UDP 端口号的 RIPng 配置

```
interface Ethernet0
 ipv6 address 2001:DB8:0:5::2/64
 ipv6 rip hill enable
 ipv6 rip summit enable
interface Ethernet1
 ipv6 address 2001:DB8:0:20::/64
 ipv6 rip hill enable
interface Ethernet2
 ipv6 address 2001:DB8:0:21::/64
 ipv6 rip summit enable
ipv6 router rip hill
  port 527 multicast-group FF02::9
```

读者在这里可以注意到，路由器 Acoma 上的进程名字和路由器 Taos 上的进程名字是不同的。RIPng 进程的名字只对本台路由器有意义，它们并不在路由更新之间进行交换。路由器 Acoma 的以太接口 Ethernet0 连接到路由器 Taos 的以太接口 Ethernet2 上。它运行了两个路由选择进程：hill 和 summit。进程 hill 也运行在 Ethernet1 的接口上。Ethernet1 接口的 IPv6 地址使用更改后的 UDP 端口号 527 通告给路由器 Taos。这是与路由器 Taos 上的 smallMountain 进程相关联的。因此，IPv6 的地址将通告给路由器 Sandia，而不是 Laguna。

示例 6-16 中显示了运行在路由器 Taos 上的 RIPng 路由选择进程的相关信息。使用命令 **show ipv6 rip** 可以显示出每一个进程的信息。注意并行路径缺省的最大数目是 16，而计时器的缺省值还没有更改。

示例 6-16 使用命令 show ipv6 rip 可以显示出运行在某台路由器上的每一个 RIPng 进程的相关信息

```
Taos#show ipv6 rip
RIP process "bigMountain", port 521, multicast-group FF02::9, pid 104
     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 1078, trigger updates 5
  Interfaces:
```

（待续）

```
    Ethernet2
    Ethernet0
  Redistribution:
    None
RIP process "smallMountain", port 527, multicast-group FF02::9, pid 117
     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 1080, trigger updates 5
  Interfaces:
    Ethernet1
    Ethernet2
  Redistribution:
    None
```

进程 bigMountain 运行在接口 Ethernet0 与 Ethernet2 上。接口 Ethernet1 和 Ethernet2 同属于进程 smallMountain，并且共同使用 UDP 端口号 527。

IPv6 路由表显示出路由器 Taos 已经把这两个进程的路由加载到它的路由表中了，参见示例 6-17。

示例 6-17 IPv6 路由表显示了来自每一个运行的 RIPng 进程的所有学习到的路由

```
Taos#show ipv6 route
IPv6 Routing Table - 15 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
C   2001:DB8:0:4::/64 [0/0]
     via ::, Ethernet1
L   2001:DB8:0:4::1/128 [0/0]
     via ::, Ethernet1
C   2001:DB8:0:5::/64 [0/0]
     via ::, Ethernet2
L   2001:DB8:0:5::1/128 [0/0]
     via ::, Ethernet2
C   2001:DB8:0:6::/64 [0/0]
     via ::, Ethernet0
L   2001:DB8:0:6::1/128 [0/0]
     via ::, Ethernet0
R   2001:DB8:0:10::/64 [120/2]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:11::/64 [120/2]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:12::/64 [120/2]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:13::/64 [120/2]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:18::/64 [120/2]
     via FE80::204:C1FF:FE50:F1C0, Ethernet0
R   2001:DB8:0:20::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Ethernet2
R   2001:DB8:0:21::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Ethernet2
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

注意示例 6-18 和示例 6-19 中显示的路由器 Laguna 和 Sandia 的路由表，读者可以看出每一台路由器都学到了属于对应的 RIPng 进程的路由。路由器 Taos 不会把 smallMountain 进程的地址转发到 bigMountain 进程中，反之亦然。

示例 6-18 IPv6 路由表显示了路由器 Laguna 从路由器 Taos 的 RIPng 进程 bigMountain 学习到的路由

```
Laguna#show ipv6 route rip
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
R   2001:DB8:0:5::/64 [120/2]
     via FE80::205:5EFF:FE6B:50A0, Ethernet0
R   2001:DB8:0:21::/64 [120/3]
     via FE80::205:5EFF:FE6B:50A0, Ethernet0
```

示例 6-19 IPv6 路由表显示了路由器 Sandia 从路由器 Taos 的 RIPng 进程 smallMountain 学习到的路由

```
Sandia#show ipv6 route rip
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
R   2001:DB8:0:5::/64 [120/2]
     via FE80::205:5EFF:FE6B:50A1, Ethernet0
R   2001:DB8:0:20::/64 [120/3]
     via FE80::205:5EFF:FE6B:50A1, Ethernet0
```

示例 6-17 中显示了路由器 Taos 学到了来自路由器 Sandia 的以太接口 Ethernet1 上的 4 个地址，这些地址属于 RIPng 进程 smallMountain：2001:db8:0:10::/64、2001:db8:0:11::/64、2001:db8:0:12::/64 和 2001:db8:0:13::/64。连接在接口 Ethernet2 上的地址 2001:db8:0:5::/64 将同时被通告到 smallMountain 和 bigMountain 进程中。在示例 6-18 中，路由器 Laguna 的路由表显示出没有加载任何 smallMountain 进程的路由。

参见示例 6-20，在路由器 Taos 上打开 RIPng 的调试信息可以看到每一个接口上发送和接收的每一个 RIPng 进程的更新消息

示例 6-20 命令 debug ipv6 rip 显示了每一个接口上发送和接收的每一个 RIPng 进程的更新消息

```
Taos#debug ipv6 rip
RIP Routing Protocol debugging is on
Taos#
RIPng: Sending multicast update on Ethernet2 for bigMountain
        src=FE80::205:5EFF:FE6B:50A0
        dst=FF02::9 (Ethernet2)
        sport=521, dport=521, length=72
        command=2, version=1, mbz=0, #rte=3
        tag=0, metric=1, prefix=2001:DB8:0:5::/64
        tag=0, metric=1, prefix=2001:DB8:0:6::/64
```

（待续）

```
        tag=0, metric=2, prefix=2001:DB8:0:18::/64
RIPng: Sending multicast update on Ethernet0 for bigMountain
        src=FE80::205:5EFF:FE6B:50A0
        dst=FF02::9 (Ethernet0)
        sport=521, dport=521, length=72
        command=2, version=1, mbz=0, #rte=3
        tag=0, metric=1, prefix=2001:DB8:0:5::/64
        tag=0, metric=1, prefix=2001:DB8:0:6::/64
        tag=0, metric=2, prefix=2001:DB8:0:21::/64
RIPng: response received from FE80::2B0:64FF:FE30:1DE0 on Ethernet1 for
  smallMountain
        src=FE80::2B0:64FF:FE30:1DE0 (Ethernet1)
        dst=FF02::9
        sport=527, dport=527, length=112
        command=2, version=1, mbz=0, #rte=5
        tag=0, metric=1, prefix=2001:DB8:0:4::/64
        tag=0, metric=1, prefix=2001:DB8:0:10::/64
        tag=0, metric=1, prefix=2001:DB8:0:11::/64
        tag=0, metric=1, prefix=2001:DB8:0:12::/64
        tag=0, metric=1, prefix=2001:DB8:0:13::/64
RIPng: response received from FE80::204:C1FF:FE50:E700 on Ethernet2 for
  smallMountain
        src=FE80::204:C1FF:FE50:E700 (Ethernet2)
        dst=FF02::9
        sport=527, dport=527, length=52
        command=2, version=1, mbz=0, #rte=2
        tag=0, metric=1, prefix=2001:DB8:0:5::/64
        tag=0, metric=1, prefix=2001:DB8:0:20::/64
RIPng: response received from FE80::204:C1FF:FE50:F1C0 on Ethernet0 for bigMountain
        src=FE80::204:C1FF:FE50:F1C0 (Ethernet0)
        dst=FF02::9
        sport=521, dport=521, length=52
        command=2, version=1, mbz=0, #rte=2
        tag=0, metric=1, prefix=2001:DB8:0:6::/64
        tag=0, metric=1, prefix=2001:DB8:0:18::/64
RIPng: response received from FE80::204:C1FF:FE50:E700 on Ethernet2 for bigMountain
        src=FE80::204:C1FF:FE50:E700 (Ethernet2)
        dst=FF02::9
        sport=521, dport=521, length=52
        command=2, version=1, mbz=0, #rte=2
        tag=0, metric=1, prefix=2001:DB8:0:5::/64
        tag=0, metric=1, prefix=2001:DB8:0:21::/64
RIPng: Sending multicast update on Ethernet1 for smallMountain
        src=FE80::205:5EFF:FE6B:50A1
        dst=FF02::9 (Ethernet1)
        sport=527, dport=527, length=72
        command=2, version=1, mbz=0, #rte=3
        tag=0, metric=1, prefix=2001:DB8:0:4::/64
        tag=0, metric=1, prefix=2001:DB8:0:5::/64
        tag=0, metric=2, prefix=2001:DB8:0:20::/64
RIPng: Sending multicast update on Ethernet2 for smallMountain
        src=FE80::205:5EFF:FE6B:50A0
        dst=FF02::9 (Ethernet2)
        sport=527, dport=527, length=132
        command=2, version=1, mbz=0, #rte=6
        tag=0, metric=1, prefix=2001:DB8:0:4::/64
        tag=0, metric=1, prefix=2001:DB8:0:5::/64
        tag=0, metric=2, prefix=2001:DB8:0:10::/64
        tag=0, metric=2, prefix=2001:DB8:0:11::/64
        tag=0, metric=2, prefix=2001:DB8:0:12::/64
        tag=0, metric=2, prefix=2001:DB8:0:13::/64
```

smallMountain 进程的更新消息通过 Ethernet1 接口发送到路由器 Sandia，通过 Ethernet2 接口发送到路由器 Acoma。bigMountain 进程的更新消息通过 Ethernet0 接口发送到路由器

Laguna，通过 Ethernet2 接口发送到路由器 Acoma。在 Ethernet2 接口上可以同时收到来自路由器 Acoma 的两个进程的更新消息。一个使用了缺省的 UDP 端口 521，另一个使用了 UDP 端口 527，后者是与 smallMountain 进程相关联的。使用 527 端口的更新包含了前缀 2001:DB8:0:20::/64。这个前缀是分配给路由器 Acoma 的 Ethernet1 接口的，它是可以被路由器 Sandia 访问的。在 Ethernet2 接口上使用 UDP 端口 521 收到的更新消息包含了前缀 2001:DB8:0:21::/64。这个前缀是分配给路由器 Acoma 的 Ethernet2 接口的，它可以分发到路由器 Taos 的 bigMountain 进程中，从而发送到路由器 Laguna。

6.4.2 案例研究：RIPng 进程的定制

路由选择进程的一些定制和 RIPv1 与 RIPv2 中的定制类似。读者可以使用全局配置命令 **ipv6 router rip** *process_name* 配置 RIPng 进程的全局参数。

管理距离、路由选择进程用于负载均分的最大路径数，以及 RIP 计时器都已经在第 5 章中讲述了。这些参数在 RIPng 中的使用方法和在 RIPv1 与 RIPv2 中的用法是相同的。RIPng 的缺省管理距离是 120，这和 RIPv1 与 RIPv2 是一样的。RIPng 用于负载均分的最大路径数的缺省值是 16。RIPng 能够负载均分的路径数最大为 64。虽然并不是所有的计时器参数的值和 RIPv1 与 RIPv2 相同，但计时器参数是相同的：每 30s 更新一次，180s 超时，保持计时器为 0，垃圾收集计时器为 120s。

参见示例 6-21，路由器 Taos 的配置更改了管理距离、最大路径数和计时器。

示例 6-21 路由器 Taos 的配置更改了管理距离、用于负载均分的最大路径数和协议计时器

```
ipv6 router rip bigMountain
  timers 10 30 30 60
  maximum-paths 8
  distance 200
```

示例 6-22 显示了命令 **show ipv6 rip** 的输出，显示了新的参数值。

示例 6-22 在路由器 Taos 上，命令 show ipv6 rip 显示了更改后的 RIPng 参数值

```
Taos#show ipv6 rip
RIP process "bigMountain", port 521, multicast-group FF02::9, pid 104
     Administrative distance is 200. Maximum paths is 8
     Updates every 10 seconds, expire after 30
     Holddown lasts 30 seconds, garbage collect after 60
     Split horizon is on; poison reverse is off
     Default routes are not generated
     Periodic updates 2513, trigger updates 7
  Interfaces:
    Ethernet2
    Ethernet0
  Redistribution:
    None
RIP process "smallMountain", port 527, multicast-group FF02::9, pid 122
     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
```

（待续）

```
     Default routes are not generated
     Periodic updates 2511, trigger updates 0
  Interfaces:
    Ethernet2
    Ethernet1
  Redistribution:
    None
Taos#
```

比较 bigMountain 进程和 smallMountain 进程中的参数值，看看哪些仍然使用缺省值。在更改计时器和管理距离时应该特别小心。所有运行同一个 RIPng 进程的路由器都必须使用同样的计时器值。管理距离虽然只是在本地路由器上具有意义，但是如果路由器上正在运行多个路由选择协议或路由选择进程，并且地址是从多个路由选择进程学习到的，那么改变管理距离需要慎重考虑。更改路由选择进程的管理距离会改变它们的优先权。通过具有较低的管理距离的路由选择进程学习到的地址才会加入到路由表中。如果从具有较高的管理距离的进程学习到同样的地址，那么只有在第一个路由条目失效时才会将这个地址加入到路由表中。示例 6-23 中的配置将这些参数值改回了原来的缺省值。

示例 6-23 在下面的配置中，路由器 Taos 的 RIPng 参数被恢复为原来的缺省值

```
ipv6 router rip bigMountain
 no timers 10 30 30 60
 no maximum-paths 8
 no distance 200
```

6.4.3 案例研究：RIPng 的度量控制

RIPng 与 RIPv1 和 RIPv2 一样，通过增加一个偏移量来调整它的度量值。但是，RIPng 不是为前缀列表中的每一项路由条目增加度量，而是通过更改相关联的接口的跳数来实现的。也就是说，RIPng 是通过在路由器相对应的接口上配置希望增加的跳数值来增加通过这个接口所通告的每一个前缀的度量值的。在缺省的情况下，RIPng 为邻居路由器通告过来的前缀度量值增加一跳。在示例 6-24 中，显示了路由器 Taos 上的 RIPng 路由表和从路由器 Sandia 收到的一个 RIPng 更新消息。这里请注意，路由表中的每个条目的度量值都是 2。路由表中的每个 RIPng 路由都是直接连接到距离只有一跳的路由器的，不论是连到路由器 Sandia 的还是 Laguna 的。路由器 Sandia 和 Laguna 是使用度量值 1 来通告它们的前缀的。路由器 Taos 为每一个接收到的路由度量值增加一跳。为了在路由器 Taos 上修改要添加到所收到的度量值，可以使用命令 **metric-offset** 来实现，参加示例 6-25。

示例 6-24 RIPng 的路由表，使用调试命令 debug ipv6 rip 显示所收到的更新消息演示了接收到的度量上增加的度量偏移量

```
Taos#show ipv6 route rip
IPv6 Routing Table - 15 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
R   2001:db8:0:18::/64[120/2]
     via FE80::204:C1FF:FE50:F1C0, Ethernet0
```

（待续）

```
R   2001:db8:0:10::/64 [120/2]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:db8:0:11::/64 [120/2]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:db8:0:12::/64 [120/2]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:db8:0:13::/64 [120/2]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:20::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Ethernet2
R   2001:DB8:0:21::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Ethernet2
Taos#debug ipv6 rip
RIP Routing Protocol debugging is on
   RIPng: response received from FE80::2B0:64FF:FE30:1DE0 on Ethernet1 for
    smallMountain
        src=FE80::2B0:64FF:FE30:1DE0 (Ethernet1)
        dst=FF02::9
        sport=521, dport=521, length=112
        command=2, version=1, mbz=0, #rte=4
        tag=0, metric=1, prefix=2001:db8:0:10::/64
        tag=0, metric=1, prefix=2001:db8:0:11::/64
        tag=0, metric=1, prefix=2001:db8:0:12::/64
        tag=0, metric=1, prefix=2001:db8:0:13::/64
```

示例 6-25 在路由器 Taos 的 Ethernet1 接口上将 RIPng 的度量偏移量增加为 3

```
interface Ethernet1
 ipv6 rip smallMountain metric-offset 3
```

在示例 6-26 中显示了路由器 Taos 的新路由表。路由器 Taos 在 Ethernet1 接口上为它所收到的每一个前缀都增加了 3 跳。

示例 6-26 在接收接口上更改度量偏移量后，RIPng 路由表显示通过这个修改度量值的接口学到的每一个前缀都具有较高的度量值

```
Taos#show ipv6 route rip
IPv6 Routing Table - 15 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
R   2001:DB8:0:10::/64 [120/4]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:11::/64 [120/4]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:12::/64 [120/4]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:13::/64 [120/4]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:18::/64 [120/2]
     via FE80::204:C1FF:FE50:F1C0, Ethernet0
R   2001:DB8:0:20::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Ethernet2
R   2001:DB8:0:21::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Ethernet2
```

这条命令允许我们调整与给定链路相关联的跳数值。通过调整后的接口所接收的所有地址将具有调整后的度量值。与之相比，RIPv1 和 RIPv2 允许我们将在接口收到的一个子网（或所有）地址加入到路由表之前增加偏移量，或者这些地址被从该接口通告出去之前可以增加

偏移量。

6.4.4 案例研究：路由汇总

与 RIPv2 一样，RIPng 的路由可以进行路由汇总后再通告给邻居。路由器 Sandia 需要通告地址 2001:DB8:0:10::/64、2001:DB8:0:11::/64、2001:DB8:0:12::/64，以及 2001:DB8:0:13::/64 的路由。这些路由可以被汇总为一条路由：2001:DB8:0:10::/62。路由器 Sandia 的路由配置参见示例 6-27。

示例 6-27 路由器 Sandia 的配置对所通告的 RIPng 地址进行了汇总

```
interface Ethernet0
 ipv6 address 2001:DB8:0:4::2/64
 ipv6 rip bigMountain enable
 ipv6 rip bigMountain summary-address 2001:DB8:0:10::/62
```

在配置了覆盖这些具体路由前缀的汇总路由地址后，这些更具体的路由前缀将被自动地抑制。在示例 6-28 中显示了路由器 Taos 的新路由表。

示例 6-28 64 位长的 4 个连续的前缀已经汇总为一条 62 位长的路由条目

```
Taos#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
R   2001:DB8:0:10::/62 [120/4]
     via FE80::2B0:64FF:FE30:1DE0, Ethernet1
R   2001:DB8:0:18::/64 [120/2]
     via FE80::204:C1FF:FE50:F1C0, Ethernet0
R   2001:DB8:0:20::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Ethernet2
R   2001:DB8:0:21::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Ethernet2
```

6.5 RIPv2 与 RIPng 的故障诊断

对于 RIPv2 协议，通常会碰到两个配置问题，即版本不匹配和认证配置错误。这两个问题都可以比较容易地从调试信息中发现，参见示例 6-29 所示。

示例 6-29 通过调试信息来发现不匹配的版本和配置错误的认证问题

```
Jemez#debug ip rip events
RIP event debugging is on
Jemez#
RIP: ignored v1 packet from 172.25.150.249 (illegal version)
RIP: ignored v2 packet from 172.25.150.249 (invalid authentication)
Jemez#
```

对于 RIPv2 或 RIPng 协议，或者任何无类别路由选择协议来说，更有可能出问题的原因是，配置了一个错误的可变长子网掩码。VLSM 并不难，但是如果 VLSM 的规划没有小心地

设计和管理，它就会带来一些非同寻常的路由选择困难。

案例研究：配置错误的 VLSM

图 6-17 中的主机 C 不能通过网络进行通信，甚至无法在本地数据链路上 ping 通其他的主机或路由器。而主机 A 和主机 B 的相互通信没有问题，能够与网络上的其他主机正常通信，但是它们都不能和主机 C 通信。这里，所有的主机都把 172.19.35.1 配置成自己的缺省网关地址。

如图 6-18 所示，当从主机 A 或主机 B 上试图去 ping 主机 C 时，发现第一个 ping 是成功的，但是后续的 ping 是失败的。显然，至少有一个 ICMP 的 Echo 请求包能够到达主机 C，并且至少有一个 Echo 响应包可以返回给 ping 的源主机，这一事实说明，网络的故障与硬件或数据链路没有太大关系。

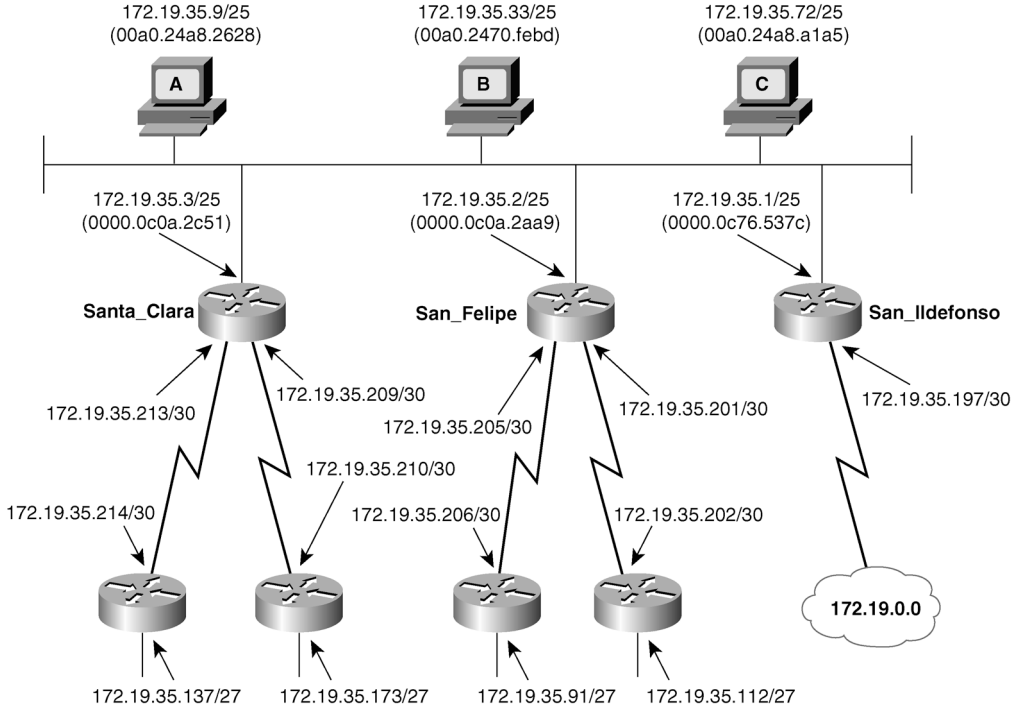


图 6-17 主机 A 和主机 B 能够通过网络进行通信，但是都不能和主机 C 通信

这个 ping 的奇怪行为可以让我们作出这样一个假设：在第一个 ping 包成功后，后续的 ping 包——不论是主机 B 发出的 Echo 请求包，还是主机 C 返回的 Echo 响应包——不知由于什么原因被误导了。因为这种情况发生在本地的数据链路上，因此应该检查一下 ARP（Address Resolution Protocol，地址解析协议）的缓冲区（cache）。

示例 6-30 和图 6-19 分别显示了主机 C 和主机 B 的 ARP 缓冲区。关于 ARP 的猜疑在这里得到证实，主机 C 的 ARP 缓冲区包含了主机 B 的正确 MAC 地址（00a0.2470.febd），但是主机 B 的缓冲区包含的与主机 C 的 IP 地址相关联的 MAC 地址是 0000.0c0a.2aa9。进一步地

观察这两个缓冲区，显示出 MAC 地址 0000.0c0a.2aa9 是路由器 San_Felipe 的本地接口的 MAC 地址，从这个信息得知：通过路由器 San_Felipe 可以到达的目的 IP 地址和 IP 地址 172.19.35.2 映射到相同的 MAC 地址上了。

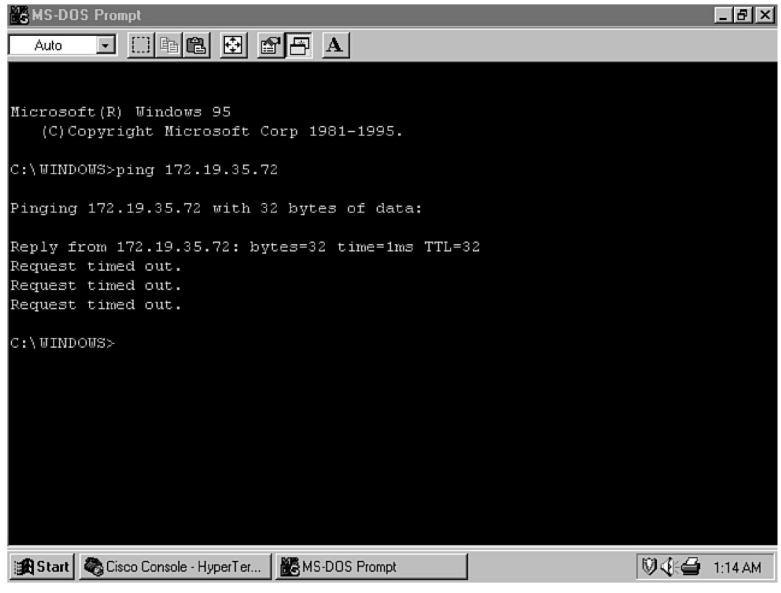


图 6-18 当主机 B 试图 ping 主机 C 时，第一个 ping 包是成功的，而后续的 ping 包是失败的

示例 6-30 主机 C 的 ARP 缓冲区正确地显示了所有地址的 MAC 地址

```
Linux 1.2.13 (Zuni.pueblo.com) (ttyp0)
Zuni login: root
Password:
Last login: Sat Nov 29 11:21:57 on tty1
Linux 1.2.13.
Zuni:~# arp -a
Address          HW type            HW address            Flags     Mask
172.19.35.112    10Mbps Ethernet    00:00:0C:0A:2A:A9     C         *
172.19.35.1      10Mbps Ethernet    00:00:0C:76:5B:7C     C         *
172.19.35.33     10Mbps Ethernet    00:A0:24:70:FE:BD     C         *
172.19.35.2      10Mbps Ethernet    00:00:0C:0A:2A:A9     C         *
172.19.35.3      10Mbps Ethernet    00:00:0C:0A:2C:51     C         *
172.19.35.9      10Mbps Ethernet    00:A0:24:A8:26:28     C         *
172.19.35.91     10Mbps Ethernet    00:00:0C:0A:2A:A9     C         *
Zuni:~#
```

现在 ping 的结果就比较清楚了。首先，主机 B 广播了一个 IP 地址为 172.19.35.72 的 ARP 请求，然后主机 C 发送一个 ARP 响应包，因而主机 B 发送的第一个 ping 包是正确的。在这期间，路由器 San_Felipe 也收到了那个 ARP 的请求包，很显然它认为自己有一条到达地址 172.19.35.72 的路由，于是路由器 San_Felipe 就用 ARP 代理（Proxy ARP）作出响应（滞后于主机 C 是因为路由器最初不得不执行一次路由的查找），这就导致主机 B 在自己的 MAC 缓存中覆盖了主机 C 的 MAC 地址。后来的 Echo 请求包被发送给路由器 San_Felipe，而路由器 San_Felipe 将把这个请求包从本地链路路由出去，最终被丢弃；连接在这个以太网链路上的协议分析仪证实了这一点（如图 6-20 所示）。

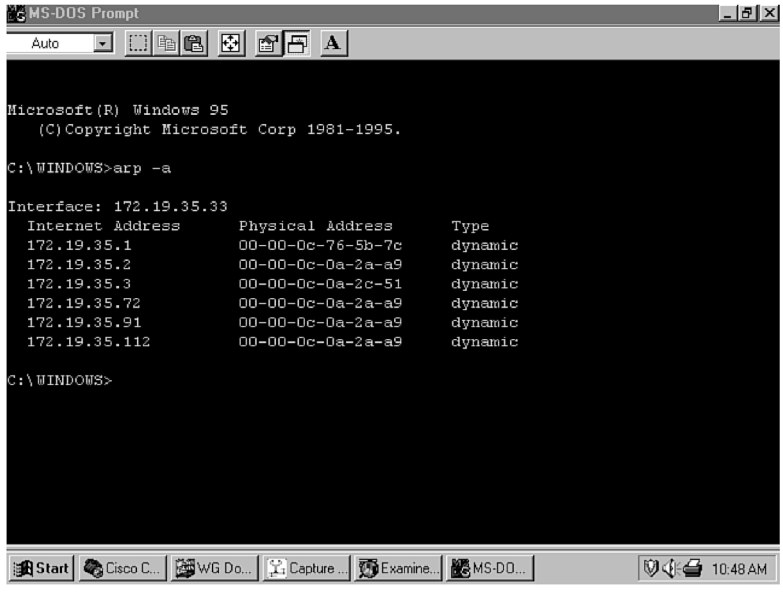


图 6-19 主机 B 的 ARP 缓冲区显示，主机 C 的 IPv4 地址被映射到了路由器 San_Felipe 的本地接口 172.19.35.2 的 MAC 地址上

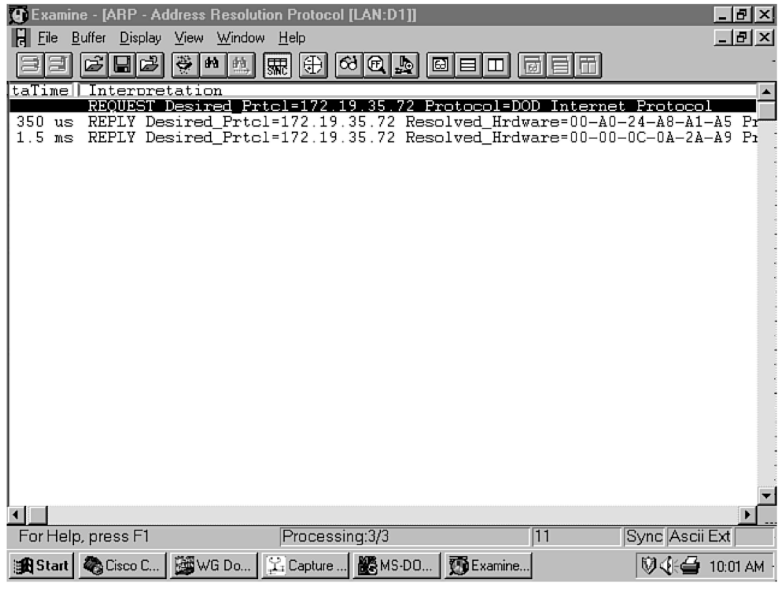


图 6-20 协议分析仪过滤出的 ARP 数据包显示，主机 B 到主机 C 的 ARP 请求和从主机 C（00a0.24a8.a1a5）与路由器 San_Felipe（0000.0c0a.2aa9）返回的响应

如果了解了故障是由路由选择问题引起的，那么余下的工作就是要找出引起路由选择问题的原因了。首先，应该确定一下每一条数据链路的子网地址，如图 6-21 所示。接着，基于二进制的表示，应该把主机 C 的 IP 地址和从路由器 San_Felipe 可达的所有子网相对照，来找出所有的地址冲突。在表 6-5 中，用粗体字显示了子网地址的最后一个八位组的子网位。

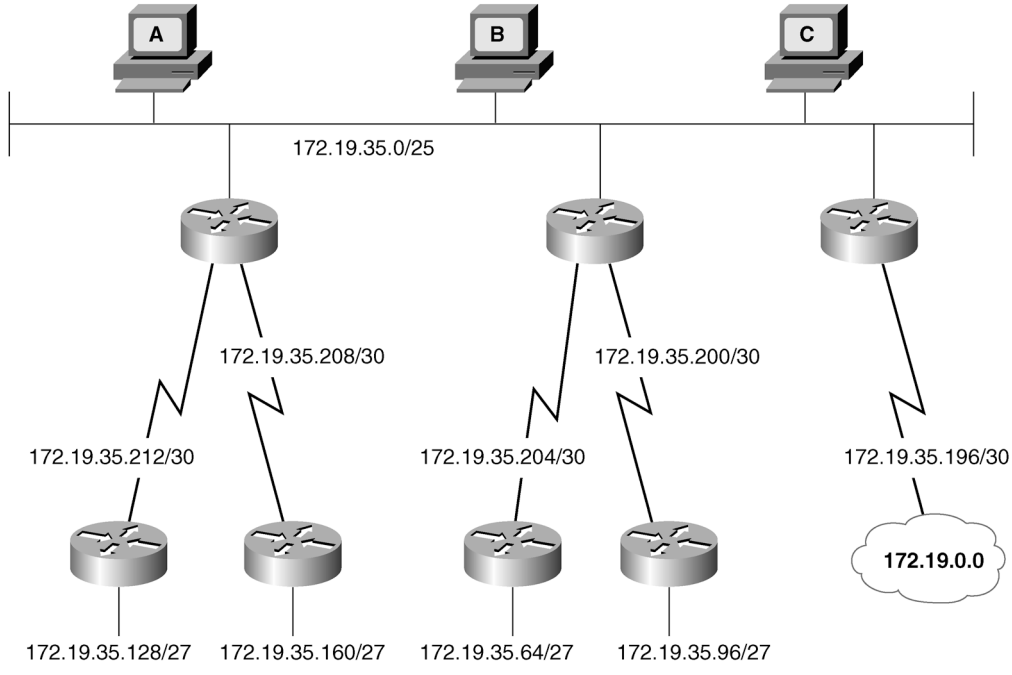


图 6-21 当分析任何地址编址的规划时，特别是 VLSM 的设计，应该先确定每一条数据链路的子网地址，这样才能发现地址冲突和重叠的问题

表 6-5 路由器 C 的 IPv4 地址，高亮度处显示了最后一个八位组的子网位

二进制表示	点分十进制表示
10101100000100110010001101001000	172.19.35.72/25
10101100000100110010001100000000	172.19.35.0/25
10101100000100110010001101000000	172.19.35.64/27
10101100000100110010001101100000	172.19.35.96/27
10101100000100110010001111001000	172.19.35.200/30
10101100000100110010001111001100	172.19.35.204/30

经过比较后，显示出子网 172.19.35.72/25 的前 3 位和 172.19.35.64/27 的前 3 位是匹配的。路由器 San_Felipe 的路由表同时拥有 172.19.35.0/25 和 172.19.35.64/27 的路由（参见示例 6-31）。当路由器收到一个要到达主机 C 的数据包时，它可以和子网 172.19.35.0/25 匹配 1 个比特位，但却可以和子网 172.19.35.64/27 匹配 3 个比特位。结果是，路由器将选择更具体的子网路由并把数据包从本地数据链路上路由出去，最后丢弃该数据包。

示例 6-31 路由器 San_Felipe 同时拥有 172.19.35.0/25 和 172.19.35.64/27 的路由，第二个路由比第一个路由能够更好地匹配主机 C 的地址

```
San_Felipe#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default,
       U - per-user static route
Gateway of last resort is 172.19.35.1 to network 0.0.0.0
```

（待续）

```
     172.19.0.0/16 is variably subnetted, 10 subnets, 3 masks
R       172.19.35.128/27 [120/1] via 172.19.35.3, 00:00:07, Ethernet0
R       172.19.35.160/27 [120/1] via 172.19.35.3, 00:00:08, Ethernet0
R       172.19.35.212/30 [120/1] via 172.19.35.3, 00:00:08, Ethernet0
R       172.19.35.208/30 [120/1] via 172.19.35.3, 00:00:08, Ethernet0
C       172.19.35.204/30 is directly connected, Serial0
C       172.19.35.200/30 is directly connected, Serial1
R       172.19.35.196/30 [120/1] via 172.19.35.1, 00:00:17, Ethernet0
C       172.19.35.0/25 is directly connected, Ethernet0
R       172.19.35.64/27 [120/1] via 172.19.35.206, 00:00:11, Serial0
R       172.19.35.96/27 [120/1] via 172.19.35.202, 00:00:23, Serial1
R* 0.0.0.0/0 [120/1] via 172.19.35.1, 00:00:18, Ethernet0
San_Felipe#
```

该故障的解决办法是，要么更改主机 C 的地址，要么更改子网 172.19.35.64。这一措施理论上听起来比较容易，但在实际的网络中，它会带来一些困难的决策，因为它涉及到这个案例研究中的网络上的其他客户。

表 6-6 中显示了网络 172.19.35.0 基于 27 位掩码的所有子网。这样做的目的是为了把最初的 4 个连续的子网捆绑成一个 25 位掩码的单个子网，以便容纳“骨干”以太网段上的 85 台主机。这种做法是可以的，因为地址编组所使用的所有子网的首个子网位都为 0，因而没有其他的地址能够引起冲突；其次，子网 172.19.35.192/27 使用 30 位的掩码来划分更小的子网，给串行链路使用，这又一次证明该做法是有效的。子网 172.19.35.128/27 和 172.19.35.160 已经被使用了，当选择子网 172.19.35.64/27 和子网 172.19.35.96/27 给两个“远程”网段使用时，错误就产生了，因为这两个子网已经被宣告了。

这个困难的决策来自于，究竟是放弃骨干以太网段的地址空间进行重新编址，还是放弃那两个远程子网各自的地址空间进行重新编址？我们选择后者，利用一个 28 位的掩码将 172.19.35.224/27 划分成两个子网，以供两个远程子网使用。

表 6-6 一个 27 位的掩码应用于子网 172.19.35.0

二进制表示	点分十进制表示
11111111111111111111111111100000	255.255.255.224
10101100000100110010001100000000	172.19.35.0/27
10101100000100110010001100100000	172.19.35.32/27
10101100000100110010001101000000	172.19.35.64/27
10101100000100110010001101100000	172.19.35.96/27
10101100000100110010001110000000	172.19.35.128/27
10101100000100110010001110100000	172.19.35.160/27
10101100000100110010001111000000	172.19.35.192/27
10101100000100110010001111100000	172.19.35.224/27

6.6 展　　望

虽然 RIPv2 协议比 RIPv1 协议有了很大的改进，但它依旧受到最大 15 跳的跳数限制，因此只能适用于小型网络。第 7 章、第 8 章和第 10 章将阐述在大型网络中使用的 3 种路由选择协议，并再结合像 VLSM 这样的设计策略，就可以成为控制大型网络的强有力的

工具。

6.7 总结表：第 6 章命令总结

命令	描述
accept-lifetime *start-time*{**infinite**|*end-time*| **duration** *seconds*}	设置一个时间段，用来指定钥匙链上的认证钥匙可被接受的有效时间
auto-summary	在网络边界打开或关闭自动路由汇总功能
debug ip rip [**events**]	启动 RIP 处理消息的调试功能
debug ipv6 rip	启动 RIPng 处理消息的调试功能
ip classless	使无类路由特性有效，即路由器能够找到最佳的匹配路由去转发数据包到目的地址，而不考虑到达目的地址的地址类别
ip rip authentication key-chain *name-of-chain*	使接口上的 RIPv2 认证有效，并指定一个所用的钥匙链的名字
ip rip authentication mode{text|md5}	指定在一个接口上使用的是明文还是 MD5 认证
ip rip receive version [1] [2]	指定一个接口可以接收的 RIP 的版本
ip rip send version [1] [2]	指定一个接口可以发送的 RIP 的版本
ip split-horizon	在接口上打开或关闭水平分隔特性
ip subnet-zero	允许接口的地址和路由选择更新使用全 0 子网
ipv6 rip *process-name* **enable**	在使用该命令的接口上运行 IPv6 RIPng 进程，并给它指定一个名字
ipv6 rip *process-name* **metric-offset** *value*	更改与入站接口相关联的度量值
ipv6 rip *process-name* **summary-address** *prefix*	把前缀长度较长的前缀汇总成较短的前缀
ipv6 router rip *process-name*	在路由器上启用 IPv6 RIPng 进程，并进入可以更改全局 RIPng 参数的配置模式
port *port-num* **multicast** *multicast-value*	更改一个 RIPng 进程的 UDP 端口号和（或）多播地址
key *number*	指定在钥匙链上的一个钥匙
key chain *name-of-chain*	指定一组钥匙
key-string *text*	指定钥匙使用的认证字符串或口令
network *network-number*	指定覆盖一个和多个运行 IGRP、EIGRP 或 RIP 协议进程的接口的网络地址
passive-interface *type number*	使一个接口不再发送路由选择更新
router rip	在路由器上启动 RIP 路由选择进程
send-lifetime *start-time*{**infinite**|*end- time*|**duration** *seconds*}	设置一个时间段，用来指定钥匙链上的认证钥匙可被发送的有效时间
show ip route [*address* [*mask*]][*protocol*[*process-ID*]]	显示当前路由表的全部或某条路由
show ipv6 rip	显示 RIPng 协议的信息
show ipv6 route rip	显示通过 RIPng 协议注入到 IPv6 路由表中的路由信息
Version	指定 RIP 路由选择进程的版本号

6.8 推荐读物

“RIP Version 2”，即 RFC 2453，Malkin.G 于 1998 年 11 月编写。

“RIPng for IPv6”，即 RFC 2080，Malkin.G 和 R.Minnear 于 1997 年 1 月编写。

6.9 复 习 题

1．RIPv2 的消息格式中包含了哪 3 个新的字段？

2．除了复习题 1 中 3 个字段定义的扩展特性外，RIPv2 相比 RIPv1 还有哪两个主要的改变？

3．RIPv2 协议使用的多播地址是什么？多播方式通告消息与广播方式相比有什么好处？

4．在 RIPv2 的消息中，路由标记字段的用途是什么？

5．在 RIPv2 的消息中，下一跳字段的用途是什么？

6．RIPv2 协议使用的 UDP 端口号是多少？

7．RIPng 协议使用的 UDP 端口号是多少？

8．什么特性要求路由选择协议必须是一个无类别路由选择协议？

9．什么特性要求路由选择协议必须使用 VLSM？

10．在 Cisco 的 RIPv2 中，可以使用哪两种类型的认证？它们都在 RFC 2453 中定义了吗？

6.10 配置练习

1．在图 6-12 的例子里，路由器 Taos 配置成可以发送版本 1 和版本 2 的更新消息，因此 Linux 主机 Pojoaque 上的“routed”进程可以理解来自路由器 Taos 的更新。除了使用 **ip rip send version** 命令，还有其他的方法配置路由器 Taos 吗？

2．一个网络分配到地址 192.168.100.0，划分这个地址来满足下面的需求：

- 1 个含有 50 台主机的子网；
- 5 个含有 10 台主机的子网；
- 1 个含有 25 台主机的子网；
- 4 个含有 5 台主机的子网。
- 10 条串行链路

3．配置图 6-22 中的 4 台路由器运行 RIP 协议。路由器 RTC 运行的版本是 IOS 10.3，并由于策略原因不能升级。

4．配置图 6-22 中的路由器 RTB 和 RTD，使其在串行链路上对交换的 RIP 更新进行认证。

5．配置图 6-22 中的路由器 RTB 和 RTD，使其在配置练习 4 的认证钥匙生效后的 3 天改用一个新的认证钥匙，这个新钥匙生效 10 小时后再改用另一个钥匙。

6．配置图 6-22 中路由器 RTA 和 RTB 运行 RIPng。在接口上分配下面的 IPv6 前缀：

RTA:

2001:DB8:0:1::/64

2001:DB8:0:2::/64

RTB:

2001:DB8:0:3::/64

2001:DB8:0:2::/64

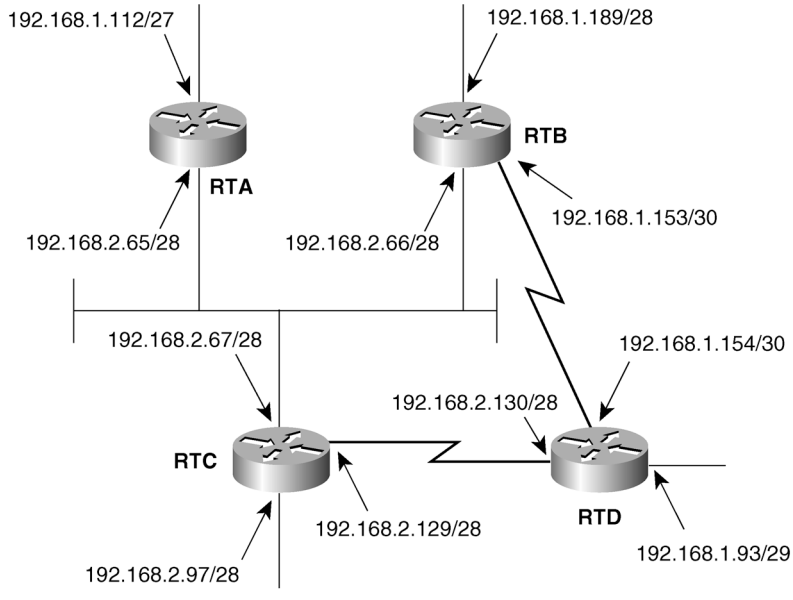


图 6-22 配置练习 3~6 的网络

6.11 故障诊断练习

1．示例 6-32～示例 6-34 显示了图 6-23 中的 3 台路由器的配置。哪些子网出现在每一台路由器的路由表中？在每一台路由器上，哪些子网是可达的？哪些子网（如果有的话）是不可达的？

2．将图 6-23 中的路由器 RTA 和 RTB 的配置改变如下：[1]

```
interface Ethernet0
ip address 192.168.13.35 255.255.255.224
ip rip receive version 1 2
```

这个改变配置的结果会有一些子网增加到路由表中吗？解释一下为什么有或为什么没有？

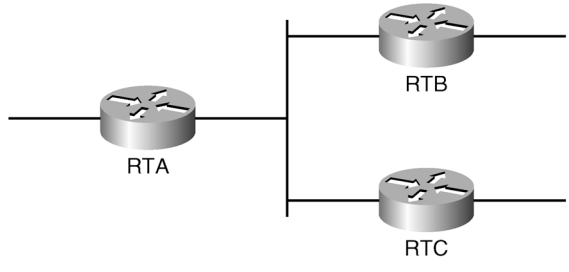


图 6-23 故障诊断练习 1~2 的网络

[1] 显示的是路由器 RTB 的配置，除了 IP 地址 192.168.13.34 外，路由器 RTA 的配置和路由器 B 是相同的。

示例 6-32 图 6-23 中路由器 RTA 的配置

```
RTA#show running-config
Building configuration...
Current configuration:
!
version 11.2
no service udp-small-servers
no service tcp-small-servers
!
hostname RTA
!
!
!
interface Ethernet0
  ip address 192.168.13.86 255.255.255.248
!
interface Serial0
  no ip address
  shutdown
!
interface Serial1
  no ip address
  shutdown
!
interface TokenRing0
  ip address 192.168.13.34 255.255.255.224
  ring-speed 16
!
router rip
  version 2
  network 192.168.13.0
!
no ip classless
!
!
line con 0
line aux 0
line vty 0 4
  login
!
end
RTA#
```

示例 6-33 图 6-23 中路由器 RTB 的配置

```
RTB#show running-config
Building configuration...
Current configuration:
!
version 11.2
no service udp-small-servers
no service tcp-small-servers
!
hostname RTB
!
!
!
interface Ethernet0
  ip address 192.168.13.90 255.255.255.240
!
interface Serial0
```

（待续）

```
  no ip address
  shutdown
!
interface Serial1
  no ip address
  shutdown
!
interface TokenRing0
  ip address 192.168.13.35 255.255.255.224
  ring-speed 16
!
router rip
  version 2
  network 192.168.13.0
!
no ip classless
!
!

line con 0
line aux 0
line vty 0 4
  login
!
end
RTB#
```

示例 6-34 图 6-23 中路由器 RTC 的配置

```
RTC#show running-config
Building configuration...
Current configuration:
!
version 11.1
service udp-small-servers
service tcp-small-servers
!
hostname RTC
!
!
!
interface Ethernet0
  ip address 192.168.13.75 255.255.255.224
!
interface Serial0
  no ip address
  shutdown
!
interface Serial1
  no ip address
  shutdown
!
interface TokenRing0
  ip address 192.168.13.33 255.255.255.224
  ring-speed 16
!
router rip
  network 192.168.13.0
!
no ip classless
!
line con 0
```

（待续）

```
line 1 8
line aux 0
line vty 0 4
  login
!
end
RTC#
```

本章包括以下主题：

- EIGRP 的前身：IGRP 协议回顾；
- 从 IGRP 到 EIGRP；
- EIGRP 的基本原理与实现；
- 配置 EIGRP；
- EIGRP 故障诊断。

第 7 章

增强型内部网关路由选择协议（EIGRP）

增强型内部网关路由选择协议（Enhanced Interior Gateway Routing protocol，EIGRP）是在 Cisco IOS 9.21 版中首次发布的，顾名思义，它是 Cisco 内部网关路由选择协议（Cisco Interior Gateway Routing protocol，IGRP）的增强版。这个命名是比较恰当的，因为它不像 RIPv2 协议那样，EIGRP 协议对 IGRP 协议所增加的扩展特性远远多于 RIPv2 对 RIPv1 的扩展。和 IGRP 协议一样，EIGRP 协议依然是一个距离矢量协议，并且使用了 IGRP 协议所用的复合度量。除此之外，EIGRP 协议和 IGRP 协议几乎没有更多的相似之处。

IGRP 协议在 IOS 软件系统的 12.2（13）T 和 12.2（R1s4）S 版本中已经停止使用。虽然是技术不断创新的时代，那些希望比 RIP 能够提供更多性能的现代网络操作人员已经转移到 EIGRP 协议或 OSPF 协议，而不是转移到改进一点性能的 IGRP 协议。但是，在学习 EIGRP 协议之前，首先了解一下该协议的发展还是有用的。

7.1 EIGRP 的前身：IGRP 协议回顾

20 世纪 80 年代中期，作为对 RIP 协议局限性的回应，Cisco 公司开发了 IGRP 协议，其中最重要的变化就是跳数的度量和 15 跳对网络口径大小的限制。IGRP 通过多种路由变量参数计算出一个复合型的度量，并提供一些“旋钮”供这些变量参数施以权重，以便反映和衡量网络的一些特征和需求。虽然跳数并不作为这些变量参数之一，但 IGRP 协议还是延续使用了跳数，并且能够实现网络最大为 255 跳的需求。

IGRP 协议相对于 RIP 协议还有其他一些优点：

- 非等价负载均衡（unequal-cost load balancing）；
- 更新周期是 RIP 协议的 3 倍时间长；
- 更新数据包格式更有效。

IGRP 协议和 EIGRP 协议的一个主要缺点是它们是 Cisco 公司的私有协议，因此，只能在 Cisco 公司的产品平台上使用，而 RIP 协议则可以作为所有平台上的任何 IP 路由选择进程的一部分来使用。

Cisco 公司开发 IGRP 协议的主要目的是创建一个功能强大的通用协议，以便使它能适应已选路由协议簇的多样性。IGRP 协议除了作为 IP 路由选择协议，它还适用于 ISO 无连接网络协议（Connectionless Network Protocol，CLNP）的路由。这个适应于多协议的性能也是 EIGRP 协议的特性，它不仅可以用来进行 IP 网络的路由，也可以用于 IPX 和 AppleTalk 网络的路由。

从宏观的角度来看，IGRP 协议继承了许多 RIP 协议的操作特点。IGRP 协议也是一个有类别距离矢量型协议，除了被水平分隔法则抑制的路由外，IGRP 将不断地周期性地向邻居路由器广播它的整张路由选择表。像 RIP 协议一样，路由器启动时，IGRP 在所有运行 IGRP 协议的接口上广播出一个请求数据包，并对收到的更新执行一个完整性的检查，用来验证更新数据包的源地址是否和收到更新的那个子网属于同一个子网。[1]新的带有可达路由度量值的路由更新条目将会被放置在路由表中，并且仅当它所带的度量值小于到达相同目的地址的原有路由条目的度量值时，才能替代原有的路由条目。IGRP 协议同样使用带毒性反转的水平分隔法则、触发更新和抑制计时器等手段来保证它的稳定性；同 RIP 协议一样，IGRP 协议也在网络边界上进行地址汇总。

与 RIP 协议不同，IGRP 协议不使用 UDP 来访问数据包，它直接通过 IP 层的协议号 9 来进行数据包访问。

7.1.1 进程域

IGRP 协议也使用进程域的概念。通过定义和跟踪多个进程域，我们可以把一个域内的通信和另一个域内的通信隔离开来。域间的流量可以通过路由重新分配（第 11 章）和路由过滤（第 13 章）来严密地控制。

图 7-1 显示了进程域和路由选择域的对照。在这里定义了两个自主系统（AS）：AS 10 和 AS 40，这些系统是路由选择域——在一个共同的管理机构下运行的一个或多个 IGP 协议的路由器集合。它们通过外部网关协议（Exterior Gateway Protocol，在这个实例中，是边界网关协议，或称为 BGP 协议）来通信。

在自主系统 AS 10 中有两个 IGRP 进程域：IGRP 20 和 IGRP 30。在 IGRP 协议内，定义了两个自主系统号 20 和 30，就此处而言，这些数字是用来区分同一个路由选择域内的两个不同路由选择进程的。进程域 IGRP 20 和进程域 IGRP 30 是通过和这两个进程域都相连的一台路由器来进行通信的。这台路由器同时运行两个 IGRP 进程，并且在这两个进程之间自动地进行路由再分配。

在 IGRP 的路由更新消息中，IGRP 把路由条目分成 3 类：内部路由（interior route）、系

[1] 这个完整性检查可以用命令 **no validate-update-sourced** 来禁止它生效。

统路由（system route）和外部路由（exterior route），每个 IGRP 的路由条目都属于这 3 个类别中的一个。

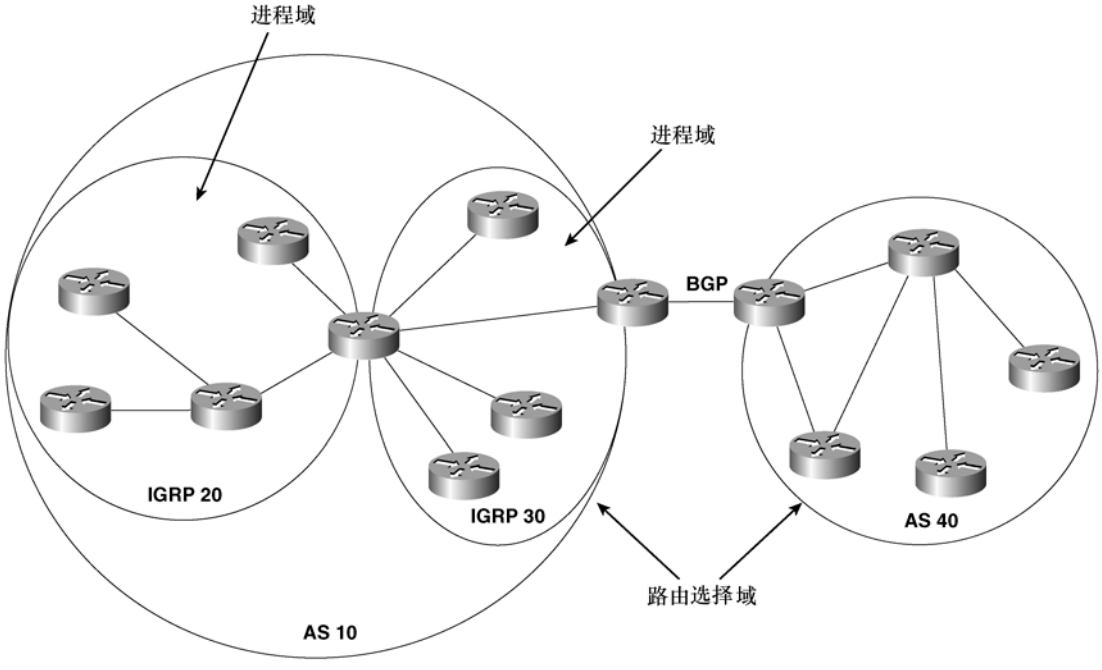


图 7-1 一个自主系统号可以指定一个路由选择域，即在一个单一的管理域内运行一个或多个 IGP 协议的一组路由器集合。在 IGRP 中，一个自主系统号也可以指定一个进程域，即依赖于一个单一的路由选择进程共享路由选择信息的一组路由器

- **内部路由（Interior Route）**——是指到达属于某个主网络的子网地址的路径，这里的主网络是指正在广播这条路由更新的数据链路的主网络地址。换句话说，作为内部路由被通告的子网对于通告路由器和接收路由器共同相连的主网络来说是“本地”的。
- **系统路由（System Route）**——是指到达在网络边界路由器上被汇总的网络地址的路径。
- **外部路由（Exterior Route）**——是指到达被标记成缺省网络（default network）的路径。对于缺省网络，路由器将直接发送所有的数据包而不对更具体的目的网络进行查找匹配。[1]缺省网络和其配置方法将在第 12 章中讲述。

图 7-2 显示了 IGRP 协议是怎样使用这 3 类路由的。路由器 LeHand 和 Tully 都与子网 192.168.2.64/26 相连，所以主网络 192.168.2.0 被认为是这两台路由器“共享”的“本地”网络。而路由器 LeHand 和子网 192.168.2.192/26 相连，显然子网 192.168.2.192/26 是连接路由器 LeHand 和 Tully 的主网络 192.168.2.0 的另一个子网。因此，路由器 LeHand 把子网 192.168.2.192/26 作为内部路由通告给路由器 Tully。

然而，与路由器 LeHand 和路由器 Thompson 相连的本地网络却是 192.168.3.0。由于路由器 LeHand 是主网络 192.168.2.0 和 192.168.3.0 的边界路由器，因此网络 192.168.2.0 被作为一条系统路由通告给路由器 Thompson，同样的，网络 192.168.3.0 也被作为一条系统路由通告给路由器 Tully。

[1] 把缺省网络分类归到外部路由是 IGRP 和 EIGRP 协议所独有的。像 RIP 和 OSPF 等开放性的协议使用地址 0.0.0.0 通告缺省网络。

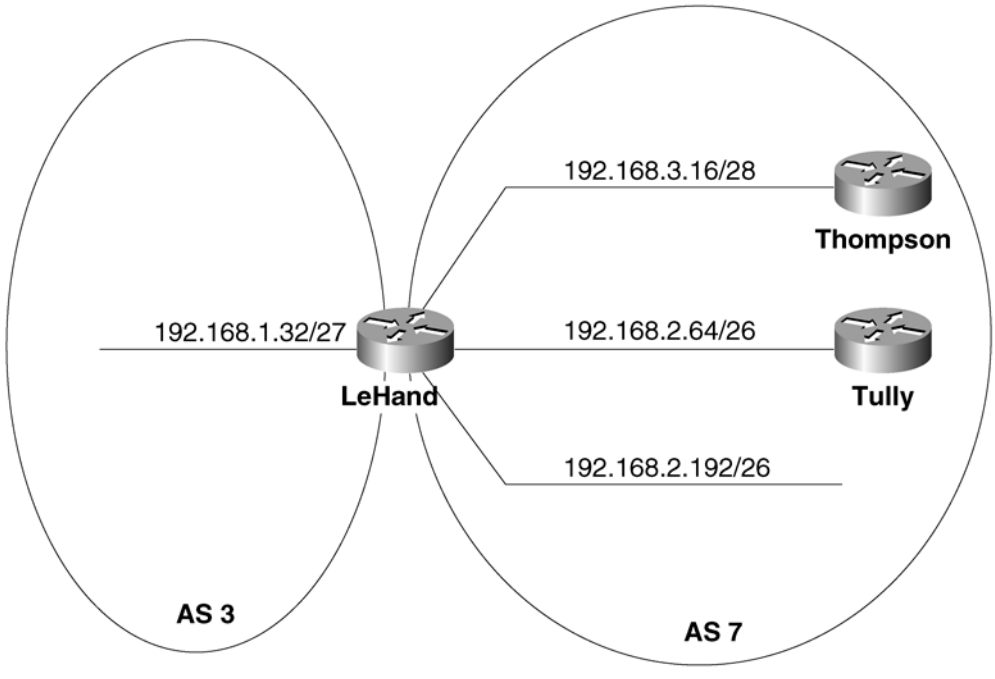


图 7-2　路由器 LeHand 把子网 192.168.2.192/26 作为一条内部路由通告给路由器 Tully，把网络 192.168.3.0 作为一条系统路由通告给路由器 Tully，并把网络 192.168.1.0 作为一条外部路由通告给路由器 Tully

网络 192.168.1.0 属于另一个自主系统，因而路由器 LeHand 把它作为一条缺省网络地址来通告，而 192.168.1.0 就被作为一条外部路由通告给路由器 Tully 和 Thompson。

7.1.2　IGRP 的计时器和稳定性

IGRP 协议的更新周期是 90s。为了防止更新计时器的同步，IGRP 针对每一个更新时间都减掉一个最大为其 20%的随机抖动变量；因此，每个更新周期所需要的时间将在 72~90s 之间变化。

当一条路由首次被用时，这条路由的无效计时器就会被设置成 270s，即是更新周期的 3 倍时间。同时，刷新计时器设置成 630s，即是更新周期时间的 7 倍长。每次接收到路由的更新报头后，这些计时器都将被重新初始化。如果在收到一条更新报头之前无效计时器的计时超时了，这条路由就会被标记成不可到达。但是，在路由器的刷新计时器超时前，这条路由还会被保留在路由选择表中，并且作为不可达的路由通告出去；如果刷新计时器超时了，这条路由才会从表中删除掉。

与 RIP 协议时长为 30s 的计时器相比，IGRP 协议使用了 90s 的计时器，这意味着 IGRP 协议的周期性更新将比 RIP 协议占用更少的带宽。然而，这是在同样的情况下，IGRP 协议比 RIP 协议的收敛更慢为代价的。例如，如果一台路由器离线了，IGRP 协议需要 3 倍于 RIP 协议的时间才能检测到这个已不存在的邻居。

如果一条路由的目的地址变为不可达，或下一跳路由器增大了到达目的地址的度量以至于引起一个触发更新的话，那么这条路由将会进入一个 280s（3 倍的更新周期加上 10s）的抑制时间状态。直到抑制计时器超时之前，有关这个目的地址新的信息都不会被路由器接受。IGRP 协议的抑制特性可以用命令 **no metric holddown** 来禁止。在一个没有路由环路的网络

拓扑中，抑制特性没有实际的意义，禁止该特性将有助于减少 IGRP 的收敛时间。

缺省的计时器可以用下面的命令来改变：

```
timers basic update invalid holddown flush [sleeptime]
```

除了 ***sleeptime*** 选项，这条命令曾在改变 RIP 协议的计时器时使用过。***Sleeptime*** 是一个周期性的毫秒（ms）级的计时器，在收到一条触发更新后，它被用来延迟一个正常的路由选择更新。

7.1.3 IGRP 的度量

与 RIP 协议相比，IGRP 协议引入的一个最重要的变化是它基于链路的特征，使用了多种度量参数，这些也继承到了 EIGRP 协议中。学习 IGRP 协议如何运用这些度量参数，对于掌握 EIGRP 协议怎样运用它的度量参数是有用的。

在 IGRP 协议中，IGRP 将根据链路的特性计算出一个“复合”的度量值，这些链路特性包括链路带宽、时延、负载和可靠性。缺省条件下，IGRP 选用路由的链路带宽和时延作为度量值。如果把数据链路想象成一个管道，那么带宽就像是这个管道的宽度，而时延就像是这个管道的长度。换句话说，带宽是数据传送能力的量度，而时延是端-端传送时间的量度。链路的另外两个特性——负载和可靠性只有在路由器上进行人工配置后才会被应用。虽然 IGRP 协议不使用 MTU（Maximum Transmission Unit，最大传输单元）作为计算复合度量值的参数，但 IGRP 也会跟踪每条路由上的最小 MTU 的大小。参见示例 7-1，可以通过命令 **show interfaces** 来观察一个特定接口上有关 IGRP 的复合度量值的大小。

示例 7-1 show interface 命令的输出包含了关于这个接口的度量的统计。这个以太网接口的度量值显示 MTU＝1 500 字节，带宽＝10Mbit/s，时延＝1 000μs，可靠性＝100%，负载=39%（最小负载）

```
Newfoundland#show interfaces ethernet 0
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0000.0c76.5b7c (bia 0000.0c76.5b7c)
  Internet address is 10.2.1.2/24
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:07, output 00:00:01, output hang never
  Last clearing of "show interface" counters never
  Queueing strategy: fifo
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     601753 packets input, 113607697 bytes, 0 no buffer
     Received 601753 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 input packets with dribble condition detected
     693494 packets output, 557731861 bytes, 0 underruns
     0 output errors, 5 collisions, 13 interface resets
     0 babbles, 0 late collision, 48 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
Newfoundland#
```

带宽用 kbit/s 单位来表示，它在计算链路的度量值时仅作为一个静态的值，没有必要反映出链路实际使用的带宽；也就是说，带宽不需要动态地去量度。例如，不论和串行接口相连的是 T1 的还是 56KB 的链路，串行接口的缺省带宽都是 1 544kbit/s。这个缺省的带宽值可以通过 **bandwidth** 命令来更改。

IGRP 的更新消息使用 3 个八位组字节来表示 IGRP“带宽”，在本书中用 BW_{IGRP} 表示，它是用因子 10^7 除以带宽得来的，因此，如果接口的带宽是 1544，那么

$BW_{IGRP}=10^7/1\ 544=6\ 476$，或 0x00194C。

时延，像带宽一样，也是一个静态特征的度量值，不需要动态地去量度。时延可以通过 **show interface** 命令显示的 DLY 参数来表示，单位是 μs（微秒）。一个接口的缺省时延可以通过命令 **delay** 进行更改，并以 10μs 作为命令配置的最小计量单位。示例 7-2 显示了用 **bandwidth** 和 **delay** 命令更改示例 7-1 中的缺省带宽和时延的情况。

示例 7-2　通过 bandwidth 和 delay 命令改变了接口 e0 的缺省度量值，使用 show interface 命令可以在输出端查看更改后的新度量值

```
Newfoundland(config)#interface e0
Newfoundland(config-if)#bandwidth 75000
Newfoundland(config-if)#delay 5
Newfoundland(config-if)#^Z
Newfoundland#
%SYS-5-CONFIG_I: Configured from console by console
Newfoundland#show interfaces ethernet0
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0000.0c76.5b7c (bia 0000.0c76.5b7c)
  Internet address is 10.2.1.2/24
  MTU 1500 bytes, BW 75000 Kbit, DLY 50 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:02, output 00:00:02, output hang never
  Last clearing of "show interface" counters never
  Queueing strategy: fifo
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     601888 packets input, 113637882 bytes, 0 no buffer
     Received 601888 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 input packets with dribble condition detected
     693646 packets output, 557884632 bytes, 0 underruns
     0 output errors, 5 collisions, 13 interface resets
     0 babbles, 0 late collision, 48 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
Newfoundland#
```

在 IGRP 的更新消息中，时延也是用 3 个八位组字节来表示，并可以通过命令 **delay** 来改变，同样是以 10μs 作为最小计量单位。为了避免误解，这个数值用 DLY_{IGRP} 来表示，以便区别于 DLY，DLY_{IGRP} 可以通过命令 **show interface** 来观察，单位是 μs（微秒）。例如，如果 DLY 的值是 50，那么

DLY_{IGRP} =DLY/10=50/10= 5，或 0x000005。

IGRP 通过设定 $DLY_{IGRP}=0$xFFFFFF 来标识一条不可到达的路由，这个数值大约为 167.8s，因此，一条 IGRP 路由端-端的最大时延是 167s。

因为 IGRP 协议使用带宽和时延来作为它的缺省度量值，因此，这些特性参数必须配置正确，并且要在所有的 IGRP 路由器的接口上统一规划配置。

除非有一个很好的理由，并且对更改这些参数的配置后所产生的结果有个清楚地理解，否则最好不要更改一个接口的带宽和时延参数。在大多数的情况下，最好保留使用这些参数的缺省值而不要加以改变。有一个值得注意的例外就是串行接口，正如本节前面所提及的，在 Cisco 路由器上，不管串行接口相连的是什么带宽的数据链路，它都会把串行接口的缺省带宽设置为 1544。这时，应该使用命令 **bandwidth** 在接口上设置链路的实际带宽。

这里请注意，很重要的一点是在 OSPF 协议中也使用带宽来计算它的度量值。因此，在一个同时运行 IGRP 和 OSPF 协议的网络中，如果要改变 IGRP 的度量值，应该使用 **delay** 来影响 IGRP 的度量值，如果改变带宽将会同时影响到 IGRP 和 OSPF。

表 7-1 中列出了一些常用接口的带宽和时延（注意，串行接口的缺省带宽总是 1544；表 7-1 显示了使用 **bandwidth** 命令来反映实际相连的链路带宽的效果）。

表 7-1　　常用 BW_{IGRP} 和 DLY_{IGRP} 的数值

介质	带宽	BW_{IGRP}	时延	DLY_{IGRP}
100M ATM	100 000kbit/s	100	100μs	10
快速以太网	100 000 kbit/s	100	100μs	10
FDDI	100 000 kbit/s	100	100μs	10
HSSI	45 045 kbit/s	222	20 000μs	2000
16MB 令牌环	16 000 kbit/s	625	630μs	63
以太网	10 000 kbit/s	1 000	1 000μs	100
T1	1 544 kbit/s	6 476	20 000μs	2 000
DS0	64 kbit/s	156 250	20 000μs	2 000
56KB	56 kbit/s	178 571	20 000μs	2 000
Tunnel	9 kbit/s	1 111 111	500 000μs	50 000

可靠性是一个动态的度量参数，它使用一个 8 位的数字来表示，255 表示 100%的可靠链路，而 1 表示最低可靠的链路。在命令 **show interface** 的输出中，可靠性被表示成 255 的分数，例如，234/255（参见示例 7-3）。

示例 7-3　这个接口显示了该接口的可靠性是 234/255，或 91.8%

```
Casablanca#show interface ethernet0
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0000.0c76.5b7c (bia 0000.0c76.5b7c)
  Internet address is 172.20.1.1 255.255.255.0
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec, rely 234/255, load 1/255
  Encapsulation ARPA, loopback not set, keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 4:00:00
  Last input 0:00:28, output 0:00:06, output hang never
  Last clearing of "show interface" counters 0:06:05
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     22 packets input, 3758 bytes, 0 no buffer
     Received 21 broadcasts, 0 runts, 0 giants
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 input packets with dribble condition detected
     125 packets output, 11254 bytes, 0 underruns
     39 output errors, 694 collisions, 0 interface resets, 0 restarts
     0 output buffer failures, 0 output buffers swapped out
Casablanca#
```

在 IGRP 的更新里，负载是一个 8 位的数字。在 **show interface** 命令的输出中表示成一个 255 的分数，例如，40/255（参见示例 7-4）；1 表示最小的负载链路，255 表示 100%的负载链路。

示例 7-4 这个接口显示了一个 40/255 的负载链路，或 15.7%

```
Yalta#show interface serial 1
Serial1 is up, line protocol is up
  Hardware is HD64570
  Internet address is 172.20.20.2 255.255.255.0
  MTU 1500 bytes, BW 56 Kbit, DLY 20000 usec, rely 255/255, load 40/255
  Encapsulation HDLC, loopback not set, keepalive set (10 sec)
  Last input 0:00:08, output 0:00:00, output hang never
  Last clearing of "show interface" counters 0:05:05
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 10000 bits/sec, 1 packets/sec
  5 minute output rate 9000 bits/sec, 1 packets/sec
     456 packets input, 397463 bytes, 0 no buffer
     Received 70 broadcasts, 0 runts, 0 giants
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     428 packets output, 395862 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets, 0 restarts
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

如果可靠性或负载被用来作为一个度量或复合度量的一部分，计算度量的算法不允许在出错率或信道占用率上突然发生变化而影响网络的不稳定。举一个例子来说明，如果在一个可能出现变化的网络上，突发的大流量可能会导致路由进入抑制（holddown）状态，而通信量的突然降低可能会触发一个更新。为了防止度量频繁的改变，可靠性和负载是基于 5min 时间常数的指数加权平均计算的，它们每 5s 被更新一次。

对于每个 IGRP 路由的复合度量，可以用下面的公式计算：

$$metric=[k1\times BW_{IGRP(min)}+(k2\times BW_{IGRP(min)})/(256-LOAD)+k3\times DLY_{IGRP(sum)}]\times[k5/(RELIABILITY+k4)]$$

在这里，$BW_{IGRP(min)}$是沿着路由路径到达目的网络的所有出站接口的 BW_{IGRP} 带宽中的最小值，而 $DLY_{IGRP(sum)}$是这条路由路径 DLY_{IGRP} 时延的总和。

系数 k1 到 k5 是可配置的加权值，它们的缺省值是：k1=k3=1，k2=k4=k5=0。这些缺省值可以通过下面的命令来更改：[1]

metric weights tos *k1 k2 k3 k4 k5*[3]

如果 k5 被设置为 0，则[k5/(RELIABILITY+k4)]这一项将不使用。[2]

使用 k1~k5 给定的缺省值，IGRP 的复合度量的计算公式将简化成缺省的度量：

$metric=BW_{IGRP(min)}+DLY_{IGRP(sum)}$。

在图 7-3 中的例子显示了网络的每个路由器接口的带宽和时延的配置，以及计算出 IGRP 路由度量的路由器之一的转发数据库。[3]

路由选择表本身只显示了已经计算出来的度量，如果需要查看 IGRP 记录的每条路由实际的参数值，可以用 **show ip route** *address* 命令，参见示例 7-5。这里路由器 Casablanca 到子

[1] tos 是 Cisco 公司最早打算使用 IGRP 协议来提供支持服务类型的路由选择时遗留下来的参数；但这个计划从来没有被采纳过，因而 tos 的值总是被设定为 0。

[2] 译者注：原来的度量计算公式将变为 $metric=k1\times BW_{IGRP(min)}+(k2\times BW_{IGRP(min)})/(256-LOAD)+k3\times DLY_{IGRP(sum)}$。

[3] 需要注意，IGRP 协议的管理距离是 100。

网 172.20.40.0/24 的路由路径上的最小带宽是 512kbit/s，最小带宽的链路接口位于路由器 Quebec 的出站接口。该路由时延的总和是(1000+20000+20000+5000)=46000μs。

$BW_{IGRP(min)} = 10^7 /512 = 19531$

$DLY_{IGRP(sum)} = 46000/10 = 4600$

$metric = BW_{IGRP(min)} + DLY_{IGRP(sum)} = 19531 + 4600 = 24131$

示例 7-5 中显示了除了跳数之外 IGRP 还记录了沿着路由最小的 MTU。MTU 不用来作度量的计算。跳数是下一跳路由器报告的跳数，仅仅用来限制网络规模的大小。缺省条件下，最大的跳数是 100，也可以通过命令 **metric maximum-hops** 配置成 1~255 之间的数值。如果一条路由超过了设置的最大跳数，那么它的时延将被设置成 0xFFFFFF，而变成一条不可达的路由。

请注意，这里所有的度量都是基于沿着路由方向的路由器出站接口计算的。例如，路由器 Yalta 到达子网 172.20.4.0/24 的路由度量是不同于路由器 Casablanca 到达子网 172.20.40.0/24 的路由度量的。这是因为，路由器 Yalta 和路由器 Quebec 之间的链路带宽的配置不同，而且到达这两个目标子网的出站接口上时延也不同。

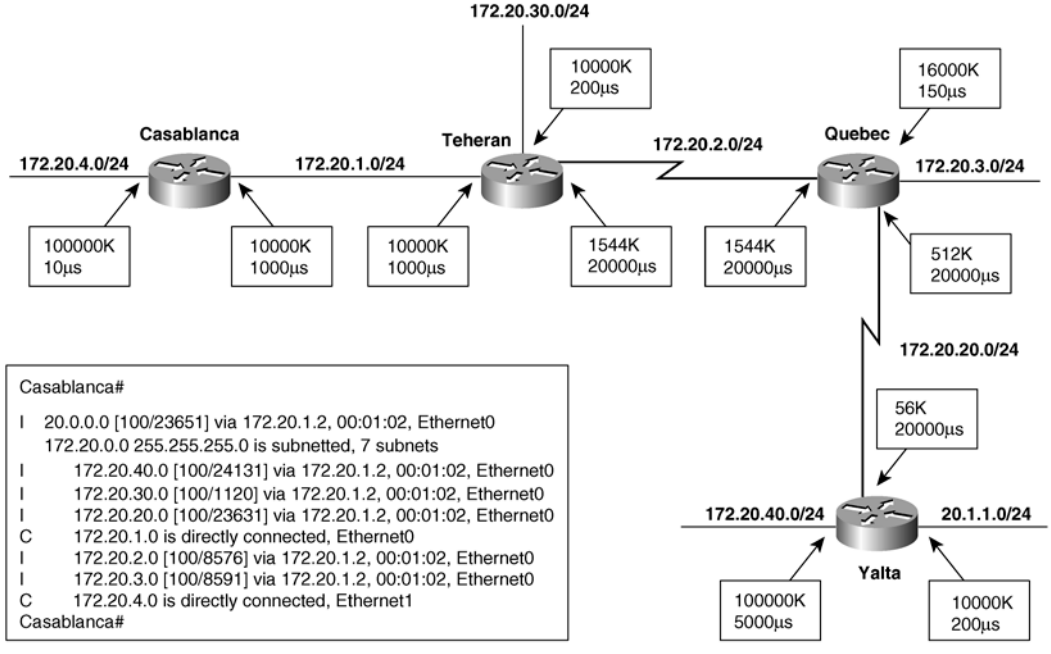


图 7-3 缺省条件下，IGRP 的度量是由时延的总和加上最小的带宽计算得来的

示例 7-5 路由器 Casablanca 到子网 172.20.40.0/24 的路由的度量由 512kbit/s 的最小带宽和 46000μs 的路由时延总和计算得出

```
Casablanca#show ip route 172.20.40.0
Routing entry for 172.20.40.0 255.255.255.0
  Known via "igrp 1", distance 100, metric 24131
  Redistributing via igrp 1
  Advertised by igrp 1 (self originated)
  Last update from 172.20.1.2 on Ethernet0, 00:00:54 ago
  Routing Descriptor Blocks:
  * 172.20.1.2, from 172.20.1.2, 00:00:54 ago, via Ethernet0
      Route metric is 24131, traffic share count is 1
      Total delay is 46000 microseconds, minimum bandwidth is 512 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

7.2 从 IGRP 到 EIGRP

最初推动开发 EIGRP 的动机仅仅是简单的为了使 IGRP 支持无类别路由选择。但是，在协议开发的早期，工作于该开发项目的工程师们接受了一些有关一个新的收敛算法的学术建议，决定在 IGRP 协议的性能扩展项目中使用这个算法。结果，EIGRP 协议除了保留 IGRP 协议引入的一些概念外，例如多种度量、协议域和非等价负载均衡，该协议和 IGRP 有明显的不同。

EIGRP 协议有时也被描述成一个具有链路状态协议行为特性的距离矢量协议。在这里，对第 4 章中的广泛论述作一个扼要的重述：距离矢量协议是路由器之间共享路由器所知道的所有信息，但仅仅限于在与之直连的邻居之间共享；而链路状态协议虽然只通告它们直连链路的信息，但是链路状态协议可以在它们的路由选择域或区域内的所有路由器上共享这些信息。

到目前为止，所讨论的所有距离矢量协议的运行都是基于 Bellman-Ford（或 Ford-Fulkerson）算法或其一些派生算法的基础之上的。因此，这些协议易于产生路由选择环路和计数无穷大的问题。结果是，这些协议必须要采取一些避免路由选择环路的措施，像水平分隔、路由毒性逆转和抑制计时器等。由于每一台路由器在向它的邻居传送路由信息之前，都必须对收到的路由信息运行路由选择算法，因此大型网络的收敛可能会变得比较慢。更为重要的是，距离矢量协议在通告路由时，如果网络核心的关键链路发生了变化，就意味着有很多发生变化的路由要进行通告。

相对于距离矢量协议，链路状态协议受到路由选择环路和有害路由选择信息的影响就小得多了。首先，链路状态数据包的转发不依赖于路由计算的执行，因而大型网络就可以更快速地进行收敛。其次，它只通告链路和链路的状态，而不通告路由，这意味着链路的变化不会引起使用这条链路的所有路由都被通告。

其他的路由选择协议执行路由的计算——不论是在给它的邻居发送距离矢量更新之前，还是在生成网络拓扑数据库之后，它们的共同特性都是单独地进行路由的计算。相反，EIGRP 协议使用了一个称为扩散计算（diffusing computations）的方法——在多台路由器之间通过一个并行的方式执行路由的计算，从而在保持无环路的拓扑时可以随时获取较快的收敛。

虽然 EIGRP 更新仍然是将距离矢量传送给它直连的邻居，但是 EIGRP 更新是非周期的、部分的和有边界的。“非周期的”意思是指更新不是按照规是的时间间隔发送的，而是在度量或网络拓扑发生变化时才发送更新。“部分的”意思是指更新只包含发生变化的路由条目，而不是路由器的所有路由条目。“有边界的”意思是指更新仅仅发送给受到影响的路由器。这些特性意味着，EIGRP 协议比典型的距离矢量协议所使用的带宽少得多，这个特点对路由选择协议在带宽较低而费用较高的广域网（WAN）链路上运行时显得特别重要。

另外一个需要关注的是，如果是在低带宽的广域网链路上进行路由选择，在路由收敛期间，路由选择信息的流量会显得比较大，这时要考虑可以使用的最大带宽。缺省情况下，EIGRP 协议使用的带宽不超过链路总带宽的 50％。后来 IOS 发布的版本允许使用命令 **ip bandwidth-percent eigrp** 来改变这个缺省的百分比。

EIGRP 协议是一个无类别的协议（也就是说，在其路由更新中的每一个路由条目都包含子网掩码）。EIGRP 协议不仅可以利用可变长子网掩码进行子网的划分（如第 6 章中所讲述的），而且可以利用可变长子网掩码进行地址的聚合（address aggregation），即一组主网络地址的汇总。

EIGRP 协议能够使用 MD5 加密校验和对 EIGRP 数据包进行认证。基本的认证和 MD5

认证已经在第 6 章中讲述过了；本章仅讲述一个配置 EIGRP 认证的实例。

最后，EIGRP 协议的一个主要特性是它不仅可以进行 IP 协议的路由选择，而且可以进行 IPX 协议和 AppleTalk 协议的路由选择。

7.3 EIGRP 的基本原理与实现

EIGRP 协议使用和 IGRP 协议相同的公式来计算它的复合度量值。但是，EIGRP 协议使用一个 256 的倍数因子扩展了度量参数，使它具有更好的度量粒度。因此，如果在一条到达目的地的路径上，配置的最小带宽是 512kbit/s，并且配置的总延迟是 46000μs，那么 IGRP 协议计算出的复合度量值应该是 24131。而 EIGRP 协议将使用 256 倍数乘以带宽和延迟参数，从而计算出的度量值是 256×24131=6177536。

如图 7-4 所示，EIGRP 协议包含以下 4 个部件：

- 依赖于协议的模块；
- 可靠传输协议（RTP）；
- 邻居发现和恢复；
- 扩散更新算法（DUAL）。

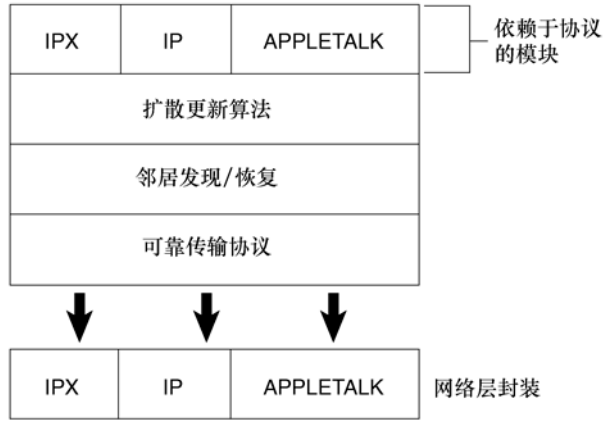


图 7-4 EIGRP 协议的 4 个主要部件。RTP 和邻居发现是使 DUAL 正确操作的更低层次上的协议。DUAL 可以在多个可路由的协议上执行路由计算

本节将讲述 EIGRP 的每一个部件，并特别注重地讲解 DUAL，最后再讨论地址聚合的内容。

7.3.1 依赖于协议的模块（Protocol-Dependent Modules）

EIGRP 协议实现了 IP 协议、IPX 协议和 AppleTalk 协议的模块，它可以担负起某一特定协议的路由选择任务。例如，IPX EIGRP 模块可以负责在 IPX 网络上与其他 IPX EIGRP 进程进行路由信息交换，并且将这些信息传递给 DUAL。另外，IPX 模块也接收和发送 SAP 信息。

正如图 7-4 所示，每个单独模块的通信量被封装在它们各自的网络层协议中；例如，对于 IPX 协议的 EIGRP 通过 IPX 协议数据包传输。

EIGRP 协议在很多情况下和其他路由选择协议自动进行路由重新分配：

- IPX EIGRP 将自动地和 IPX RIP 协议、NLSP 协议进行路由重新分配；
- AppleTalk EIGRP 将自动地和 AppleTalk RTMP 协议进行路由重新分配；
- 如果 IGRP 进程和 EIGRP 进程在同一个自主系统内，那么 IP EIGRP 也将自动地和 IGRP 进行路由重新分配。

EIGRP 和其他 IP 路由选择协议之间的路由重新分配将在第 11 章中讨论。

配置关于 IPX 和 AppleTalk 的 EIGRP 超出了本书的讲解范围，请参阅 Cisco 配置手册以获取更多的信息。

7.3.2 可靠传输协议

可靠传输协议（Reliable Transport Protocol，RTP）用来管理 EIGRP 数据包的发送和接收。可靠的发送是指发送是有保障的而且数据包是有序的发送的。有保障的发送是依赖 Cisco 公司私有的算法来实现的，这个私有的算法被称为“可靠组播（reliable multicast）”，它使用保留的 D 类地址 224.0.0.10。每一个接收可靠组播数据包的邻居都会发送一个单播的确认数据包。

有序的发送是通过在每个数据包中包含两个序列号来实现的。每一个数据包都包含一个由发送该数据包的路由器分配的序列号，这个序列号在每台路由器发送一个新的数据包时递增 1。另外，发送路由器会把最近从目的路由器收到的数据包的序列号放在该数据包中。

在一些实例中，RTP 也可以使用不可靠的发送，不需要确认，而且在使用不可靠发送的 EIGRP 数据包中不包含序列号。

EIGRP 协议使用多种类型的数据包，所有这些数据包都通过 IP 头部的协议号 88 来标识。

- **Hello**（**Hello**）——用于邻居发现和恢复进程。Hello 数据包使用组播方式发送，而且使用不可靠的发送方式。
- **确认**（**Acknowledgments，ACK**）——是不包含数据的 Hello 数据包。ACK 总是使用单播方式和不可靠的发送方式。
- **更新**（**Update**）——用于传递路由更新信息。不像 RIP 协议和 IGRP 协议的更新，EIGRP 协议的这些更新数据包只在必要的时候传递必要的信息，而且仅仅传递给需要路由信息的路由器。当只有某一指定的路由器需要路由更新时，更新数据包就是单播发送的；当有多台路由器需要路由更新时，更新数据包就是组播发送的，例如，路由的度量和拓扑发生变化的时候。更新数据包总是使用可靠的发送方式。
- **查询**（**Query**）**和答复**（**Reply**）——是 DUAL 有限状态机（DUAL finite state machine）用来管理它的扩散计算的。查询消息可以使用组播方式或者单播方式发送，而回复消息总是单播方式发送的。查询和回复数据包都使用可靠的发送方式。
- **请求**（**Request**）——最初打算提供给路由服务器使用的数据包类型。但是这个应用从来没有实现过，在这里提到请求数据包主要是因为在一些老的文档中可能会提及它们。

如果任何数据包通过可靠的方式组播出去，而没有从邻居那里收到一个 ACK 数据包，那么这个数据包就会以单播方式被重新发送给那个没有响应的邻居。如果经过 16 次这样的单播重传还没有收到一个 ACK 数据包的话，那么这个邻居就会被宣告为无效。

在从组播方式切换到单播方式之前等待一个 ACK 时间可以由组播流计时器（multicast flow timer）指定。后续的单播之间的时间可以由重传超时（Retransmission Timeout，RTO）

指定。对于每一个邻居，组播流计时器和重传超时都可以通过平均回程时间（Smooth Round-Trip Time，SRTT）来计算。SRTT 是一个用来衡量路由器发送 EIGRP 数据包到邻居和从邻居那里接收到该数据包的确认所花费的平均时间，以毫秒（ms）为单位。关于 SRTT、RTO 和组播流计时器的精确值的计算公式是私有版权的。

下面两小节将讲述使用不同数据包类型的 EIGRP 的部件。

7.3.3 邻居发现和恢复

因为 EIGRP 协议的更新消息是非周期的，因此有一个发现和跟踪邻居的方法是非常重要的；在这里，邻居是指网络上直连的通告 EIGRP 的路由器。在大多数网络中，Hello 数据包是以组播方式每 5s 发送一次的，其中减掉一个很小的随机时间差用来防止更新的同步。在多点的 X.25、帧中继和 ATM 接口上，由于它们的接入链路速率通常是 T1 或更低的速率，因此它们的 Hello 数据包是以单播方式每 60s 发送一次的。[1]这个比较长的 Hello 数据包时间间隔也缺省地使用在 ATM SVC 和 ISDN PRI 的接口上。在所有的实例中，Hello 数据包都是不进行确认的。缺省的 Hello 数据包的时间间隔可以在每个接口上使用命令 **ip hello-interval eigrp** 进行更改。

当一台路由器从它的邻居路由器收到一个 Hello 数据包时，这个数据包将包含一个抑制时间（holdtime）。这个抑制时间会告诉本路由器，在它收到后续的 Hello 数据包之前等待的最长时间。如果抑制计时器超时了，路由器还没有收到 Hello 数据包，那么将宣告这个邻居不可到达，并且通知 DUAL 这个邻居丢失了。在缺省的情况下，抑制时间是 Hello 时间间隔的 3 倍，也就是说，对于低速的非广播多路访问（NBMA）网络来说是 180s，对于其他所有的网络来说是 15s。这个缺省值可以通告在每个接口上配置命令 **ip hold-time eigrp** 来更改。EIGRP 协议具有在 15s 以内检测邻居丢失的能力，对照 RIP 协议的 180s 和 IGRP 协议的 270s 所花费的时间，显然这是一个对 EIGRP 的快速收敛起很大作用的因素。

每一个邻居的相关信息都记录在一个邻居表中。参见示例 7-6 所示，邻居表（neighbor table）记录了邻居路由器的 IP 地址和收到邻居 Hello 数据包的接口。邻居通告的抑制时间作为 SRTT 和邻居关系建立时间（uptime）也记录在邻居表中，这里的邻居关系建立时间是指从邻居第一次被添加到邻居表后到现在所经过的时间。重传超时（RTO）是指在一个组播方式的数据包发送失败后，路由器等待一个单播方式发送的数据包的确认时间，单位是毫秒（ms）。如果一个 EIGRP 的更新、查询或答复数据包被发送出去，那么这个数据包的一个拷贝就会放在一个重传队列里排队。如果重传超时了还没有收到确认数据包，那么重传队列中数据包的另一个拷贝将被再次发送出去。队列计数（Q Count）就是标识在这个重传队列中等待发送的数据包数量的。从邻居收到的最新的更新、查询或答复数据包的序列号也记录在了邻居表中。可靠传输协议 RTP 跟踪这些序列号，以确保来自邻居的数据包不是无序的。最后，H 列记录了这台路由器所学到的邻居的顺序号。

示例 7-6 show ip eigrp neighbors 命令用来观察 IP EIGRP 的邻居表

```
Wright#show ip eigrp neighbors
IP-EIGRP neighbors for process 1
```

（待续）

[1] 点到点的子接口每 5s 发送一次 Hello 消息。

```
H   Address     Interface   Hold  Uptime    SRTT   RTO   Q   Seq
                            (sec)           (ms)        Cnt  Num
3   10.1.1.2    Et0           10  09:01:27   12    200   0   5
2   10.1.4.2    Se1           13  09:02:11   23    200   0   11
1   10.1.2.2    Et1           14  09:02:12    8    200   0   15
0   10.1.3.2    Se0           12  09:02:12   21    200   0   13
Wright#
```

7.3.4 扩散更新算法

扩散更新算法（Diffusing Update Algorithm，DUAL）是一个收敛算法，它代替了用于其他距离向量协议使用的 Bellman-Ford 或 Ford-Fulkerson 算法。DUAL 算法背后的设计思想是，即使暂时的路由选择环路也会对一个网络的性能造成损害。DUAL 最初是由 E. W. Dijkstra 和 C. S. Scholten 提议的[1]，指的是为了随时能够打破路由环路，而使用扩散计算去执行一个分布式最短路径路由选择。虽然很多研究人员对 DUAL 算法的发展作出了贡献，但是最显著的贡献来自于 J. J. Garcia-Luna-Aceves 的工作。[2]

1. DUAL：预备概念

为了能够正确地操作 DUAL，较低层的协议必须确保满足下面的几个条件：[3]

- 一个节点需要在有限的时间内检测到一个新邻居的存在或一个相连邻居的丢失。
- 在一个正在运行的链路上传送的所有消息应该在一个有限的时间内正确地收到，并且包含正确的序列号。
- 所有的消息，包括改变链路的代价、链路失败和发现新邻居的通告，都应该在一个有限的时间内一次一个地处理，并且应该被有顺序地检测到。

Cisco 的 EIGRP 协议使用邻居发现/恢复和可靠传输协议（RTP）来确定这些前提条件。在介绍 DUAL 之前，先来介绍几个术语和概念。

（1）邻接（adjacency）

刚启动时，路由器使用 Hello 数据包发现它的邻居并标识自己给邻居识别。当邻居被发现时，EIGRP 协议将试图和它的邻居形成一个邻接关系。邻接是指两个互相交换路由信息的邻居之间形成的一条逻辑的关联关系。一旦邻接成功地建立，路由器就可以从它们的邻居那里接收路由更新消息了。这里的路由更新消息包括发送路由器所知道的所有路由和这些路由的度量值。对于每一条路由，路由器都将会基于它邻居通告的距离（distance）和到它的邻居的链路代价计算出一个距离。

（2）可行距离（Feasible Distance，FD）

到达每一个目的地的最小度量将作为该目的网络的可行距离。例如，路由器可能得到 3

---

[1] Edsger W. Dijkstra 和 C. S. Scholten 编写的“Termination Detection for Diffusing Computations.”Information Processing Letters, Vol. 11, No. 1,pp. 1–4: 29 August 1980。

[2] J. J. Garcia-Luna-Aceves.“A Unified Approach for Loop-Free Routing Using Link States or Distance Vectors,”ACM SIGCOMM Computer Communications Review, Vol. 19, No. 4, pp. 212–223: September 1989。
J.J.Garcia-Luna-Aceves.“Loop-Free Routing Using Diffusing Computations,”IEEE//ACM Transactions on Networking, Vol.1,No.1,February 1993。

[3] J.J. Garcia-Luna-Aceves.“Area-Based, Loop-Free Internet Routing.”Proceedings of IEEE INFOCOMM 94. Toronto, Ontario, Canada, June 1994。

条到达子网 172.16.5.0 不同的路由，这 3 条路由计算所得的度量分别是 380672、12381440 和 660868；380672 为可行距离（FD），因为它是经计算到达子网 172.16.5.0 的最小度量。

（3）可行性条件（Feasibility Condition，FC）

可行性条件就是需要满足下面这样的条件——本地路由器的一个邻居路由器所通告的到达一个目的网络的距离是否小于本地路由器到达相同目的网络的可行距离（FD）。

（4）可行后继路由器（Feasible Successor，FS）

如果本地路由器的邻居路由器所通告的到达目的网络的距离满足了 FC，那么这个邻居就会成为该目的网络的一个可行后继路由器。[1] 例如，假定一台路由器到达子网 172.16.5.0 的可行距离是 380672，而它的邻居路由器所通告的到达该目的子网的路由路径的距离是 355072，那么这台邻居路由器就满足 FC，因而成为一台可行后继路由器；如果邻居路由器所通告的距离是 380928，那么这台邻居路由器就不满足 FC，因而也就不能成为一台可行后继路由器。

可行后继路由器和可行性条件的概念是避免环路的一项核心技术，因为可行后继路由器总是“下游路由器（downstream）”（也就是说，可行后继路由器到达目的地的度量距离比本地路由器的可行距离（FD）更短），所以路由器从来不会选择一条导致反过来还要经过它本身的路径——像这样的路径一般有一个大于本地路由器 FD 的距离。

存在一个或多个可行后继路由器的每一个目的网络，将与下面的每一项一起被记录在一个称为“拓扑结构表（topological table）”的表中：

- 目的网络的可行距离；
- 所有的可行后继路由器；
- 每一个可行后继路由器所通告的到达目的网络的通告距离；
- 本地路由器所计算的经过每一个可行后继路由器到达目的网络的距离，也就是基于可行后继路由器所通告的到达目的子网的距离和本地路由器与该可行后继路由器之间相连链路的成本计算所得的距离；
- 与发现每一个可行后继路由器的网络相连的接口。[2]

（5）后继路由器（successor）

对于在拓扑结构表中列出的每一个目的网络，将选用拥有最小度量值的路由并放置到路由表中。通告这条路由的邻居就成为一个后继路由器，或者是到达目的网络的数据包的下一跳路由器。

举一个例子可以帮助我们来澄清这些术语，但首先简要地描述一下本小节这个例子中用到的网络还是必要的。图 7-5 显示了一个基于 EIGRP 的网络，这个网络将在本小节和后续的 3 个小节中使用。[3] 把命令 **metric weights 0 0 0 1 0 0** 添加到 EIGRP 的进程中，因此只使用延迟来进行路由的度量计算。命令 **delay** 使用图中每一个链路上标注的数值。例如，路由器 Wright 和路由器 Langley 的接口和子网 10.1.3.0 相连，那么接口配置的延迟就是 2。这些做法将给下面的例子带来方便和简化。

---

1 后继路由器简单的理解就是指到达目的网络更近一跳的路由器，换句话说，就是下一跳路由器。

2 事实上，这个接口并未明确地显示在路由表中。更确切地说，它是邻居路由器自身的属性。这个约定意味着通过多条并行链路的相同路由器将被 EIGRP 协议看作是多个邻居。

3 在本小节和后续几小节演示的几个图例以及使用的网络示例改编自 Garcia-Luna 先生的“Loop-Free Routing Using Diffusing Computations”，并得到了他的许可。

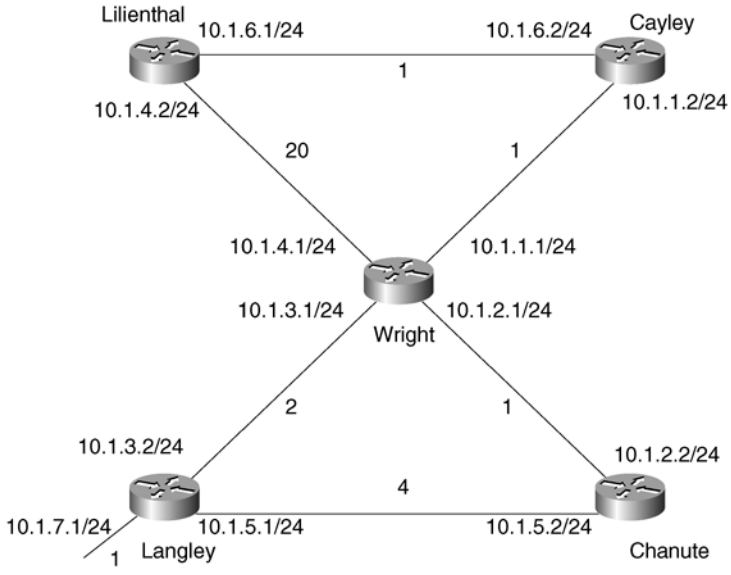


图 7-5　本小节和后续的两个小节的图例和例子都是基于 EIGRP 协议的网络

需要指出的是，虽然这里使用的延迟参数为了简化而不太实用，但是进行度量的方法还是很实用的。很多参数是通过接口命令 **bandwidth** 所指定的带宽进行计算的。如命令 **ip bandwidth-percent eigrp** 就是直接应用于 EIGRP 的；但 OSPF 的代价并不是直接地应用于 OSPF 的。 因此，除了需要设置串行链路的带宽与其实际带宽相符，还应该避免改变带宽的配置。如果需要改变一个接口的度量来影响 EIGRP（或 IGRP 协议）的路由选择，还应该使用命令 **delay**。这样可以避免很多令人头痛的问题发生。

在示例 7-7 中，可以使用命令 **show ip eigrp topology** 来查看路由器 Langley 的拓扑结构表。图 7-5 中显示的 7 个子网的每个子网和这些子网的可行后继路由器，都列在拓扑结构表中。例如，子网 10.1.6.0 的可行后继路由器 10.1.3.1（路由器 Wright）和 10.1.5.2（路由器 Chanute），分别通过接口 S0 和 S1 到达。

示例 7-7　路由器 Langley 的拓扑结构表

```
Langley#show ip eigrp topology
IP-EIGRP Topology Table for process 1

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - Reply status

P 10.1.3.0/24, 1 successors, FD is 512
         via Connected, Serial0
P 10.1.2.0/24, 1 successors, FD is 768
         via 10.1.3.1 (768/256), Serial0
         via 10.1.5.2 (1280/256), Serial1
P 10.1.1.0/24, 1 successors, FD is 768
         via 10.1.3.1 (768/256), Serial0
         via 10.1.5.2 (1536/512), Serial1
P 10.1.7.0/24, 1 successors, FD is 256
         via Connected, Ethernet0
P 10.1.6.0/24, 1 successors, FD is 1024
         via 10.1.3.1 (1024/512), Serial0
         via 10.1.5.2 (1792/768), Serial1
```

（待续）

```
P 10.1.5.0/24, 1 successors, FD is 1024
         via Connected, Serial1
P 10.1.4.0/24, 1 successors, FD is 5632
         via 10.1.3.1 (5632/5120), Serial0
         via 10.1.5.2 (6400/5376), Serial1
Langley#
```

圆括号中的两个度量也都是和每个可行后继路由器相关联的。第一个数字是本地路由器计算得出的路由器 Langley 到达目的网络的度量值。第二个数字是邻居通告的度量值。例如，图 7-5 中路由器 Langley 经过路由器 Wright 到达子网 10.1.6.0 的度量值是 256×（2+1+1）=1024，而邻居路由器 Wright 通告的到达这个目的子网的度量值是 256×（1+1）=512。通过路由器 Chanute 到达上面相同目的子网的这两个度量值分别是 256×（4+1+1+1）=1792 和 256×（1+1+1）=768。

可见，从路由器 Langley 到达子网 10.1.6.0 的最小度量值是 1024，因此，这个度量值就是可行距离（FD）。示例 7-8 中显示了路由器 Langley 的路由表，可以从中看出所选用的后继路由器。

示例 7-8　基于最小的度量距离计算，路由器 Langley 的路由表显示了每个可知的目的网络选用的单台后继路由器

```
Langley#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
     10.0.0.0/8 is subnetted, 7 subnets
C       10.1.3.0 is directly connected, Serial0
D       10.1.2.0 [90/768] via 10.1.3.1, 00:32:06, Serial0
D       10.1.1.0 [90/768] via 10.1.3.1, 00:32:07, Serial0
C       10.1.7.0 is directly connected, Ethernet0
D       10.1.6.0 [90/1024] via 10.1.3.1, 00:32:07, Serial0
C       10.1.5.0 is directly connected, Serial1
D       10.1.4.0 [90/5632] via 10.1.3.1, 00:32:07, Serial0
Langley#
```

对于路由器 Langley 的每一条路由，只有一台后继路由器，而示例 7-9 中路由器 Cayley 的拓扑结构表却显示了到达目的子网 10.1.4.0 的两台后继路由器。这是因为在路由器 Cayley 所计算的度量值中，有两条路由的度量值都匹配它的可行距离。因此这两条路由都存在于示例 7-10 中的路由表中，并且路由器 Cayley 执行等价负载均衡。

示例 7-9　路由器 Cayley 的路由拓扑结构表显示了到达目的子网 10.1.4.0 的两台后继路由器

```
Cayley#show ip eigrp topology
IP-EIGRP Topology Table for process 1
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - Reply status
P 10.1.3.0/24, 1 successors, FD is 768
         via 10.1.1.1 (768/512), Ethernet0
P 10.1.2.0/24, 1 successors, FD is 512
         via 10.1.1.1 (512/256), Ethernet0
P 10.1.1.0/24, 1 successors, FD is 256
```

（待续）

```
         via Connected, Ethernet0
P 10.1.7.0/24, 1 successors, FD is 1024
         via 10.1.1.1 (1024/768), Ethernet0
P 10.1.6.0/24, 1 successors, FD is 256
         via Connected, Serial0
P 10.1.5.0/24, 1 successors, FD is 1536
         via 10.1.1.1 (1536/1280), Ethernet0
P 10.1.4.0/24, 2 successors, FD is 5376
         via 10.1.6.1 (5376/5120), Serial0
         via 10.1.1.1 (5376/5120), Ethernet0
Cayley#
```

示例 7-10　在到达目的子网 10.1.4.0 的两台后继路由器之间将执行等价负载均衡

```
Cayley#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
     10.0.0.0/24 is subnetted, 7 subnets
D       10.1.3.0 [90/768] via 10.1.1.1, 00:01:19, Ethernet0
D       10.1.2.0 [90/512] via 10.1.1.1, 00:01:19, Ethernet0
C       10.1.1.0 is directly connected, Ethernet0
D       10.1.7.0 [90/1024] via 10.1.1.1, 00:01:19, Ethernet0
C       10.1.6.0 is directly connected, Serial0
D       10.1.5.0 [90/1536] via 10.1.1.1, 00:01:19, Ethernet0
D       10.1.4.0 [90/5376] via 10.1.1.1, 00:01:19, Ethernet0
                 [90/5376] via 10.1.6.1, 00:01:19, Serial0
Cayley#
```

示例 7-11 中的路由器 Chanute 的拓扑结构表显示了几条仅仅拥有一台可行后继路由器的路由。例如，到达子网 10.1.6.0 的路由拥有一个距离为 768 的可行距离（FD），因而路由器 Wright（10.1.2.1）是惟一的可行后继路由器。路由器 Langley 有一条到达子网 10.1.6.0 的路由，但是它的度量值是 256×（2+1+1）=1024，大于 FD，因此，路由器 Langley 到达子网 10.1.6.0 的路由不满足可行性条件（FC），因而路由器 Langley 也就没有资格成为一台可行后继路由器。

示例 7-11　从路由器 Chanute 可达的几个子网只有惟一的一台可行后继路由器

```
Chanute#show ip eigrp topology
IP-EIGRP Topology Table for process 1
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - Reply status
P 10.1.3.0/24, 1 successors, FD is 768
         via 10.1.2.1 (768/512), Ethernet0
         via 10.1.5.1 (1536/512), Serial0
P 10.1.2.0/24, 1 successors, FD is 256
         via Connected, Ethernet0
P 10.1.1.0/24, 1 successors, FD is 512
         via 10.1.2.1 (512/256), Ethernet0
P 10.1.7.0/24, 1 successors, FD is 1024
         via 10.1.2.1 (1024/768), Ethernet0
         via 10.1.5.1 (1280/256), Serial0
P 10.1.6.0/24, 1 successors, FD is 768
         via 10.1.2.1 (768/512), Ethernet0
P 10.1.5.0/24, 1 successors, FD is 1024
```

（待续）

```
        via Connected, Serial0
P 10.1.4.0/24, 1 successors, FD is 5376
        via 10.1.2.1 (5376/5120), Ethernet0
Chanute#
```

如果一台可行后继路由器通告的一条路由在本地路由器上所计算的度量比当前后继路由器的度量小，那么这台可行后继路由器就成为后继路由器。下面的情况可能会引起这种情形的发生：

- 发现一条新的路由；
- 一条后继路由器路由的度量值增加后超过了可行后继路由器的度量值；
- 一条可行后继路由器路由的度量值减小后小于后继路由器的度量值。

例如，示例 7-12 中显示了路由器 Lilienthal 到达子网 10.1.3.0 的后继路由器是路由器 Cayley（10.1.6.2）。假设路由器 Lilienthal 和路由器 Wright 之间的链路代价减小到 1，那么由于路由器 Wright（10.1.4.1）正在通告一条到达子网 10.1.3.0 的距离是 512 的路由，同时根据路由器 Lilienthal 和路由器 Wright 之间新的链路代价值，路由器 Lilienthal 计算出经过路由器 Wright 到达目的子网的度量现在变成了 768。因此，路由器 Wright 将替代路由器 Cayley 成为到达子网 10.1.3.0 的后继路由器。

示例 7-12 路由器 Lilienthal 的拓扑结构表

```
Lilienthal#show ip eigrp topology
IP-EIGRP Topology Table for process 1
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - Reply status
P 10.1.3.0/24, 1 successors, FD is 1024
         via 10.1.6.2 (1024/768), Serial0
         via 10.1.4.1 (5632/512), Serial1
P 10.1.2.0/24, 1 successors, FD is 768
         via 10.1.6.2 (768/512), Serial0
         via 10.1.4.1 (5376/256), Serial1
P 10.1.1.0/24, 1 successors, FD is 512
         via 10.1.6.2 (512/256), Serial0
         via 10.1.4.1 (5376/256), Serial1
P 10.1.7.0/24, 1 successors, FD is 1280
         via 10.1.6.2 (1280/1024), Serial0
         via 10.1.4.1 (5888/768), Serial1
P 10.1.6.0/24, 1 successors, FD is 256
         via Connected, Serial0
P 10.1.5.0/24, 1 successors, FD is 1792
         via 10.1.6.2 (1792/1536), Serial0
         via 10.1.4.1 (6400/1280), Serial1
P 10.1.4.0/24, 1 successors, FD is 5120
         via Connected, Serial1
Lilienthal#
```

其次，我们假定路由器 Lilienthal 发现了一个新的邻居，并且这个邻居正在通告一条到达子网 10.1.3.0 距离为 256 的路由。由于这个距离小于当前的可行距离（FD），因而这个新的邻居就成为一台可行后继路由器。进一步假定路由器 Lilienthal 到达这个新邻居的链路代价是 256，那么路由器 Lilienthal 将计算得出经过这个新邻居到达子网 10.1.3.0 的度量值为 512。这个度量值小于经过路由器 Wright 的度量值，因此这台新的邻居路由器将成为到达子网 10.1.3.0 的后继路由器。

由于可行后继路由器减少了扩散计算的数量，并提高了网络的性能，因此可行后继路由器十分重要。可行后继路由器也对降低重新收敛的次数有一定的贡献。如果到达后继路由器

的一条链路失效了，或者链路的代价增加并超过了可行距离（FD），那么这台路由器将首先在它的拓扑结构表中查找可行后继路由器，如果发现存在一台可行后继路由器，那么它就成为后继路由器；这种选择通常发生在次秒级范围内。路由器只有在找不到任何一台可行后继路由器的情况下，才开始进行扩散计算。成功进行 EIGRP 设计的关键在于，确保对所有的目的地来说总是存在一台可行后继路由器。

下面小节将给出一个更正式的规则集合，用来说明路由器是何时以及怎么样查找可行后继路由器的。这个规则集合称为 DUAL 有限状态机。

2. DUAL 有限状态机

当一个 EIGRP 的路由器不执行扩散计算时，每一条路由都处于被动状态（passive state）。参见前面部分出现的所有的拓扑结构表，每一条路由左边的关键字就是用来指出路由的被动状态的。

在产生输入事件（input event）的任何时候，路由器都会重新评估一条路由的可行后继路由器的列表，这将在最后小节中讲述。一个输入事件可以是：

- 直连链路的代价发生变化；
- 直连链路的状态（up 或 down）发生变化；
- 收到一个更新数据包；
- 收到一个查询数据包；
- 收到一个答复数据包。

路由器重新评估的第一步是，在本地路由器上执行一个本地计算（local computation），也就是对于所有的可行后继路由器，重新计算到达目的地的距离。可能的结果有下面几种：

- 如果拥有最低的度量距离的可行后继路由器和已经存在的后继路由器不同，那么可行后继路由器将成为后继路由器；
- 如果新的度量距离小于 FD，那么就更新 FD；
- 如果新的度量距离和已经存在的度量距离不同，那么将向所有的邻居发送更新。

当路由器执行一个本地计算时，路由依然保持被动状态。如果本地路由器发现了一台可行后继路由器，那么将发送一个更新消息给它所有的邻居，但不改变路由的状态。

如果在拓扑结构表中没有发现任何一台可行后继路由器的话，那么路由器将开始进行扩散计算，而且路由器的路由状态改变成活动状态（active state）。在扩散计算完成和路由的状态返回到被动状态之前，路由器不能：

- 改变路由的后继路由器；
- 改变正在通告的路由的距离；
- 改变路由的 FD；
- 开始进行路由的另一个扩散计算。

如图 7-6 所示，路由器是通过向它所有的邻居发送查询来开始一个扩散计算的，查询中包含一个到达目的地的新的本地路由器计算的距离。收到查询后，每一台邻居路由器将执行它自己的本地计算：

- 如果该邻居拥有到达目的地的一台或多台可行后继路由器，它将发送一个答复给原来发送查询的路由器。答复中将包含这台邻居路由器所计算的它到达目网络的最小距离。

- 如果一个邻居没有可行后继路由器，它将把路由的状态改变为活动状态，并且开始进行扩散计算。

对于每一台接收查询的邻居路由器，本地路由器将设置一个答复状态标记（reply status flag(r)）来不断地跟踪所有未处理的查询。当本地路由器收到所有发送到邻居路由器的查询的答复时，扩散计算就完成了。

在一些实例中，路由器没有收到发出的每一个查询的答复。例如，这有可能发生在一个拥有很多低速带宽或质量较差的链路的大型网络当中。在扩散计算的开始，一个活动计时器（active timer）被设置为 3min。[1]如果在活动计时器计时超时后还没有收到希望收到的所有答复，那么这条路由就被宣告“卡”在活动状态（Stuck-In-Active，SIA）。这些没有答复的邻居将从邻居表中删除，并且扩散计算认为这个邻居回应了一个无穷大的度量。

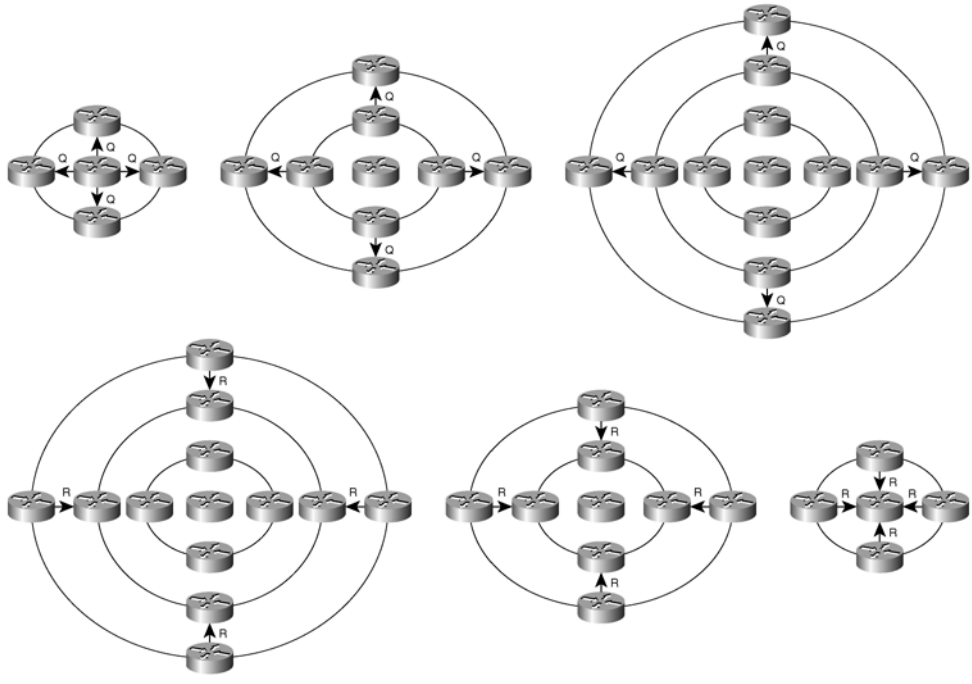


图 7-6 扩散计算在查询被发送时扩大，而在答复被收到时收缩

活动计时器的缺省配置是 3min，这个时长可以通过命令 **timers active-time** 来改变或使其无效。由于查询的丢失而造成邻居的删除显然会带来负面的影响，但是，在一个稳定的、设计良好的网络中，SIA 的情形是从来不会发生的。本章的故障诊断一节中将更详细地讨论 SIA 及 SIA 过程最近的一个增强特性，即使用两个新的 EIGRP 消息：SIA Query 和 SIA Reply 消息。这两个消息可以减少 SIA 故障的机会，也可以在发生故障时，使它们的故障诊断变得容易。

在扩散计算完成时，始发路由器会将 FD 设置成无穷大，这样可以确保任何答复到达目的地时有限距离的邻居路由器都满足 FC，并成为一台可行后继路由器。对于这些答复消息，度量都是由答复消息中所通告的距离加上与发送答复的邻居路由器相连的链路代价计算得出的。选择一台后继路由器是基于最低的度量值的，而且该度量值被设置为 FD。任何一台可

[1] 译者注：在一些早期的 IOS 软件版本中，缺省的活动计时器设置为 1min。

行后继路由器如果不满足该新的可行距离（FD）的可行性条件（FC）的话，就会从拓扑结构表中被删除。注意，在收到所有的答复之前不会选择后继路由器。

因为有多种类型的输入事件（input events）能够引起一条路由改变它的状态，所以当一条路由处于活动状态时就说明可能发生了一些类型的输入事件。DUAL 定义了多种活动状态。查询始发标记（query origin flag（O））用来指出当前的状态。图 7-7 和表 7-2 中显示了完整的 DUAL 有限状态机。

有两个例子可以帮助我们阐明 DUAL 的处理过程。图 7-8 中显示了例子的网络拓扑，这里只需要关注到达子网 10.1.7.0 的每一台路由器的路径；参考图 7-5 中指定的地址。在数据链路上，箭头指出了每一台路由器用来到达子网 10.1.7.0 的后继路由器。在圆括号中显示的分别是每一台路由器到达子网的本地计算距离、路由器的 FD、答复状态标记（reply status flag（r））和查询始发标记（query origin flag（O））。在后面使用这个网络的图示和例子中，活动路由器（active router）都用一个圆圈指出。

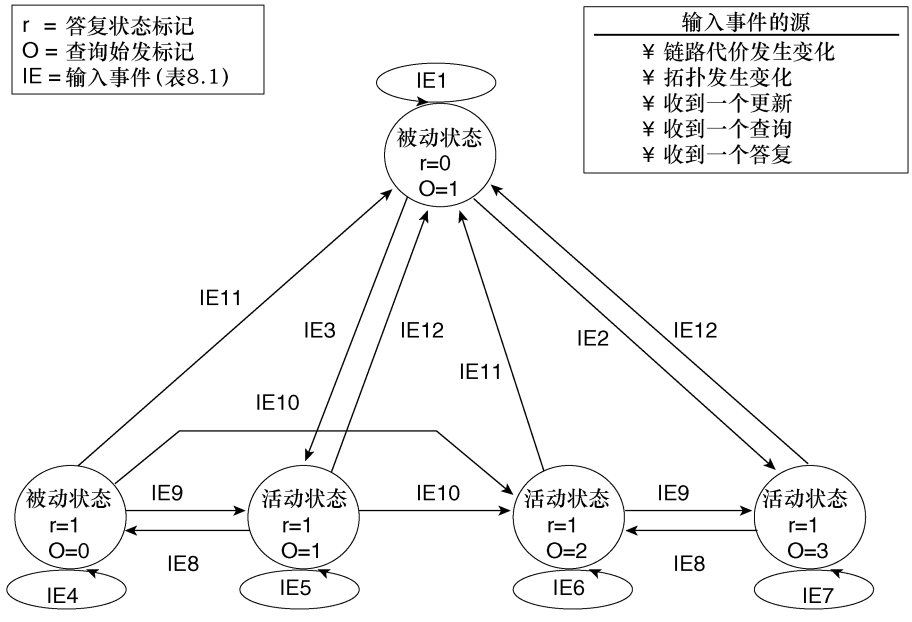


图 7-7　DUAL 有限状态机。查询始发标记（query origin flag（O））标识出扩散计算当前的状态。参看表 7-2 关于每一个输入事件（IE）的解释

表 7-2　　DUAL 有限状态机的输入事件

输入事件	描述
IE1	满足 FC 或者目的地不可到达的任何输入事件
IE2	从后继路由器收到了查询消息；不满足 FC
IE3	除了来自后继路由器查询的其他输入事件；不满足 FC
IE4	除了最新的答复或来自后继路由器的查询的输入事件
IE5	除了最新的答复、来自后继路由器的查询或到达目的地距离的增加的输入事件
IE6	除了最新答复的输入事件
IE7	除了最新答复或到达目的地距离的增加的输入事件
IE8	到达目的地距离的增加

续表

输入事件	描述
IE9	收到了最新的答复；当前 FD 不满足 FC
IE10	从后继路由器收到了查询
IE11	收到了最新的答复；FC 和当前 FD 匹配
IE12	收到了最新的答复；设置 FD 为无限大

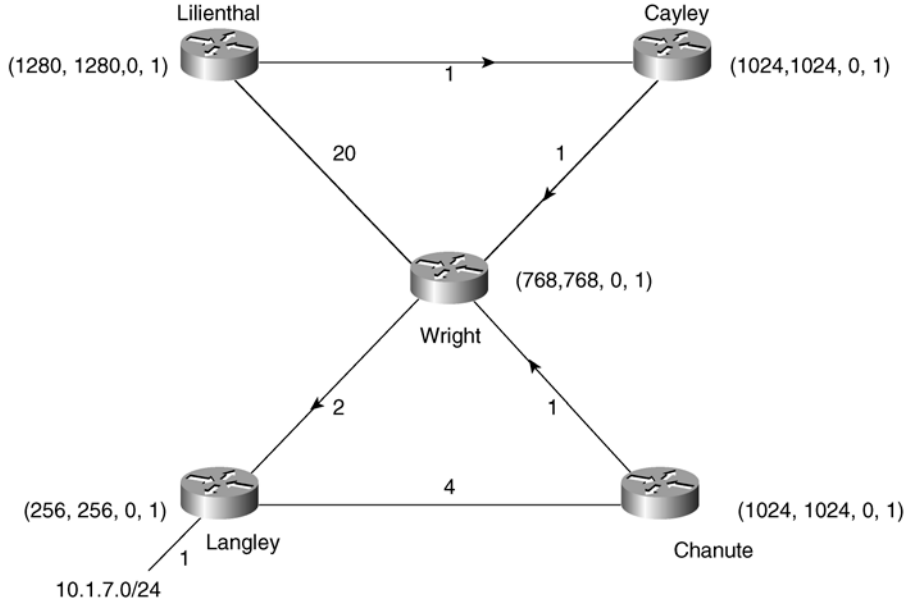


图 7-8 到达子网 10.1.7.0 的所有路由都是被动状态的，通过 r=0 和 O=1 指示

3. 扩散计算：范例 1

这个例子关注的是路由器 Cayley 和它到达子网 10.1.7.0 的路由。在图 7-9 中，路由器 Cayley 和 Wright（10.1.1.1）之间的链路是失效的。EIGRP 把失效的链路当作一条拥有无穷大距离的链路。[1]路由器 Cayley 检查它的拓扑结构表，来查找到达子网 10.1.7.0 的可行后继路由器，但是没有找到，参见示例 7-9 所示。

如图 7-10 所示，路由器 Cayley 的路由变成了活动状态。这条路由的距离和 FD 也变为不可到达的了，并且路由器 Cayley 把一个包含新距离的查询发送给它的邻居路由器 Lilienthal。路由器 Cayley 关于路由器 Lilienthal 的答复状态标记被设置为 1，用来指出期待一个答复，因此输入事件是一个查询的未接受状态（IE3），O=1。

一旦收到了查询消息，路由器 Lilienthal 将执行一个本地计算，如图 7-11 所示。参见示例 7-12，由于路由器 Lilienthal 拥有到达子网 10.1.7.0 的可行后继路由器，因而它的路由将不会变为活动状态。路由器 Wright 成为新的后继路由器，而且路由器 Lilienthal 将发送一个含有它经过路由器 Wright 到达子网 10.1.7.0 的距离的答复消息。因为到达子网 10.1.7.0 的距离已经增加了，而且路由不处在活动状态，因此路由器 Lilienthal 的 FD 没有变化。

[1] 无穷大距离可以用 0xFFFFFFFF 或 4294967295 来表示。

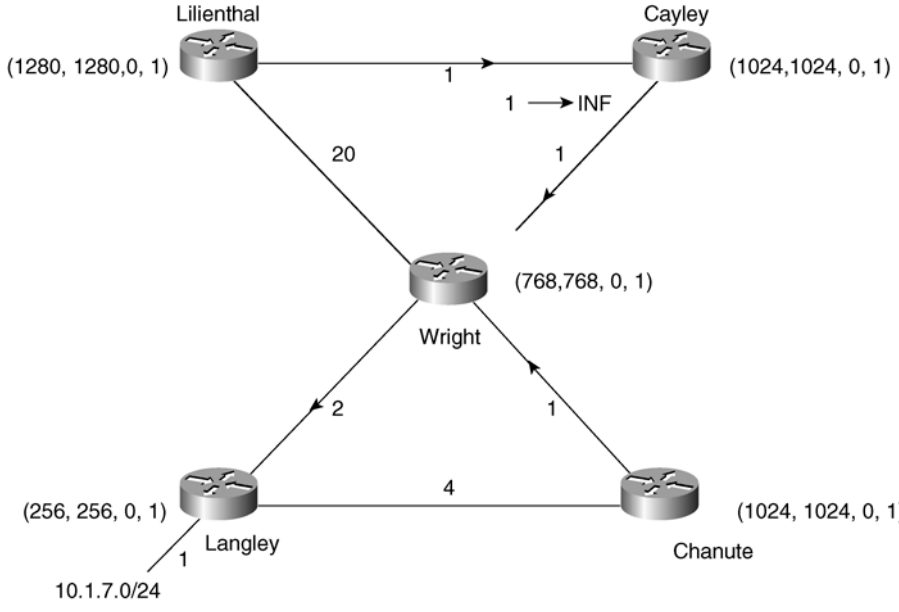


图 7-9　路由器 Cayley 和 Wright 间的链路是失效的，因而路由器 Cayley 没有一台到达子网 10.1.7.0 的可行后继路由器

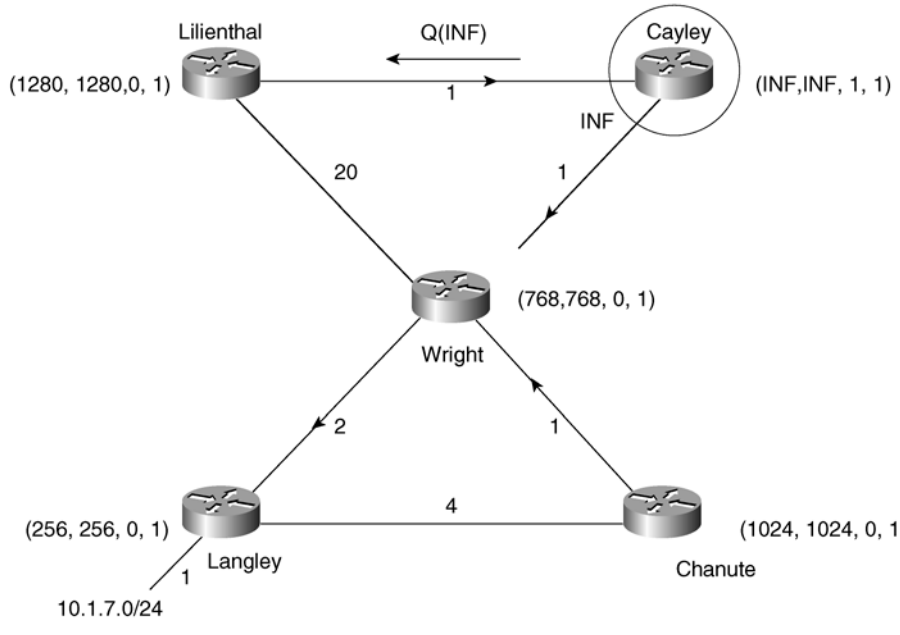


图 7-10　路由器 Cayley 到达子网 10.1.7.0 的路由变成活动状态，并且为了确定可行后继路由器而去查询路由器 Lilienthal

一旦从路由器 Lilienthal 收到一个答复消息，路由器 Cayley 就设置 r=0，路由也就变成了被动状态，如图 7-12 所示。路由器 Lilienthal 成为路由器 Cayley 的新后继路由器，FD 也将设置成新的距离。最后，路由器 Cayley 发送给路由器 Lilienthal 一个包含其本地计算度量的更新。路由器 Lilienthal 也发送新的更新来通告它的新度量值。

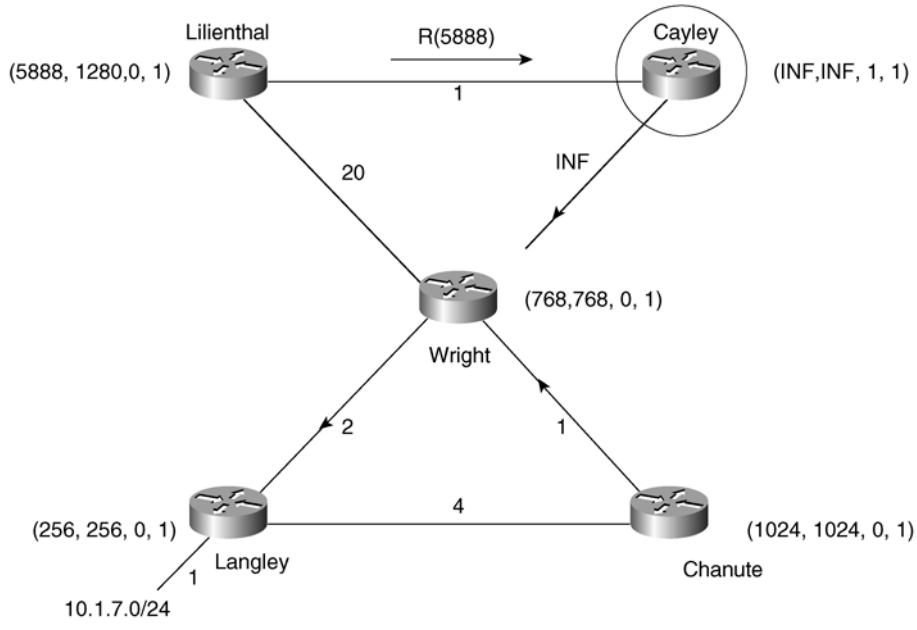


图 7-11 路由器 Lilienthal 有一个到达子网 10.1.7.0 的可行后继路由器。它执行一个本地计算，向路由器 Cayley 发送一个包含其经过路由器 Wright 到达目的子网的应答数据包，而且发送一个更新给路由器 Wright

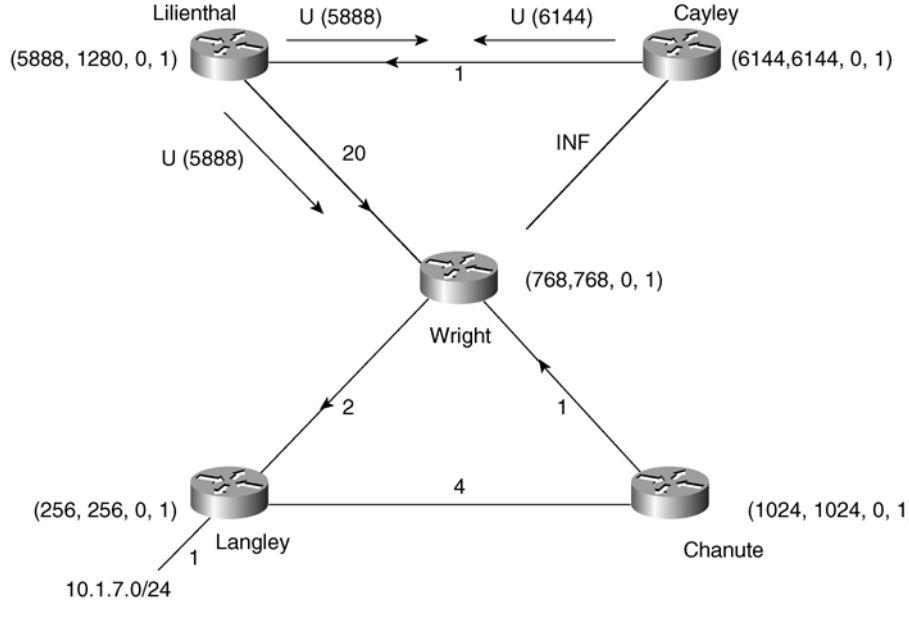


图 7-12 路由器 Cayley 到达子网 10.1.7.0 的路由变为被动状态，而且发送一个更新给路由器 Lilienthal

EIGRP 协议包的行为可以通过调试命令 **debug eigrp packets** 来观察。缺省情况下，路由器将会显示所有的 EIGRP 数据包。由于大量 Hello 和 ACK 调试信息的输出可能很难去跟踪，因此该命令允许使用关键字选项以便只显示指定的数据包类型。在示例 7-13 中，命令 **debug eigrp packets query reply update** 用来在路由器 Cayley 上观察本例中描述的事件的数据包行为。

示例 7-13　在本例中描述的 EIGRP 数据包事件可以通过这些调试信息进行观察

```
Cayley#debug eigrp packet update query reply
EIGRP Packets debugging is on
    (UPDATE, QUERY, REPLY)
B#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0, changed state to down
EIGRP: Enqueueing QUERY on Serial0 iidbQ un/rely 0/1 serno 45-49
EIGRP: Enqueueing QUERY on Serial0 nbr 10.1.6.1 iidbQ un/rely 0/0 peerQ un/rely
0/0 serno 45-49
EIGRP: Sending QUERY on Serial0 nbr 10.1.6.1
  AS 1, Flags 0x0, Seq 45/64 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
45-49
EIGRP: Received REPLY on Serial0 nbr 10.1.6.1
  AS 1, Flags 0x0, Seq 65/45 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Enqueueing UPDATE on Serial0 iidbQ un/rely 0/1 serno 50-54
EIGRP: Enqueueing UPDATE on Serial0 nbr 10.1.6.1 iidbQ un/rely 0/0 peerQ un/rely
 0/0 serno 50-54
EIGRP: Sending UPDATE on Serial0 nbr 10.1.6.1
  AS 1, Flags 0x0, Seq 46/66 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
50-54
EIGRP: Received UPDATE on Serial0 nbr 10.1.6.1
  AS 1, Flags 0x0, Seq 67/46 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
```

- **标记（Flags）**——在输出的调试信息中，指出 EIGRP 包头中标记的状态；有关 EIGRP 包头的内容请参阅 7.3.5 小节。0x0 表示没有标记被设置。0x1 表示设置了初始化（initialization）位，在一个新的邻居关系中，当附加的路由条目是首个时，这个标记就被设置。0x2 表示设置了条件接收位（conditional receive bit），这个标记用在私有的可靠组播算法中。
- **序列号（Seq）**——表示一个数据包序列号/确认序列号。
- **idbq**——表示在接口上的输入队列数据包数/输出队列数据包数。
- **iidbq**——表示在接口上等待传送的不可靠组播数据包数/等待传送的可靠组播数据包数。
- **peerQ**——表示在接口上等待传送的不可靠单播数据包数/等待传送的可靠单播数据包数。
- **serno**——表示一个指向某条路由的双重连接的序列号的指针。这个指示器使用内部（或私有）机制，用来在一个快速变化的拓扑中跟踪正确的路由信息。

4. 扩散计算：范例 2

这个例子所关注的是路由器 Wright 和它到达子网 10.1.7.0 的路由。虽然在这里描述的输入事件（在扩散更新计算的期间内链路的延迟变化了两次）在现实的网络中未必会发生，但是这个例子显示了 DUAL 算法是怎样控制多种度量变化的。

在图 7-13 中，把路由器 Wright 和 Langley 之间的链路代价由 2 变成 10。经过路由器 Langley 到达 10.1.7.0 的距离现在超过了路由器 Wright 的可行距离，这将引起路由器 Wright 开始进行本地度量计算。这时度量将被更新，除了和链路代价发生改变的链路相连的邻居路由器 Langley 外，路由器 Wright 会向它所有的邻居发送更新消息，如图 7-14 所示。

请注意，路由器 Langley 是到达子网 10.1.7.0 的惟一可行后继路由器，这是因为路由器 Chanute 的本地计算度量高于路由器 Wright 的 FD（1024>768）。路由器 Wright 和 Langley 之间链路度量的增加引起路由器 Wright 在它的拓扑结构表中去查找一台新的后继路由器。由于路由器 Wright 在它的拓扑结构表中可以查到的惟一可行后继路由器是路由器 Langley，因此，路由器 Wright 到达子网 10.1.7.0 的路由将变为活动状态。查询消息将被发送到邻居路由器，如图 7-15 所示。

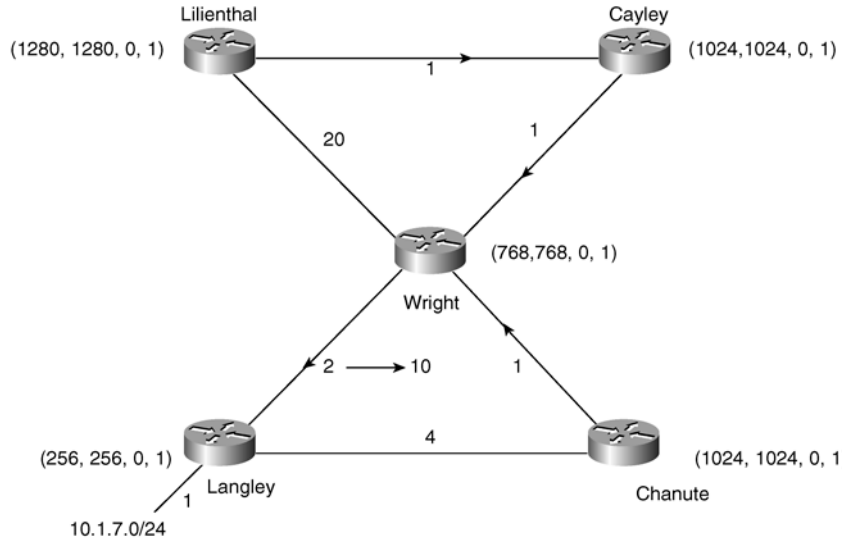


图 7-13 路由器 Cayley 到达子网 10.1.7.0 的路由变为被动状态，并且向路由器 Lilienthal 发送一个更新消息

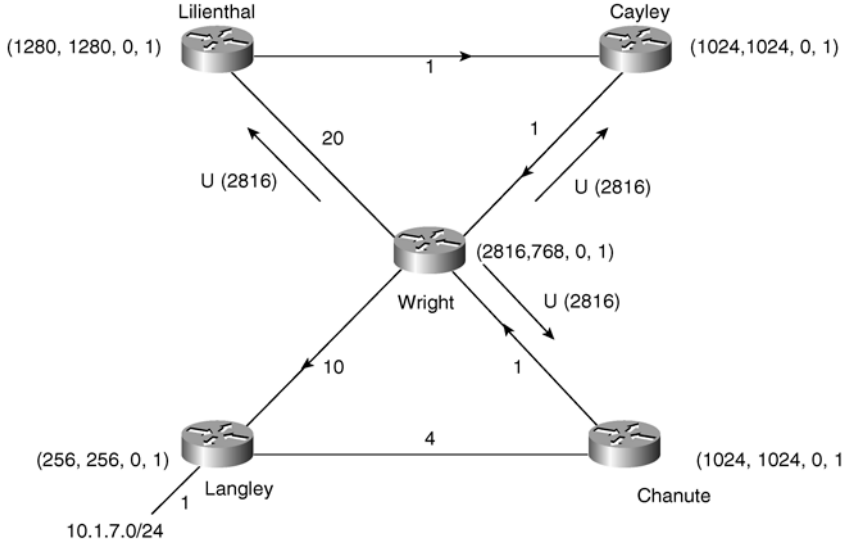


图 7-14 路由器 Wright 发送含有新度量值的更新消息到除了路由器 Langley 之外的所有邻居路由器

同时，在图 7-14 中，路由器 Wright 发出的更新消息将引起路由器 Cayley、Lilienthal 和 Chanute 分别执行一个本地计算。

在路由器 Cayley 上，现在经过路由器 Wright 的路由距离超过了路由器 Cayley 的可行距离（FD）（2816>1024）。因而，路由变为活动状态并向它的邻居路由器发送查询消息。

在图 7-15 中，路由器 Lilienthal 正在使用路由器 Cayley 作为一台后继路由器，但是还没有收到从路由器 Cayley 发出的查询消息。因此，路由器 Lilienthal 只不过重新计算了经过路由器 Wright 的路径的度量值，发现它不再满足可行性条件，从而从拓扑结构表中删除了这条路径。

在路由器 Chanute 看来，路由器 Wright 是它的后继路由器。因为路由器 Wright 所通告的距离不再满足路由器 Chanute 的可行性条件（FC）（2816>1024），并且因为路由器 Chanute

还有一台可行后继路由器（参见示例 7-11），因此，路由器 Chanute 将会把路由器 Wright 从它的拓扑结构表中删除。随后，路由器 Langley 变成了路由器 Chanute 的后继路由器，度量值被更新后，路由器 Chanute 向它的邻居路由器发送更新消息（参见图 7-15）。而路由器 Chaute 上的路由则从来没有变为活动状态。

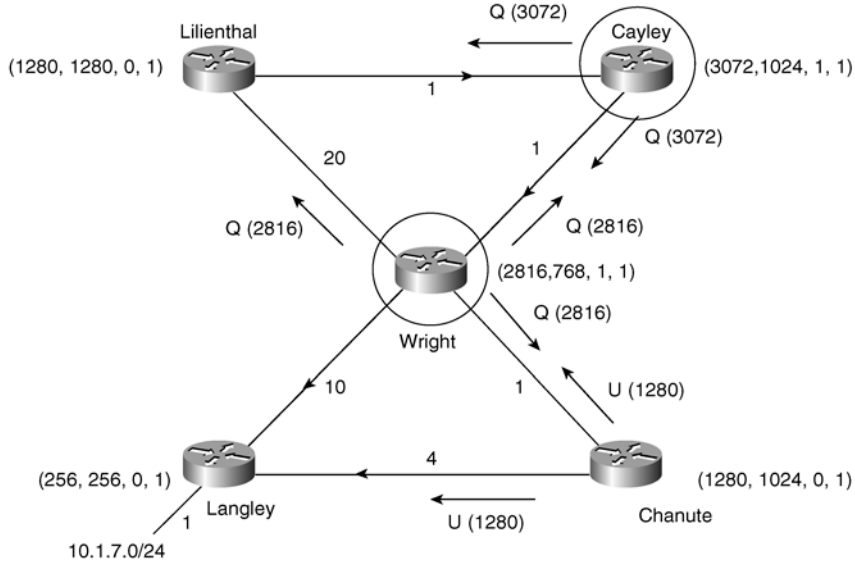


图 7-15　路由器 Wright 到达 10.1.7.0 的路由变为活动状态，Wright 向它的邻居发出查询，以便选择一个可行后继路由器。为了响应来自路由器 Wright 的早期更新，路由器 Cayley 使它的路由成为活动状态并向它的邻居路由器发出查询；同样，路由器 Chanute 也改变了它的度量值并发送出更新

路由器 Cayley、Lilienthal 和 Chanute 分别以不同的答复来响应发自于路由器 Wright 的查询，如图 7-16 所示。

路由器 Cayley 已经是活动状态，因为输入事件是来自于后继路由器的查询，这个查询最初标记为 2（O=2），参见图 7-7 和表 7-2。

路由器 Lilienthal 一旦收到路由器 Wright 的查询，就发送一个含有经过路由器 Cayley 的距离的应答。然而，就在刚发出这个应答消息后，路由器 Lilienthal 收到了来自路由器 Cayley 的查询，这时，路由器 Lilienthal 的可行距离超出了，因而度量值将被更新；路由器 Lilienthal 使路由变为活动状态，并向它的邻居路由器发出查询。

路由器 Chanute 已经把它的后继路由器切换到路由器 Langley，并且只发出一个应答消息。

当上述的一切正在继续进行的时候，图 7-16 中显示了路由器 Wright 和 Langley 之间的链路代价又从 10 增加到了 20。路由器 Wright 将又重新计算基于这个新的链路代价到达子网 10.1.7.0 的路由度量值，但是由于该路由此时是处于活动状态的，因而，在这条路由变成被动状态之前，它所通告的距离和可行距离（FD）都不能改变。

根据图 7-7 和表 7-2，在路由处于活动状态的时候，到达目的地的距离如果增加了将会使 O=0，如图 7-17 所示。路由器 Wright 将响应来自于路由器 Lilienthal 的查询。路由器 Wright 所报告的距离还是它到达子网 10.1.7.0 的路由起初变为活动状态时的距离（记住，当路由处于活动状态时，路由器所通告的距离不能改变）。路由器 Cayley 也发送一个应答来响应路由

器 Lilienthal 的查询。

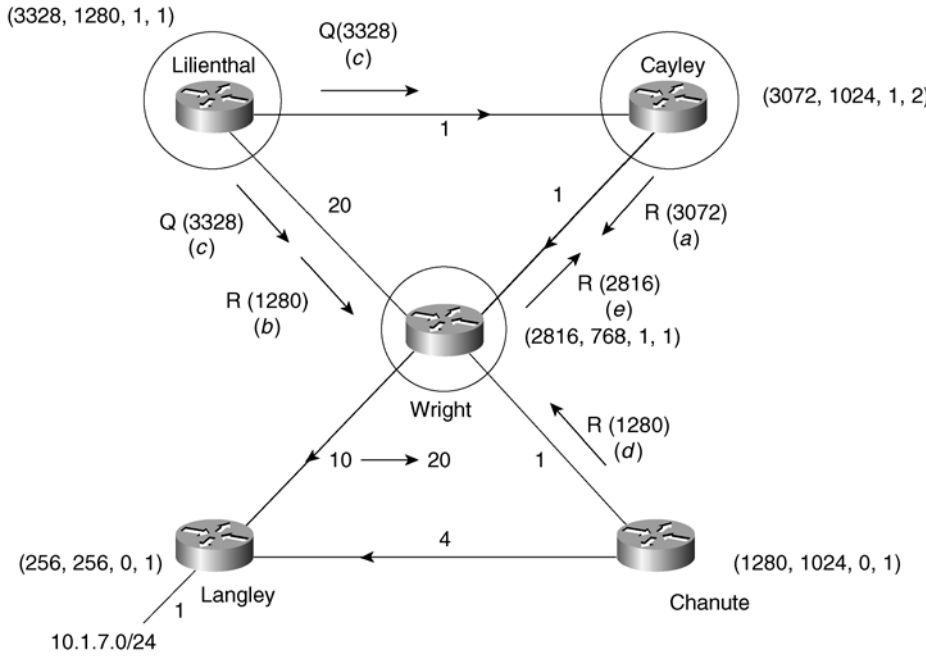


图 7-16 路由器 Cayley（a）答复来自路由器 Wright 的查询。路由器 Lilienthal（b）答复来自路由器 Wright 的查询，并且（c）使路由变成活动状态，同时发送查询来响应路由器 Cayley 的查询。路由器 Chanute（d）回复路由器 Wright 的查询。路由器 Wright（e）回复路由器 Cayley 的查询

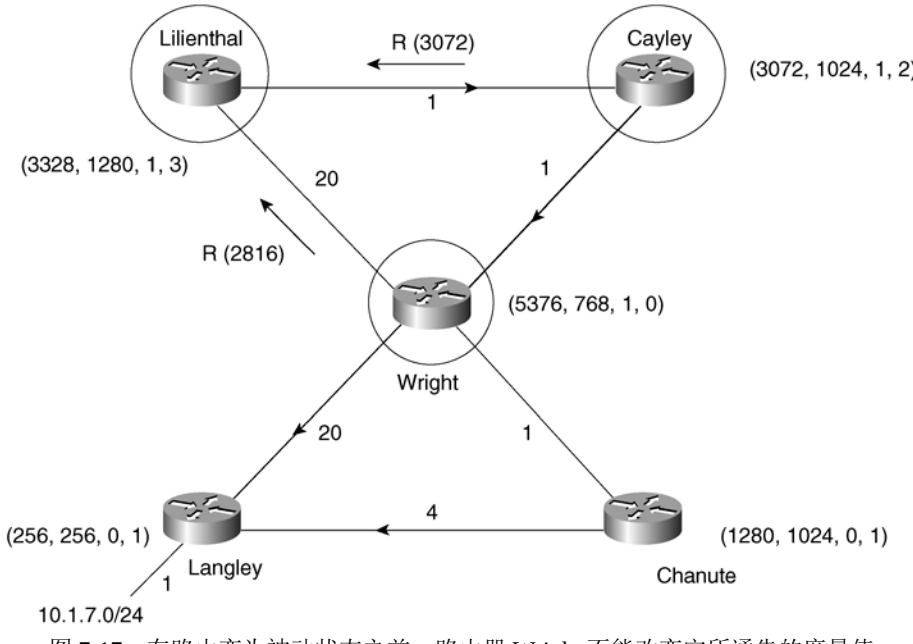


图 7-17 在路由变为被动状态之前，路由器 Wright 不能改变它所通告的度量值

路由器 Lilienthal 在收到了它发送的所有查询的答复后，将把路由的状态转变成被动状态，如图 7-18 所示。这时，路由就可以设置新的可行距离（FD）了。由于路由器 Cayley 所通告的路由距离低于路由器 Lilienthal 的可行距离，因而它仍然保持是后继路由器。路由器

Lilienthal 也发送一个答复来响应路由器 Cayley 的查询。

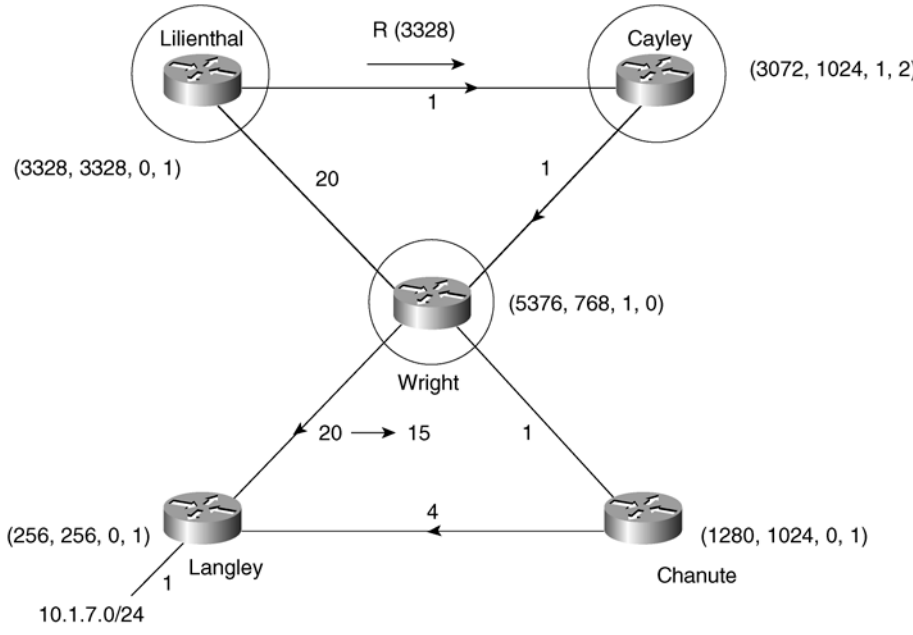


图 7-18　收到了所希望得到的最后一个答复后，路由器 Lilienthal 将使它的路由转变为被动状态（r=0，O=1）

图 7-18 中显示了路由器 Wright 和 Langley 之间的链路距离再次从 20 改变成 15。路由器 Wright 也再次计算出它的路由的本地距离为 4096，如图 7-19 所示。如果在路由变为被动状态之前，路由器 Wright 收到一个查询，那么它将仍然通告含有距离为 2816 的路由，2816 是在路由变为活动状态时的距离值。

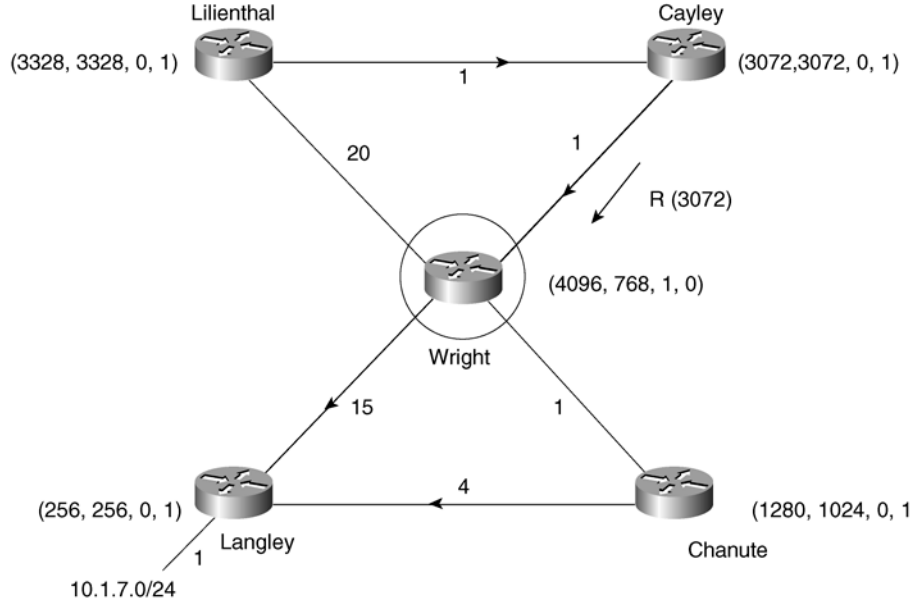


图 7-19　收到了所希望得到的最后一个答复后，路由器 Cayley 将使它的路由状态转变为被动状态

当路由器 Cayley 收到它所发送的查询的答复时，它到达子网 10.1.7.0 的路由也将变成被动状态，如图 7-19 所示，将设置一个新的可行距离（FD）。虽然路由器 Wright 在本地计算的度量值是 4096，但是它所通告的最新度量值却是 2816。因此，路由器 Wright 满足路由器 Cayley 的可行性条件（FC），从而变为到达子网 10.1.7.0 的后继路由器，并发送一个答复给路由器 Wright。

在图 7-20 中，路由器 Wright 收到了它所发出的每一个查询的答复后，它的路由状态就变为被动状态了。路由器 Wright 选择了路由器 Chanute 作为它的新后继路由器，并且把可行距离（FD）改变为路由器 Chanute 所通告的距离与它和邻居路由器（Chanute）之间的链路代价的总和。路由器 Wright 发送一个更新给它所有的邻居路由器，并通告它在本地所计算的新度量值。

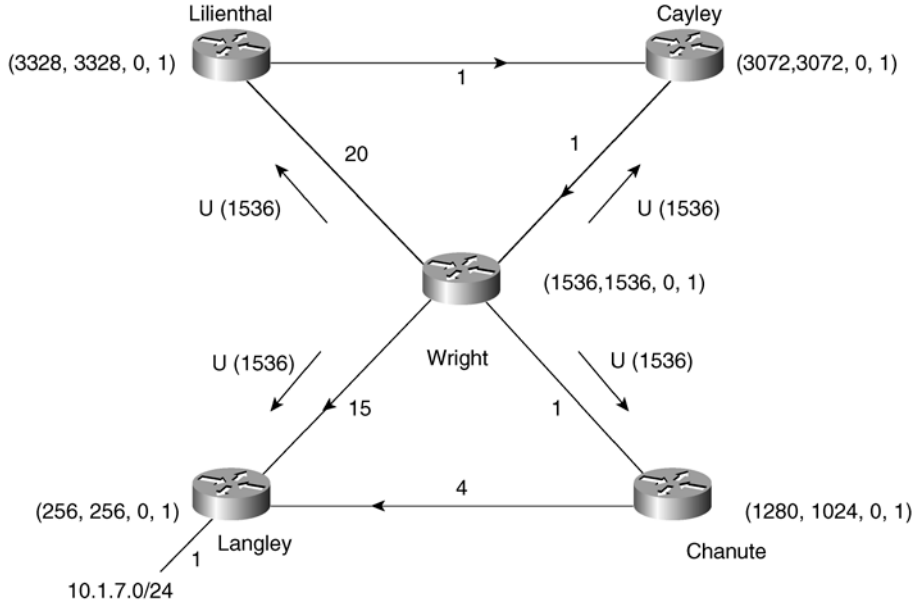


图 7-20 路由器 Wright 转变到被动状态，选择路由器 Chanute 作为它的后继路由器，同时改变 FD，并且更新所有的邻居路由器

路由器 Cayley 使用路由器 Wright 作为它的后继路由器。当它收到一个来自路由器 Wright 的并有较低代价的更新时，它将改变它在本地的计算度量和 FD，并且更新它的邻居路由器，如图 7-21 所示。

来自路由器 Cayley 的更新并不影响路由器 Wright，这是因为路由器 Cayley 不满足路由器 Wright 的 FC。在路由器 Lilienthal 上的更新会引起一个本地计算。

如图 7-22 所示，路由器 Lilienthal 减小了它的度量值及 FD，并更新它的邻居。

虽然，扩散计算算法可以通过更加详细的描述或阅读一些读物以便完全的理解，但是在这里所讲述的内容和前面的例子也包含了扩散计算算法的主要核心内容：

- 任何时间，发生一个输入事件，就会执行一个本地计算。
- 如果在路由器的拓扑结构表中发现了一台或多台可行后继路由器，那么将使用具有最低度量代价的可行后继路由器作为它的后继路由器。

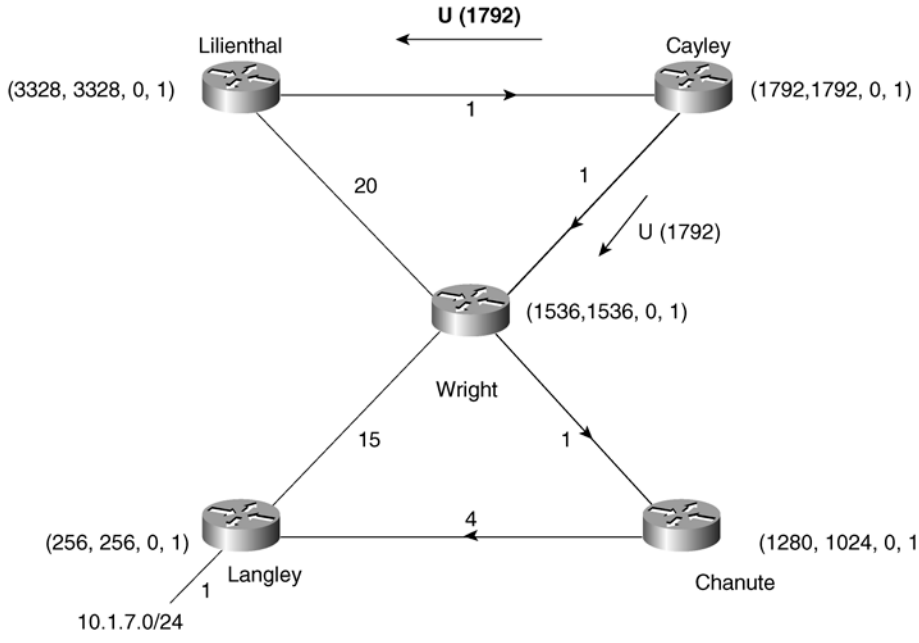


图 7-21　路由器 Cayley 重新计算它的度量值，根据路由器 Wright 通告的较低的代价更改它的 FD，并更新它的邻居路由器

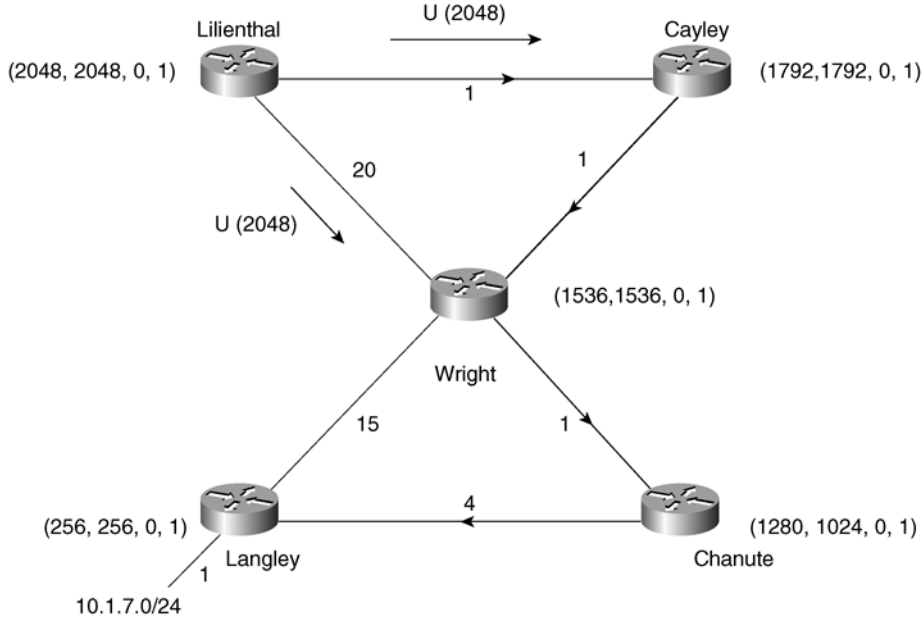


图 7-22　路由器 Lilienthal 重新计算它的度量，基于来自路由器 Cayley 的更新改变它的 FD，并更新它的邻居路由器

- 如果没有发现可行后继路由器，那么将使它的路由变成活动状态，向它的邻居路由器发送查询消息，以便确定一个可行后继路由器。
- 在所有的查询被答复响应之前，或者活动计时器计时超时之前，将保持路由状态为活动状态。
- 如果扩散计算的结果无法发现一个可行后继路由器，那么将宣告这个目的地不可到达。

7.3.5 EIGRP 的数据包格式

EIGRP 协议数据包的 IP 头部指定它的协议号是 88，最大长度可以是传输该数据包接口的 IP 最大传输单元（MTU）的大小——通常是 1500 字节。紧接着 IP 头部后面的是 EIGRP 协议头部，EIGRP 协议头部后面是类型/长度/数值（Type/Length/Value，TLV）这 3 个参数的不同组合。这些 TLV 不仅携带路由条目的信息，而且提供多个字段来管理 DUAL 算法的处理、组播的先后次序和 IOS 软件版本。

1. EIGRP 包头

图 7-23 中显示了 EIGRP 的头部，它是每个 EIGRP 数据包的开始部分。

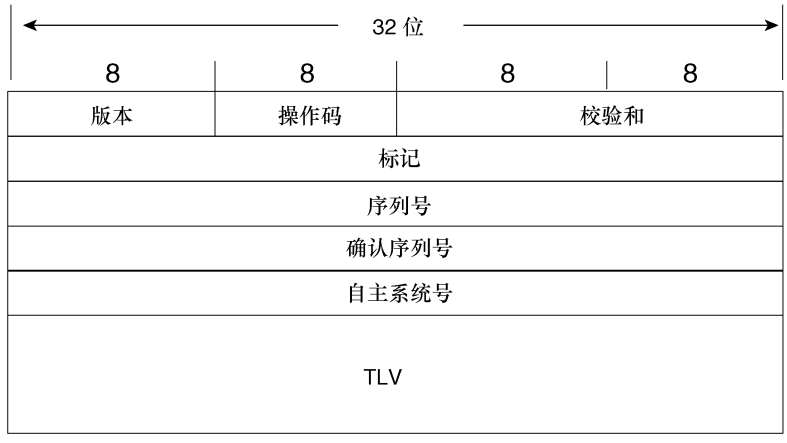


图 7-23 EIGRP 包头的格式

- **版本号（Version）**——指出发起 EIGRP 进程的具体版本。EIGRP 协议本身的版本自发布后还没有改变过。
- **操作码（Opcode）**——指出 EIGRP 数据包的类型，这显示在表 7-3 中。虽然表中包含了 IPX SAP 数据包类型，但 IPX EIGRP 的讨论已经超出了本书的范围。

表 7-3 EIGRP 的数据包类型

操作码（Opcode）	类型（Type）
1	更新（Update）
3	查询（Query）
4	答复（Reply）
5	问候（Hello）
6	IPX SAP
10	SIA 查询
11	SIA 答复

- **校验和（Checksum）**——标准的 IP 校验和。它是基于除了 IP 头部的整个 EIGRP 数据包来计算的。
- **标记（Flags）**——目前包括两个标记。大部分的位设置为是 Init 位，也就是设置为 0x00000001，指出附加的路由条目是新的邻居关系的开始。第二位设置为

0x00000002，表示条件接收位（conditional receive bit），并使用在一个私有的可靠组播算法中。

- **序列号（Sequence）**——是一个用在 RTP 中的 32 位序列号。
- **确认序列号（ACK）**——是本地路由器从邻居路由器那里收到的最新的一个 32 位序列号。一个包含有非零 ACK 字段的 Hello 数据包将被看作是一个 ACK 数据包，而不看作一个 Hello 数据包。注意，如果数据包本身是单播的，这里的 ACK 字段只能是非零的，因为确认数据包从来都不是组播的。
- **自主系统号（Autonomous System Number）**——指定一个 EIGRP 协议域的标识号。

跟在 EIGRP 头部后面的就是 TLV 字段，表 7-4 中列出了多种类型的 TLV 字段。虽然在本书中不讲述 IPX 协议和 AppleTalk 协议类型，但在下面的表中也包含了进来。每一个 TLV 字段都包含一个表 7-4 中列出的 2 个八位组字节的类型号、一个指定 TLV 字段长度的 2 个八位组字节的字段和一个由类型决定其格式的可变字段。

表 7-4 类型/长度/数值（TLV）的类型

数值	TLV 类型
一般的 **TLV** 类型	
0x0001	EIGRP 参数
0x0003	序列（Sequence）
0x0004	软件版本*
0x0005	下一个组播序列
IP 特有的 **TLV** 类型	
0x0102	IP 内部路由
0x0103	IP 外部路由
AppleTalk 特有的 **TLV** 类型	
0x0202	AppleTalk 内部路由
0x0203	AppleTalk 外部路由
0x0204	Apple Talk 电缆配置
IPX 特有的 **TLV** 类型	
0x0302	IPX 内部路由
0x0303	IPX 外部路由

*这个数据包指出是软件的老版本在运行（软件版本为 0）还是软件从 IOS10.3(11)、11.0(8)和 11.1(3)起的新版本（软件版本为 1）在运行。

2. 一般的 TLV 字段

这些 TLV 字段可以携带 EIGRP 的管理信息而不需要指定任何一个可路由的协议。带参数的 TLV 用来传递度量权重和抑制时间，如图 7-24 所示。序列、软件版本和下一个组播序列等 TLV 是用于 Cisco 的私有可靠性组播算法的，该内容已超出了本书的讲述范围。

3. IP 特有的 TLV 字段

每一个内部路由和外部路由的 TLV 都包含一个路由条目。每个更新、查询和答复数据包都至少包含一个路由 TLV。

内部路由和外部路由的 TLV 包括了路由的度量信息。就像早先提到的，EIGRP 协议使用的度量与 IGRP 协议使用的度量相同，只是扩大了 256 倍。

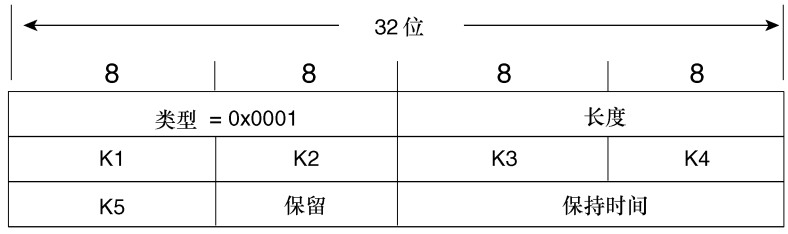
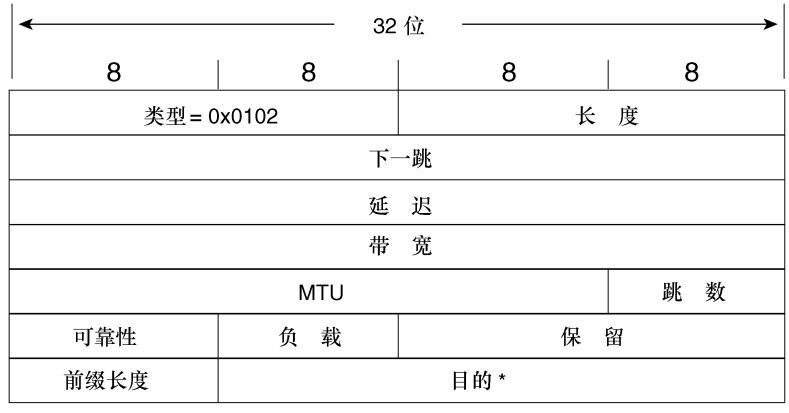


图 7-24 带 EIGRP 参数的 TLV

（1）IP 内部路由的 TLV

内部路由是指在 EIGRP 自主系统内部可以到达目的地的路径。内部路由的 TLV 格式如图 7-25 所示。



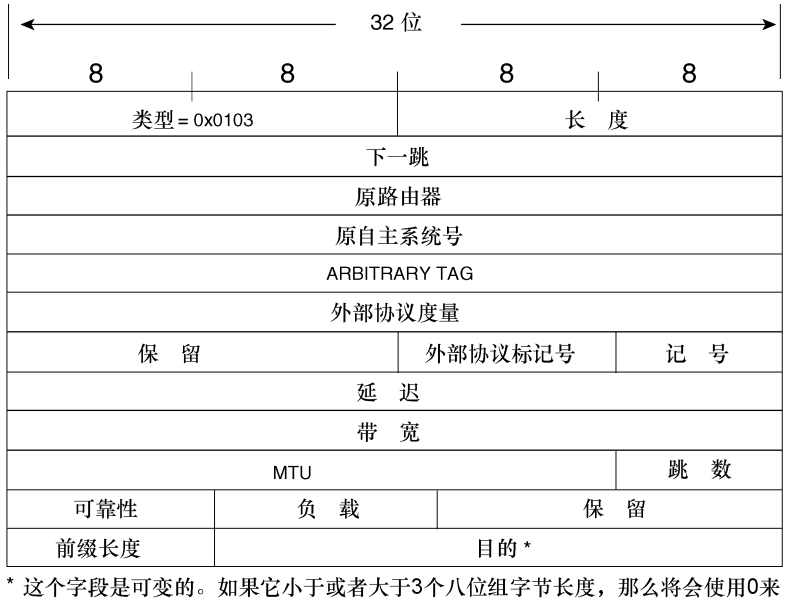
* 这个字段是可变的。如果它小于或者大于3个八位组字节长度，那么将会使用0来填充TLV，以使它达到下一个4个八位组字节的边界。例如，假设目的地址是10.1，那么目的字段将是2个八位组字节和一个0x00 的填充项。如果目的地址是 192.168.16.64，那么目的字段将是4个八位组字节和一个0x000000的填充。

图 7-25 IP 内部路由 TLV

- **下一跳（Next Hop）**——是指下一跳 IP 地址。这个地址可能是、也可能不是始发路由器的地址。
- **延迟（Delay）**——是指所配置的以 10μs 为单位表示的延迟总和。注意，不像 IGRP 数据包中 24 位的字段，这个字段是 32 位的。这个更大的字段可以容纳 EIGRP 使用的大 256 倍的延迟。一个 0xFFFFFFFF 的延迟标识一个不可到达的路由。
- **带宽（Bandwidth）**——就是 $256\times BW_{IGRP(min)}$，或者用 2 560 000 000 除以沿着路由方向的所有接口所配置的最小带宽。像延迟一样，这个字段也比 IGRP 的带宽字段多 8 位。
- **MTU**——是指沿着到达目的地的路由上所有链路中最小的最大传输单元。虽然 EIGRP 数据包中包含了这个参数，但是它从来没有在度量值的计算中使用过。
- **跳数（Hop Count）**——是一个在 0x01～0xFF 之间的数字，表示到达目的地的路由的跳数。路由器将通告与之直连网络的跳数为 0 跳；后续的路由器将记录并通告相对于下一跳路由器的路由。
- **可靠性（Reliability）**——是一个在 0x01～0xFF 之间的数字，用来反映沿着到达目的地的路由上接口的出站误码率的总和，每 5min 通过一个指数的加权平均来计算。

0xFF 表示 100%的可靠链路。

- **负载（Load）**——是一个在 0x01～0xFF 之间的数字，用来反映沿着到达目的地的路由上接口的出站负载的总和，每 5min 通过一个指数的加权平均来计算。0x01 表示一条最小负载的链路。
- **保留字段（Reserved）**——一个未使用的字段并且总是设置为 0x0000。
- **前缀长度（Prefix Length）**——指出一个地址掩码中的网络位的个数。
- **目的地址（Destination）**——表示一个路由的目的地址。虽然在图 7-25 和图 7-26 中显示的字段只是一个 3 个八位组字节长的字段，但是这个字段针对不同的地址是可变的。例如，假如有一条到达目的地址 10.1.0.0/16 的路由，它的前缀长度是 16，因而目的地址只需要一个包含 10.1 的 2 个八位组字节的字段。假如一条到达目的地址 192.168.17.64/27 的路由，它的前缀长度是 27，因而目的地址将需要一个包含 192.168.17.64 的 4 个八位组字节的字段。如果这个字段没有 3 个八位组字节的长度，那么 TLV 将增加 0 来填充这个字段，以便使这个字段达到 4 个八位组字节的边界长。



* 这个字段是可变的。如果它小于或者大于3个八位组字节长度，那么将会使用0来填充TLV，以使它达到下一个4个八位组字节的边界。例如，假设目的地址是10.1，那么目的字段将是2个八位组字节和一个0x00的填充项。如果目的地址是192.168.16.64，那么目的字段将是4个八位组字节和一个0x000000的填充。

图 7-26　IP 外部路由的 TLV

（2）IP 外部路由的 TLV

外部路由是指到达 EIGRP 自主系统外部的目的地址的一条路径，或者是一条通过路由重新分配注入到 EIGRP 域内的路由。图 7-26 显示了外部路由 TLV 字段的格式。

- **下一跳（Next Hop）**——就是路由的下一跳 IP 地址。在一个多路访问的网络中，正在通告路由的路由器可能不是到达目的地的最佳下一跳路由器。例如，一个在以太网链路上宣告 EIGRP 路由的路由器也可能宣告 BGP 的路由，同时也可能把从 BGP 学到的路由通告到 EIGRP 的自主系统。因为以太链路上的其他路由器并不宣告 BGP 路由，因此，它们也无法得知 BGP 宣告者的接口是一个最佳的下一跳地址。下一跳

字段允许同时宣告两种路由协议的路由器告诉它的 EIGRP 邻居——“使用地址 A.B.C.D 代替我的接口地址作为它的下一跳”。

- **原路由器（Originating Router）**——是一个 IP 地址，或者重分配外部路由到 EIGRP 自主系统的路由器 ID。
- **原自主系统号（Originating Autonomous System Number）**——是指始发路由的路由器所在的自主系统号。
- **Arbitrary Tag**——可以用来携带一组路由映射的标记。如要了解路由映射的用法，请参见第 14 章的内容。
- **外部协议度量（External Protocol Metric）**——顾名思义，这是一个外部协议的度量。在和 IGRP 协议之间进行重分配时，这个字段用来跟踪 IGRP 协议的度量值。
- **保留字段（Reserved）**——一个未使用的字段并且总是设置为 0x0000。
- **外部协议 ID（External Protocol ID）**——用来标识外部路由是从哪一个协议学习到的。表 7-5 列出了这个字段的可能值。

表 7-5 外部协议 ID 字段的数值

代码	外部协议
0x01	IGRP
0x02	EIGRP
0x03	静态路由
0x04	RIP
0x05	Hello
0x06	OSPF
0x07	IS-IS
0x08	EGP
0x09	BGP
0x0A	IDRP
0x0B	直连链路

- **标记（Flags）**——目前仅定了两个标记。如果这个 8 位字段最右边的第一位设置了（0x01），该路由就是外部路由。如果右边的第二位设置了（0x02），该路由就是一个候选的缺省路由。缺省路由的讲述请参见第 12 章的内容。

其余的字段描述了度量和目的地址。这些字段的含义与在内部路由 TLV 中讲述的相同字段的含义是一样的。

7.3.6 地址聚合

在第 1 章中介绍了子网划分的操作方法——为了使多条链路可以使用一个主网络地址，将地址掩码扩展到了主机地址空间。第 6 章介绍了可变子网掩码的操作方法——地址掩码的使用扩展到了子网中，甚至在子网中又创建了更多的新子网。

从相反的观点来看，子网地址也可以考虑成为一组更小的子网的汇总，而一个主网络地址可以看作一组子网的汇总。在每个这样的实例中，汇总都是通过减少子网掩码的长度来完成的。

地址聚合是打破主网络地址分类限制的进一步汇总措施。聚合的地址表示了一组数字上连续的网络地址，或称为超网（supernet）。[1]图 7-27 中显示了一个聚合地址的例子。

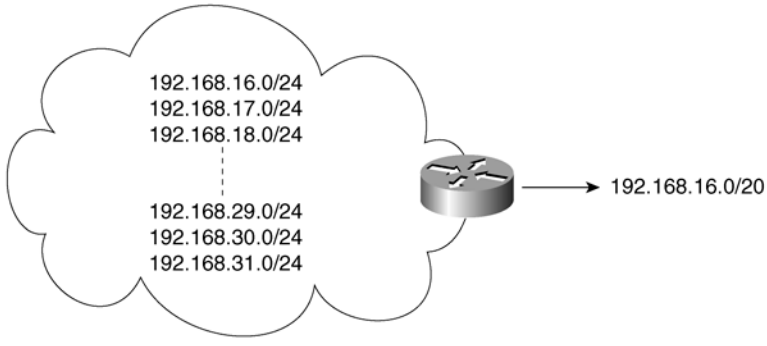


图 7-27　这组网络地址可以看作是单个的聚合地址或聚合子网

图 7-28 显示了是怎样得出图 7-27 中的聚合地址的。对于一组网络地址，寻找出所有网络地址的共同位并对这些位进行掩码。被掩码覆盖的部分就是聚合地址。

```
11111111111111111111111100000000 = 24 位掩码
11000000101010000001000000000000 = 192.168.16.0/24
11000000101010000001000100000000 = 192.168.17.0/24
11000000101010000001001000000000 = 192.168.18.0/24
11000000101010000001001100000000 = 192.168.19.0/24
11000000101010000001010000000000 = 192.168.20.0/24
11000000101010000001010100000000 = 192.168.21.0/24
11000000101010000001011000000000 = 192.168.22.0/24
11000000101010000001011100000000 = 192.168.23.0/24
11000000101010000001100000000000 = 192.168.24.0/24
11000000101010000001100100000000 = 192.168.25.0/24
11000000101010000001101000000000 = 192.168.26.0/24
11000000101010000001101100000000 = 192.168.27.0/24
11000000101010000001110000000000 = 192.168.28.0/24
11000000101010000001110100000000 = 192.168.29.0/24
11000000101010000001111000000000 = 192.168.30.0/24
11000000101010000001111100000000 = 192.168.31.0/24
11000000101010000001000000000000 = 192.168.16.0/20
```

图 7-28　聚合地址是由对一组数字上连续的网络地址的所有共同位进行掩码而得出的

当设计一个超网时，有一点很重要，就是超网的成员地址应该由原来掩码位的一个完整和连续的地址集合组成。例如，在图 7-28 中，聚合地址的 20 位掩码比成员地址的掩码少 4 位。对于这 4 个“不同”的位，注意，它们包括了 0000～FFFF 之间二进制位组合的每一种可能性。按照这个设计规则如果失败的话，将会引起寻址方案冲突、减小聚合路由的性能，并且可能导致路由选择环路和路由选择“黑洞”。

汇总寻址的一个明显的好处就是节省网络资源。由于通告更少的路由从而节省了带宽，而处理更少的路由则节省了 CPU 的周期。更为重要的是，由于减小了路由表的大小而节省了内存的使用。

无类别路由选择、VLSM 和聚合寻址都是通过创建层次化的地址来达到最大限度地节省网络资源的。与 IGRP 协议不同，EIGRP 协议支持所有这些寻址策略。在图 7-29 中，Treetop Aviation 的工程部门分配了 16 个 C 类的地址，这些地址已经根据需要分配到不同

[1] 更正确地说，聚合应该是任何一组地址的汇总。这里声明，本书中所提及的聚合地址是指一组主网络地址的汇总。

的子部门中。

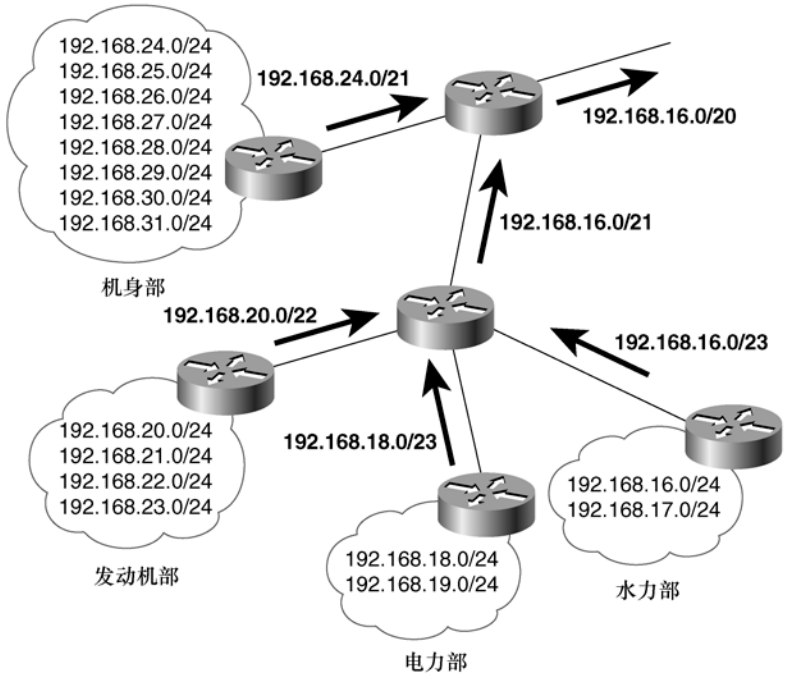


图 7-29 在 Treetop Aviation 中，一个更大部门中的几个子部门正在聚合地址。依次地，整个部门将通过单个聚合地址 192.168.16.0/20 通告出去

发动机、电力和水力部门的聚合地址是它们自己聚合到单个地址 192.168.16.0/21 中去的。这个地址和机身部门的聚合地址一起被聚合到一个单一的地址 192.168.16.0/20 中，这个地址也表示了整个工程部门的地址。

其他的部门也可以进行类似地表示。例如，假设 Treetop Aviation 总共有 8 个部门，而每个部门分配的地址与工程部门的相似，那么在最高层次的骨干路由器只有很少的 8 条路由，如图 7-30 所示。

层次化的地址设计将继续应用于每个部门的每一个子部门中，通过子网化划分成更小的单独的网络地址；VLSM 也可以用来进一步划分子网。路由选择协议将在网络的边界上自动地汇总子网，这和前面章节的讲述是一致的。

在 Internet 上，地址聚合也允许地址的节省和地址的分层。对于以指数速度增长的 Internet，有两点需要关注：可供使用的 IP 地址（尤其是 B 类地址）的消耗和存储 Internet 路由选择信息所需要的巨大的数据库。

这个问题的一种解决方案就是使用称为无类别域间路由选择（CIDR）的方法。[1] 在 CIDR 下，C 类地址的聚合由 IANA 机构分配给国际上不同的地址分配权威机构，像亚太地区的亚太网络信息中心（Asia Pacific Network Information Centre，APNIC），北美地区的美洲 Internet 编号注册局（American Registry for Internet Numbers，ARIN），以及欧洲地区的 Rèseaux IP Européens（RIPE）机构。这些地址聚合是按照地域进行分配管理的，如表 7-6 所示。

[1] V. Fuller, T. Li, J. I. Yu, and K. Varadhan.“Classless Inter-Domain Routing （CIDR）: An Address Assignment and Aggregation Strategy.”RFC 1519,1993 年 9 月。

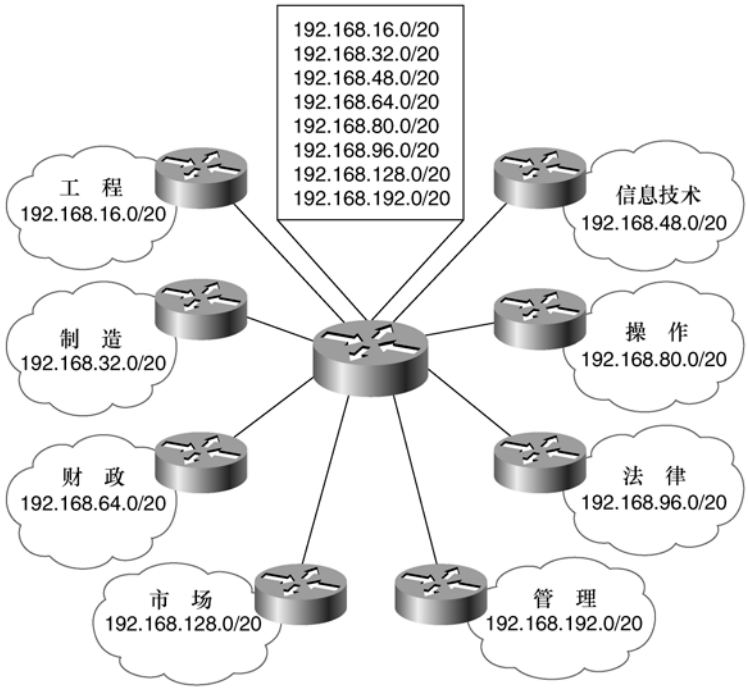


图 7-30　虽然在这个网络中有 128 个主网络地址并且可能覆盖了 32 000 台主机，但是骨干路由器的路由表中只有 8 条聚合地址

表 7-6　　CIDR 地址按国际上的地理区域分配

地区	地址范围
Multiregional	192.0.0.0–193.255.255.255
欧洲	194.0.0.0–195.255.255.255
其他	196.0.0.0–197.255.255.255
北美	198.0.0.0–199.255.255.255
中、南美	200.0.0.0–201.255.255.255
环太平洋地区（Pacific Rim）	202.0.0.0–203.255.255.255
其他	204.0.0.0–205.255.255.255
其他	206.0.0.0–207.255.255.255

这些地址分配权威机构轮流地把他们自己管理的那部分地址分配给本地的网络服务提供商（ISP）。当一个组织申请 IP 地址并且所需的地址小于 32 个子网和 4096 台主机时，将可以分配给它一组连续的称为 CIDR 块（CIDR block）的 C 类地址。

这样，各个组织团体所属的 Internet 路由器就可以把单一的汇总地址通告给他们的 ISP。反过来，ISP 也可以把其自己所有的地址聚合起来，因此在理论上可以把世界上一个区域内的所有 ISP 的地址都汇总到表 7-6 中所表示的地址中去。目前所了解到的 Internet 全球路由表的大小接近 200 000 条路由，显然并没有很好的坚持使用 CIDR 技术。尽管如此，CIDR 技术也是一个很好的理念。在规范 IPv6 地址分配给地区管理机构方面也进行了类似的努力。我们只能希望在 IPv6 方面的努力能够比在 IPv4 地址分配方面更加成功。

7.3.7 EIGRP 和 IPv6

正当撰写本书第二版时，EIGRP 协议还不支持 IPv6 协议。但是，对 EIGRP 协议进行扩展从而支持 IPv6 协议的努力也在进行，也许在读者阅读到本书的时候将会实现这些扩展。由于 EIGRP 数据包使用 TLV 传送数据，因此扩展该协议支持 IPv6 将可以通过增加专门的 IPv6 TLV 来简单的实现。

7.4 配置 EIGRP

本节的案例研究演示了一个基本的 EIGRP 配置，并讲述了路由汇总的技巧和与 IGRP 协议之间的互操作性。

7.4.1 案例研究：EIGRP 的基本配置

EIGRP 只需要两个步骤就可以启动一个 EIGRP 的路由选择进程：

步骤 1：使用 **router eigrp** *process-id* 命令启动 EIGRP 进程。

步骤 2：使用 **network** 命令来指定运行 EIGRP 协议的每个主网络。

EIGRP 进程 ID 号可以是 1～65 535（0 不允许使用）之间的任何一个数字，只要对必须共享路由信息的所有路由器上的所有 EIGRP 进程 ID 号是相同的，那么网络管理员可以随意地选用进程 ID 号。另外，这个进程号也可以是公共分配的自主系统号。图 7-31 显示了一个简单的网络，图中 3 台路由器的配置参见示例 7-14~示例 7-16。

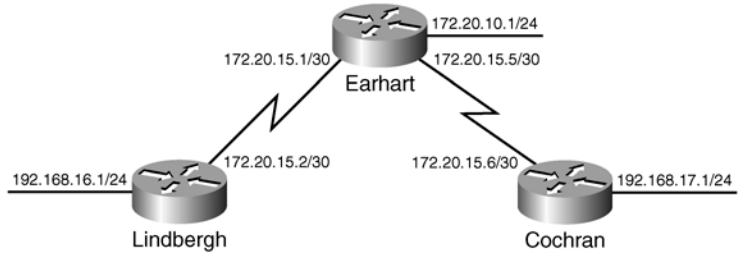


图 7-31 与 IGRP 协议不同，EIGRP 协议可以支持这个网络的 VLSM 要求

示例 7-14 路由器 Earhart 的配置

```
router eigrp 15
 network 172.20.0.0
```

示例 7-15 路由器 Cochran 的配置

```
router eigrp 15
 network 172.20.0.0
 network 192.167.17.0
```

示例 7-16　路由器 Lindbergh 的配置

```
router eigrp 15
 network 172.20.0.0
 network 192.167.16.0
```

路由器 Earhart 的路由表参见示例 7-17。这个路由表显示了 EIGRP 协议缺省的管理距离是 90，并且表中的网络 172.20.0.0 被划分为不同的子网。

示例 7-17　路由器 Earhart 的路由表

```
Earhart#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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
     172.20.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       172.20.10.0/24 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
D    192.168.17.0/24 [90/2195456] via 172.20.15.6, 00:00:01, Serial0.1
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:01, Serial0.2
Earhart#
```

与本章前面的一些例子不同，图 7-31 中的网络使用了缺省度量，这样在一个更为实际的环境中复习一下 EIGRP 的度量计算也许是有用的。

跟踪从路由器 Earhart 到达网络 192.168.16.0 的路由，这条路由的路径穿过了一个串行接口和一个以太接口，每个接口的度量都是缺省的配置数值。这条路由路径的最小带宽是串行接口上的带宽，[1]而时延是这两个接口时延的总和。请参考表 7-1：

$BW_{EIGRP(min)}$ = 256×6 476=1 657 856

$DLY_{EIGRP(sum)}$ = 256× (2 000+100)=537 600

因此，

Metric=1 657 856+537 600=2 195 456

7.4.2　案例研究：非等价负载均衡

在和 RIP 协议同样的 CEF/快速交换/处理交换（CEF/fast/process switching）转发机制的限制下，EIGRP 可以在最多 16 条等价的路由路径[2]上实现等价负载均衡。但与 RIP 协议不同的是，EIGRP 协议也可以实现非等价的负载均衡。在图 7-32 中，路由器 Earhart 和路由器 Cochran 中间另外增加了一条串行接口，它的带宽配置为 256kbit/s。这样做的目的是，为了使路由器 Earhart 在这两条链路上可以执行非等价负载均衡，即根据它们链路度量大小的反比来分配这两条链路上数据流量的负载大小。

观察路由器 Earhart 通过 S0.1 接口到达网络 192.168.17.0 的路由，可以看出最小的带宽是 1 544 kbit/s（假定路由器 Cochran 的以太网接口使用的是快速以太网的缺省带宽，即

[1] 记住串行接口的缺省带宽是 1544kbit/s。

[2] 缺省的路径是 4 条，可以参见 7.4.3 小节来获取进一步的详细信息。

100 Mbit/s）。根据表 7-1 中所标明的，串行接口和快速以太网接口的 $DLY_{EIGRP(sum)}$为 256×（2000+10）=514560。$BW_{EIGRP(min)}$为 256×（10^7/1544）=1657856，因此，这条路由的复合度量值是 514560+1657856=2172416。

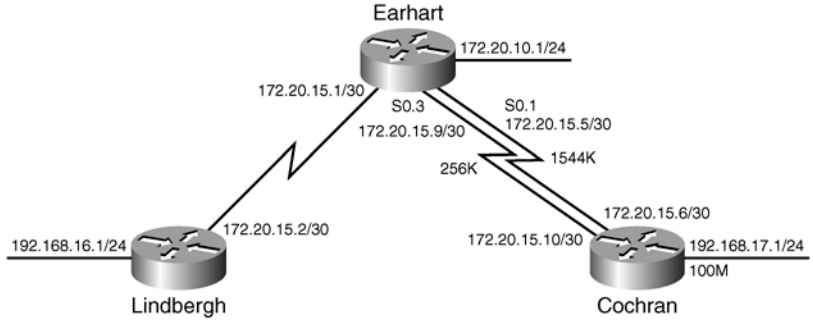


图 7-32 可以配置 EIGRP 协议在链路之间实现非等价负载均衡，例如路由器 Earhart 和 Cochran 之间的链路

路由器 Earhart 通过 S0.3 接口到达网络 192.168.17.0 的最小带宽是 256kbit/s。它的 $DLY_{EIGRP(sum)}$与上面的第一条路由相同，因此，这条路由的复合度量值是 256×（10^7/256）+514560=10514432。如果不做进一步的配置，EIGRP 协议将会简单地选择路径代价最小的路径。示例 7-18 显示了路由器 Earhart 仅仅使用经过串行接口 S0.1 的那条路径，它的度量值为 2172416。

示例 7-18 路由器 Earhart 仅仅使用最小路径代价的链路到达网络 192.168.17.0。要实现非等价负载均衡则需要另外的配置

```
Earhart#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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
     172.20.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       172.20.10.0/27 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
C       172.20.15.8/30 is directly connected, Serial0.3
D    192.168.17.0/24 [90/2172416] via 172.20.15.6, 00:00:09, Serial0.1
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:09, Serial0.2
Earhart#
```

差异变量（**variance**）命令用来确定哪些路由在非等价负载均衡中是可以使用的。**Variance** 定义了一个倍数因子，用来表示一条路由的度量值和最小代价路由的差异程度。任何路由的度量值如果超过了最小代价路由的度量值乘以 **Variance** 的值，那么这条路由将不被使用。

Variance 的缺省值是 1，这意味着如果要实现负载均衡，多条路由的度量值必须是相同的。Variance 必须是整数。

路由器 Earhart 通过 S0.3 接口的路由的度量值是通过 S0.1 接口的路由的 10514432/2172416=4.8 倍。因此，为了在这两条链路上实现非等价的负载均衡，路由器 Earhart 上的 variance 值应该是 5。EIGRP 的配置参见示例 7-19。

示例 7-19　在路由器 Earhart 的 EIGRP 配置中，使用数值为 5 的差异变量来实现非等价的负载均衡

```
router eigrp 15
 network 172.20.0.0
 variance 5
```

在路由器 Earhart 上将 variance 指定为 5 以后，它的路由表就包含了第二条代价较高的路由（参见示例 7-20）。在非等价负载均衡中的路由经常会碰到以下 3 种情况：

- 增加到负载共享“组”中的路由条数不能超过最大路径条数（maximum-paths）的限制。
- 下一跳路由器必须在度量值上更接近目的网络。换句话说，在下一跳路由器上到达目的网络的度量值必须小于本地路由器到达该目的网络的度量值。到达目的网络更近的下一跳路由器，通常被称为下游路由器（downstream router）。
- 最小路由代价的度量值乘以 variance 后，必须大于所增加的非最小路由代价的度量值。

示例 7-20　从路由器 Earhart 到达网络 192.168.17.0 的第二条路径的复合度量值是 10514432，或者说是最小代价的路由度量的 4.8 倍。如果 variance 的值设置为不小于 5 的话，那么 EIGRP 协议将把第二条路径加入到路由表中

```
Earhart(config)#router eigrp 15
Earhart(config-router)#variance 5
Earhart(config-router)#^Z
Earhart#clear ip route *
Earhart#show ip route
Gateway of last resort is not set
     172.20.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       172.20.10.0/24 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
C       172.20.15.8/30 is directly connected, Serial0.3
D    192.168.17.0/24 [90/2172416] via 172.20.15.6, 00:00:02, Serial0.1
                     [90/10514432] via 172.20.15.10, 00:00:02, Serial0.3
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:02, Serial0.2
Earhart#
```

关于按每目的地（per destination）和按每数据包（per packet）进行负载均衡的规则，已经在第 3 章中讨论过了，同样也适用于这里。如果数据包转发是快速交换（fast swithing）或缺省配置的 CEF 交换，就按照每目的地进行负载均衡；如果数据包转发是处理交换（process switching）或更改的 CEF 交换，就按照每数据包进行负载均衡。示例 7-21 显示了从路由器 Earhart 发出 20 个 ping 包的调试输出结果；在这里，CEF 和快速交换已经通过命令 **no ip cef** 和 **no ip route-cache** 关闭了，因此路由器将执行按每数据包的非等价负载均衡。每 5 个数据包通过 1544kbit/s 的链路（下一跳是 172.20.15.6）发送过后，就会有 1 个数据包通过 256kbit/s 的链路（下一跳是 172.20.15.10）发送，与这两条路径大约 5∶1 的度量比值是相一致的。

示例 7-21 路由器执行的是按每数据包的负载均衡，即每通过低代价的链路发送 5 个数据包，就会通过高代价的链路发送 1 个数据包

```
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.3), g=172.20.15.10,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.3), g=172.20.15.10,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.3), g=172.20.15.10,
    len 100, forward
Earhart#
```

如果 **variance** 设置为 1，那么 EIGRP 将只会把到达目的网络最小代价的路由加入到它的路由表中。然而，在一些情况下，例如，有时为了减小收敛的时间或帮助故障排错，即使没有发生负载均衡，也要将所有可用的路由都加入到路由表中。所有的数据包应该使用路径代价最小的路由，只有当主路径失效时，才被切换到下一个最好的路径。这里有一个隐含的缺省命令 **traffic-share balanced**（也就是说，这个命令存在，但是不需要保留在配置文件里面）。在路由表中存在多条路径时，为了使路由器只使用最小代价的路径，可以把缺省的配置改成 **traffic-share min**。如果有多条相等的最小代价的路径，并且配置了 **traffic-share min**，那么 EIGRP 将执行等价负载均衡。

7.4.3 案例研究：设置最大的路径数

EIGRP 协议可以进行负载均衡的路由路径的最大条数可以用 **maximum-paths** 命令来设置。在 12.3（2）T 版本和后来发布的 12.3(T)版本的 IOS 软件中，路径条数可以是 1~16 之间的任何值，而在这之前的早期版本中路径条数可以是 1~6 之间的任何值。所有版本的缺省值是 4。

图 7-33 显示了从路由器 Earhart 到达网络 172.18.0.0 的 3 条不同代价的并行路由。网络管理员可能仅仅希望在这些路由中的两条路由上进行负载均衡，只有当这些路由中的任何一条被确认失效时，才使用第三条路由替代它。

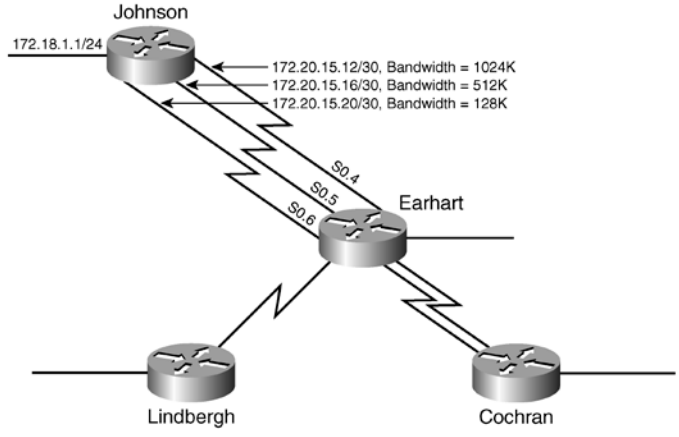


图 7-33　同时配置 **maximum-paths** 和 **variance** 命令，用来在路由器 Earhart 和 Johnson 之间 3 条链路中的其中两条上进行负载均衡。如果任何一条链路失效，第三条链路将替代失效的链路

从路由器 Earhart 上来看这 3 条链路的度量值分别是：

- 通过 S0.4 接口：256×（9765 +（2000 + 10））= 3014400；
- 通过 S0.5 接口：256×（19531 +（2000 + 10））= 5514496；
- 通过 S0.6 接口：256×（78125+（2000 + 10））= 20514560。

通过 S0.6 接口的度量值是最小代价路径度量值的 6.8 倍，因此，**variance** 的值应该是 7。路由器 Earhart 的 EIGRP 的配置参见示例 7-22 所示。

示例 7-22　路由器 Earhart 的配置。使用 variance 命令和 maximum-paths 命令实现指定最大路径数的非等价的负载均衡

```
router eigrp 15
 variance 7
 network 172.20.0.0
 maximum-paths 2
```

Variance 命令确保到达网络 172.18.0.0 的 3 条路由都是可用的；**maximum-paths** 命令用来限制负载均衡“组”中最多只有两条最佳的路由。在示例 7-23 中可以看到这种配置的结果。第一个路由表显示了路由器 Earhart 在 3 个度量中代价最低的两个链路上（即通过 S0.4 和 S0.5 接口的两条链路）进行负载均衡的情形。在通过 S0.4 的链路失效后，第二个路由表中显示了路由器在通过 S0.5 和 S0.6 的两条链路上进行负载均衡的情形。在每一种情况下，路由器都是执行和这两条路径的度量值成反比的负载均衡。

示例 7-23　路由器 Earhart 的路由表显示了同时使用 variance 和 maximum-paths 命令配置到达网络 172.18.0.0 的负载均衡时，3 条链路的某一条链路失效前后的不同结果

```
Earhart#debug eigrp neighbor
EIGRP Neighbors debugging is on
Earhart#show ip route
```

（待续）

```
Gateway of last resort is not set

D    172.18.0.0/16 [90/5514496] via 172.20.15.18, 00:00:16, Serial0.5
                   [90/3014400] via 172.20.15.14, 00:00:16, Serial0.4
     172.20.0.0/16 is variably subnetted, 7 subnets, 2 masks
C       172.20.15.20/30 is directly connected, Serial0.6
C       172.20.15.16/30 is directly connected, Serial0.5
C       172.20.10.0/24 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
C       172.20.15.12/30 is directly connected, Serial0.4
C       172.20.15.8/30 is directly connected, Serial0.3
D    192.168.17.0/24 [90/2195456] via 172.20.15.6, 00:00:18, Serial0.1
                     [90/10537472] via 172.20.15.10, 00:00:18, Serial0.3
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:19, Serial0.2
Earhart#
1w6d: EIGRP: Holdtime expired
1w6d: EIGRP: Neighbor 172.20.15.14 went down on Serial0.4
Earhart#show ip route
Gateway of last resort is not set
D    172.18.0.0/16 [90/5514496] via 172.20.15.18, 00:00:09, Serial0.5
                   [90/20514560] via 172.20.15.22, 00:00:09, Serial0.6
     172.20.0.0/16 is variably subnetted, 7 subnets, 2 masks
C       172.20.15.20/30 is directly connected, Serial0.6
C       172.20.15.16/30 is directly connected, Serial0.5
C       172.20.10.0/24 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
C       172.20.15.12/30 is directly connected, Serial0.4
C       172.20.15.8/30 is directly connected, Serial0.3
D    192.168.17.0/24 [90/2195456] via 172.20.15.6, 00:00:26, Serial0.1
                     [90/10537472] via 172.20.15.10, 00:00:26, Serial0.3
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:26, Serial0.2
Earhart#
```

在网络中配置并行的路径时应该小心谨慎。太多的并行路径在某个链路失效时会增加 EIGRP 收敛的时间，这是因为邻居路由器的数目也增加了，因此增加了查询的范围。

7.4.4 案例研究：多个 EIGRP 进程

在上述案例的网络中增添两台新的路由器 Bleriot 和 Post。在网络中创建两个 EIGRP 进程域，这两个进程域之间不进行通信。图 7-34 显示了这两个自主系统和其相关联的链路。

路由器 Post、Johnson、Lindbergh 和 Earhart 的配置都是相当简单的，路由器 Johnson、Earhart 和 Lindbergh 运行 EIGRP 15，而路由器 Post 将运行 EIGRP 10。在路由器 Bleriot 上的配置参见示例 7-24 所示。

示例 7-24 路由器 Bleriot 上配置了 EIGRP 15 和 EIGRP 10

```
router eigrp 15
 network 172.20.0.0
!
router eigrp 10
 network 10.0.0.0
```

每个 EIGRP 进程都只在指定网络的接口上运行。在路由器 Cochran 上，除了 FA0/0 之外的所有接口都属于网络 172.20.0.0，参见示例 7-25 所示。

使用 **passive-interface** 命令是为了防止 EIGRP Hello 发送到不属于它们的数据链路上去。

请注意，由于路由器 Cochran 的接口属于网络 172.20.0.0，因此 **passive-interface** 命令可以用来限制不必要的路由选择协议流量。对于 EIGRP 协议来说，这个命令阻塞了不必要的 Hello 消息。这样将不会形成邻接关系，也不会发送其他的 EIGRP 流量。

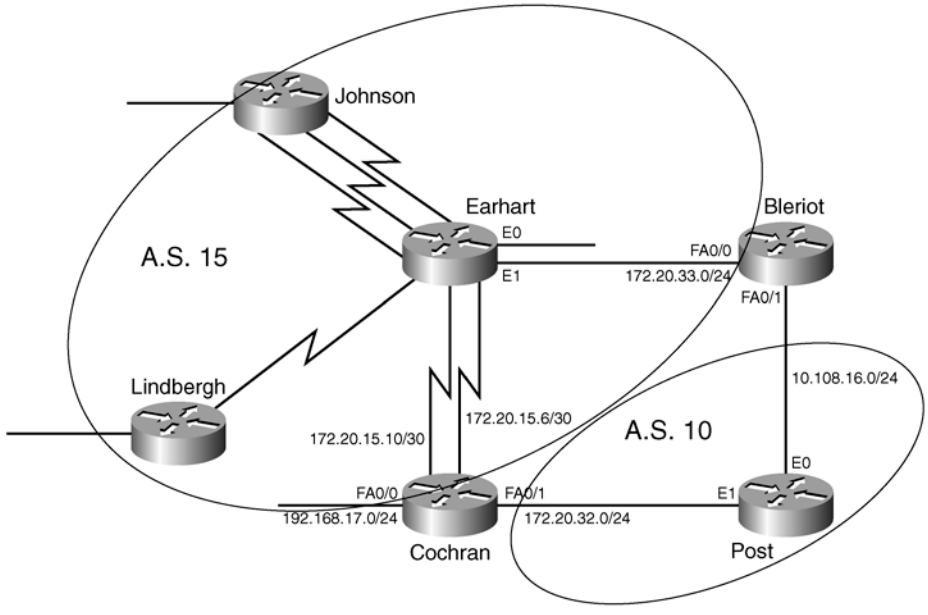


图 7-34　路由器 Bleriot 和 Cochran 上分别运行多个 EIGRP 进程，以便使用这个 IGP 协议创建各自的的自主系统（AS 10 和 AS 15）

示例 7-25　在路由器 Cochran 的 EIGRP 15 和 EIGRP 10 配置中要求配置被动接口，这是由于 Cochran 相连的两个接口上使用了相同的主网络号

```
router eigrp 15
 passive-interface Fastethernet0/1
 network 172.20.0.0
!
router eigrp 10
 passive-interface Serial0/0.1
 passive-interface Serial 0/0.2
 network 172.20.0.0
```

参见示例 7-26 所示，路由器 Bleriot 的邻居表显示了 EIGRP 10 进程的一个邻居 10.108.16.2 和 EIGRP 15 进程的一个邻居 172.20.33.1。

示例 7-26　使用命令 **show ip eigrp neighbor** 可以显示与每个 EIGRP 进程相关联的邻居

```
Bleriot#show ip eigrp neighbors
IP-EIGRP neighbors for process 15
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   172.20.33.1             Fa0/0             11 00:31:19   34   204  0  44
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   10.108.16.2             Fa0/1             10 00:19:21  289  1734  0  6
Bleriot#
```

在使用被动接口的地方，EIGRP 的网络语句可以配置通配符掩码位。这里的通配符掩码

位指定了在 EIGRP 进程中标识所包括的接口时使用的该地址的位数。

路由器 Cochran 的配置变化参见示例 7-27 所示。

示例 7-27 在路由器 Cochran 的 EIGRP 15 和 EIGRP 10 的配置中，使用了该网络的通配符掩码位来缩小运行给定 EIGRP 进程的接口范围

```
router eigrp 15
 network 172.20.15.0 0.0.0.255
!
router eigrp 10
 network 172.20.32.0 0.0.0.255
```

参见示例 7-27，在路由器 Cochran 的配置中，指定了前 3 个八位组为 172.20.15 的地址的接口运行 EIGRP 15，而前 3 个八位组为 172.20.32 的接口运行 EIGRP 10。

7.4.5 案例研究：关闭自动路由汇总

在缺省的情况下，EIGRP 协议和前面章节所讲述的协议一样，在网络边界上进行路由汇总。但是和前面那些协议不同的是，EIGRP 的自动路由汇总可以被关闭。

示例 7-28 中显示了路由器 Bleriot 的路由表。这里请注意，地址 172.20.32.0 的路由是通过接口 FE0/0 学习到的，但这条路径却是经过了路由器 Bleriot 的快速以太网链路和一条从路由器 Earhart 到 Cochran 的串行接口。而经过路由器 Post，与接口 FE0/1 相连的路径到达该目的地仅仅只有一个快速以太网链路的度量值。更仔细地查看这个路由表，我们发现经过 FE0/1 接口学习到的路由地址除了 10.108.16.0，只有 172.20.0.0/16 这一条路由地址了。在经过地址 10.0.0.0 处的网络边界后，通告给路由器 Bleriot 之前，地址 72.20.32.0 已经被汇总。为了使路由器 Bleriot 能够通过路由器 Post 转发数据包到 172.20.32.0，可以使用命令 **no auto-summary** 在路由器 Post 上关闭自动路由汇总。

示例 7-28 在某些情况下，EIGRP 的自动汇总会引起次优化的路由选择

```
Bleriot#show ip route
Gateway of last resort is not set
D    172.18.0.0/16 [90/3016960] via 172.20.33.1, 00:26:42, FastEthernet0/0
     172.20.0.0/16 is variably subnetted, 10 subnets, 4 masks
D       172.20.32.0/24 [90/2174976] via 172.20.33.1, 00:14:46, FastEthernet0/0
C       172.20.33.0/24 is directly connected, FastEthernet0/0
D       172.20.15.20/30
           [90/20514560] via 172.20.33.1, 00:20:28, FastEthernet0/0
D       172.20.15.16/30
           [90/5514496] via 172.20.33.1, 00:20:28, FastEthernet0/0
D       172.20.10.0/27 [90/284160] via 172.20.33.1, 00:43:34, FastEthernet0/0
D       172.20.15.4/30 [90/2172416] via 172.20.33.1, 00:26:59, FastEthernet0/0
D       172.20.15.0/30 [90/2172416] via 172.20.33.1, 00:27:18, FastEthernet0/0
D       172.20.0.0/16 [90/284160] via 10.108.16.2, 00:14:50, FastEthernet0/1
D       172.20.15.12/30
           [90/3014400] via 172.20.33.1, 00:26:49, FastEthernet0/0
D       172.20.15.8/30
           [90/10514432] via 172.20.33.1, 00:20:47, FastEthernet0/0
     10.0.0.0/24 is subnetted, 1 subnets
C       10.108.16.0 is directly connected, FastEthernet0/1
D    192.168.16.0/24 [90/2198017] via 172.20.33.1, 00:27:33, FastEthernet0/0
D    192.168.17.0/24 [90/2174976] via 172.20.33.1, 00:00:38, FastEthernet0/0
Bleriot#
```

路由器 Post 的配置参见示例 7-29 所示。

示例 7-29　路由器 Post 的配置中关闭了 EIGRP 的自动路由汇总功能。

```
router eigrp 10
 network 10.0.0.0
 network 172.20.0.0
 no auto-summary
```

路由器 Bleriot 新的路由表参见示例 7-30 所示。

示例 7-30　在关闭了 EIGRP 的自动路由汇总后，在路由表中就可以看到远端地址的子网了

```
Bleriot#show ip route
Gateway of last resort is not set
D    172.18.0.0/16 [90/3016960] via 172.20.33.1, 00:35:27, FastEthernet0/0
     172.20.0.0/16 is variably subnetted, 9 subnets, 3 masks
D       172.20.32.0/24 [90/284160] via 10.108.16.2, 00:00:55, FastEthernet0/1
C       172.20.33.0/24 is directly connected, FastEthernet0/0
D       172.20.15.20/30
           [90/20514560] via 172.20.33.1, 00:29:13, FastEthernet0/0
D       172.20.15.16/30
           [90/5514496] via 172.20.33.1, 00:29:13, FastEthernet0/0
D       172.20.10.0/27 [90/284160] via 172.20.33.1, 00:52:19, FastEthernet0/0
D       172.20.15.4/30 [90/2172416] via 172.20.33.1, 00:35:44, FastEthernet0/0
D       172.20.15.0/30 [90/2172416] via 172.20.33.1, 00:36:03, FastEthernet0/0
D       172.20.15.12/30
           [90/3014400] via 172.20.33.1, 00:35:34, FastEthernet0/0
D       172.20.15.8/30
           [90/10514432] via 172.20.33.1, 00:29:20, FastEthernet0/0
     10.0.0.0/24 is subnetted, 1 subnets
C       10.108.16.0 is directly connected, FastEthernet0/1
D    192.168.16.0/24 [90/2198016] via 172.20.33.1, 00:36:06, FastEthernet0/0
D    192.168.17.0/24 [90/2174976] via 172.20.33.1, 00:00:38, FastEthernet0/0
Bleriot#
```

在图 7-35 中显示了另外一种情况，说明关闭路由汇总是有用的。

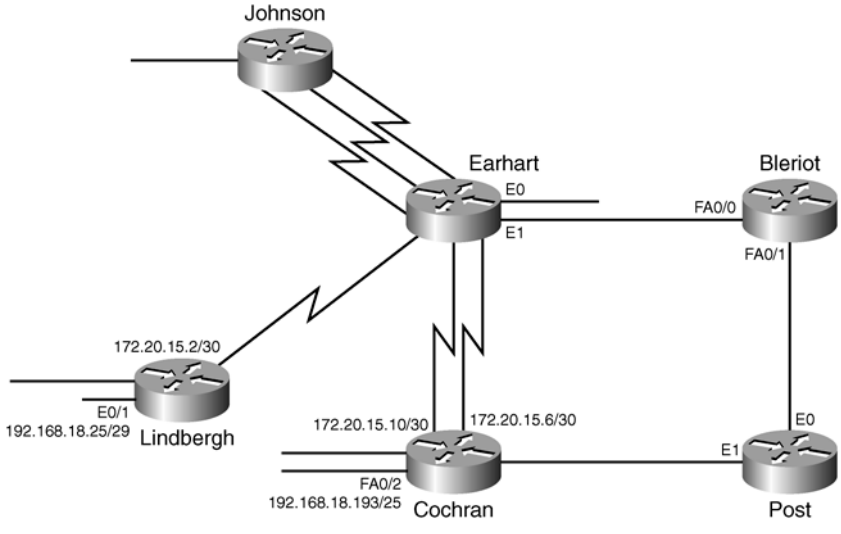


图 7-35　在路由器 Cochran 和 Lindbergh 上关闭自动路由汇总功能来防止到达网络 192.168.18.0 的不确定的路由

在路由器 Cochran 和路由器 Lindbergh 上分别添加了新的以太网链路，并且给它们分配的地址形成了不连续的子网。这两台路由器的缺省行为是把它们自己当作主网络 192.168.18.0 和 172.20.0.0 之间的边界路由器。结果，路由器 Earhart 将在它的两个串行接口上收到到达网络 192.168.18.0 的汇总路由通告。这个结果会产生一个路由选择不明确的情况：路由器 Earhart 记录了到达网络 192.168.18.0 的两条等价路径，要到达那些以太子网之一的数据包可能会、也可能不会被路由到正确的链路上去。

使用命令 **no auto-summary** 来关闭自动汇总功能后，路由器 Lindbergh 的配置参见示例 7-31 所示。

示例 7-31 路由器 Lindbergh 的配置关闭了自动汇总功能

```
router eigrp 15
 network 172.20.0.0
 network 192.168.16.0
 network 192.168.18.0
 no auto-summary
```

在路由器 Lindberghhe 和 Cochran 上关闭了自动路由汇总功能后，分离的子网 192.168.18.24/29 和 192.168.18.128/25 将可以被通告到网络 172.20.0.0 中去，这样就在路由器 Earhart 上排除了不确定的路由。

7.4.6 案例研究：末梢路由选择

现在回忆一下本章早些时候讲述的 DUAL 算法。当一台路由器的 EIGRP 拓扑表变坏的时候（可能是度量值变大了，可能是后继路由器不再可达了），如果没有可行后继可以到达该地址，那么这个路由条目将进入 active 状态，同时这台路由器会向它的所有邻居路由器发送查询数据包。如图 7-36 所示，假如路由器 Earhart 到路由器 Yeager 的链路中断了，路由器 Earhart 就会发送查询数据包到它的所有邻居，包括路由器 Johnson 和 Lindbergh，以便找到具有到达路由器 Yeager 的路径的邻居。路由器 Earhart 收到它所发出的所有请求那个路由条目的查询响应之前，它不会在拓扑表中改变那个 active 状态的条目。如果路由器 Earhart 到达 Lindbergh 的链路又出问题了，而这时 Earhart 还没有收到有关查询路由器 Yeager 的地址的任何答复，那么即使路由器 Earhart 和 Yeager 之间的链路恢复正常，路由器 Yeager 的地址也会继续保持 active 状态。

如图 7-36 所示，在这个网络中，路由器 Johnson 和 Lindbergh 没有任何后门路由到达其它任何站点。它们在设计为星型（hub-and-spoke）拓扑的网络中称为分支（spoke）路由器。这些路由器不用来为网络中的任何地址提供透传路径（transit path）。如果路由器 Lindbergh 或 Johnson 需要向非本地的站点地址转发数据包时，那么数据包就会被转发到路由器 Earhart 上。例如，路由器 Lindbergh 知道一条到达地址 172.20.10.0 的路径，这条路径是经过路由器 Earhart 的。为了减少网络不稳定的风险，不需要向路由器 Johnson 发送有关网络上其他目的地地址的查询。路由器 Johnson 和 Lindbergh 可以配置为末梢路由选择（stub routing）。

一台具有 EIGRP 末梢邻居的路由器将不会向它的末梢发送查询，因此这就排除了配置末梢的远端站点引起“卡在活动状态”的情形发生的机会，也降低了网络中其余部分路由选择的不稳定。

下面路由器 Johnson 被配置为一台 EIGRP 末梢路由器。Johnson 有关末梢路由器的配置

参见示例 7-32 所示。

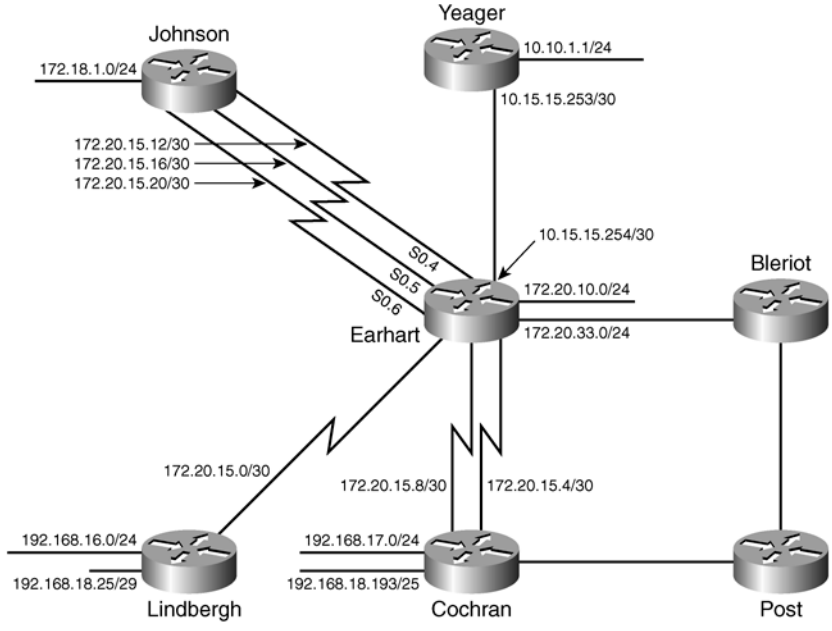


图 7-36 路由器 Yeager 通过添加到网络中的单条链路连接到路由器 Earhart

示例 7-32 路由器 Johnson 的 EIGRP 末梢路由器配置

```
router eigrp 15
 eigrp stub
```

路由器 Earhart 作为中心路由器，不需要改变配置。

命令 **eigrp stub** 会使路由器 Johnson 仅仅发送包含与它直连的路由和汇总路由的更新消息。路由器 Johnson 使用下面的命令可以配置包括下列路由的任何组合，如直连路由、汇总路由、静态路由，或者重新分配到 EIGRP 内的路由，等等：

```
eigrp stub {connected | redistributed | static | summary | receive-only}
```

路由器 Johnson 也可以利用 **receive-only** 选项配置为在更新消息中不发送任何路由信息。在使用 **receive-only** 选项的配置下，远端的路由器在更新消息中将不包括任何地址。路由器 Johnson 将不得不利用其他方式把与它直连的地址通告到网络的其余部分，以便使流量可以到达那个地址的站点，这可以通过在路由器 Earhart 上配置静态路由来实现。

参见示例 7-33 所示，可以在中心路由器上使用命令 **show ip eigrp neighbor detail** 检验配置为末梢路由器的邻居。路由器 Earhart 的输出信息显示路由器 Johnson 配置成为一台末梢路由器。

示例 7-33 命令 show ip eigrp neighbor detail 显示了配置为 EIGRP 末梢路由器的邻居路由器

```
Earhart#show ip eigrp neighbor detail
IP-EIGRP neighbors for process 15
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
```

（待续）

```
                                    (sec)         (ms)       Cnt Num
7   172.20.33.2             Et1           11 00:00:10   12   200  0  13
   Version 12.1/1.2, Retrans: 0, Retries: 0
6   10.15.15.253            Et2           11 00:00:10   12   200  0  6
   Version 12.3/1.2, Retrans: 1, Retries: 0
5   172.20.15.22            Se0.6         11 00:00:13  298  1788  0  73
   Version 12.3/1.2, Retrans: 0, Retries: 0
   Stub Peer Advertising ( CONNECTED SUMMARY ) Routes
4   172.20.15.14            Se0.4         11 00:00:13  927  5000  0  81
   Version 12.3/1.2, Retrans: 0, Retries: 0
   Stub Peer Advertising ( CONNECTED SUMMARY ) Routes
3   172.20.15.18            Se0.5         10 00:00:13  817  4902  0  80
   Version 12.3/1.2, Retrans: 0, Retries: 0
   Stub Peer Advertising ( CONNECTED SUMMARY ) Routes
0   172.20.15.2             Se0.2         11 00:15:48  274  1644  0  13
   Version 12.3/1.2, Retrans: 0, Retries: 0
2   172.20.15.10            Se0.3         13 00:45:40   72   570  0  59
   Version 12.3/1.2, Retrans: 0, Retries: 0
1   172.20.15.6             Se0.1         11 00:46:01   61   366  0  57
   Version 12.3/1.2, Retrans: 0, Retries: 0
Earhart#
```

路由器Earhart具有3个末梢路由器。其中有3条链路连接到路由器Johnson：172.20.15.22、172.20.15.14，以及 172.20.15.17。

现在，假定为了增加与路由器 Lindbergh 相连的 LAN 通过网络核心区域的流量的冗余性，我们在路由器Lindbergh和Cochran之间增加一条新的链路，如图7-37所示。在路由器Lindbergh配置作为一台末梢路由器之前，如果路由器 Cochran 与 Earhart 之间的链路出现故障，路由器 Cochran 将会发送查询到路由器 Lindbergh，希望在它的拓扑表中找到到达目的地址的另一条路径，在这里实例中的地址是 172.20.10.0。路由器 Lindbergh 将会响应一个确定的应答，因为在 Lindbergh 的拓扑表中具有一条经过路由器 Earhart 学到的网络 172.20.10.0 的路由条目。因此，从路由器 Cochran 到达 172.20.10.1 的流量将经过路由器 Lindbergh。

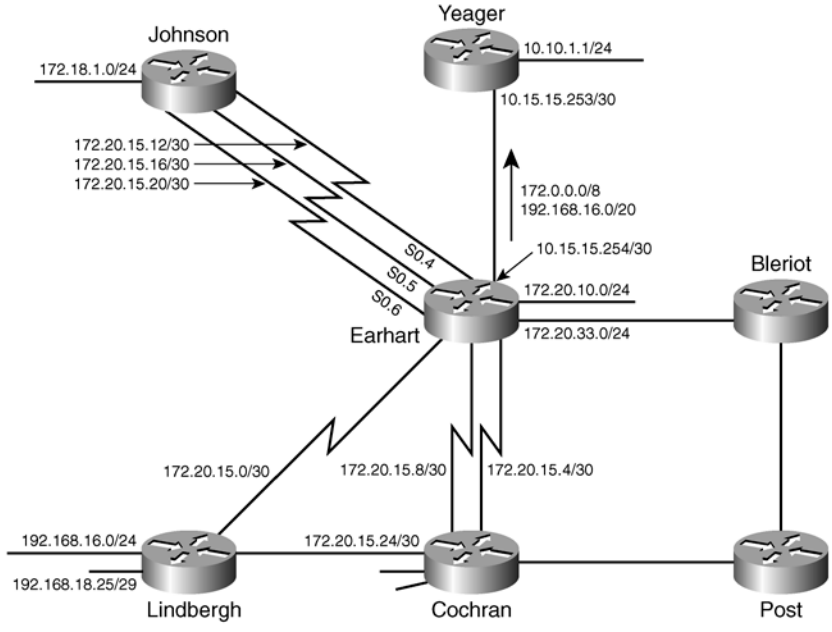


图 7-37 在路由器 Lindbergh 与 Cochran 之间增加了一条新的链路，用来增加 Lindbergh 所连接的 LAN 网络到核心网络之间流量的冗余性

虽然这是另一条可选的路径，但是经过一个远程站点转发流量并不总是明智的选择。在这个实例中，路由器 Lindbergh 的那条链路是用来作为从它所连接的地址到核心网络地址之间流量的冗余链路的，而不是希望 Lindbergh 作为一个中间透传路由器。它的带宽也许不够用来提供中间透传的路由。末梢路由选择可以简单地解决这个问题。作为一台末梢路由器，没有任何查询消息发送到路由器 Lindbergh 上，因而路由器 Lindbergh 也不会使自己成为路由器 Cochran 的另一条可用的路径。而且，路由器 Lindbergh 只会发送包括以下这些路由的更新：直连路由、汇总路由、静态路由，或重新分配路由，而不会包括像 172.20.10.0 这样的远端路由。

在示例 7-34 中显示了路由器 Lindbergh（连接到 Cochran 的 Serial0/0.4 接口）没有配置为末梢路由器时，路由器 Cochran 的 EIGRP 邻居。假定路由器 Cochran 到路由器 Earhart 的两条链路都中断了。参见示例 7-35，路由器 Cochran 随后的拓扑表显示所有地址都是经过路由器 Lindbergh（Serial0/0.4）转发的。

示例 7-34　路由器 Cochran 的 EIGRP 邻居表显示了它的邻居路由器。路由器 Lindbergh 不是一台末梢路由器

```
Cochran#show ip eigrp neighbor detail
IP-EIGRP neighbors for process 15
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
2   172.20.15.26            Se0/0.4           10 00:00:18 1152  5000  0  34
   Version 12.3/1.2, Retrans: 0, Retries: 0
1   172.20.15.9             Se0/0.2           14 00:41:58  104   624  0  205
   Version 12.1/1.2, Retrans: 1, Retries: 0
0   172.20.15.5             Se0/0.1           11 00:49:12   99   594  0  206
   Version 12.1/1.2, Retrans: 2, Retries: 0
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   172.20.32.2             Fa0/1             13 00:42:01   60   360  0  14
   Version 12.1/1.2, Retrans: 2, Retries: 0
Cochran#
```

示例 7-35　在路由器 Cochran 与 Earhart 之间的主要链路都发生故障后，所有的地址都通过具有双连接的分支路由器 Lindbergh 访问了

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 15: Neighbor 172.20.15.5 (Serial0/0.1) is down:
interface down
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 15: Neighbor 172.20.15.9 (Serial0/0.2) is down:
interface down
Cochran#show ip eigrp topology
IP-EIGRP Topology Table for AS(15)/ID(192.168.17.1)
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 10.0.0.0/8, 1 successors, FD is 2707456
        via 172.20.15.26 (2707456/2195456), Serial0/0.4
P 192.168.18.24/29, 1 successors, FD is 2195456
        via 172.20.15.26 (2195456/281600), Serial0/0.4
P 192.168.16.0/24, 1 successors, FD is 2195456
        via 172.20.15.26 (2195456/281600), Serial0/0.4
P 192.168.17.0/24, 1 successors, FD is 28160
        via Connected, FastEthernet0/0
P 172.20.32.0/24, 1 successors, FD is 28160
```

（待续）

```
        via Connected, FastEthernet0/1
P 172.20.33.0/24, 1 successors, FD is 2707456
        via 172.20.15.26 (2707456/2195456), Serial0/0.4
P 172.20.15.20/30, 1 successors, FD is 21536000
        via 172.20.15.26 (21536000/21024000), Serial0/0.4
P 172.20.15.16/30, 1 successors, FD is 6535936
        via 172.20.15.26 (6535936/6023936), Serial0/0.4
P 172.20.15.24/30, 1 successors, FD is 2169856
        via Connected, Serial0/0.4
P 172.20.10.0/27, 1 successors, FD is 2707456
        via 172.20.15.26 (2707456/2195456), Serial0/0.4
P 172.20.15.4/30, 1 successors, FD is 3193856
        via 172.20.15.26 (3193856/2681856), Serial0/0.4
P 172.20.15.0/30, 1 successors, FD is 2681856
        via 172.20.15.26 (2681856/2169856), Serial0/0.4
P 172.20.15.12/30, 1 successors, FD is 4035840
        via 172.20.15.26 (4035840/3523840), Serial0/0.4
P 172.18.0.0/16, 1 successors, FD is 4038400
        via 172.20.15.26 (4038400/3526400), Serial0/0.4
P 172.20.15.8/30, 1 successors, FD is 11535872
        via 172.20.15.26 (11535872/11023872), Serial0/0.4
P 192.168.18.128/25, 1 successors, FD is 28160
        via Connected, FastEthernet0/2
P 10.108.16.0/24, 1 successors, FD is 284160
        via 172.20.32.2 (284160/281600), FastEthernet0/1
P 172.20.15.24/30, 1 successors, FD is 2169856
        via Connected, Serial0/0.4
Cochran#
```

现在，将路由器 Lindbergh 配置为一台 EIGRP 的末梢路由器，参见示例 7-36 所示。

示例 7-36　路由器 Lindbergh 配置为一台 EIGRP 末梢路由器

```
router eigrp 15
 eigrp stub
```

示例 7-37 显示了路由器 Cochran 的 EIGRP 邻居表，示例 7-38 显示了路由器 Cochran 到 Earhart 的两条链路再次中断后 Cochran 的 EIGRP 拓扑表。

示例 7-37　路由器 Cochran 的 EIGRP 邻居表显示了它的邻居路由器。路由器 Lindbergh 是一台末梢路由器。路由器 Cochran 运行的是比 Earhart 更新的 IOS 版本（12.3）。请注意在这里被抑制的查询是明确表明的

```
Cochran#show ip eigrp neighbor detail
IP-EIGRP neighbors for process 15
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
2   172.20.15.26            Se0/0.4           10 00:01:08   56   336  0  20
   Version 12.3/1.2, Retrans: 2, Retries: 0
   Stub Peer Advertising ( CONNECTED SUMMARY ) Routes
   Suppressing queries
1   172.20.15.9             Se0/0.2           11 00:29:46   96   576  0  111
   Version 12.1/1.2, Retrans: 1, Retries: 0
0   172.20.15.5             Se0/0.1           10 00:37:00   96   576  0  110
   Version 12.1/1.2, Retrans: 2, Retries: 0
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   172.20.32.2             Fa0/1             13 00:29:50   51   306  0  10
   Version 12.1/1.2, Retrans: 2, Retries: 0
Cochran#
```

示例 7-38　在路由器 Cochran 与 Earhart 之间的主要链路都发生故障后，只有与路由器 Lindbergh 相连的地址可以通过串口 Serial0/0.4 到达

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 15: Neighbor 172.20.15.5 (Serial0/0.1) is down:
interface down
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 15: Neighbor 172.20.15.9 (Serial0/0.2) is down:
interface down

Cochran#show ip eigrp topology
IP-EIGRP Topology Table for AS(15)/ID(192.168.17.1)
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 192.168.18.24/29, 1 successors, FD is 2195456
        via 172.20.15.26 (2195456/281600), Serial0/0.4
P 192.168.16.0/24, 1 successors, FD is 2195456
        via 172.20.15.26 (2195456/281600), Serial0/0.4
P 192.168.17.0/24, 1 successors, FD is 28160
        via Connected, FastEthernet0/0
P 172.20.32.0/24, 1 successors, FD is 28160
        via Connected, FastEthernet0/1
P 172.20.15.24/30, 1 successors, FD is 2169856
        via Connected, Serial0/0.4
P 172.20.15.0/30, 1 successors, FD is 2681856
        via 172.20.15.26 (2681856/2169856), Serial0/0.4
P 192.168.18.128/25, 1 successors, FD is 28160
        via Connected, FastEthernet0/2
P 10.108.16.0/24, 1 successors, FD is 284160
        via 172.20.32.2 (284160/281600), FastEthernet0/1
P 172.20.15.24/30, 1 successors, FD is 2169856
        via Connected, Serial0/0.4
Cochran#
```

配置路由器 Lindbergh 作为 EIGRP 末梢路由器可以防止它在核心链路发生故障期间变成中间透传的路由器。

配置 EIGRP 的末梢路由选择可以最大限度地减少查询的数量，这样极大地增加了 EIGRP 网络的扩展性，而网络中断时间要求相关地址处于 active 状态。

末梢路由选择取消了发送到末梢路由器的查询，但是它并没有为末梢点隐藏网络其余部分的拓扑。路由器 Earhart 能够对末梢点隐藏网络其余部分的拓扑。由于到每个子网的所有数据包总是转发到中心路由器的，因此这些末梢路由器就不需要了解关于每个子网的信息。路由器 Earhart 可以通过汇总地址完成这项任务。

7.4.7　案例研究：地址汇总

在图 7-37 所示的网络中，路由器 Yeager 具有单条链路连接到路由器 Earhart 上。路由器 Earhart 必须通告到路由器 Yeager 的 6 个地址能够被汇总为两条聚合的地址。路由器 Earhart 的配置参见示例 7-39 所示。

示例 7-39　路由器 Earhart 的配置汇总路由到路由器 Yeager

```
interface Ethernet2
 ip address 10.15.15.254 255.255.255.252
 ip summary-address eigrp 15 172.0.0.0 255.0.0.0
 ip summary-address eigrp 15 192.168.16.0 255.255.240.0
```

命令 **ip summary-address eigrp** 将会自动地抑制更具体的到达路由器 Yeager 的网络的通告。示例 7-40 显示了配置聚合地址前后路由器 Yeager 的路由表。即使在这么一个小的网络中，通过 EIGRP 学到的路由条目数也减少了一半；在一个大型网络中，压缩路由表的大小和存储它们所需要的内存是很重要的。

示例 7-40 在路由器 Earhart 上配置汇总地址前后所显示的路由器 Yeager 的路由表

```
Yeager#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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

D    172.18.0.0/16 [90/3040000] via 10.15.15.254, 00:13:07, Ethernet0
D    172.20.0.0/16 [90/307200] via 10.15.15.254, 00:13:07, Ethernet0
     10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.10.1.0/24 is directly connected, Ethernet1
C       10.15.15.252/30 is directly connected, Ethernet0
     192.168.17.0/27 is subnetted, 1 subnets
D       192.168.17.0 [90/2198016] via 10.15.15.254, 00:03:57, Ethernet0
D    192.168.16.0/24 [90/2221056] via 10.15.15.254, 00:01:51, Ethernet0
     192.168.18.0/24 is variably subnetted, 2 subnets, 2 masks
D       192.168.18.24/29 [90/2221056] via 10.15.15.254, 00:13:09, Ethernet0
D       192.168.18.128/25 [90/2198016] via 10.15.15.254, 00:13:09, Ethernet0

Yeager#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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

     10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.10.1.0/24 is directly connected, Ethernet1
C       10.15.15.252/30 is directly connected, Ethernet0
D    172.0.0.0/8 [90/307200] via 10.15.15.254, 00:00:57, Ethernet0
D    192.168.16.0/20 [90/2198016] via 10.15.15.254, 00:00:57, Ethernet0
```

7.4.8 认证

MD5 加密校验和是 EIGRP 协议惟一支持的认证方式，初看起来，这和 RIPv2 或 OSPF 协议相比灵活性较差，因为后两种协议同时支持 MD5 认证和明文口令认证。然而，明文口令认证应该只使用在邻居设备不支持比较安全的 MD5 认证的时候。由于 EIGRP 协议仅会在两台 Cisco 设备之间互相宣告，因此这种情况不会发生。

配置 EIGRP 协议的认证有以下几个步骤：

步骤 1：定义一个带有名字的钥匙链。

步骤 2：在钥匙链上定义一个或一组钥匙。

步骤 3：在接口上启用认证并指定使用的钥匙链。

步骤 4：可选地配置钥匙的管理。

钥匙链的配置和管理已经在第 6 章中讲述过。EIGRP 协议认证是在接口上配置命令 **ip authentication key-chain eigrp** 和命令 **ip authentication mode eigrp md5** 来启用和连接到一个钥匙链上的。[1]

参考图 7-37，在路由器 Cochran 到路由器 Earhart 的接口上启动 EIGRP 认证的配置参见示例 7-41 所示。

示例 7-41 配置路由器 Cochran 与路由器 Earhart 使用 MD5 认证

```
key chain Edwards
 key 1
 key-string PanchoBarnes
!
interface Serial0/0.1
 ip address 172.20.15.6 255.255.255.252
 ip authentication key-chain eigrp 15 Edwards
 ip authentication mode eigrp 15 md5
interface Serial0/0.2
 ip address 172.20.15.10 255.255.255.252
 ip authentication key-chain eigrp 15 Edwards
 ip authentication mode eigrp 15 md5
```

在路由器 Earhart 上也要作相似的配置。关于钥匙链的管理命令 **accept-lifetime** 和 **send-lifetime** 的使用方法已经在第 6 章中讲述过了。

7.5 EIGRP 故障诊断

RIP 协议的路由信息交换的故障诊断是一个相当简单的过程。路由选择更新要么传播出去了要么没有传播出去，要么包含了精确的路由信息要么没有包含。EIGRP 协议复杂性的增加也意味着故障排除过程的复杂性增加了。在 EIGRP 协议的故障排除中，必须验证邻居表和邻接关系的正确性，检查 DUAL 算法的查询/响应过程的正确性，并考虑在自动汇总时对 VLSM 的影响。

本节的案例研究讲述了一系列典型的事件，可以用来追踪一个 EIGRP 的故障。随后的一个案例研究将讲述在一个大型的 EIGRP 网络中引起网络不稳定的一些偶然原因。

7.5.1 案例研究：邻居丢失

图 7-38 中显示了一个小型的 EIGRP 网络。用户抱怨子网 192.168.16.224/28 无法到达。仔细察看路由表，发现在路由器 Grissom 上有一些错误（参见示例 7-42 所示）。[2]

---

[1] 虽然 MD5 认证是 EIGRP 协议目前惟一可用的认证模式，但是使用命令 **ip authentication mode eigrp md5** 可以为将来出现其他可用的认证模式作预先考虑。

[2] 当排除一个网络的故障时，检查所有路由器接口的地址配置是否属于正确的子网是一种好习惯。

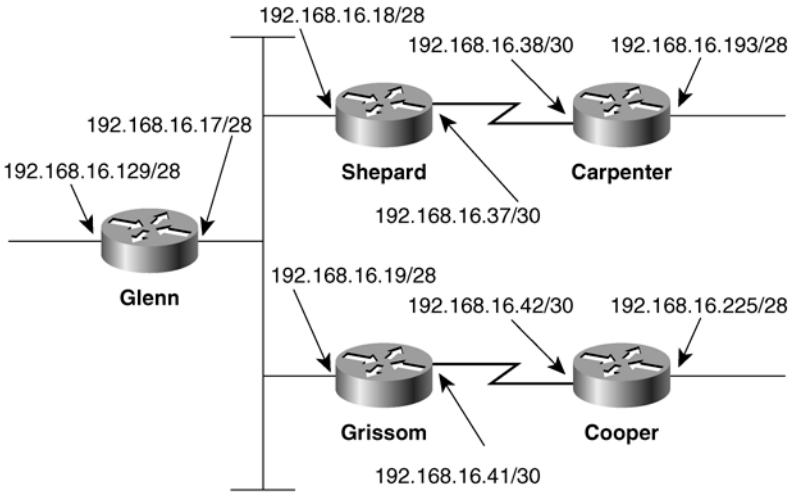


图 7-38 在这个例子的 EIGRP 网络中，通过路由器 Grissom 不能到达子网 192.168.16.224/28

示例 7-42 路由器 Shepard 和 Grissom 的路由表显示了路由器 Grissom 的 EIGRP 进程没有通告和接收到关于子网 192.168.16.16/28 的路由

```
Grissom#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
     192.168.16.0/24 is variably subnetted, 3 subnets, 2 masks
C       192.168.16.40/30 is directly connected, Serial0
C       192.168.16.16/28 is directly connected, Ethernet0
D       192.168.16.224/28 [90/2195456] via 192.168.16.42, 01:07:26, Serial0
```

```
Shepard#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
     192.168.16.0/24 is variably subnetted, 4 subnets, 2 masks
C       192.168.16.36/30 is directly connected, Serial0
C       192.168.16.16/28 is directly connected, Ethernet0
D       192.168.16.192/28 [90/2297856] via 192.168.16.38, 01:07:20, Serial0
D       192.168.16.128/28 [90/307200] via 192.168.16.17, 01:07:20, Ethernet0
```

从示例 7-42 中的两个路由表中可以得到以下的信息：

- 路由器 Shepard 的路由表中没有子网 192.168.16.40/30 和子网 192.168.16.224/28 的信息，虽然路由器 Grissom 的路由表中含有这些信息。
- 路由器 Grissom 的路由表中没有包含任何应该由路由器 Glenn 或 Shepard 通告的子网。
- 路由器 Shepard 的路由表中包含了由路由器 Glenn 通告的子网（并且路由器 Glenn 的路由表中包含了路由器 Shepard 通告的子网，虽然它的路由表没有包括在示例 7-42 中）。

从以上的观察信息可以得出结论，路由器 Grissom 没有正确地接收和通告关于子网

192.168.16.16/28 的路由。

在这些可能引起故障的原因中，应该首先来察看一些最简单和直接的原因。这些原因有：

- 接口地址或者掩码配置不正确；
- EIGRP 的进程 ID 号不正确；
- **network** 语句遗漏或者不正确。

在这个实例中，没有发现 EIGRP 或地址的配置错误。

接下来，仔细检查一下邻居表，分别在路由器 Grissom、Shepard 和 Glenn 上察看邻居表，参见示例 7-43 所示，有两个事实比较明显：

- 路由器 Grissom（192.168.16.19）在它的邻居的邻居表中，但是它的邻居却不在路由器 Grissom 的邻居表中。
- 整个网络已经运行了 5 个多小时了，这个信息可以从除了路由器 Grissom 外的所有邻居的 uptime 统计信息反映出来。然而，路由器 Grissom 的 uptime 显示大约是 1min。

示例 7-43　路由器 Shepard 和 Glenn 发现路由器 Grissom 是它们的邻居，但 Grissom 却看不到它们。这意味着路由器 Shepard 和 Glenn 可以接收来自 Grissom 的 Hello 消息，但是 Grissom 收不到路由器 Shepard 和 Glenn。

```
Grissom#show ip eigrp neighbors
IP-EIGRP neighbors for process 75
H   Address            Interface   Hold   Uptime    SRTT   RTO   Q   Seq
                                   (sec)             (ms)       Cnt  Num
0   192.168.16.42      Se0           11   05:27:11   23    200   0   8
Shepard#show ip eigrp neighbors
IP-EIGRP neighbors for process 75
H   Address            Interface   Hold   Uptime    SRTT   RTO   Q   Seq
                                   (sec)             (ms)       Cnt  Num
1   192.168.16.19      Et0           12   00:01:01    0   5000   1   0
2   192.168.16.17      Et0           11   05:27:33    8    200   0   6
0   192.168.16.38      Se0           14   05:27:34   22    200   0   10
Glenn#show ip eigrp neighbors
IP-EIGRP neighbors for process 75
H   Address            Interface   Hold   Uptime    SRTT   RTO   Q   Seq
                                   (sec)             (ms)       Cnt  Num
1   192.168.16.19      Et0           14   00:00:59    0   8000   1   0
2   192.168.16.18      Et0           10   05:30:11    9     20   0   7
0   192.168.16.130     Et1           12   05:30:58    6     20   0   7
```

假设路由器 Grissom 在路由器 Shepard 的邻居表中，则路由器 Shepard 必须接收来自路由器 Grissom 的 Hello 消息。然而，路由器 Grissom 显然不接收来自路由器 Shepard 的 Hello 消息。没有 Hello 消息的双向交换，就不能形成一个邻接关系，路由信息也就无法进行交换。

更仔细地检查路由器 Shepard 和路由器 Glenn 的邻居表，更加证实了这个假设：

- 路由器 Grissom 的 SRTT 为 0，表示从来没有一个数据包在路由路径上进行过往返（round-trip）。
- 路由器 Grissom 的 RTO 分别增加到了 5s 和 8s。
- 有一个关于路由器 Grissom 的数据包在队列中等待发送。
- 关于路由器 Grissom 记录的序列号为 0，表示从来没有从路由器 Grissom 接收到可靠的数据包。

这些因素都表明，这两台路由器都在试图向路由器 Grissom 发送可靠的数据包，但是却

没有收到一个 ACK 确认消息。

在示例 7-44 中，在路由器 Shepard 上使用命令 **debug eigrp packets** 可以更好地观察到底发生了什么。这样的话，所有的 EIGRP 数据包类型都将显示出来了，但是可以使用第二个调试命令：**debug ip eigrp neighbor 75 192.168.16.19**。该命令是在第一个命令的基础上增加了一个过滤器。它告诉 **debug eigrp packet** 只需要显示 EIGRP 75（图 7-38 中那些路由器的进程 ID 号）的 IP 数据包并且只显示和邻居 192.168.16.19（路由器 Grissom）有关的那些数据包。

示例 7-44 调试命令 debug ip eigrp neighbor 用来控制命令 debug eigrp packet 所显示的数据包

```
Shepard#debug eigrp packets
EIGRP Packets debugging is on
   (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK)
Shepard#debug ip eigrp neighbor 75 192.168.16.19
IP Neighbor target enabled on AS 75 for 192.168.16.19
IP-EIGRP Neighbor Target Events debugging is on
EIGRP: Sending UPDATE on Ethernet0 nbr 192.168.16.19, retry 14, RTO 5000
   AS 75, Flags 0x1, Seq 22/0 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
1-4
EIGRP: Received HELLO on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
EIGRP: Sending UPDATE on Ethernet0 nbr 192.168.16.19, retry 15, RTO 5000
AS 75, Flags 0x1, Seq 22/0 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
1-4
EIGRP: Received HELLO on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
EIGRP: Sending UPDATE on Ethernet0 nbr 192.168.16.19, retry 16, RTO 5000
   AS 75, Flags 0x1, Seq 22/0 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
1-4
EIGRP: Received HELLO on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
EIGRP: Retransmission retry limit exceeded
EIGRP: Received HELLO on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0
EIGRP: Enqueueing UPDATE on Ethernet0 nbr 192.168.16.19 iidbQ un/rely 0/1 peerQ
un/rely 0/0 serno 1-4
EIGRP: Sending UPDATE on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x1, Seq 23/0 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
1-4
```

示例 7-44 中显示了路由器 Shepard 正在接收来自路由器 Grissom 的 Hello 数据包。它也显示了路由器 Shepard 正在试图发送更新消息给路由器 Grissom，而路由器 Grissom 却不确认它们。这样进行了第 16 次重试后，将显示一条“重传尝试次数限制超出”（Retransmission retry limit exceeded）的消息。这个超出限制的计数可以从路由器的邻居表中看出，因为所显示的路由器 Grissom 的 uptime 时间都较低——一旦超过重传尝试次数的限制，路由器 Grissom 将从路由器 Shepard 的邻居表中删除。但是，由于路由器 Shepard 依然可以接收到来自路由器 Grissom 的 Hello 数据包，所以路由器 Grissom 又将在路由器 Shepard 的邻居表中迅速地再次出现，这个处理过程又将再次开始。

示例 7-45 显示了在路由器 Shepard 上调试命令 **debug eigrp neighbors** 的输出信息。这个命令没有指定具体的 IP 地址，而是改为显示 EIGRP 的邻居事件了。在这里，显示了前面段落里描述的邻居事件的两种情况：路由器 Grissom 在超出重传次数限制时被宣告丢失，但一旦收到下一个 Hello 数据包时又立即“找回来”了。

示例 7-45　命令 **debug eigrp neighbors** 显示了邻居事件

```
Shepard#debug eigrp neighbors
EIGRP Neighbors debugging is on
Shepard#
EIGRP: Retransmission retry limit exceeded
EIGRP: Holdtime expired
EIGRP: Neighbor 192.168.16.19 went down on Ethernet0
EIGRP: New peer 192.168.16.19
EIGRP: Retransmission retry limit exceeded
EIGRP: Holdtime expired
EIGRP: Neighbor 192.168.16.19 went down on Ethernet0
EIGRP: New peer 192.168.16.19
```

虽然示例 7-44 中显示正在向路由器 Grissom 发送更新数据包，但是在示例 7-46 的路由器 Grissom 上，通过对 EIGRP 数据包的观察来看，这台路由器并没有收到那些发送给它的数据包。因为路由器 Grissom 可以和路由器 Cooper 成功地交换 Hello 数据包，因而路由器 Grissom 的 EIGRP 进程肯定是在运行的。因此怀疑是路由器 Grissom 的以太网接口发生故障了。在示例 7-47 中，观察路由器的配置文件，发现在以太网接口 E0 处配置了一个作为入站过滤器的访问列表。

示例 7-46　路由器 Grissom 正在和路由器 Cooper 通过接口 S0 交换 Hello 数据包，并正在从接口 E0 发送出 Hello 数据包。但是，路由器 Grissom 却没有在接口 E0 上收到任何 EIGRP 的数据包

```
Grissom#debug eigrp packets
EIGRP Packets debugging is on
     (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK)
Grissom#
EIGRP: Sending HELLO on Serial0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Received HELLO on Serial0 nbr 192.168.16.42
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Sending HELLO on Ethernet0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Sending HELLO on Serial0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Received HELLO on Serial0 nbr 192.168.16.42
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Sending HELLO on Ethernet0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Sending HELLO on Serial0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Sending HELLO on Ethernet0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Received HELLO on Serial0 nbr 192.168.16.42
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Sending HELLO on Serial0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Sending HELLO on Ethernet0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
```

示例 7-47　入站访问列表正在拒绝 EIGRP 数据包

```
interface Ethernet0
 ip address 192.168.16.19 255.255.255.240
 ip access-group 150 in
!
!
access-list 150 permit tcp any any established
```

（待续）

```
access-list 150 permit tcp any host 192.168.16.238 eq ftp
access-list 150 permit tcp host 192.168.16.201 any eq telnet
access-list 150 permit tcp any host 192.168.16.230 eq pop3
access-list 150 permit udp any any eq snmp
access-list 150 permit icmp any 192.168.16.224 0.0.0.15
```

当在路由器 Grissom 的 E0 接口上收到 EIGRP 的数据包时，这些数据包首先要通过访问列表 150 的过滤。由于这些数据包和访问列表中的任何一项都不匹配，因此它们被丢弃。这个问题可以通过在访问列表后附加一条匹配条目来解决（参见示例 7-48）：

```
access-list 150 permit eigrp 192.168.16.16 0.0.0.15 any
```

示例 7-48 当在访问列表后增加一个条目来允许 EIGRP 数据包通过时，路由器 Grissom 的邻居表和路由表显示出了它现在拥有可达所有子网的路由

```
Grissom#show ip eigrp neighbors
IP-EIGRP neighbors for process 75
H   Address                 Interface    Hold Uptime    SRTT    RTO    Q    Seq
                                         (sec)          (ms)          Cnt  Num
2   192.168.16.17           Et0             10 00:06:20    4    200    0    41
1   192.168.16.18           Et0             14 00:06:24   15    200    0    85
0   192.168.16.42           Se0             10 06:22:56   22    200    0    12
Grissom#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
   192.168.16.0/24 is variably subnetted, 6 subnets, 2 masks
C     192.168.16.40/30 is directly connected, Serial0
D     192.168.16.36/30 [90/2195456] via 192.168.16.18, 00:06:27, Ethernet0
C     192.168.16.16/28 is directly connected, Ethernet0
D     192.168.16.224/28 [90/2195456] via 192.168.16.42, 00:06:12, Serial0
D     192.168.16.192/28 [90/2323456] via 192.168.16.18, 00:06:27, Ethernet0
D     192.168.16.128/28 [90/307200] via 192.168.16.17, 00:06:12, Ethernet0
Grissom#
```

7.5.2 “卡”在活动状态的邻居

当本地路由器的一条路由变为活动状态并且向它的邻居路由器发送查询时，在本地路由器收到每个查询的答复之前，这条路由将一直保持活动状态。但是如果一个邻居失效或其他无效的情况发生将没有办法作出答复，那么究竟会发生什么呢？答案是这条路由将会永久地停留在活动状态。活动计时器和 SIA-Retransmit 计时器将设计用来防止这种情况的发生。当发送一个查询时，就会设置活动计时器和 SIA-Retransmit 计时器。如果路由器的 IOS 软件（IOS 版本早于 12.2[4.1]）不支持 SIA-Retransmit 计时器，则只能使用活动计时器。如果在收到查询的答复消息之前，活动计时器超时了，这条路由就被宣告“卡”在活动状态（stuck-in-active），这个邻居也就被推断为失效，并且从邻居表中刷新掉。[1]SIA 路由和任何其他经过这个邻居的路由也都会从路由表中删除。DUAL 算法将会认为这个邻居已经答复了一个含有无穷大度量的消息。

事实上，这一系列的事件应该从来不会发生。在活动计时器超时很久以前，丢失的 Hello

[1] 正如前面所提及的，缺省的活动计时时间是 3min，并且可以通过命令 **timer active-time** 来更改。

数据包就应该识别出一个无效的邻居了。

但是在一个大型的 EIGRP 网络中，究竟发生了什么，使一个查询消息就像电池广告中的小兔子一样，一直保持向前继续？回忆一下，查询会引起扩散计算的不断增大，反之，答复将会引起扩散计算的不断减小，如图 7-6 所示。这样，查询最终将必然会到达网络的边界，而答复最终也必然开始往回收缩，但是，如果扩散计算的直径增大到足够大，活动计时器将可能在收到所有的答复前超时。结果，从邻居表中刷新掉一个合法的邻居将明显会带来网络的不稳定。

当一个邻居神秘地从邻居表中消失后，随后又重新出现，或者用户抱怨总是断断续续地不能到达目的地时，SIA 路由可能就是故障所在了。检查路由器的错误记录日志是找出是否出现 SIA 路由的一个好途径，参见示例 7-49 所示。

示例 7-49 这个错误记录日志中的最后一条记录显示了一条 SIA 消息

```
Gagarin#show logging
Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
    Console logging: level debugging, 3369 messages logged
    Monitor logging: level debugging, 0 messages logged
    Trap logging: level informational, 71 message lines logged
    Buffer logging: level debugging, 3369 messages logged
Log Buffer (4096 bytes):
   ...
   ...
   ...
DUAL: dual_rcvupdate(): 10.51.1.0/24 via 10.1.2.1 metric 409600/128256
DUAL: Find FS for dest 10.51.1.0/24. FD is 4294967295, RD is 4294967295 found
DUAL: RT installed 10.51.1.0/24 via 10.1.2.1
DUAL: Send update about 10.51.1.0/24. Reason: metric chg
DUAL: Send update about 10.51.1.0/24. Reason: new if
DUAL: dual_rcvupdate(): 10.52.1.0/24 via 10.1.2.1 metric 409600/128256
DUAL: Find FS for dest 10.52.1.0/24. FD is 4294967295, RD is 4294967295 found
%DUAL-3-SIA: Route 10.11.1.0/24 stuck-in-active state in IP-EIGRP 1. Cleaning up
Gagarin#
```

当追踪 SIA 路由产生的原因时，应该仔细关注路由器的拓扑结构表。如果路由处于活动状态，那么就应该注意邻居路由器仍然没有收到查询内容。例如，在示例 7-50 中显示了一个含有几条处于活动状态的路由的拓扑结构表。注意，这里大多数的路由已经有 15s 的时间处于活动状态了，而另一个路由（10.6.1.0）进入活动状态则已经有 41s 的时间了。

示例 7-50 这个拓扑结构表显示了几个处于活动状态的路由，它们都在等待来自邻居路由器 10.1.2.1 的答复

```
Gagarin#show ip eigrp topology
IP-EIGRP Topology Table for process 1
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply
       r - Reply Status
A 10.11.1.0/24, 0 successors, FD is 3072128000, Q
    1 replies, active 00:00:15, query-origin: Local origin
    Remaining replies:
        via 10.1.2.1, r, Ethernet0
A 10.10.1.0/24, 0 successors, FD is 3584128000, Q
    1 replies, active 00:00:15, query-origin: Local origin
    Remaining replies:
        via 10.1.2.1, r, Ethernet0
A 10.9.1.0/24, 0 successors, FD is 4096128000, Q
```

（待续）

```
    1 replies, active 00:00:15, query-origin: Local origin
    Remaining replies:
        via 10.1.2.1, r, Ethernet0
A 10.2.1.0/24, 1 successors, FD is Inaccessible, Q
    1 replies, active ve 00:00:15, query-origin: Local origin
    Remaining res:
        via 10.1.2.1, r, Ethernet0
P 10.1.2.0/24, 1 successors, FD is 281600
        via Connected, Ethernet0
A 10.6.1.0/24, 0 successors, FD is 3385160704, Q
    1 replies, active 00:00:41, query-origin: Local origin
    Remaining replies:
        via 10.1.2.1, r, Ethernet0
A 10.27.1.0/24, 0 successors, FD is 3897160704, Q
--More-
```

也注意到在每个条目中，邻居路由器 10.1.2.1 都带有一个答复状态标记（r），这表明该邻居的答复仍然没有被收到。邻居路由器本身或者和邻居路由器相连的链路可能没有问题，但是在网络拓扑中该信息指出了应该将追查继续下去的方向。

通常在一个大型的 EIGRP 网络中引起 SIA 的原因是网络拥塞严重、数据链路带宽较低以及路由器内存过低或 CPU 利用率负荷过大，等等。如果这些有限的资源必须处理数量很大的查询的话，这个问题将会进一步恶化。

冒然地调整接口上的带宽参数可能会引起另外的 SIA 路由。回忆一下在设计 EIGRP 网络时，EIGRP 使用的带宽一般不超过链路可用带宽的 50%。这个限制意味着 EIGRP 的调节是和所配置的带宽相关联的。如果试图在处理路由选择时人为地降低带宽，EIGRP 进程所需求的带宽将可能会极度缺乏。假如运行的是 IOS 软件的 11.2 版本或后续的版本，则可以使用命令 **ip bandwidth-percent eigrp** 来调整带宽使用的百分比。

例如，假设一个接口和一个带宽为 56kbit/s 的串行链路相连，但是链路带宽被设置成了 14kbit/s。EIGRP 协议应该限制自己使用的带宽应在这个数值的 50%之内，或者说就是在 7kbit/s 之内。示例 7-51 中的命令将把 EIGRP 协议使用的带宽百分比调整到 200%，即 14kbit/s 的 200%，也就是实际 56kbit/s 链路带宽的 50%：

示例 7-51 路由器的配置调整了 EIGRP 占用的配置带宽的百分比

```
interface Serial 3
 ip address 172.18.107.210 255.255.255.240
 bandwidth 14
 ip bandwidth-percent eigrp 1 200
```

也可以使用命令 **timers active-time** 来增大活动计时器的周期，这样在某些情况下可以帮助避免 SIA 路由，但是采取这种办法应当仔细考虑对网络路由再次收敛的影响。

SIA-Retransmit 计时器是一个新的计时器，它和两个新的 EIGRP 数据包类型——SIA 查询和 SIA 答复一起帮助使 SIA 减少到最少，并对响应查询确实有问题的链路上的邻居进行重置。

考虑图 7-39 中的网络。对于路由器 Mercury，EIGRP 路由会引导到达网络 172.16.100.0 的流量经过路由器 Apollo。路由器 Vostok 不是一台可行后继路由器，因为从路由器 Vostok 到网络 172.16.100.0 的度量值太大。路由器 Vostok 到达 172.16.100.0 的流量会引导经过路由器 Mercury 和 Apollo。路由器 Soyuz 不是一台可行后继路由器，因为从 Soyuz 到达网络 172.16.100.0 的度量值太大了。

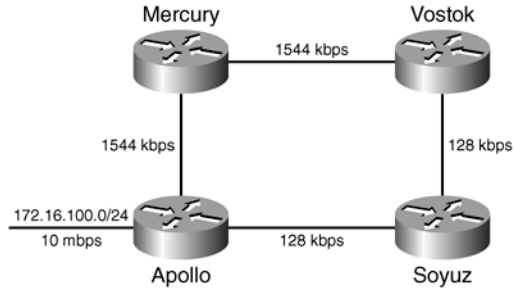


图 7-39　路由器 Mercury 并不把路由器 Vostok 作为到达网络 172.16.100.0 的一台可行后继路由器

如图 7-40 所示，当路由器 Mercury 和 Apollo 之间的链路发生故障时，路由器 Mercury 会把地址 172.16.100.0（和经过邻居 Apollo 学习到的任何其他的地址）变为活动状态，并发送一个查询到路由器 Vostok。路由器 Vostok 也会把该地址置为活动状态，并发送一个查询到路由器 Soyuz。这时活动计时器将被设置。另外，SIA-Retransmit 计时器也会被设置。SIA-Retransmit 计时器设置成活动计时器的数值的一半，通常是 90s。

在 SIA-Retransmit 计时器超时以后，路由器 Mercury 发送一个 SIA 查询到路由器 Vostok。路由器 Vostok 发送一个 SIA 答复到路由器 Soyuz。路由器 Vostok 将会发送一个 SIA 答复响应来自路由器 Mercury 的 SIA 查询。路由器 Mercury 会重置活动计时器和 SIA-Retransmit 计时器。在接收 SIA 答复消息时这些路由器会发送最多 3 个 SIA 查询（假定从始发地址查询的地方没有收到答复消息），然后才会重置一台邻居路由器。因此在一台邻居路由器响应 SIA 查询的时候（大约 6min，假定缺省的活动计时器为 180s），它不应该被宣告为 stuck-in-active 和重置。这对于在一个大型网络中对查询作出响应已经给了足够的时间了。

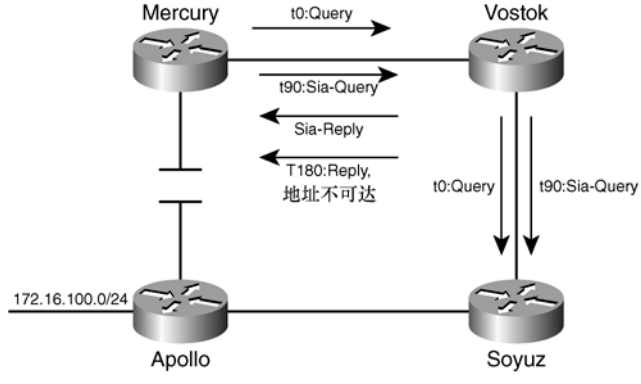


图 7-40　SIA 查询和 SIA 答复用来避免 SIA 情况

但是，在路由器 Vostok 到 Soyuz 之间的链路上会存在一个问题，虽然它允许使用足够多的 Hello 数据包用来保持邻居路由器的活动状态，但是在 SIA-Retransmit 的计时周期内还没有收到来自路由器 Vostok 的 SIA 答复。如果在 SIA 查询的 90s 计时时间内没有收到 SIA 答复，从而对始发地址查询的响应也不会收到，路由器 Vostok 将会重置邻居路由器 Soyuz，并答复路由器 Mercury 始发的查询宣告那个地址不可到达。

SIA-Retransmit 计时器可以完成两件事情。如果邻居路由器正在对 SIA 查询进行响应，那么将会给大型网络更多的时间来对地址的查询作出响应。如果邻居路由器没有进

行响应，那么邻居路由器将被重置。只有从它的邻居没有收到响应的路由器才会重置它的邻接关系。在引入 SIA-Retransmit 计时器之前，即使网络下游的某处出现了问题，在活动计时器超时后如果还没有收到对一个活动查询的响应，那么路由器将会重置它的邻居邻接关系。

一个好的网络设计应该是解决像 SIA 路由这类网络不稳定性的最好方法。全面考虑使用灵活的地址分配、路由过滤、缺省路由和路由汇总，在一个大型的网络中创建一些边界来限制扩散计算的大小和范围。第 13 章将包含一个这种网络设计的例子。

7.6 展　　望

当比较 EIGRP 协议和 OSPF 协议时，人们经常说 EIGRP 协议的优势在于它的配置比较简单。这种看法在很多网络的情况中是正确的，但是本章故障处理一节的讲述显示了当网络的规模增大时，对划分 EIGRP 网络的拓扑将做出更多地努力。相反地，正如下一章所描述的，非常复杂的 OSPF 在配置一个大型网络时反而显得简单了。

7.7 总结表：第 7 章命令总结

命令	描述
accept-lifetime *start-time*{**infinite**|*end-time*|**duration** *seconds*}	设置一个时间段，用来指定钥匙链上的认证钥匙可被接受的有效时间
auto-summary	在网络边界上打开或关闭自动路由汇总功能，这个命令缺省的配置为打开
bandwidth *kilobits*	在接口上指定带宽参数，单位是 kbit/s。在一些路由选择协议中用来计算度量值，但它不影响数据链路实际的带宽
debug eigrp packets	显示 EIGRP 数据包的活动行为
debug ip eigrp neighbor *process-id address*	在命令 **debug eigrp packets** 基础上增加一个过滤器，告诉路由器只显示选定的进程 ID 和邻居的 IP 数据包
delay *tens-of-microseconds*	在接口上指定延迟参数，单位是 10μs。
eigrp stub {connected|redistributed|static|summary| receive-only}	配置一台分支路由器作为 EIGRP 末梢
ip authentication key-chain eigrp *process-id key-chain*	在一个运行 EIGRP 协议的接口上配置一个钥匙链，并指定一个钥匙链所使用的名字
ip authentication mode eigrp *process-id* **md5**	在一个接口上启动 EIGRP 协议使用的认证类型
ip bandwidth-percent eigrp *process-id percent*	配置 EIGRP 协议所使用的带宽百分比，缺省配置是 50%
ip hello-interval eigrp *process-id seconds*	配置 EIGRP 的 Hello 数据包的时间间隔
ip hold-time eigrp *process-id seconds*	配置 EIGRP 的抑制时间
ip summary-address eigrp *process-id address mask*	配置路由器发送一个汇总的 EIGRP 通告
key *number*	指定钥匙链上的一个钥匙
key chain *name-of-chain*	指定一组认证钥匙
key-string *text*	指定钥匙使用的认证字符串或口令
metric weights *tos k1 k2 k3 k4 k5*	指定在 IGRP 和 EIGRP 协议中计算复合度量值时，对带宽、负载、延迟和可靠性等参数所使用的权重

续表

命令	描述
network *network-number*	指定覆盖一个或多个运行 IGRP、EIGRP 或 RIP 协议进程接口的网络地址
passive-interface *type number*	使一个接口不再发送广播或组播的路由选择更新
router eigrp *process-id*	在路由器上启动 EIGRP 路由选择进程
send-lifetime *start-time*{**infinite**|*end- time*|**duration** *seconds*}	设置一个时间段，用来指定钥匙链上的认证钥匙可被发送的有效时间
show ip eigrp neighbors [*type number*]	用来显示 EIGRP 的邻居表
show ip eigrp topology [*process-id*|[[*ip address*]*mask*]]	用来显示 EIGRP 的拓扑结构表
timers active-time {*minutes*|**disabled**}	改变或关闭缺省的 3min 的活动状态计时
traffic-share {balanced|min}	指定 IGRP 协议或 EIGRP 协议路由选择进程是否使用非等价负载均衡或只使用等价负载均衡
variance *multiplier*	指定一个倍数来表示一条路由与最小代价路径的度量值的差别程度，确定是否可以依然包含在非等价负载均衡“组”中

7.8　复 习 题

1．EIGRP 协议是一个距离矢量协议还是一个链路状态路由选择协议？

2．EIGRP 协议在一条链路上使用的可配置最大带宽是多少？这个带宽百分比可以更改吗？

3．EIGRP 协议和 IGRP 协议在计算复合度量时有什么不同？

4．EIGRP 协议的 4 个基本部件是什么？

5．在 EIGRP 协议的上下文环境中，术语“可靠的分发（reliable delivery）”是什么意思？有哪两种方法可以确保 EIGRP 数据包的可靠分发？

6．有什么机制可以确保路由器正在接收的是最新的路由条目？

7．EIGRP 协议使用的组播 IP 地址是什么？

8．EIGRP 协议使用的数据包类型是什么？

9．缺省情况下，EIGRP 协议发送 Hello 数据包的时间间隔是多少？

10．缺省的抑制时间是多少？

11．邻居表和拓扑结构表的不同之处是什么？

12．什么是可行距离？

13．什么是可行性条件？

14．什么是可行后继路由器？

15．什么是后继路由器？

16．处于活动状态的路由和处于被动状态的路由有什么不同之处？

17．引起一个被动状态的路由变成活动状态的条件是什么？

18．引起一个活动状态的路由变成被动状态的条件是什么？

19．Stuck-in-active 是什么意思？

20．子网划分和地址聚合有什么不同之处？

7.9 配置练习

1．写出图 7-41 中路由器 A、B 和 C 的 EIGRP 配置，并使用进程 ID 5。

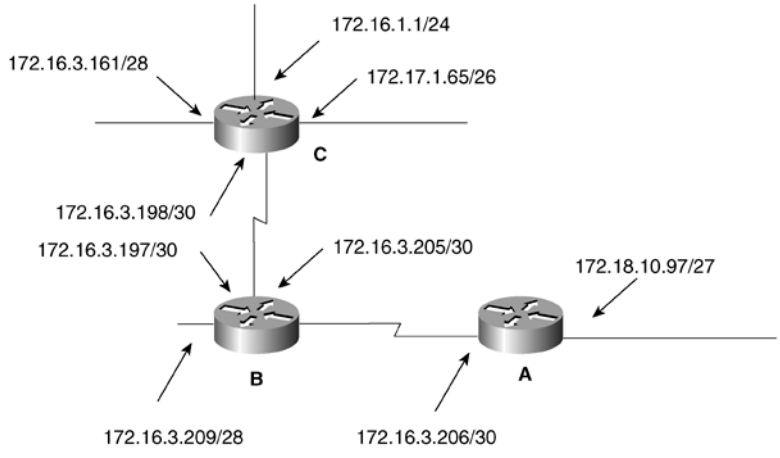


图 7-41 配置练习 1 和练习 2 的网络

2．在图 7-41 中，连接路由器 A 和路由器 B 的串行接口都是 S0 接口。配置这两台路由器之间的认证，假设从今天开始，有两天的时间使用第一个钥匙。然后配置第二个钥匙，在第一个钥匙使用过后 30 天开始使用。

在图 7-42 中增添了路由器 D。将这台路由器添加到配置练习 2 所写的配置中。

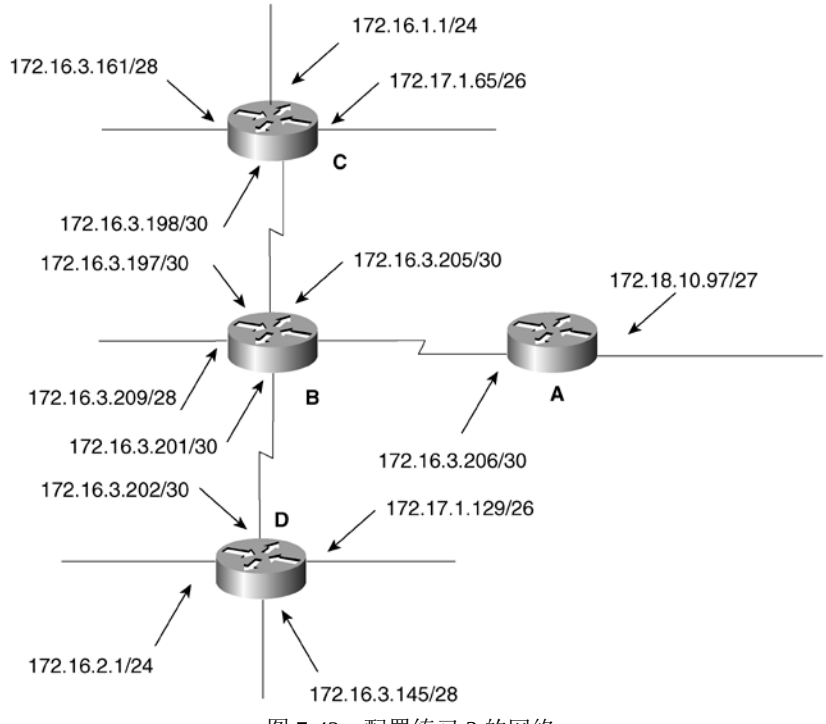


图 7-42 配置练习 3 的网络

3．在图 7-43 中增添了路由器 F，配置这台路由器，在与配置练习 2 和练习 3 中配置的路由器之间运行 EIGRP 协议。

4．在图 7-43 的网络中任何可能的地方配置路由汇总。

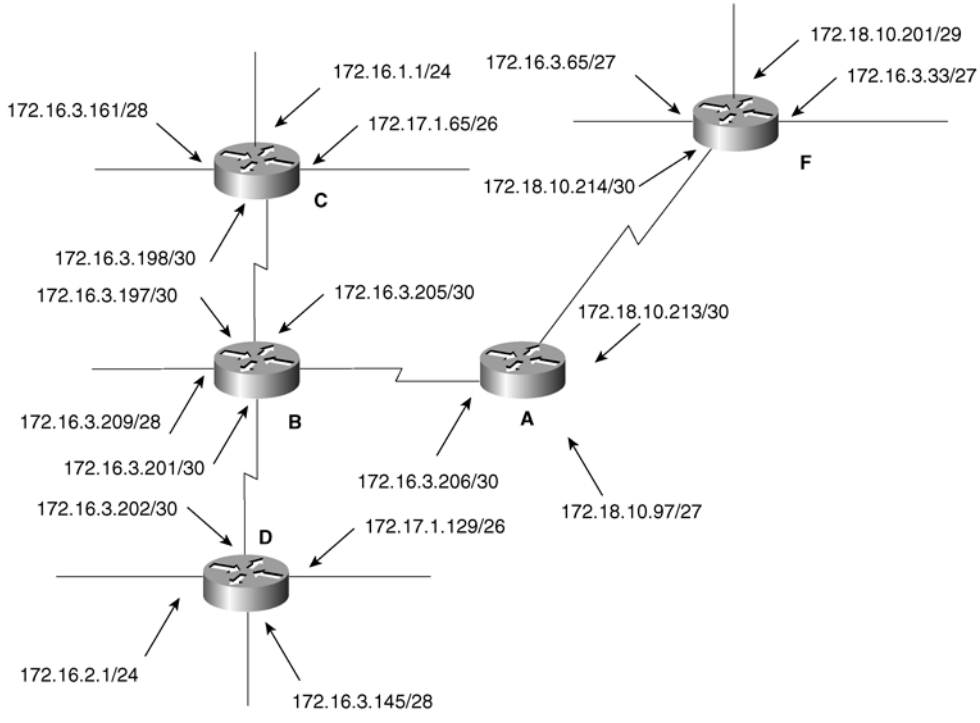


图 7-43　配置练习 4 和练习 5 的网络

7.10　故障排除练习

1．表 7-7 显示了在图 7-44 中的每个接口上使用 **show interface** 命令显示的数值。哪一台路由器将作为路由器 F 到达子网 A 的后继路由器？

表 7-7　　使用 **show interface** 命令显示的图 7-44 中所有接口的度量值

路由器	接口	带宽（kbit/s）	延迟(μs)
A	E0 F0 T0 S0 S1	10000 100000 16000 512 1544	1000 100 630 20000 20000
B	T0 S0	16000 1544	630 20000
C	E0 S0	10000 1544	1000 20000
D	E0 F0 S0	10000 100000 1544	1000 100 20000

续表

路由器	接口	带宽 BW(k)	延迟 DLY(μs)
E	E0 S0	10000 1544	1000 20000
F	F0 S0	100000 512	100 20000
G	E0 E1 F0 S0	10000 10000 100000 56	1000 1000 100 20000

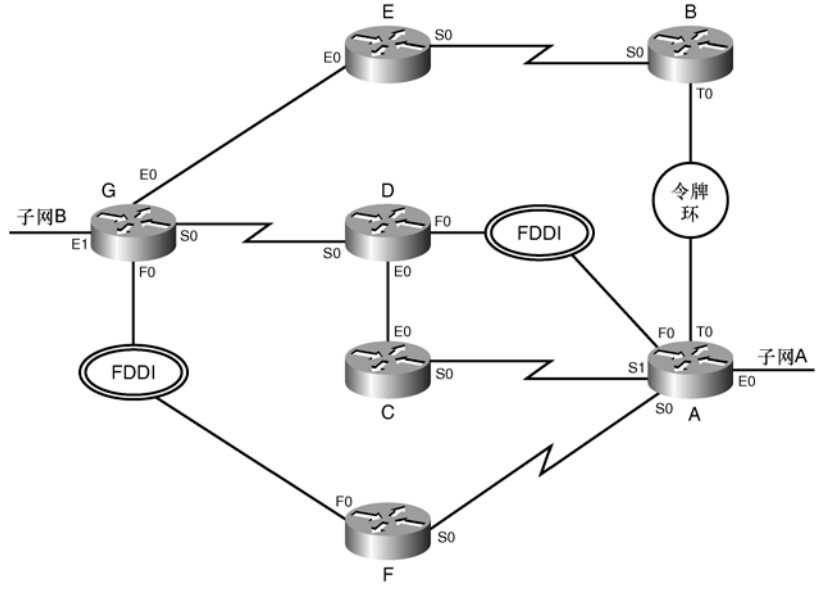


图 7-44　故障排除练习 1～5 的网络

2．在图 7-44 中，路由器 C 到达子网 A 的可行距离是什么？

3．在图 7-44 中，路由器 G 到达子网 A 的可行距离是什么？

4．在图 7-44 中，在路由器 G 的拓扑结构表中，有哪些路由器是作为可行后继路由器的？

5．在图 7-44 中，路由器 A 到达子网 B 的可行距离是什么？

本章主要包括以下主题：

- OSPF 协议的基本原理与实现；
- OSPF 协议的配置；
- OSPF 协议的故障诊断。



第 8 章

开放最短路径优先协议（OSPFv2）

开放最短路径优先（Open Shortest Path First，OSPF）协议是由 Internet 工程任务组（Internet Engineering Task Force，IETF）开发的路由选择协议，用来替代存在一些问题的 RIP 协议。现在，OSPF 协议是 IETF 组织建议使用的内部网关协议（IGP）。OSPF 协议是一个链路状态协议，正如它的命名所描述的，OSPF 使用 Dijkstra 的最短路径优先（SPF）算法，而且是开放的。这里所说的开放是指它不属于任何一个厂商或组织所私有。OSPF 协议的发展经过了几个 RFC，所有这些相关的 RFC 都是由 John Moy 撰写的。RFC 1131 详细说明了 OSPF 协议版本 1，这个版本从来没有在实验平台以外使用过。OSPF 协议版本 2，也就是目前 IPv4 协议仍然使用的版本，最初是在 RFC 1247 中说明的，最新是在 RFC 2328 中说明的。

像所有的链路状态协议一样，OSPF 协议和距离矢量协议相比，一个主要的改善在于它的快速收敛，这使得 OSPF 协议可以支持更大型的网络，并且不容易受到有害路由选择信息的影响。OSPF 协议的其他一些特性有：

- 使用了区域的概念，这样可以有效地减少路由选择协议对路由器的 CPU 和内存的占用；划分区域还可以降低路由选择协议的通信量，这使构建一个层次化的网络拓扑成为可能；
- 完全无类别地处理地址问题，排除了不连续子网这样的有类别路由选择协议的问题；
- 支持无类别路由表查询、VLSM 和用来进行有效地址管理的超网技术；
- 支持无大小限制的、任意的度量值；

- 支持使用多条路径的效率更高的等价负载均衡；[1]
- 使用保留的组播地址来减小对不宣告 OSPF 的设备的影响；
- 支持更安全的路由选择认证；
- 使用可以跟踪外部路由的路由标记。

OSPF 协议也支持具有服务类型（Type of Service，ToS）的路由选择能力，但是它从来没有被广泛地实施过。基于这个原因，RFC 2328 已经在 OSPF 协议中删除了该 ToS 路由选择选项。

8.1 OSPF 的基本原理与实现[2]

从一个非常概括的角度来看，OSPF 协议的操作是比较容易解释的：

1．宣告 OSPF 的路由器从所有启动 OSPF 协议的接口上发出 Hello 数据包。如果两台路由器共享一条公共数据链路，并且能够相互成功协商它们各自 Hello 数据包中所指定的某些参数，那么它们就成为了邻居（Neighbor）。

2．邻接关系（Adjacency），可以想象成为一条点到点的虚链路，它是在一些邻居路由器之间构成的。OSPF 协议定义了一些网络类型和一些路由器类型的邻接关系。邻接关系的建立是由交换 Hello 信息的路由器类型和交换 Hello 信息的网络类型决定的。

3．每一台路由器都会在所有形成邻接关系的邻居之间发送链路状态通告（Link State Advertisement，LSA）。LSA 描述了路由器所有的链路、接口、路由器的邻居以及链路状态信息。这些链路可以是到一个末梢网络（stub network，是指没有和其他路由器相连的网络）的链路、到其他 OSPF 路由器的链路、到其他区域网络的链路，或是到外部网络（从其他的路由选择进程学习到的网络）的链路。由于这些链路状态信息的多样性，OSPF 协议定义了许多 LSA 类型。

4．每一台收到从邻居路由器发出的 LSA 的路由器都会把这些 LSA 记录在它的链路状态数据库当中，并且发送一份 LSA 的拷贝给该路由器的其他所有邻居。

5．通过 LSA 泛洪扩散到整个区域，所有的路由器都会形成同样的链路状态数据库。

6．当这些路由器的数据库完全相同时，每一台路由器都将以其自身为根，使用 SPF 算法来计算一个无环路的拓扑图，以描述它所知道的到达每一个目的地的最短路径（最小的路径代价）。这个拓扑图就是 SPF 算法树。

7．每一台路由器都将从 SPF 算法树中构建出自己的路由表。[3]

当所有的链路状态信息洪泛到区域内的所有路由器上，并且邻居检验它们的数据库也相同（也就是说，链路状态数据库已经同步），从而成功创建路由表时，OSPF 协议就变成了一个“安静”的协议。邻居之间交换的 Hello 数据包称为 keepalive，并且每隔 30min 重传一次

---

[1] 更准确地说，RFC 建议称为等价多路径——多条等价路径的发现和使用，但是 RFC 并没有指明路由选择协议如何路由转发单个数据包通过这些多条不同的路径。在 Cisco 路由器上，OSPF 的实现执行的是前面章节描述过的等价负载均衡。

[2] 由于 OSPF 协议的术语和概念的相互关系，本章在完整地描述它们的定义之前将会频繁地使用这些术语和概念。建议读者多阅读几遍本章的内容，而不是仅仅阅读一遍，以便确保对 OSPF 协议的操作有一个完全的理解。复习第 4 章中的 4.3 节也是很有帮助的。

[3] 受到路由过滤影响的是利用链路状态数据库计算路由的基本过程，而不是邻居之间交换路由的基本过程。如需了解更详细的内容，请参见第 13 章的内容。

LSA。如果网络拓扑稳定，那么网络中将不会有什么活动或行为发生。

8.1.1 邻居和邻接关系

在发送任何 LSA 通告之前，OSPF 路由器都必须首先发现它们的邻居路由器并建立起邻接关系。邻居路由器，连同每一台邻居路由器所在的链路（接口）和维护邻居路由器的一些必要的其他信息都被记录在一个邻居表里，参见示例 8-1 所示。

示例 8-1 邻居表记录了所有宣告 OSPF 协议的邻居路由器

```
Monet#show ip ospf neighbor
Neighbor ID     Pri   State     Dead Time   Address         Interface
192.168.30.70     1   FULL/DR   00:00:34    192.168.17.73   Ethernet0
192.168.30.254    1   FULL/DR   00:00:34    192.168.32.2    Ethernet1
192.168.30.70     1   FULL/BDR  00:00:34    192.168.32.4    Ethernet1
192.168.30.30     1   FULL/ -   00:00:33    192.168.17.50   Serial0.23
192.168.30.10     1   FULL/ -   00:00:32    192.168.17.9    Serial1
192.168.30.68     1   FULL/ -   00:00:39    192.168.21.134  Serial2.824
192.168.30.18     1   FULL/ -   00:00:30    192.168.21.142  Serial2.826
192.168.30.78     1   FULL/ -   00:00:36    192.168.21.170  Serial2.836
```

一台 OSPF 路由器对其他 OSPF 路由器的跟踪需要每台路由器都提供一个路由器 ID（Router ID），路由器 ID 在 OSPF 区域内惟一标识一台路由器的 IP 地址。Cisco 路由器通过下面的方法得到它们的路由器 ID：

（1）如果使用 **router-id** 命令手工配置 Router ID，就使用 Router ID。

（2）如果没有手工配置 Router ID，路由器就选取它所有的四环（loopback）接口上数值最高的 IP 地址。

（3）如果路由器没有配置 IP 地址的 loopback 接口，那么路由器将选取它所有的物理接口上数值最高的 IP 地址。用作路由器 ID 的接口不一定非要运行 OSPF 协议。

使用 loopback 接口作为路由器 ID 有两个好处：

- loopback 接口比任何其他物理接口更稳定。一旦路由器启动成功，这个环回接口就处于活动状态，只有整个路由器失效时它才会失效。
- 网络管理员在预先分配和识别作为路由器 ID 的地址时有更多的回旋余地。

在 Cisco 路由器上，即使路由器的这个用作路由器 ID 的物理接口随后失效或被删除，OSPF 协议也会继续使用原来的物理接口作为路由器 ID（参见本章后面的 8.2.2 小节的内容）。因此，loopback 接口的稳定性只是一个次要的优点，loopback 接口的一个主要好处在于它具有更好控制路由器 ID 的能力。

OSPF 路由器利用 Hello 数据包通告它的路由器 ID 来开始建立和邻居的关系。

1. Hello 协议

Hello 协议服务于以下几个目的：

- 它是发现邻居路由器的方法；
- 在两台路由器成为邻居之前，需要通告这两台路由器必须相互认可的几个参数；
- Hello 数据包在邻居路由器之间担当 keepalive 的角色；
- 它确保了邻居路由器之间的双向通信；

- 它用来在一个广播网络或非广播多路访问（NBMA）网络上选取指定路由器（Designated Router，DR）和备份指定路由器（Backup Designated Router，BDR）。

宣告 OSPF 的路由器周期性地从启动 OSPF 协议的每一个接口上发送 Hello 数据包。该周期性的时间段称为 Hello 时间间隔（HelloInterval），它的配置是基于路由器的每一个接口的。在 Cisco 路由器上，对于广播型网络使用的缺省 Hello 时间间隔是 10s，对于非广播型网络缺省是 30s。这个值可以通过命令 **ip ospf hello-interval** 来更改。如果一台路由器在一个称为路由器无效时间间隔（RouterDeadInterval）的时间段内还没有收到来自邻居的 Hello 数据包，那么它将宣告它的邻居路由器无效。在 Cisco 路由器中，路由器无效时间间隔的缺省值是 Hello 时间间隔的 4 倍，并且这个值可以通过命令 **ip ospf dead-interval** 来更改。[1]

每一个 Hello 数据包都包含以下信息：

- 始发路由器的路由器 ID（RouterID）；
- 始发路由器接口的区域 ID（AreaID）；
- 始发路由器接口的地址掩码；
- 始发路由器接口的认证类型和认证信息；
- 始发路由器接口的 Hello 时间间隔；
- 始发路由器接口的路由器无效时间间隔；
- 路由器的优先级；
- 指定路由器（DR）和备份指定路由器（BDR）；
- 标识可选性能的 5 个标记位；
- 始发路由器的所有有效邻居的路由器 ID。这个列表仅仅包含这样一些所谓有效的邻居路由器——即在最近的路由器无效时间间隔内，始发路由器接口可以从其接收到 Hello 数据包的那些邻居。

本小节概述了上面列出的多数信息的含义和用法。随后的章节将会详细地讲述指定路由器（DR）、备份指定路由器（BDR）、路由器的优先级和阐明 Hello 数据包的详细格式。当一台路由器从它的邻居路由器收到一个 Hello 数据包时，它将检验该 Hello 数据包携带的区域 ID、认证信息、网络掩码、Hello 间隔时间、路由器无效时间间隔以及可选项的数值是否和接收接口上配置的对应值相匹配。如果它们不匹配，那么该数据包将被丢弃，而且邻接关系也无法建立。

如果所有的参数都匹配，那么这个 Hello 数据包就被认为是有效的。而且，如果始发路由器的路由器 ID 已经在接收该 Hello 数据包的接口的邻居表中列出，那么路由器无效时间间隔计时器将被重置。如果始发路由器的路由器 ID 没有在邻居表中列出，那么就把这个路由器 ID 加入到它的邻居表中。

无论何时，路由器发送一个 Hello 数据包时，都会在这个数据包中列出传送该数据包的链路上所出现的所有邻居的路由器 ID。如果一台路由器收到了一个有效的 Hello 数据包，并在这个 Hello 数据包中发现了自己的路由器 ID，那么这台路由器就认为是双向通信（two-way communication）建立成功了。

[1] 在 RFC 2328 中没有为 HelloInterval 或者 RouterDeadInterval 设置一个必需的值，虽然它建议采用 10s 和 4×HelloInterval。

一旦双向通信成功建立，邻接关系也就可能建立了。然而，正如前面所提及的，并不是所有的邻居路由器都会成为邻接对象。一个邻接关系的形成与否是依赖于和这两台互为邻居的路由器所连网络的类型的。另外，网络类型也影响 OSPF 数据包传送的方式。因此，在讲述邻接关系之前，讲述网络类型是必要的。

2. 网络类型

OSPF 协议定义了以下 5 种网络的类型：

（1）点到点网络（point-to-point）；

（2）广播型网络（broadcast）；

（3）非广播多路访问（NBMA）网络；

（4）点到多点网络（point-to-multipoint）；

（5）虚链路（virtual links）。

- 点到点网络（point-to-point）

点到点网络，像 T1、DS-3 或 SONET 链路，是连接单独一对路由器的。在点到点网络上的有效邻居总是可以形成邻接关系。在这些网络上的 OSPF 数据包的目的地址也总是保留的 D 类地址 224.0.0.5，这个组播地址称为 AllSPFRouters。[1]

- 广播型网络（broadcast）

广播型网络，像以太网、令牌环网和 FDDI，也可以更确切地定义为广播型多址网络，以便区别于 NBMA 网络。广播型网络是多址的网络，因而它们可以连接多于两台的设备。而且由于它们是广播型的，所以连接在这种网络上的所有设备都可以接收到个别传送的数据包。在广播型网络上的 OSPF 路由器正如下面“指定路由器和备份指定路由器”中所讲述的，会选举一台指定路由器和一台备份指定路由器。Hello 数据包像所有始发于 DR 和 BDR 的 OSPF 数据包一样，是以组播方式发送到 AllSPFRouters（目的地址是 224.0.0.5）的。携带这些数据包的数据帧的目的介质访问控制（MAC）地址是 0100.5E00.0005。其他所有的路由器都将以组播方式发送链路状态更新数据包和链路状态确认数据包（将在后面讲述）到保留的 D 类地址 224.0.0.6，这个组播地址称为 AllDRouters。携带这些数据包的数据帧的目的 MAC 地址是 0100.5E00.0006。

- 非广播多路访问（NBMA）网络

NBMA 网络，像 X.25、帧中继和 ATM 等，可以连接两台以上的路由器，但是它们没有广播数据包的能力。一台在 NBMA 网络上的路由器发送的数据包将不能被其他与之相连的路由器收到。结果是，在这些网络上的路由器有必要增加另外的配置来获得它们的邻居。在 NBMA 网络上的 OSPF 路由器需要选举 DR 和 BDR，并且所有的 OSPF 数据包都是单播的。

- 点到多点网络（point-to-multipoint）

点到多点网络是 NBMA 网络的一个特殊配置，可以被看作是一群点到点链路的集合。在这些网络上的 OSPF 路由器不需要选举 DR 和 BDR，OSPF 数据包以单播方式发送给每一个已知的邻居。

- 虚链路（virtual links）

[1] 这个规则的一个例外是重传的 LSA 数据包，它们在所有的网络类型中都是使用单播方式发送的。这个例外的情况将在后面的“可靠的泛洪扩散：确认”部分讲述。

虚链路将在后面部分讲述，它可以被路由器认为是没有编号的点到点网络的一种特殊配置。在虚链路上 OSPF 数据包是以单播方式发送的。

除了以上那 5 种网络类型外，应该注意的是，所有的网络也都可以归纳到下面两种更普通的网络类型之一：

- **传送网络（Transit Network）**——与两台或两台以上的路由器相连。这种网络仅仅传送那些“只需仅仅通过”的数据包，也就是这样的一些数据包——它们的始发网络和目的网络都不同于当前的传送网络。
- **末梢网络（Stub Network）**[1]——仅仅和一台路由器相连。末梢网络上的数据包总是有一个源地址或者目的地址属于这个末梢网络。也就是说，末梢网络上的所有数据包要么始发于这个末梢网络上的某个设备，要么终止于这个末梢网络上的某个设备。OSPF 协议在末梢网络上通告主机路由（就是网络掩码为 255.255.255.255 的路由）。loopback 接口也可以认为是末梢网络，并当作主机路由来通告。[2]

3. 指定路由器和备份指定路由器

对于 OSPF 协议来说，在多址网络上有关 LSA 的泛洪扩散（flooding，将在后面的章节讲述）方面还存在两个问题：

- 在构建相关路由器之间的邻接关系时，会创建很多不必要的 LSA。假设在一个多址网络上有 n 台路由器，那么就会构成 $n(n-1)/2$ 个邻接关系，如图 8-1。每台路由器都会通告出 $n-1$ 条 LSA 信息到与之存在邻接关系的邻居路由器，再加上 1 个网络 LSA，这样计算的最终结果是，这个网络上将产生出 n^2 个 LSA 通告。
- 多址网络本身的泛洪扩散显得比较混乱。某一台路由器向与它存在邻接关系的所有邻居发出 LSA，同样地，这些邻接的邻居路由器又向与它有邻接关系的邻居的邻居发出这个 LSA，这样将会在同一个网络上创建很多个相同 LSA 的副本。

为了在一个多路访问网络避免这些问题的发生，可以在多路访问网络上选举一台指定路由器。这台指定路由器将完成以下工作：

- 描述这个多路访问网络和 OSPF 区域内其余与其相连的路由器；
- 管理这个多路访问网络上的泛洪扩散过程。

DR 背后的一种概念是广播链路本身被认为是一个“伪节点”，或者虚拟路由器。当 SPF 树进行计算的时候，把链路看作一个节点，与该链路相连的路由器也是连接到这个节点上的。从与伪节点相连的路由器到这个伪节点的代价就是该路由器与这个广播链路相连的接口的出站代价，但是从伪节点到任何与之相连的路由器的代价都为 0。通过这种方式，所有路径的代价都不会受到伪节点的影响。

网络中的每一台路由器都会与 DR 形成一个邻接关系，如图 8-2 所示，DR 在特定的网络 LSA 中表示为一个伪节点。请记住，一台路由器可能是它所连接的其中一个多路访问网络的指定路由器，也可能不是它所连接的另一个多路访问网络的指定路由器。换句话说，指定路由器是路由器接口的特性，而不是整个路由器的特性。

---

[1] 注意，不要把 stub 网络和本章后面章节讲述的 stub 区域的概念相混淆。

[2] 从 IOS 11.3 版本开始，这个缺省的行为可以在 loopback 接口上增加使用命令 **ip ospf network point-to-point** 来改变。这将可以使 loopback 接口的地址作为一个子网地址来通告。

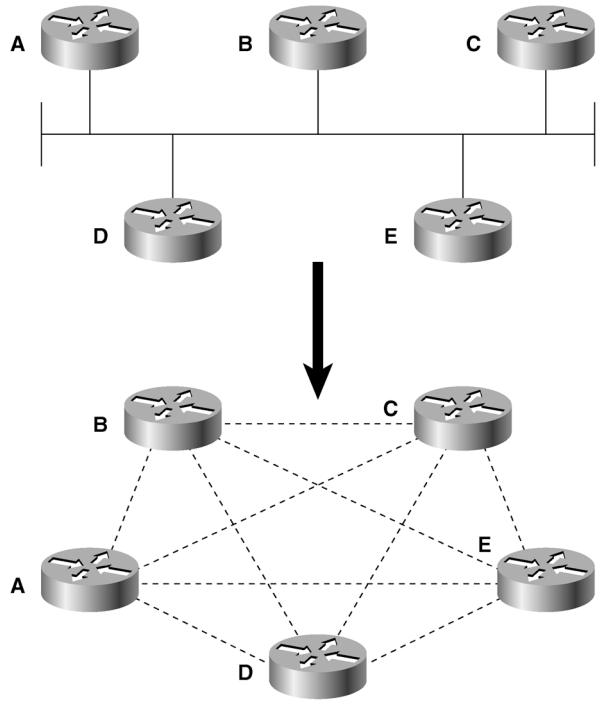


图 8-1 在 OSPF 网络上，如果要在每一台路由器和它的邻居路由器之间形成完全网状的 OSPF 邻接关系，那么这 5 台路由器之间将需要形成 10 个邻接关系；这个网络还将会产生 25 条 LSA 通告

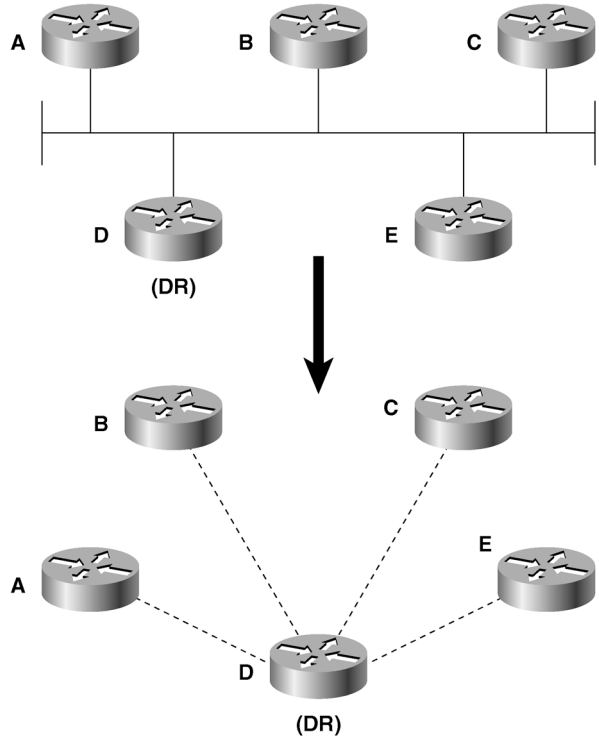


图 8-2 指定路由器描述了一个多址网络。网络上的其他路由器都将和这个指定路由器（DR）构成邻接关系，而不是它们互相之间构成邻接关系

到目前所描述的为止，可以看出关于指定路由器的一个重要问题是，如果一台指定路由器失效了，就必须选取一台新的指定路由器。同时，网络上的所有路由器也要重新建立新的邻接关系，并且网络上所有的路由器必须根据新选出的指定路由器进行同步它们的网络数据库（邻接关系创建过程中的一部分）。当所有上述的过程发生时，网络将无法有效地传送数据包。

为了避免这个问题，在网络上除了选取指定路由器，还再选取一台备份指定路由器（BDR）。这样，网络上所有的路由器将和指定路由器（DR）与备份指定路由器（BDR）同时形成邻接关系。DR 和 BDR 之间也将互相形成邻接关系。这时，如果 DR 失效了，BDR 将成为新的 DR。但是由于网络上其余的路由器已经和 BDR 形成了邻接关系，因此网络可以将无法传送数据的影响降低到最小。

DR 和 BDR 的选取是通过一个接口状态机的方式触发的，接口状态机将在后面讲述。为了能够使选取的处理过程可以进行，需要满足以下一些前提条件：

- 每台路由器的每一个多点访问接口都有一个路由器的优先级（Router Priority），用一个 8 位的无符号整数来表示，范围大小是 0～255。在 Cisco 路由器上，缺省的优先级是 1。基于每一个多点访问的接口都可以通过命令 **ip ospf priority** 来更改。具有 0 优先级的路由器将不能成为 DR 或者 BDR。
- Hello 数据包包含了表示始发路由器指定的路由器优先级的字段，也包含了表示路由器认为可能是 DR 和 BDR 的相关接口的 IP 地址的字段。
- 当一个接口在一个多址网络上开始有效时，它将把它的 DR 和 BDR 的地址设置为 0.0.0.0。同时它也将等待计时器（wait timer）的值设置为等于路由器无效时间间隔（RouterDeadInterval）。
- 在多址网络上已经存在的接口将把 DR 和 BDR 的地址记录入一个接口数据结构表中，接口数据结构表将在后面的章节中讲述。

DR 和 BDR 的选取过程如下描述：

（1）在路由器和它的邻居路由器之间首先建立双向通信（2-way communication），接着检查每台邻居路由器发送的 Hello 数据包的优先级、DR 和 BDR 字段。列出所有具有 DR 和 BDR 选取资格的路由器的列表（也就是说，路由器的优先级要大于 0，并且它的邻居状态至少要是 2-way 的）；接着，所有的路由器将宣称自己是 DR 路由器（Hello 数据包的 DR 字段是它们自身接口的地址）；所有的路由器也将宣称它们自己是 BDR 路由器（Hello 数据包的 BDR 字段是它们自身接口的地址）。除非没有选取资格，路由器计算时也将在这个具有选取资格路由器的列表中包括它本身。

（2）从具有选取资格的路由器的列表中，创建一个还没有宣告为 DR 路由器的所有路由器的子集（宣告自己为 DR 路由器的路由器不能被选取为 BDR 路由器）。

（3）如果在这个子集中的一台或者多台邻居路由器，它们在 Hello 数据包的 BDR 字段包含了它们自己的接口地址，那么具有最高优先级的邻居路由器将被宣告为 BDR 路由器。在优先级相同的条件下，具有最高路由器 ID 的邻居路由器将被选作 BDR 路由器。

（4）如果在这个子集中没有路由器宣称自己是 BDR 路由器，那么具有最高优先级的邻居路由器将被宣告为 BDR 路由器。在优先级相同的条件下，具有最高路由器 ID 的邻居路由器将被选作 BDR 路由器。

（5）如果一台或多台具有选取资格的路由器在 Hello 数据包的 DR 字段包含它们自己的接口地址，那么具有最高优先级的邻居路由器将被宣告为 DR 路由器。在优先级相同的条

件下，具有最高路由器 ID 的邻居路由器将被选作 DR 路由器。

（6）如果没有路由器宣称自己是 DR 路由器，那么新选取的 BDR 路由器将成为 DR 路由器。

（7）如果正在执行计算的路由器是新选取的 DR 或 BDR 路由器，或者它不再是 DR 或 BDR 路由器了，那么将重复以上的 2～6 步骤。

简单地说，当一台 OSPF 路由器有效（active）并去发现它的邻居路由器时，它将去检查有效的 DR 和 BDR 路由器。如果 DR 和 BDR 路由器存在的话，这台路由器将接受已经存在的 DR 和 BDR 路由器。如果 BDR 路由器不存在，将执行一个选取过程，选出具有最高优先级的路由器作为 BDR 路由器。如果存在多台路由器具有相同的优先级，那么在数值上具有最高路由器 ID 的路由器将被选中。如果没有有效的 DR 路由器存在，那么 BDR 路由器将被选举为 DR 路由器，然后再执行一个选取过程选取 BDR 路由器。

这里需要注意的是，路由器的优先级可以影响一个选取过程，但是它不能强制更换已经有效的 DR 或 BDR 路由器。也就是说，在已经选取了 DR 和 BDR 路由器后，如果一台具有更高优先级的路由器变为有效的，那么这台新的路由器将不会替换 DR 或 BDR 路由器的任何一台。因此，在一个多访问网络上，最先初始化启动的两台具有 DR 选取资格的路由器将成为 DR 和 BDR 路由器。

一旦 DR 和 BDR 路由器选取成功，其他的路由器（称为 DRothers）将只和 DR 及 BDR 路由器之间形成邻接关系。所有的路由器将继续以组播方式发送 Hello 数据包到 AllSPFRouters（组播地址是 224.0.0.5），因此它们能够跟踪它们的邻居路由器，但是 DRothers 路由器只以组播方式发送更新数据包到 AllDRouters（组播地址是 224.0.0.6）。只有 DR 和 BDR 路由器去侦听这个地址，反过来，DR 路由器将使用组播地址 224.0.0.5 泛洪扩散更新数据包到 DRothers。

请注意，如果在一个多访问网络上只有惟一的一台具有选取资格的路由器相连，那么这台路由器将成为 DR 路由器，而且在这个网络上没有 BDR 路由器。其他所有的路由器都将只和这台 DR 路由器建立邻接关系。如果没有具有选取资格的路由器和一个多访问网络相连，那么这个网络上将没有 DR 或者 BDR 路由器，而且也不建立任何邻接关系。在这种情况下，网络上所有路由器的邻居状态都将停留在“2-Way”状态（将在后面的“邻居状态机”部分中阐述）。

关于 DR 和 BDR 路由器所需要完成的更多功能将在随后的章节作更多地完整描述。

4. OSPF 接口

链路状态协议的基本要点是它涉及到了路由器之间的链路和那些链路的状态。在 Hello 数据包发送及在邻接关系建立之前，以及在 LSA 通告发送之前，一台 OSPF 路由器必须了解它自己的链路情况。OSPF 协议通过路由器的接口信息来了解链路信息，所以在讲述 OSPF 协议时会混用接口和链路这两个术语，而不会仔细区分它们的含义。本小节将讲述 OSPF 协议接口的数据结构和 OSPF 协议接口的不同状态。

（1）OSPF 接口数据结构

运行 OSPF 协议的路由器将为每一个启动 OSPF 协议的接口维护一个数据结构。参见示例 8-2 所示，使用 **show ip ospf interface** 命令观察路由器接口的数据结构的内容。[1]

[1] 根据所运行的 IOS 软件版本而定，该命令输出的信息可能比这里讨论的更多，但这里的信息是每一个 OSPF 接口最基本的信息。

示例 8-2 可以使用命令 show ip ospf interface 来观察与路由器接口相关的 OSPF 协议的具体信息。在这个例子中，接口连接到点到点类型的网络

```
Renoir#show ip ospf interface Serial1.738
Serial1.738 is up, line protocol is up
   Internet Address 192.168.21.21/30, Area 7
   Process ID 1, Router ID 192.168.30.70, Network Type POINT_TO_POINT, Cost: 781
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
      Hello due in 00:00:07
   Neighbor Count is 1, Adjacent neighbor count is 1
      Adjacent with neighbor 192.168.30.77
   Message digest authentication enabled
      Youngest key id is 10
```

路由器接口的数据结构信息说明如下：

- **IP Address and Mask（IP 地址和掩码）**——这个信息是路由器接口所配置的 IP 地址和掩码。始发于这个接口的 OSPF 数据包将把这个地址作为源地址。参见示例 8-2 所示，这个地址/掩码组合是 192.168.21.21/30。
- **Area ID（区域 ID）**——就是接口所在的区域，也就是这个接口所属的网络指定的区域 ID。始发于这个接口的 OSPF 数据包将使用这个区域 ID。参见示例 8-2 所示，这里所显示的接口区域 ID 是 7。
- **Process ID（进程 ID）**——这个特性是 Cisco 公司特有的属性，不是 OSPF 协议开放标准的一部分。Cisco 路由器依赖这个特性能够在同一台路由器中运行多个 OSPF 进程，并且使用这个进程 ID 来区分这些 OSPF 进程。进程 ID 的概念仅在所配置的路由器上有效，而在该路由器之外没有意义。参见示例 8-2 所示，这里的进程 ID 是 1。
- **Router ID（路由器 ID）**——参见示例 8-2 所示，这里的路由器 ID 是 192.168.30.70。
- **Network Type（网络类型）**——和这个接口相连的网络类型：广播型、点到点类型、NBMA、点到多点类型或虚链路等。在示例 8-2 中，网络的类型是点到点。[1]
- **Cost（代价）**——是指从该接口发送出去的数据包的出站接口代价。链路代价是 OSPF 协议的度量，并使用 16 位无符号的整数表示，范围在 1～65 535 之间。Cisco 公司使用的缺省代价是 10^8/BW，表示为一个整数，在这里 BW 是指在接口上配置的带宽，而 10^8 是 Cisco 路由器使用的参考带宽。示例 8-2 中所示的接口配置了一个 128kbit/s 的带宽（示例中没有显示），所以它的代价是 10^8/128kbit/s=781。

路由器接口的代价值可以通过命令 **ip ospf cost** 来改变，当在一个多厂商产品的网络环境中配置 Cisco 路由器时，这个命令变得十分重要。例如，其他厂商的路由器在其所有的接口上使用的缺省代价是 1（实际上就是把 OSPF 的代价映射为跳数）。如果网络中所有的路由器没有使用同一种计算代价的方式来指定 OSPF 的代价，那么 OSPF 协议的路由选择将出现不正确的、次优的，或其他一些不确定的情形。

使用 10^8 作为接口的参考带宽在现代一些带宽高于 100Mbit/s（例如 OC-3 或吉比特以太网）的网络介质中会产生一个问题。10^8/100M=1，这意味着更高带宽的传输介质在 OSPF 协议中将会计算出一个小于 1 的分数，这在 OSPF 协议中是不允许的。因此，任何代价的计算结果如果是小于 1 的分数，都将四舍五入为 1。但是，如果我们的网络是由高带宽链路组成

[1] 请注意，在这里和这个接口相连的是帧中继网络。但是由于这是一个点到点的子接口，因此，OSPF 协议使用点到点类型替代了 NBMA 的网络类型。

的，这就意味着所有接口的代价都变成了 1，从而计算最短路径就变成了基于最小的路由器跳数了。为了修正这个问题，Cisco 提供了命令 **auto-cost reference-bandwidth**，该命令允许管理者更改缺省的参考带宽。

其他一些接口数据结构的参数信息如下：

- **InfTransDelay**——这个信息是指 LSA 从路由器的接口发送后经历的时间，以秒数计算，当 LSA 从路由器接口发出后将会引起这个参数值不断地增大。参见示例 8-2 所示，在图中它是以 Transmit Delay 来显示的，并且在 Cisco 路由器上缺省值为 1s。InfTransDelay 可以通过命令 **ip ospf transmit-delay** 来改变。
- **State（状态）**——这个接口的功能状态将在后面的“OSPF 接口状态机”部分讲述。
- **Router Priority（路由器优先级）**——用来选择 DR 和 BDR 的一个 8 位无符号整数，范围是 0～255。在示例 8-2 中没有显示路由器的优先级是因为这里的网络类型是点到点的，而在这种网络类型中不需要选取 DR 和 BDR。参见示例 8-3 所示，它显示了同一台路由器上另一个 OSPF 接口，从图中可以看出，这个接口与一个广播型网络相连接，因此，在这个网络上将会选取 DR 和 BDR。在 Cisco 路由器上，路由器的优先级缺省为 1，并且可以通过命令 **ip ospf priority** 来改变。

示例 8-3　这里所显示的接口和一个广播型网络相连，并且这台路由器是该广播型网络上的 DR

```
Renoir#show ip ospf interface Ethernet0
Ethernet0 is up, line protocol is up
   Internet Address 192.168.17.73/29, Area 0
   Process ID 1, Router ID 192.168.30.70, Network Type BROADCAST, Cost: 10
   Transmit Delay is 1 sec, State DR, Priority 1
   Designated Router (ID) 192.168.30.70, Interface address 192.168.17.73
   Backup Designated router (ID) 192.168.30.80, Interface address 192.168.17.74
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
      Hello due in 00:00:03
   Neighbor Count is 1, Adjacent neighbor count is 1
      Adjacent with neighbor 192.168.30.80 (Backup Designated Router)
   Message digest authentication enabled
      Youngest key id is 10
```

- **Designated Router（指定路由器）**——对于和路由器接口相连的网络的指定路由器（DR），路由器将同时记录下指定路由器的 Router ID 和它与这个共享网络相连的接口地址信息。注意，在示例 8-2 中并没有显示出指定路由器，因为只有在多址网络类型中才会显示出指定路由器。在示例 8-3 中，这里的指定路由器是 192.168.30.70，它所连接的接口地址是 192.168.17.73。在示例 8-3 中查看一下 Router ID、接口地址和接口状态，就会发现路由器 Renoir 就是 DR。
- **Backup Designated Router（备份指定路由器）**——对于和路由器接口相连的网络的 BDR，路由器也将同时记录下指定路由器的 Router ID 和它与这个共享网络相连的接口地址信息。参见示例 8-3，这里的 BDR 是 192.168.30.80，而它所连接的接口地址是 192.168.17.74。
- **HelloInterval**——是指在接口上传送两个 Hello 数据包之间的周期性间隔时间，以秒（s）来表示。这个周期时间是在从接口发送的 Hello 数据包中通告的。Cisco 路由

器在广播型网络上的缺省值为 10s，而在非广播型网络上的缺省值为 30s，并且可以通过命令 **ip ospf hello-interval** 来改变。在示例 8-3 中，HelloInterval 是通过 Hello 表示的，并在这里使用了路由器配置的缺省值。

- **RouterDeadInterval**——是指在宣告邻居路由器无效之前，本地路由器从与一个接口相连的网络上侦听到来自于邻居路由器的一个 Hello 数据包所经历的时间，以秒（s）来表示。RouterDeadInterval 是在从接口发送的 Hello 数据包中通告的。在 Cisco 公司的路由器上，这个时间缺省的是 HelloInterval 的 4 倍，并且可以通过命令 **ip ospf dead-interval** 来改变。在示例 8-3 中，RouterDeadInterval 是通过 Dead 表示的，并在这里使用了路由器配置的缺省值。
- **Wait Timer（等待计时器）**——在开始选取 DR 和 BDR 之前，路由器等待邻居路由器的 Hello 数据包通告 DR 和 BDR 的时长。等待计数器的时间长度就是 RouterDeadInterval 的时间。在示例 8-2 中的等待时间是没有意义的，因为那个接口是和一个点到点的网络相连的，没有 DR 和 BDR 的选取问题。
- **RxmtInterval**——是指在没有得到确认的情况下，路由器重传 OSPF 数据包将要等待的时间长度，以秒（s）来表示。在示例 8-3 中，这个时间是通过 retransmit 来表示的，并在这里使用了 Cisco 路由器缺省配置的时间——5s。路由器接口的 RxmtInterval 可以通过命令 **ip ospf retransmit-interval** 来改变。
- **Hello Timer（Hello 计时器）**——这个计时器的初始值由 HelloInterval 来设置。当它计时超时后，路由器将从接口上发送出一个 Hello 数据包。在示例 8-3 中显示了 Hello Timer 将在 3s 后超时。
- **Neighboring Routers（邻居路由器）**——是指和这个接口相连的网络上有效邻居路由器（这里的有效邻居路由器是指在 RouterDeadInterval 的时间内可以收到来自于它们的 Hello 数据包的那些邻居路由器）的列表。示例 8-4 显示了同一台路由器另一个接口的信息。在这里，和这个接口相连的网络上应该可以学习到 5 台邻居路由器，但是只有两台邻居路由器是有邻接关系的（只有建立邻接关系的邻居路由器的 Router ID 才会显示出来）。作为一个 DRother 路由器，在这个网络只能和 DR 与 BDR 路由器建立邻接关系，这和多址网络中使用 DR 的规则是一致的。

示例 8-4 在这个网络上，路由器可以看到 5 台邻居路由器，但是只和 DR 与 BDR 路由器建立了邻接关系

```
Renoir#show ip ospf interface Ethernet1
Ethernet1 is up, line protocol is up
   Internet Address 192.168.32.4/24, Area 78
   Process ID 1, Router ID 192.168.30.70, Network Type BROADCAST, Cost: 10
   Transmit Delay is 1 sec, State DROTHER, Priority 1
   Designated Router (ID) 192.168.30.254, Interface address 192.168.32.2
   Backup Designated router (ID) 192.168.30.80, Interface address 192.168.32.1
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
      Hello due in 00:00:01
   Neighbor Count is 5, Adjacent neighbor count is 2
      Adjacent with neighbor 192.168.30.80 (Backup Designated Router)
      Adjacent with neighbor 192.168.30.254 (Designated Router)
   Message digest authentication enabled
      Youngest key id is 10
```

- **AuType**——描述了在网络上使用的认证类型。OSPF 协议认证的类型可以是 Null（没有认证）、简单口令或者加密认证（MD 认证）。在示例 8-4 中可以看出这里使用了消息摘要（MD）认证方式。如果使用了 Null 认证方式，在使用命令 **show ip ospf interface** 时将不会显示认证方式和密钥信息。
- **Authentication Key（认证密钥）**——如果在路由器的接口上启用的是简单认证方式，那么认证密钥就是一个 64 位的口令；如果在路由器的接口上启用的是加密认证方式，那么认证密钥就是一个消息摘要密钥。示例 8-4 中显示了“youngest key ID”是 10。这说明了一个事实，就是加密认证允许在路由器的一个接口上配置多个密钥，从而可以确保便捷、安全地改变密钥。

示例 8-5 中显示了一个和 NBMA 网络相连的接口。注意，在这里 HelloInterval 的值是 NBMA 网络类型缺省值 30s，而 RouterDeadInterval 的值缺省是 4 倍的 HelloInterval。

示例 8-5　这个接口是和 NBMA 网络类型的帧中继网络相连的，并且它是这个网络的 BDR

```
Renoir#show ip ospf interface Serial3
Serial3 is up, line protocol is up
   Internet Address 192.168.16.41/30, Area 0
   Process ID 1, Router ID 192.168.30.105, Network Type NON_BROADCAST, Cost: 64
   Transmit Delay is 1 sec, State BDR, Priority 1
   Designated Router (ID) 192.168.30.210, Interface address 192.168.16.42
   Backup Designated router (ID) 192.168.30.105, Interface address 192.168.16.41
   Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
      Hello due in 00:00:08
   Neighbor Count is 1, Adjacent neighbor count is 1
      Adjacent with neighbor 192.168.30.210 (Designated Router)
```

花费一些时间来比较一下示例 8-2～示例 8-5 中所示的信息会很有收获的。所有这 4 个接口都是在同一台路由器上，但是它们却又在不同类型的网络环境中，从而扮演不同的角色。在每一个实例中所示的接口状态都表明了不同网络上 OSPF 路由器的不同角色。在下一个小节中，将会讲述多种接口状态和接口状态机。

（2）OSPF 接口状态机

一个启用 OSPF 协议的接口在它变成完全有效之前，将会在几种接口状态中间发生转换。这些接口状态是失效、点到点、等待、DR、备份、DRother 和 Loopback 等。

- **失效（Down）**——这是初始化的接口状态。在这个阶段，接口不起任何作用，只是将所有接口的参数设置成它们各自的初始数值，因而，在接口上没有任何路由协议的通信量进行发送和接收。
- **点到点**——这种接口状态仅仅适用于和点到点、点到多点以及虚电路等网络类型相连的接口。当接口的状态切换到该状态时，这种接口就开始起作用了。这时，路由器的接口将开始每隔 HelloInterval 的时间发送一次 Hello 数据包，并尝试和接口链路另一端相连的邻接路由器建立邻接关系。
- **等待（Waiting）**——这种接口状态仅仅适用于和广播型、NBMA 等网络类型相连的接口。当接口的状态切换到这个状态时，这个接口将开始发送和接收 Hello 数据包，并设置等待计时器的值。而路由器将在接口处于这个状态的时候，试图去识别网络上的 DR 和 BDR。
- **指定路由器（DR）**——在这种状态下，该路由器是所连网络的指定路由器（DR），

并将和所在多址网络上的其他路由器建立邻接关系。

- **备份（Backup）**——在这种接口状态下，该路由器就是所连网络的备份指定路由器（BDR），并且和所在多址网络上的其他路由器建立邻接关系。
- **DRother**——在这种接口状态下，该路由器既不是所连网络上的 DR 路由器，也不是 BDR 路由器。虽然它将跟踪网络上所有的路由器，但仅仅会和网络上的 DR 路由器和 BDR 路由器建立邻接关系。
- **Loopback**——在这种接口状态下，路由器的接口通过软件或硬件的方式成为环回（loopback）的。虽然一个接口在该种状态下不能发送数据包，但是接口的地址还是可以通过路由器 LSA 通告出去的（将在后面的部分讲述），因此，进行测试用的数据包能够发现到达这个接口的路径。

图 8-3 中显示了 OSPF 接口的状态和引起接口状态发生转换的输入事件。关于输入事件的描述请查看表 8-1。

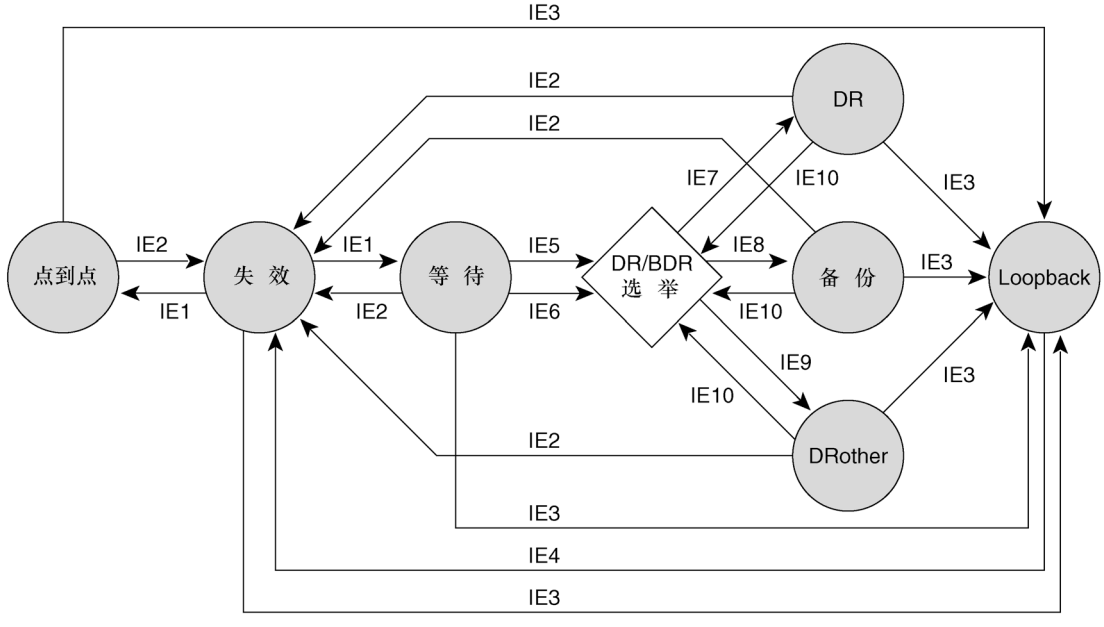


图 8-3 OSPF 接口状态机；查看表 8-1 中关于输入事件（IE）的说明

表 8-1 接口状态机的输入事件

输入事件	描述
IE1	底层协议指明该网络接口是可操作的
IE2	底层协议指明该网络接口是不可操作的
IE3	网络管理系统或底层协议指明该网络接口打环路后是有效的（looped up）
IE4	网络管理系统或底层协议指明该网络接口打环路后是无效的（looped down）
IE5	收到 Hello 数据包，在 Hello 数据包中，要么始发邻居路由器把自身作为 BDR 列出，要么始发邻居路由器把自身作为 DR 列出而不指明 BDR
IE6	等待计时器超时
IE7	路由器被所在的网络选取为 DR 路由器
IE8	路由器被所在网络选取为 BDR 路由器

续表

输入事件	描述
IE9	路由器没有被所在网络选取为 DR 或者 BDR 路由器
IE10	在网络中一组有效的邻居路由器发生了变化。这些变化可能是下列变化之一： （1）和一个邻居路由器之间建立了双向通信 （2）和一个邻居路由器之间丢失了双向通信 （3）收到一个 Hello 数据包，在该 Hello 数据包中始发路由器重新把它自身作为 DR 或 BDR 路由器列出 （4）收到来自于 DR 路由器的 Hello 数据包，在该 Hello 数据包中路由器不再把它自身作为 DR 列出 （5）收到来自于 BDR 路由器的 Hello 数据包，在该 Hello 数据包中路由器不再把它自身作为 BDR 列出 （6）在 RouterDeadInterval 超时后，还没有从 DR 或 BDR 收到 Hello 数据包

5. OSPF 邻居

前面的章节已经讨论了路由器和与之相连的数据链路之间的关系。虽然一台路由器与其他路由器之间的相互操作和关系在讲述 DR 和 BDR 路由器选取的章节中已经做了一些讨论，但是在那些章节介绍 DR 路由器选取过程的主要目的还是围绕着建立一种与网络的联系。本小节的重点将是讲述网络中的路由器与它的邻接路由器之间的关系。邻居之间建立关联关系的最终目的是为了形成邻居路由器之间的邻接关系，最终可以顺利地传送路由选择信息。

要成功建立一个邻接关系，通常需要下面 4 个阶段：

- 邻居路由器发现阶段；
- 双向通信阶段（bidirectional communication）——当两台互为邻居的路由器在它们的 Hello 数据包中都互相列出了它们对方的路由器 ID（Router ID）时，路由器就认为双向通信完成了；
- 数据库同步阶段（database synchronization）——路由器之间将进行交换数据库描述（database description）、链路状态请求、链路状态更新和链路状态确认数据包（将在后续的章节讲述）信息，以便确保在邻居路由器的链路状态数据库中包含有相同的数据库信息。执行这一步骤的目的是使其中一台邻居路由器成为“主路由器”（master），而使另一台路由器成为“从路由器”（slave）。“主路由器”将控制数据库描述数据包的信息交换；
- 完全邻接阶段（full adjacency）。

在前面的介绍中，邻居关系的建立和维持都是通过交换 Hello 数据包来实现的。在广播类型和点到点类型的网络里，Hello 数据包以组播方式发送给组播地址 AllSPFRouters（224.0.0.5）。在 NBMA 类型、点到多点和虚链路类型的网络里，Hello 数据包以单播方式发送给每台单独的邻居路由器。单播的发送方式意味着，路由器首先必须知道邻居路由器的存在，这可以通过手工配置的方式或使用逆向地址解析协议（Inverse ARP）之类的底层协议来发现。关于在这些网络类型中的邻居的配置方法将会在相应的章节中讲述。

在 NBMA 类型的网络中，路由器是每经过 PollInterval 的时间给它邻居状态为 down 的邻居发送一次 Hello 数据包，但是在其他的各种网络类型中，路由器都是每经过 HelloInterval 的时间给它的邻居路由器发送一次 Hello 数据包。在 Cisco 路由器中，NBMA 网络中 PollInterval 的缺省值是 120s。

（1）邻居数据结构

OSPF 路由器在每个 OSPF 接口的接口数据结构中保存的信息可以用来为每一种类型的网络构成 Hello 数据包的内容。路由器通过发送包含这些信息的 Hello 数据包，可以将自己通告给它的邻居路由器。同样的，对于每一台邻居路由器来说，路由器也将维护一个邻居数据结构表，用来表示从其他路由器学习到的 Hello 数据包的信息。

在示例 8-6 中，使用命令 **show ip ospf neighbor** 可以观察到路由器单个邻居的邻居数据结构中的一些信息。[1]

示例 8-6 OSPF 路由器通过邻居数据结构来描述与每个邻居的每次会话

```
Seurat#show ip ospf neighbor 10.7.0.1
 Neighbor 10.7.0.1, interface address 10.8.1.2
    In the area 0 via interface Ethernet0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 10.8.1.1 BDR is 10.8.1.2
    Options is 0x52
    LLS Options is 0x1 (LR)
    Dead timer due in 00:00:30
    Neighbor is up for 09:55:04
    Index 1/3, retransmission queue length 0, number of retransmission 1
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
```

事实上，每台邻居路由器的数据结构中所记录的信息要比示例 8-6 中所显示的信息更多。[2] 邻居数据结构所含信息如下所述：

- **Neighbor ID（邻居路由器 ID）**——邻居路由器的 ID。参见示例 8-6 所示，邻居路由器 ID 是 10.7.0.1。
- **Neighbor IP Address（邻居 IP 地址）**——是指和网络相连的邻居路由器的接口 IP 地址。当 OSPF 数据包以单播方式发送给邻居路由器时，这个地址就是目的地址。参见示例 8-6 所示，这里的邻居路由器 IP 地址是 10.8.1.2。
- **Area ID（区域 ID）**——为了使两台路由器能够互为邻居路由器，路由器收到的 Hello 数据包中所带的区域 ID 必须和路由器接收接口配置的区域 ID 要匹配。示例 8-6 中所示的邻居路由器的区域 ID 是 0（0.0.0.0）。
- **Interface（接口）**——是指与邻居路由器所在网络相连的接口，也就是说，邻居路由器可以通过该接口到达。在示例 8-6 中，该邻居路由器是通过 Ethernet0/0 接口到达的。
- **Neighbor Priority（邻居优先级）**——这一项表示邻居路由器的优先级，并在邻居路由器的 Hello 数据包中通告。所谓的优先级是在 DR 和 BDR 的选取过程使用的。在示例 8-6 的邻居路由器的优先级是 1，这是 Cisco 路由器的缺省值。
- **State（状态）**——这一项指的是从本地路由器角度看到的邻居路由器的功能状态，邻居的状态将在下面的“邻居状态机”部分讲述。示例 8-6 中邻居的状态为 Full。
- **Designated Router（指定路由器）**——这个地址包含在邻居路由器发送的 Hello 数据包的 DR 字段里面。示例 8-6 中的 DR 是 10.8.1.1。
- **Backup Designated Router（备份指定路由器）**——这个地址包含在邻居路由器发

[1] 请与示例 8-1 中的用法进行比较；看看有什么不同。

[2] 当你使用命令 **show ip ospf interface** 时，根据你所运行的 IOS 软件版本而定，你可能会看到某些与示例 8-6 中显示的不同的信息。

送的 Hello 数据包的 BDR 字段中。示例 8-6 中的 BDR 是 10.8.1.2。

- **PollInterval**——这个值只用于 NBMA 网络上相关的邻居路由器。因为在 NBMA 网络上，邻居路由器可能无法自动地被本地路由器发现，因此，如果邻居状态是失效（Down）的，那么路由器将每经过 PollInterval 的时间就会发送一个 Hello 数据包给它的邻居路由器。这里的 PollInterval 的时长比 HelloInterval 的时间要长一些。在示例 8-6 中所示的 NBMA 网络上的邻居路由器显示出它的 PollInterval 时间是 120s——这是 Cisco 路由器的缺省值。
- **Neighbor Options（邻居路由器可选项）**——这是邻居路由器支持的一些可选的 OSPF 性能。关于这些可选项的介绍将在 Hello 数据包格式的讲述中讨论。示例 8-6 中可选项字段的值是 0x52。
- **Inactivity Timer（失效计时器）**——这是一个时长为 RouterDeadInterval（这个参数是在接口数据结构中定义的）的计时器。无论何时，只要从邻居路由器收到一个 Hello 数据包，这个计时器就会被重新设置。如果在这个失效计时器超时了还没从邻居路由器那里收到一个 Hello 数据包，那么该邻居路由器将被宣告为失效（down）。参见示例 8-6 所示，在这里失效计时器用 Dead Timer 来表示，并且将在 30s 后超时。

在邻居数据结构中还有一些信息使用命令 **show ip ospf neighbor** 没有显示出来，这些没有显示的信息说明如下：

- **Master/Slave（主/从）**——在 ExStart 状态下，邻居之间协商的主/从关系将用来控制数据库的同步问题。
- **DD Sequence Number（数据库描述序列号）**——是指当前正在向邻居路由器发送的数据库描述序列号。
- **Last Received Database Description Packet（最后收到的数据库描述数据包）**——这个数据包记录了初始化位（Initialize）、后继位（More）和主/从位（Master/Slave）、可选项，以及最后收到的数据库描述数据包的序列号等信息。这个信息可以用来确定下一个数据库描述数据包是否是重复的。
- **Link State Retransmission List（链路状态重传列表）**——这是在邻接关系建立后，OSPF 已经进行泛洪扩散（flood）但还没有得到确认的 LSA 的列表。当 LSA 还没有得到确认或邻接关系被破坏的时候，LSA 将每经过 RxmtInterval 的时间就重传一次，这里的 RxmtInterval 是在接口的数据结构中定义的。在示例 8-6 中显示的不是链路状态重传列表，它显示的是当前列表中 LSA 的数目（“retransmission queue length”），这里是 0。
- **Database Summary List（数据库摘要列表）**——这一项是指在数据库同步期间，数据库描述数据包中向邻居路由器发送的 LSA 列表。当路由器进入到信息交换状态（Exchange state）时，这些 LSA 将会构成链路状态数据库。
- **Link State Request List（链路状态请求列表）**——这个列表记录了来自邻居路由器的数据库描述数据包的 LSA，这些 LSA 要比在路由器链路状态数据库中的 LSA 更加新。而链路状态请求数据包发送给邻居这些 LSA 的拷贝。当路由器通过链路状态更新数据包收到请求的 LSA 时，请求列表就会减小，最终将变为空列表。

（2）邻居状态机

OSPF 路由器需要邻居路由器在几种邻居状态之间转换后（在邻居数据结构中讲述），才能形成邻居之间的完全邻接关系（Full Adjacent）。

- **失效状态（Down）**——这是一个邻居会话的初始状态，用来指明在最近一个 RouterDeadInterval 的时间内还没有收到来自邻居路由器的 Hello 数据包。除非在 NBMA 网络中的那些邻居路由器，否则，Hello 数据包是不会发送给那些失效的邻居路由器的。在 NBMA 网络环境中，Hello 数据包是每隔 PollInterval 的时间发送一次的。如果一台邻居路由器从其他更高一些的邻居状态转换到了失效状态，那么路由器将会清空链路状态重传列表、数据库摘要列表和链路状态请求列表。
- **尝试状态（Attempt）**——这种状态仅仅适用于 NBMA 网络上的邻居，在 NBMA 网络上邻居路由器是手工配置的。当 NBMA 网络上具有 DR 选取资格的路由器和其邻居路由器相连的接口开始变为有效（Active）时，或者当这台路由器成为 DR 或 BDR 时，这台具有 DR 选取资格的路由器将会把邻居路由器的状态转换到 Attempt 状态。在 Attempt 状态下，路由器将使用 HelloInterval 的时间代替 PollInterval 的时间来作为向邻居发送 Hello 数据包的时间间隔。
- **初始状态（Init）**——这一状态表明在最近的 RouterDeadInterval 时间里路由器收到了来自邻居路由器的 Hello 数据包，但是双向通信仍然没有建立。路由器将会在 Hello 数据包的邻居字段中包含这种状态下或更高状态的所有邻居路由器的路由器 ID。
- **双向通信状态（2-Way）**——这一状态表明本地路由器已经在来自邻居路由器的 Hello 数据包的邻居字段中看到了它自己的路由器 ID，这也就意味着，一个双向通信的会话已经成功建立了。在多址网络中，邻居路由器必须在这个状态或更高状态时才能有资格被选作该网络上的 DR 或 BDR。如果在 Init 状态下从邻居路由器那里收到一个数据库描述数据包，也可以引起邻居状态直接转换到 2-Way 状态。
- **信息交换初始状态（ExStart）**——在这一状态下，本地路由器和它的邻居将建立起主/从关系，并确定数据库描述数据包的序列号，以便为数据库描述数据包的信息交换做准备。在这里，具有最高路由器 ID 的邻居路由器将成为“主”路由器。
- **信息交换状态（Exchange）**——在这一状态下，本地路由器将向它的邻居路由器发送可以描述它整个链路状态数据库信息的数据库描述数据包。同时，在这个 Exchange 的状态下，本地路由器也会发送链路状态请求数据包给它的邻居路由器，用来请求最新的 LSA。
- **信息加载状态（Loading）**——在这一状态下，本地路由器将会向它的邻居路由器发送链路状态请求数据包，用来请求最新的 LSA 通告。虽然在 Exchange 状态下已经发现了这些最新的 LSA 通告，但是本地路由器还没有收到这些 LSA 通告。
- **完全邻接状态（Full）**——在这一状态下，邻居路由器之间将建立起完全邻接关系，这种邻接关系出现在路由器 LSA 和网络 LSA 中。

在图 8-4～图 8-6 中，显示了 OSPF 协议的邻居状态和引起这些邻居状态发生转换的输入

事件。在表 8-2 中对这些输入事件作了详细描述，并在表 8-3 中定义了点。在图 8-4 中显示了从最初始的功能状态到最完全的功能状态一步一步向更高状态转换的一般过程。在图 8-5 和图 8-6 中显示了 OSPF 协议邻居状态机的整个转换过程。

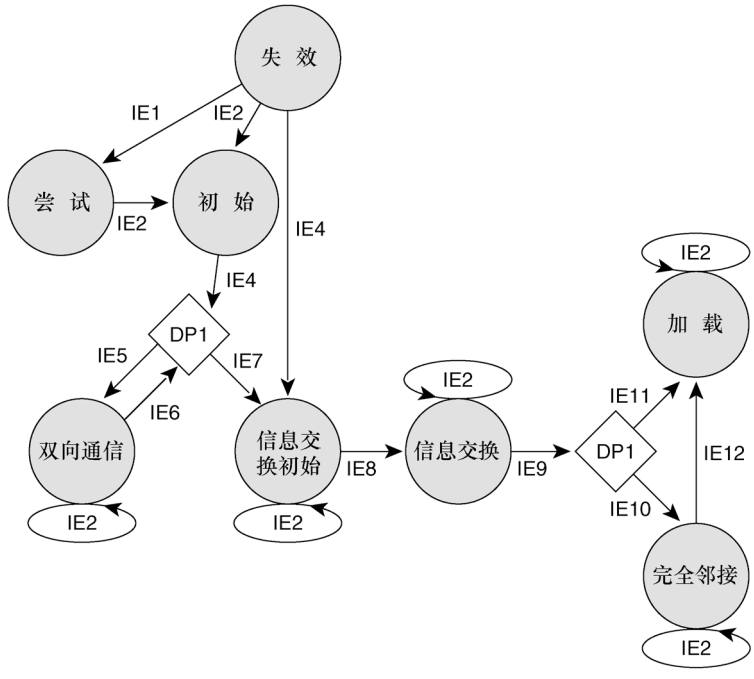


图 8-4 在 OSPF 协议的邻居状态机中，一台邻居路由器从失效状态到完全邻接状态所经过的一系列状态转换

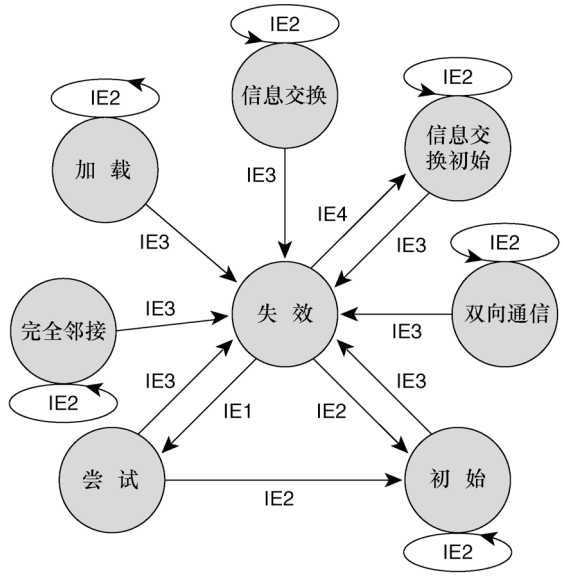


图 8-5 在 OSPF 协议的邻居状态机中，一台邻居路由器从失效状态到初始状态的转换

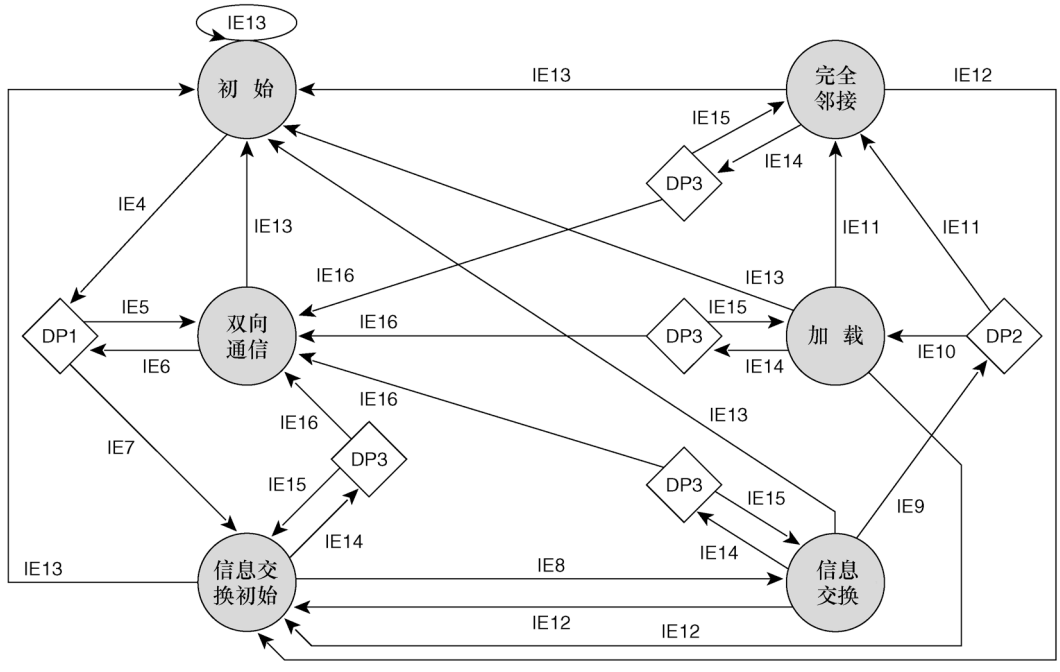


图 8-6 在 OSPF 协议的邻居状态机中，一台邻居路由器从初始状态到完全邻接状态的转换

表 8-2 图 8-4、图 8-5 和图 8-6 的输入事件

输入事件	描述
IE1	这个输入事件只发生在与 NBMA 网络相连的邻居路由器上，并可以通过下列所述的情况之一触发该输入事件： （1）与 NBMA 网络相连的路由器接口开始变为有效（active），并且邻居路由器有资格进行 DR 的选取 （2）本地路由器变为 DR 路由器或者 BDR 路由器，并且邻居路由器没有资格进行 DR 的选取
IE2	从邻居路由器那里收到一个有效的 Hello 数据包
IE3	根据底层的协议、来自于 OSPF 进程本身的明确指令或者无效计时器的超时等影响使邻居路由器不再可达
IE4	本地路由器在邻居路由器发送的 Hello 数据包的邻居字段列表中开始看到它自己的路由器 ID，或者是从邻居路由器收到了数据库描述数据包
IE5	邻居路由器不能建立邻接关系
IE6	这个输入事件可以由下面两种情况的任何一种触发： （1）邻居状态开始转换到 2-way 状态； （2）接口状态发生变化
IE7	不能和这台邻居路由器形成邻接关系
IE8	已经成功建立主/从关系，并且已经交换数据库描述序列号
IE9	完成了数据库描述数据包的信息交换
IE10	链路状态请求列表非空，存在要请求的条目
IE11	链路状态请求列表为空
IE12	邻接关系将被中断并接着重新开始。这个输入事件可以由下面几种情况的任何一种触发： （1）接收到一个数据库描述序列号不匹配的数据库描述数据包 （2）接收到一个含可选项字段的设置和最后一个数据库描述数据包的可选项字段设置不同的数据库描述 （3）接收到一个含初始状态位（Init 位）的设置和最初的数据包不同的数据库描述 （4）接收到一个含 LSA 不在本地路由器的链路状态数据库里的链路状态请求数据包
IE13	从邻居路由器收到一个 Hello 数据包，但是这个 Hello 数据包的邻居字段中没有列出接收该数据包的路由器的路由器 ID

续表

输入事件	描述
IE14	当接口状态变化时将产生这个输入事件
IE15	与该邻居之间现有的或者形成的邻接关系应该继续
IE16	与该邻居之间现有的或者形成的邻接关系不应该继续

表 8-3 图 8-4 ~ 图 8-6 中的判定点

判定点	描述
DP1	是否应该与这台邻居路由器建立一个邻接关系？如果满足下面所描述的条件中的一个或多个，那么将建立邻接关系： （1）网络类型是点到点的 （2）网络类型是点到多点的 （3）网络类型是虚链路 （4）本地路由器是邻接路由器所在的网络上的 DR （5）本地路由器是邻接路由器所在的网络上的 BDR （6）邻居路由器是 DR （7）邻居路由器是 BDR
DP2	关于这台邻居路由器的链路状态请求列表是否是空的
DP3	与这台邻居路由器之间现有的或者形成的邻接关系是否应该继续

（3）建立一个邻接关系

除非邻居路由器之间 Hello 数据包的参数不匹配，一般情况下，在点到点、点到多点和虚链路类型的网络上邻居路由器之间总是可以形成邻接关系的。而在广播型网络和 NBMA 网络上，将需要选取 DR 和 BDR 路由器，DR 和 BDR 路由器将和所有的邻接路由器形成邻接关系，但是在 DRothers 路由器之间没有邻接关系存在。

在一个邻接关系的创建过程中，OSPF 协议使用以下 3 种数据包类型：

- 数据库描述数据包（类型 2）；
- 链路状态请求数据包（类型 3）；
- 链路状态更新数据包（类型 4）。

这些数据包类型的格式将在 8.1.7 小节中详细讲解。

数据库描述数据包对于邻接关系的建立过程来说非常重要。正如它的名字所暗示的，该数据包携带了始发路由器的链路状态数据库中的每一个 LSA 的一个简要描述。这些描述不是关于 LSA 的完整描述，而仅仅是它们的头部——这些信息对于接收路由器判定在它自己的数据库中的 LSA 通告是否是最新的拷贝来说已经是足够的了。另外，在数据库描述数据包中有 3 个标记位用来管理邻接关系的建立过程：

- I 位，或称为初始位（Initial bit），当需要指明所发送的是第一个数据库描述数据包时，该位设置为 1；
- M 位，或称为后继位（More bit），当需要指明所发送的还不是最后一个数据库描述数据包时，该位设置为 1；
- MS 位，或称为主/从位（Master/Slave bit），当数据库描述数据包始发于一个“主”路由器时，该位设置为 1。

当两台邻居路由器在 ExStart 状态开始进行主/从关系协商时，它们都将通过发送一个 MS 位设置为 1 的空的数据库描述数据包来宣称自己是“主”路由器。这两个数据库描述数据包的数据库描述序列号是由发出这两个数据包的路由器根据当时应该使用到的序列号来确定

的。具有较低路由器 ID 的邻居路由器将成为“从”路由器，并且回复一个 MS 位设置为 0 的数据库描述数据包——这个数据库描述数据包的序列号设置为“主”路由器的序列号。同时，这个数据库描述数据包也将是第一个携带 LSA 摘要信息的数据包。当主/从关系协商完成后，邻居状态也将转换到 Exchange 状态。

在 Exchange 状态，邻居路由器开始同步它们的链路状态数据库，同步链路状态数据库的操作是通过描述它们各自的链路状态数据库的所有条目来实现的。数据库摘要列表由路由器的链路状态数据库中所有的 LSA 通告的头部组成，而本地路由器将向它的邻居路由器发送包含这些 LSA 头部列表的数据库描述数据包。

如果本地路由器发现它的邻居路由器有一条 LSA 通告不在它自己的链路状态数据库中，或者邻居路由器含有比已知 LSA 通告更新的拷贝，那么本地路由器将把这条 LSA 放入它的链路状态请求列表中。随后，本地路由器将发出一个链路状态请求数据包去请求一个关于刚才讨论的 LSA 的完整拷贝。链路状态更新数据包将会传送这些被请求的 LSA 的信息。当本地路由器收到关于这些被请求的 LSA 之后，它将从自己的链路状态请求列表中删除这些 LSA 的条目。

在更新数据包中传送的所有的 LSA 必须单独地进行确认。因此，路由器将把这些传送的 LSA 放入它的链路状态重传列表中。当这些 LSA 被确认后，路由器就从它的链路状态重传列表中删除它们。LSA 可以通过下面两种方法之一来确认：

- **显式确认（Explicit Acknowledgment）**——确认收到包含这个 LSA 头部的链路状态确认数据包。
- **隐式确认（Implicit Acknowledgment）**——确认收到包含这个 LSA 的相同实例（没有其他更加新的 LSA）的更新数据包。

“主”路由器将控制数据库的同步过程，并确保每次只有一个数据库描述数据包是未处理的。当“从”路由器收到一个从“主”路由器发出的数据库描述数据包后，“从”路由器将通过发送一个具有相同序列号的数据库描述数据包来确认那个数据包。如果“主”路由器在 RxmtInterval（这个参数在接口数据结构一节中已经介绍）的时间内没有收到一个关于未处理的数据库描述数据包的确认，那么“主”路由器将会发送该数据包的一份新拷贝。

“从”路由器发送数据库描述数据包仅仅用来响应它从“主”路由器那里收到的数据库描述数据包。如果它所收到的数据库描述数据包具有一个新的序列号，那么“从”路由器将发送一个具有相同序列号的数据库描述数据包。如果它所收到的数据库描述数据包的序列号和在这之前已确认的数据库描述数据包相同，那么这个确认数据包就是重发的。

当数据库同步过程完成后，将会出现下面两种状态转换的其中一种：

- 如果链路状态请求列表中仍然还有一些 LSA 条目，那么路由器将把邻居的状态转换到加载（Loading）状态。
- 如果链路状态请求列表为空，那么路由器将会把邻居的状态转换到完全邻接（Full）状态。

如果“主”路由器已经发送过可以完整地描述它自己的链路状态数据库所必要的所有数据库描述数据包，并且从“从”路由器收到一个 M 位设置为 0 的数据库描述数据包，那么这时“主”路由器就认为数据库的同步过程已完成。如果“从”路由器接收到一个 M 位设置为 0 的数据库描述数据包，并且向“主”路由器发送一个确认的 M 位也设置为 0 的数据库描述数据包的话（也就是说，“从”路由器已经完全描述了它自己的链路状态数据库），那么这时“从”路由器就认为数据库的同步过程已完成。由于“从”路由器必须确认每一个收到的数据库描述数据包，因此“从”路由器总是最先得知同步过程完成了。

图 8-7 中显示了一个邻接关系的创建过程。这个例子是直接从 RFC2328 中引用而来的。

在图 8-7 中演示了链路状态数据库同步过程中的下列步骤：

（1）在多路访问的网络上，路由器 RT1 变为有效状态，并发送一个 Hello 数据包。由于它还没有学习到任何邻居，因而这个 Hello 数据包的邻居字段是空的，而 DR 和 BDR 字段设置为 0.0.0.0。

（2）一旦从路由器 RT1 收到上面的 Hello 数据包，路由器 RT2 就会为 RT1 创建一个邻居数据结构，并将 RT1 的状态设置为初始状态（Init）。路由器 RT2 将发送一个 Hello 数据包给路由器 RT1，并在这个 Hello 数据包的邻居字段里设置 RT1 的路由器 ID。同样的，作为 DR，路由器 RT2 也将把 Hello 数据包的 DR 字段设置成它自己的接口地址。

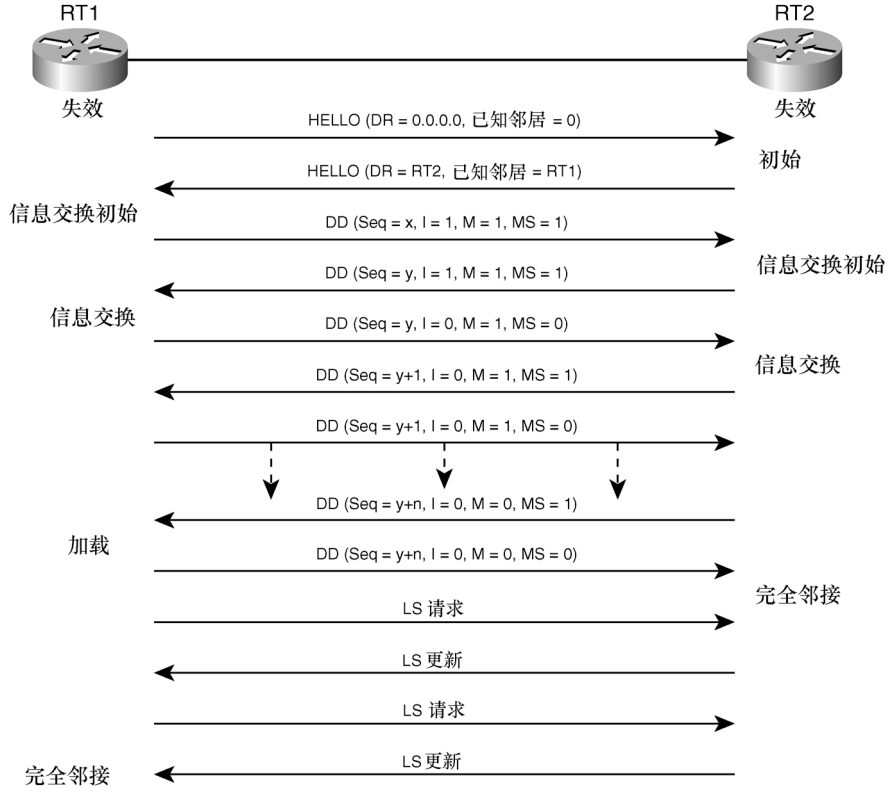


图 8-7 链路状态数据库同步过程和相关的邻居状态

（3）当路由器 RT1 接收到来自路由器 RT2 的 Hello 数据包，并看到自己的路由器 ID 时（参见表 8-2 的输入事件 IE4），RT1 将为路由器 RT2 创建一个邻居数据结构，并把 RT2 的状态设置为 ExStart 状态，以便开始进行主/从关系的协商。接着，路由器 RT1 产生一个空的数据库描述数据包（没有包含 LSA 的摘要），并把数据库描述的序列号设置为 x。同时设置初始位（I 位）来指明这个数据包是路由器 RT1 用来进行本次信息交换（Exchange）的最初的数据库描述数据包，并设置后继位（M 位）来指明这个数据包不是最后的数据库描述数据包，最后还要设置主从位（MS 位）来指明路由器 RT1 声称自己是“主”路由器。

（4）路由器 RT2 一旦收到来自 RT1 的数据库描述数据包，就会把 RT1 的状态转换到 ExStart 状态。接着，它将发送一个响应的数据库描述数据包，并把这个数据库描述数据包的序列号设置为 y。由于路由器 RT2 拥有比 RT1 更高的路由器 ID，因此它将把自己的 MS 位设置为 1。就像最初的那个数据库描述数据包一样，这个数据包用来进行主从关系协商，因此也是空的。

（5）当这两台邻居路由器同意 RT2 是主路由器后，路由器 RT1 就把路由器 RT2 的状态转换为 Exchange 状态。路由器 RT1 将产生一个数据库描述数据包，这个数据包的序列号使用 RT2 的数据库描述数据包的序列号 y，并设置 MS 位为 0 用来指明 RT1 是“从”路由器。同时，该数据包将会传送路由器 RT1 的链路状态摘要列表中的 LSA 头部。

（6）路由器 RT2 一旦收到来自 RT1 的数据库描述数据包，就会把它的邻居状态转换到 Exchange 状态。接着，它将发送一个数据库描述数据包，这个数据包包含路由器 RT2 自己的链路状态摘要列表中的 LSA 头部，并使它的数据库描述序列号增加到 y+1。

（7）当路由器 RT1 从路由器 RT2 收到上述的数据库描述数据包后，路由器 RT1 就会发送一个包含相同序列号的确认数据包。这个过程将一直延续，路由器 RT2 发送一个单一的数据库描述数据包，接着等待从 RT1 发出的包含相同序列号的确认数据包，然后 RT2 再发送下一个数据库描述数据包，直到路由器 RT2 发出包含最后一个 LSA 摘要的数据库描述数据包，并把这个数据包的 M 位设置为 0。

（8）收到上述数据包并且确信它所发出的确认数据包包含它自己最后的 LSA 摘要后，路由器 RT1 就会认为 Exchange 过程已经完成。然而，路由器 RT1 的链路状态请求列表中还存在 LSA 条目，因此，它将转换到信息加载状态（Loading）。

（9）当路由器 RT2 收到 RT1 的最后一个数据库描述数据包时，路由器 RT2 将把 RT1 的状态转换为完全邻接状态（Full），这是因为在它的链路状态请求列表中已经没有 LSA 条目了。

（10）路由器 RT1 发送链路状态请求数据包，而路由器 RT2 通过链路状态更新数据包发送被请求的 LSA 通告，这个过程一直持续到路由器 RT1 的链路状态请求列表变为空。然后，路由器 RT1 也将把路由器 RT2 的状态转换到完全邻接状态。

这里要注意，如果路由器的链路状态请求列表中还有 LSA 条目，它并不需要等待 Loading 状态才发送链路状态请求数据包。事实上，当邻居状态还依旧是 Exchange 状态时路由器就可以发送链路状态请求数据包了。因此，同步过程也许不像图 8-7 中描绘的那么整齐有序，但是却更有效。

图 8-8 中显示了使用协议分析仪捕获到的两台邻居路由器之间正在创建邻接关系的过程。虽然链路状态请求数据包和链路状态更新数据包正在被发送，但这时两个邻居仍然处于 Exchange 状态；注意这里的初始位、后继位、主从位和序列号反映了实际网络环境中的处理过程，这和图 8-7 中描绘的一般过程是一致的。



图 8-8　这个协议分析仪的捕获界面显示了正在创建邻接关系的过程

在示例 8-7 中，使用调试命令 **debug ip ospf adj** 得到的输出结果显示了图 8-8 中正在创建邻接关系的过程，这是在其中一台路由器（路由器 ID 是 192.168.30.175）上观察到的结果。

示例 8-7 调试命令的输出结果显示了图 8-8 中的邻接事件，这是从其中一台路由器上观察到的结果

```
Degas#debug ip ospf adj
OSPF adjacency events debugging is on
OSPF: Rcv DBD from 192.168.30.70 on Ethernet0 seq 0x20E0 opt 0x2 flag 0x7 len 32
state INIT
OSPF: 2 Way Communication to 192.168.30.70 on Ethernet0,
  state 2WAY
OSPF: Neighbor change Event on interface Ethernet0
OSPF: DR/BDR election on Ethernet0
OSPF: Elect BDR 192.168.30.70
OSPF: Elect DR 192.168.30.175
   DR: 192.168.30.175 (Id) BDR: 192.168.30.70 (Id)
OSPF: Send DBD to 192.168.30.70 on Ethernet0 seq 0xB17 opt 0x2 flag 0x7 len 32
OSPF: First DBD and we are not SLAVE
OSPF: Rcv DBD from 192.168.30.70 on Ethernet0 seq 0xB17 opt 0x2 flag 0x2 len 92
state EXSTART
OSPF: NBR Negotiation Done. We are the MASTER
OSPF: Send DBD to 192.168.30.70 on Ethernet0 seq 0xB18 opt 0x2 flag 0x3 len 72
OSPF: Database request to 192.168.30.70
OSPF: Rcv DBD from 192.168.30.70 on Ethernet0 seq 0xB18 opt 0x2 flag 0x0 len 32
state EXCHANGE
OSPF: Send DBD to 192.168.30.70 on Ethernet0 seq 0xB19 opt 0x2 flag 0x1 len 32
OSPF: Rcv DBD from 192.168.30.70 on Ethernet0 seq 0xB19 opt 0x2 flag 0x0 len 32
state EXCHANGE
OSPF: Exchange Done with 192.168.30.70 on Ethernet0
OSPF: Synchronized with 192.168.30.70 on Ethernet0,
state FULL
```

在图 8-8 中，在同步过程结束的时候，可以观察到一系列的链路状态更新数据包和链路状态确认数据包。这些数据包都是 LSA 泛洪扩散过程的一部分，这将在下面一小节中讲述。

6. 泛洪扩散（Flooding）

整个 OSPF 的拓扑图可以描绘成一组互连的路由器或一组互连的节点，这里所说的互连不是指物理的链路而是指逻辑的邻接关系（如图 8-9 所示）。为了使这些节点能够在这个逻辑拓扑上完全地进行路由选择，每一个节点都必须拥有一个关于该拓扑结构的相同拓扑图。这个拓扑图就是拓扑数据库。

OSPF 拓扑数据库更熟知的一种叫法是链路状态数据库。这个数据库由路由器可以接收到的所有 LSA 组成。在拓扑图中发生的一个变化将可以表示为一条或多条 LSA 的变化。泛洪扩散（Flooding）过程就是将这些变化的或新的 LSA 发送到整个网络中去，以确保每一个节点的数据库都可以更新，最终保持所有其他节点的数据库的统一性。

泛洪扩散过程将会使用到下面两种类型的 OSPF 数据包：

- 链路状态更新数据包（Link State Update packets，类型 4）；
- 链路状态确认数据包（Link State Acknowledgment packets，类型 5）。

正如图 8-10 中所显示的那样，每一个链路状态更新数据包和确认数据包都可以携带多个 LSA。虽然 LSA 本身是泛洪扩散到整个网络的，但是更新数据包和确认数据包却只在具有邻接关系的两个节点之间传送。

在点到点的网络中，路由器是以组播方式将更新数据包发送到组播地址 AllSPFRouters（224.0.0.5）的。在点到多点和虚链路的网络上，路由器是以单播方式将更新数据包发送到邻接邻居的接口地址的。

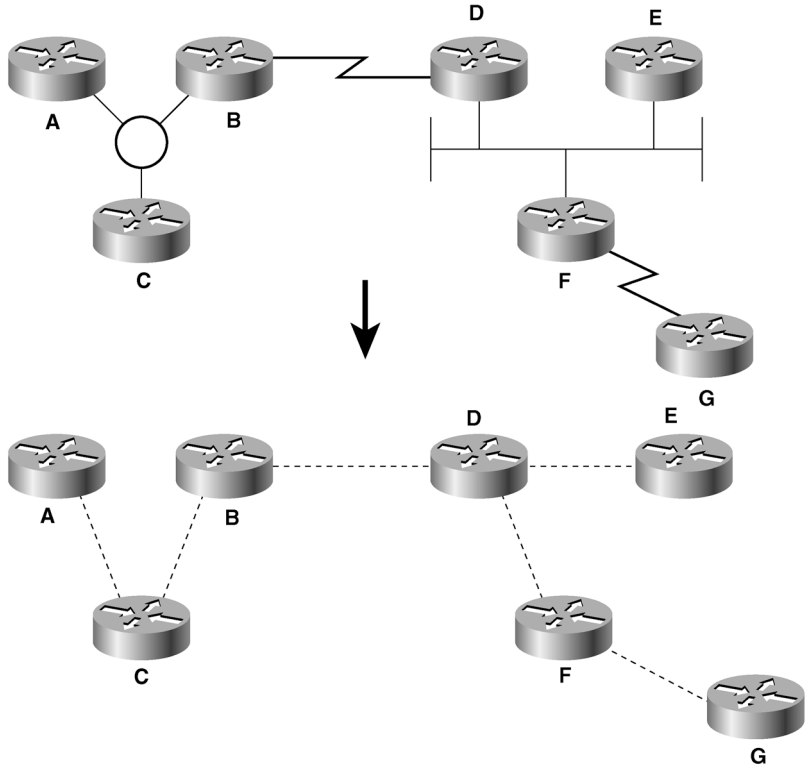


图 8-9　OSPF 协议把一组通过数据链路相连的路由器看作是一组逻辑上通过邻接关系相连的节点

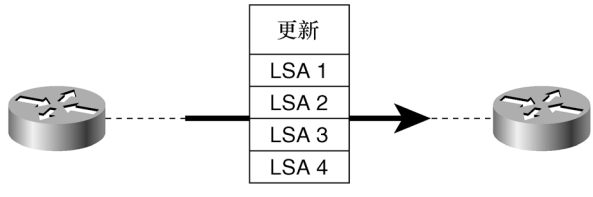


图 8-10　LSA 可以在链路状态更新数据包中发送，从而穿过节点之间的邻接

在广播型的网络上，DRothers 路由器只能和 DR 与 BDR 路由器形成邻接关系，因此，更新数据包将发送到组播地址 AllDRouters（224.0.0.6）。相应地，DR 路由器也将以组播方式发送包含 LSA 的更新数据包到网络上所有与之建立邻接关系的路由器，这里使用的组播地址是 AllSPFRouters。接着，所有的路由器将从所有其他的接口上泛洪扩散 LSA（如图 8-11 所示）。虽然 BDR 路由器也使用组播方式收到和记录了来自 DRothers 路由器的 LSA 通告，但是它不会再重复泛洪扩散或者确认这些 LSA，除非 DR 路由器失效了它才会这么做。在 NBMA 网络上存在同样的 DR/BDR 的功能特性，只是 LSA 是以单播方式从 DRothers 路由器发送给 DR 和 BDR 路由器的，并且 DR 路由器也是以单播方式发送该 LSA 的拷贝到所有与之建立邻接关系的邻居路由器的。

因为完全相同的链路状态数据库信息是正确操作 OSPF 最基本的前提，因此 LSA 的泛洪

扩散必须是可靠的。发送 LSA 的路由器必须要确认它们发出的 LSA 是否被成功接收了，而接收 LSA 的路由器也必须确认它们正在接收的 LSA 信息是正确的。

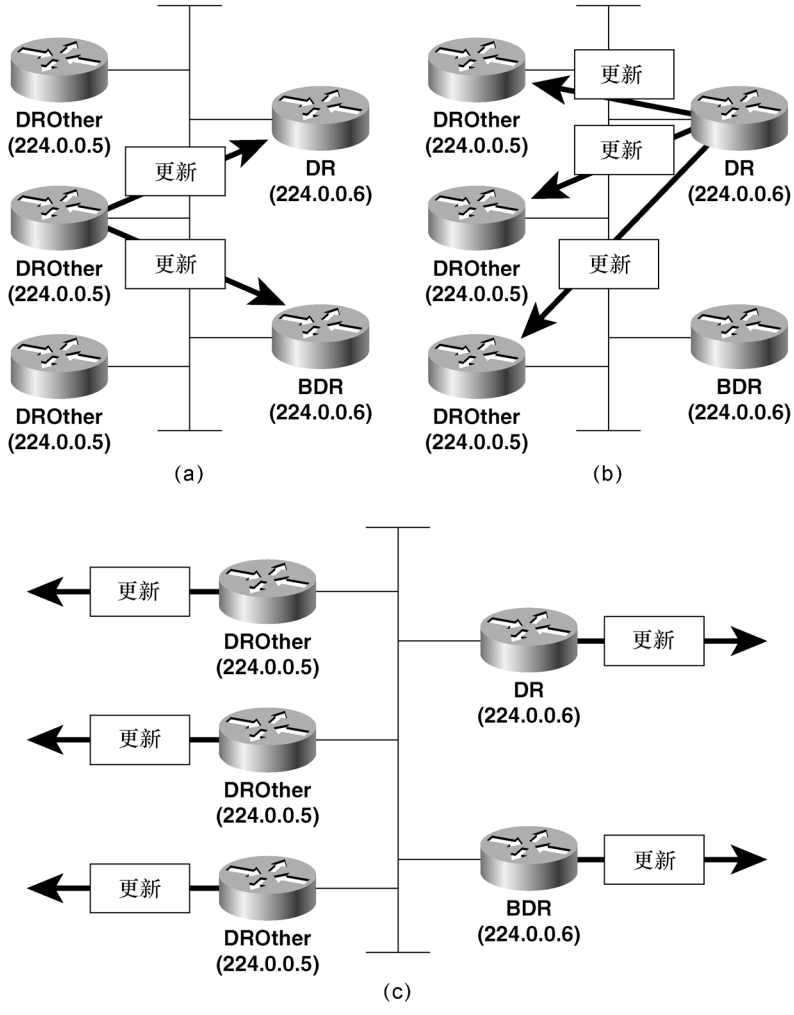


图 8-11 在广播型网络上，DRother 路由器只向 DR 和 BDR 路由器发送 LSA（a）；而 DR 路由器将再把这个 LSA 泛洪扩散到所有的与之有邻接关系的邻居路由器（b）；接着，所有的路由器在它们其他所有的接口上泛洪扩散这个 LSA（c）

（1）可靠的泛洪扩散：确认

对于可靠的泛洪扩散来讲，每一个单独传送的 LSA 都必须被确认。在这里，确认可以有隐式确认（Implicit Acknowledgment）或显式确认（Explicit Acknowledgment）两种方式。

邻居路由器可以通过向始发更新数据包的路由器回送包含那个 LSA 的拷贝信息的更新数据包，来作为对所收到的 LSA 的隐式确认。在一些情况下，隐式确认方式比显式确认方式更有效。例如，当邻居路由器正打算向始发路由器发送更新数据包的时候。

邻居路由器的显式确认是指通过发送一个链路状态确认数据包来确认收到 LSA 的方式。而且，可以使用单个链路状态确认来确认多个 LSA 通告。这个链路状态确认数据包不需要携带完整的 LSA 信息，而只是需要携带 LSA 的头部就足以完全识别这些 LSA 了。

当一台路由器开始发送一个 LSA 时，会把这个 LSA 的一份拷贝放进它所发送的每个邻

居的链路状态重传列表中。这个 LSA 通告每隔 RxmtInterval 的时间重传一次，一直到该 LSA 得到确认，或者一直到这个邻接关系中断。不论是哪一种网络类型，包含重传的链路状态更新数据包总是以单播方式发送。

确认可以是有时延的（delayed）或直接（direct）的。通过延迟一个确认的方法，更多的 LSA 通告可以通过单个链路状态确认数据包来确认。在一个广播型的网络上，来自多台邻居路由器的 LSA 可以由单个组播的链路状态确认数据包来确认。一个被延迟的确认数据包的延迟时间必须小于 RxmtInterval 的时长，从而避免不必要的数据包重传。一般的情况下，在不同的网络类型上使用于链路状态更新数据包的单播/组播地址约定也可以适用于链路状态确认。

直接的确认总是立即发送并且是以单播方式发送的。直接的确认将在出现下面的两种情况下发送：

- 从邻居路由器收到了重复的 LSA，可能表明邻居还没有收到这个 LSA 的一个确认。
- LSA 的老化时间（age）达到最大生存时间（MaxAge，将在下一小节介绍），说明在接收路由器的链路状态数据库里已经没有这个 LSA 的实例（instance）。

（2）可靠的泛洪扩散：序列号、校验和、老化时间

每一个 LSA 都包含 3 个值用来确保在每个数据库中保存的 LSA 是最新的。这 3 个数值是序列号、校验和以及老化时间（age）。

OSPF 协议使用线性的序列号空间（已在第 4 章中讲述）和 32 位有符号的序列号，这里序列号的大小从 InitialSequenceNumber（0x80000001）到 MaxSequenceNumber（0x7fffffff）。当一台路由器始发一条 LSA 通告时，它将设置这个 LSA 的序列号为 InitialSequenceNumber。每当这台路由器产生该 LSA 的一个新实例（Instance）时，该路由器就会将它的序列号增加 1。

如果当前 LSA 的序列号最大值是 MaxSequenceNumber 并且又必须创建这个 LSA 的一个新实例时，这台路由器就必须开始从所有的数据库中清除老的 LSA。这一操作是通过设置现有 LSA 的年龄或老化时间（age）为最大生存时间（MaxAge，将在后面的章节介绍），并且重新泛洪扩散它到所有的邻接节点来实现的。一旦所有的邻接邻居路由器确认过这个“提前老化”的 LSA 后，也就可以泛洪扩散这个 LSA 的一个含有 InitialSequenceNumber 序列号的新实例。

校验和是一个使用 Fletcher 算法计算得到的 16 位整数。[1]这个校验和的计算除了 Age 字段（因为这个 age 字段在 LSA 从一个节点到另一个节点时都会发生变化，因此如果校验和也计算这个字段的话，将在每一个节点上都需要重新计算校验和）外，将覆盖整个 LSA 数据包。驻留在链路状态数据库中的每个 LSA 的校验和每 5min 将检验一次，以便确保这个 LSA 在数据库中没有被破坏。

老化时间（age）是一个用来指明 LSA 的生存时间的 16 位无符号整数，以秒为单位计，范围是 0～3600（1h，也就是最大生存时间）。一台路由器在始发一个 LSA 时，它就把老化时间设置为 0。而当泛洪扩散的 LSA 经过一台路由器时，LSA 的老化时间就会增加一个由 InfTransDelay 设定的秒数。在 Cisco 路由器中，InfTransDelay 设定的缺省值为 1s，这个数值

[1] Alex McKenzie, “ISO Transport Protocol Specification ISO DP 8073,” RFC 905, April 1984, Annex B.

可以通过命令 **ip ospf transmit-delay** 来改变。当 LSA 驻留在路由器的数据库中时，LSA 的老化时间同样也会增大。

当一条 LSA 通告的老化时间达到最大生存时间时，LSA 将被重新泛洪扩散，并且随后会从路由器的数据库中清除该条 LSA。当一台路由器需要从所有路由器的数据库中清除一条 LSA 时，它会提前把这条 LSA 的老化时间设置为最大生存时间并重新泛洪扩散这个 LSA。在这里，只有始发这条 LSA 的路由器才可以提前使这条 LSA 老化。

示例 8-8 中显示了一个链路状态数据库的部分信息，从中可以观察到每一条 LSA 的老化时间、序列号和校验和。有关链路状态数据库和不同类型的 LSA 的详细讨论将在 8.1.3 小节中介绍。

示例 8-8 在链路状态数据库中记录了每一条 LSA 的老化时间、序列号和校验和。老化时间是以秒来计算的

```
Manet#show ip ospf database
       OSPF Router with ID (192.168.30.43) (Process ID 1)
                Router Link States (Area 3)
Link ID         ADV Router      Age   Seq#        Checksum Link Count
192.168.30.13   192.168.30.13   910   0x80000F29  0xA94E   2
192.168.30.23   192.168.30.23   1334  0x80000F55  0x8D53   3
192.168.30.30   192.168.30.30   327   0x800011CA  0x523    8
192.168.30.33   192.168.30.33   70    0x80000AF4  0x94DD   3
192.168.30.43   192.168.30.43   1697  0x80000F2F  0x1DA1   2
```

当收到某条相同的 LSA 的多个实例时，路由器将通过下面的算法来确定哪个是最新的 LSA 实例：

1．比较 LSA 实例的序列号。拥有最大的序列号的 LSA 就是最新的 LSA。

2．如果 LSA 实例的序列号相同， 那么将会比较它们的校验和。拥有最大的无符号校验和的 LSA 就是最新的 LSA。

3．如果 LSA 实例的校验和也相同，那么将进一步比较它们的老化时间。如果只有一条 LSA 拥有大小为最大生存时间的老化时间，那么就认为这条 LSA 是最新的 LSA。

4．如果这些 LSA 的老化时间之间的差别多于 15min（称做 MaxAgeDiff）， 那么拥有较小的老化时间的 LSA 将是最新的 LSA。

5．如果上述的条件都无法区分最新的 LSA，那么这两个 LSA 就被认为是相同的。

8.1.2 区域（Area）

到目前为止，读者应该对 OSPF 协议有一定的了解了。OSPF 协议由于使用了多个数据库和复杂的算法，因而相比前面几章介绍的路由选择协议而言，它将会耗费路由器更多的内存和更多的 CPU 处理能力。当网络的规模不断增大时，对路由器的性能要求就会显得比较重要甚至达到了路由器性能的极限。另一方面，虽然 LSA 的泛洪扩散比 RIP 协议中周期性的、全路由表的更新更加有效率，但是对于一个大型的网络来说，它依然给大量数据链路带来了无法承受的负担。SPF 算法本身并没有特别的解决办法。LSA 的泛洪扩散和数据库的维护等相关的处理仍然大大加重了 CPU 的负担。

OSPF 协议可以利用区域的概念来缩小这些不利的影响。在 OSPF 协议的环境下，区域（Area）是一组逻辑上的 OSPF 路由器和链路，它可以有效地把一个 OSPF 域分割成几个子域

（如图 8-12 所示）。在一个区域内的路由器将不需要了解它们所在区域外部的拓扑细节。在这种环境下：

- 路由器仅仅需要和它所在区域的其他路由器具有相同的链路状态数据库，而没有必要和整个 OSPF 域内的所有路由器共享相同的链路状态数据库。因此，在这种情况下，链路状态数据库大小的缩减就降低了对路由器内存的消耗。
- 链路状态数据库的减小也就意味着处理较少的 LSA，从而也就降低了对路由器 CPU 的消耗。
- 由于链路状态数据库只需要在一个区域内进行维护，因此，大量的 LSA 泛洪扩散也就被限制在一个区域里面了。

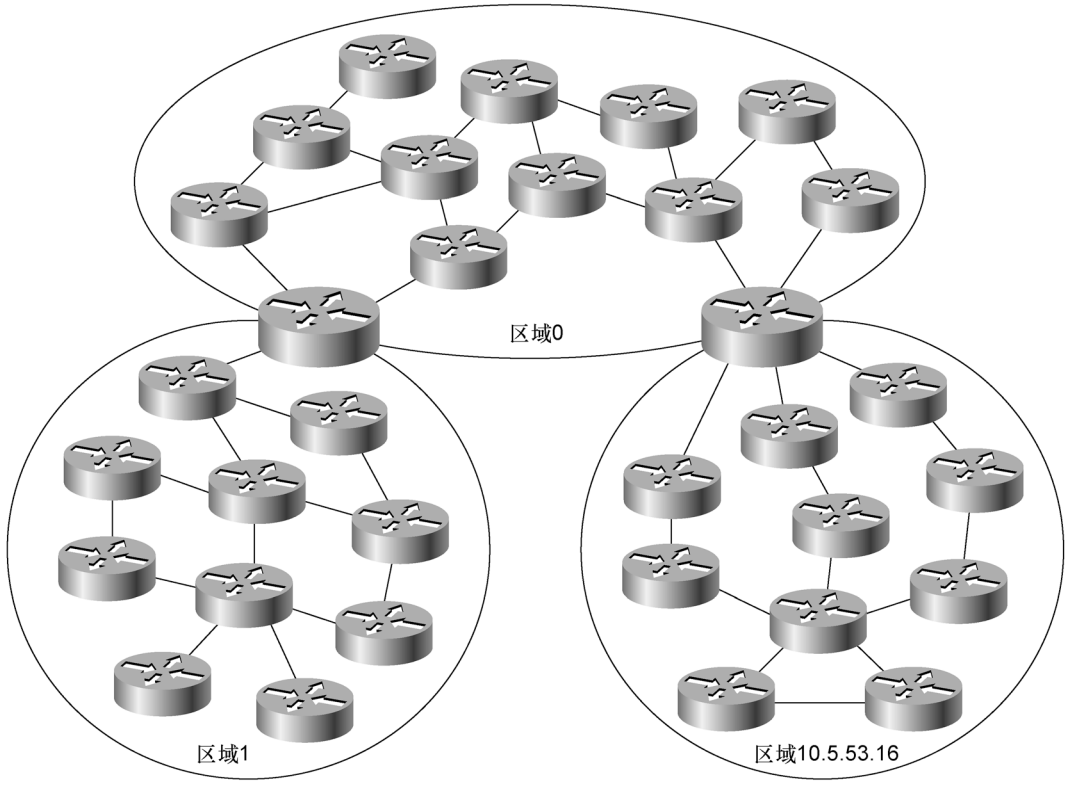


图 8-12　OSPF 区域是一组逻辑上的 OSPF 路由器。每一个区域都是通过它自己的链路状态数据库来描述的，而且每台路由器也都只需要维护路由器本身所在的区域的链路状态数据库

区域是通过一个 32 位的区域 ID（Area ID）来识别的。正如图 8-12 所显示的，区域 ID 可以表示成一个十进制的数字，也可以表示成一个点分十进制的数字。在 Cisco 路由器中这两种表示方式都可以使用。 到底选用哪一种格式来标识一个具体的区域 ID，通常是根据使用的方便性来选择。例如，区域 0 和区域 0.0.0.0 的使用效果是相同的，区域 16 和 0.0.0.16，区域 271 和 0.0.1.15，等等。在上述这些实例当中，我们应该首先选用十进制的表示方式。然而，如果要在区域 3232243229 和区域 192.168.30.29 两种表示方式中选择一种格式的话，那么后面一种格式可能是比较好的一种选择。

定义了下面 3 种与区域相关的通信量的类型：

- **域内通信量（Intra-Area Traffic）**——是指由在单个域内的路由器之间交换的数据

包构成的通信量。

- **域间通信量（Inter-Area Traffic）**——是指由在不同区域的路由器之间交换的数据包构成的通信量。
- **外部通信量（External Traffic）**——是指由 OSPF 域内的路由器和其他路由选择域的路由器之间交换的数据包构成的通信量。

区域 0（或者区域 0.0.0.0）是为骨干域保留的区域 ID 号。骨干区域（Backbone Area）的任务是汇总每一个区域的网络拓扑到其他所有的区域。正是由于这个原因，所有的域间通信量都必须通过骨干区域，非骨干区域之间不能直接交换数据包。

大多数 OSPF 协议的设计者对于单个区域所能支持的路由器的最大数量都有一个个人认为较适当的粗略的经验值。单个区域所支持的路由器最大数量的范围大约是 30～200。但是，在一个区域内实际加入的路由器数量要比单个区域所能容纳的路由器最大数量小一些。这是因为还有更为重要的一些因素影响着这个数量，诸如一个区域内链路的数量、网络拓扑的稳定性、路由器的内存和 CPU 性能、路由汇总的有效使用和注入到这个区域的汇总 LSA 的数量，等等。正是由于这些因素，有时在一些区域里包含 25 台路由器可能都已经显得比较多了，而在另一些区域内却可以容纳多于 500 台的路由器。

只使用单个区域来设计一个小型的 OSPF 网络是非常合理的。不论区域数量的多少，如果一个区域的数据链路非常之少，以至于没有冗余链路存在的话，那么一些潜在的故障就会发生。如果这样一个区域被分割开来，那么就有可能使网络的通信服务中断。被分割的区域（或称为分段区域，Partitioned Area）将在后面的小节中更详细地介绍。

1. 路由器的类型

路由器也像通信量一样可以被分成和区域相关的几个类型。所有的 OSPF 路由器都是下面 4 种路由器类型中的一种，如图 8-13 所示。

- **内部路由器（Internal Router）**——是指所有接口都属于同一个区域的路由器。
- **区域边界路由器（Area Border Routers，ABR）**——是指连接一个或者多个区域到骨干区域的路由器，并且这些路由器会作为域间通信量的路由网关。因而，ABR 路由器总是至少有一个接口是属于骨干区域的，而且必须为每一个与之相连的区域维护不同的链路状态数据库。正因为这个原因，ABR 路由器通常需要比一般的内部路由器更多的内存和更高性能的路由处理器。ABR 路由器将会汇总与它相连的区域的拓扑信息给骨干区域，然后又将这些汇总信息传送给其他的区域。
- **骨干路由器（Backbone Router）**——是指至少有一个接口是和骨干区域相连的路由器。这个定义意味着 ABR 路由器也可以是骨干路由器，但是，如图 8-13 中显示，并不是所有的骨干路由器都是 ABR 路由器。另外，如果一台内部路由器的所有接口都属于区域 0，那么这台内部路由器也是一台骨干路由器。
- **自主系统边界路由器（Autonomous System Boundary Router，ASBR）**——可以认为是 OSPF 域外部的通信量进入 OSPF 域的网关路由器，也就是说，ASBR 路由器是用来把其他路由选择协议（例如，图 8-13 中显示的 BGP 协议和 EIGRP 协议进程）学习到的路由，通过路由选择重分配的方式注入到 OSPF 域的路由器。一台 ASBR 路由器可以是位于 OSPF 域的自主系统内部的任何路由器，它可以是一台内部路由器、骨干路由器或者 ABR 路由器。

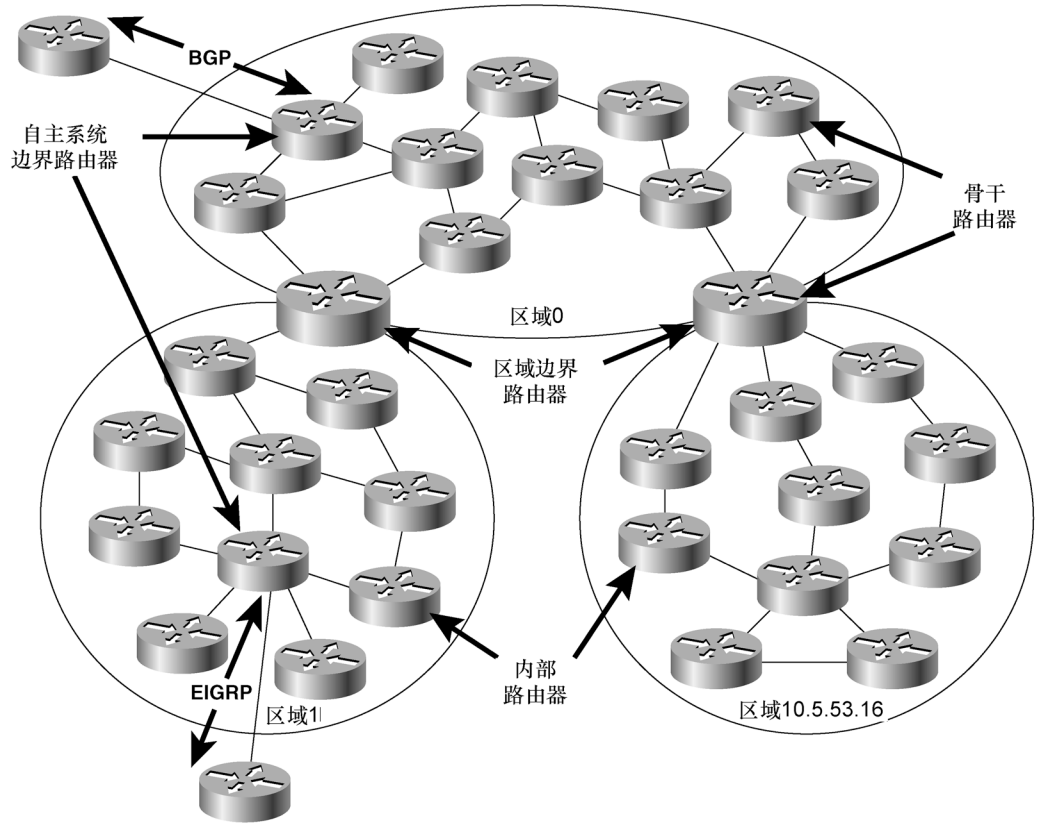


图 8-13　所有的 OSPF 路由器都可以被归类到下面 4 种路由器类型之一——内部路由器、骨干路由器、区域边界路由器（ABR）或自主系统边界路由器（ASBR）。注意前 3 种路由器类型的任何一种都可能成为一台 ASBR

2. 分段区域

分段区域（partitioned area）是指由于链路失效而使一个区域的一个部分和其他部分隔离开来的情形。如果一个非骨干的区域变成分段区域，并且在这个分段区域的任何一段区域里的所有路由器当中都还能发现一台 ABR 路由器，如图 8-14 所示，那么这个分段区域将不会产生中断通信服务的情况。骨干区域仅仅会把这个分段区域看成两个单独的区域。但是，从这个分段区域的任何一段区域到另一段区域的域内通信量将变为域间通信量，这些通信量将通过骨干区域而绕开该分段区域。这里要注意，分段区域和孤立区域（isolated area）是不同的，孤立区域没有链路路径和网络相连。

如果一个骨干区域本身变成了分段区域，那么将会带来更加麻烦的问题。如图 8-15 中所示，一个分段的骨干区域将把原来的骨干区域隔离成两部分区域，并在这两部分区域上创建两个单独的 OSPF 域。

如图 8-16 所示，图中显示了一些更好的区域设计方法。在区域 0 和区域 2 之间设计了两条数据链路，这样任何一条链路失效都不会使它们变成分段的区域。但是，区域 2 也有一个设计缺陷就是如果 ABR 路由器失效了，那么这个区域就会被孤立。区域 3 使用了两台 ABR 路由器，在这里，任何单条链路的失效或单个 ABR 的失效都不会隔离这个区域的任何部分。

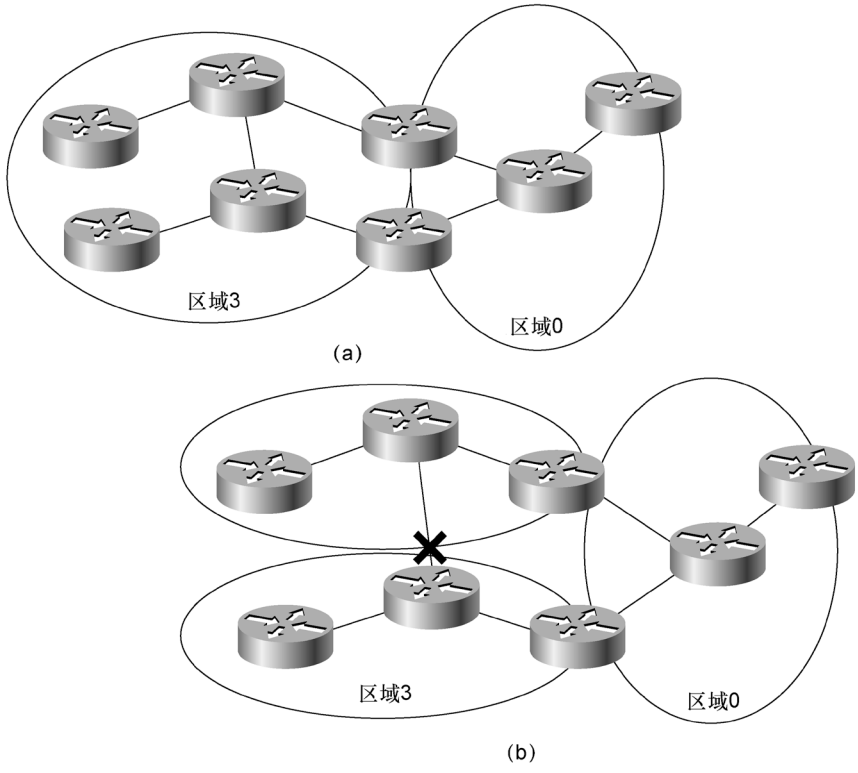


图 8-14 （a）区域 3 通过两台 ABR 路由器和骨干区域（区域 0）相连。（b）区域 3 的一条链路失效了将会创建一个分段区域，但是区域 3 内的所有路由器都仍然可以到达一台 ABR 路由器。在这种情况下，数据的通信量仍然可以在这个分段区域的两边进行转发

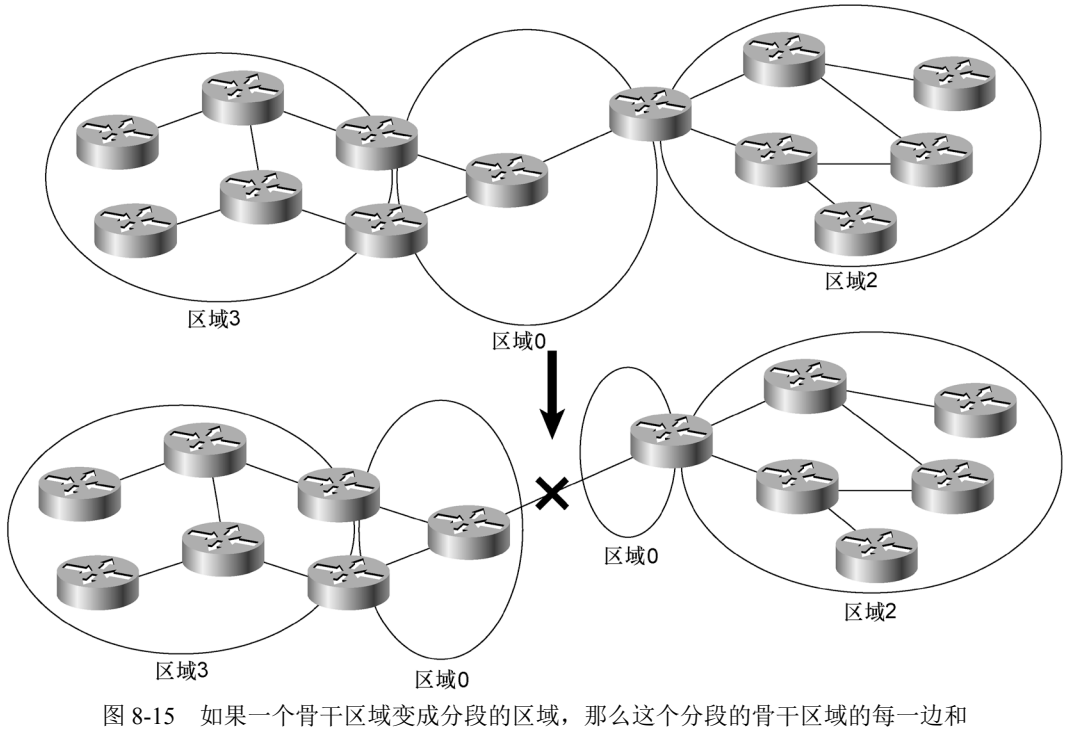


图 8-15 如果一个骨干区域变成分段的区域，那么这个分段的骨干区域的每一边和与之相连的区域都将和另外一边的部分隔离开来

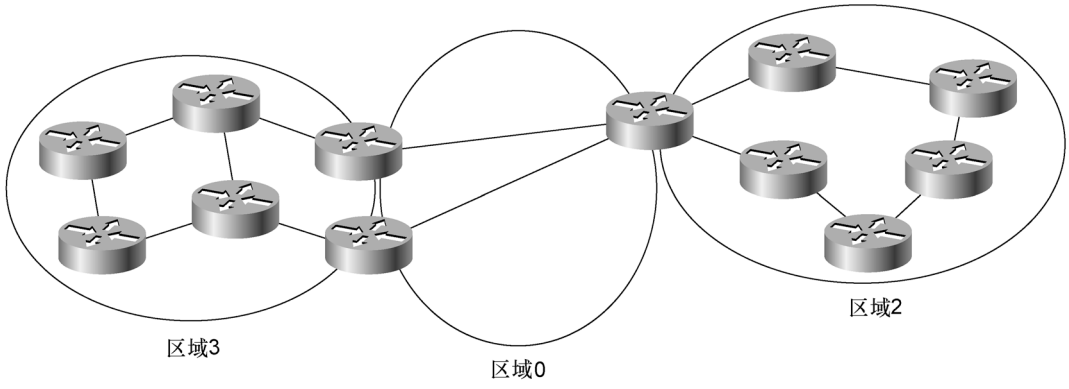


图 8-16　在区域 0 和区域 2 之间，单条链路的失效不会隔离这个区域。在区域 3 中，单条链路或单台 ABR 路由器的失效也不会隔离区域 3

3. 虚链路

虚链路（virtual link）是指一条通过一个非骨干区域连接到骨干区域的链路。虚链路主要应用于以下几种目的：

（1）通过一个非骨干区域连接一个区域到骨干区域（如图 8-17 所示）。

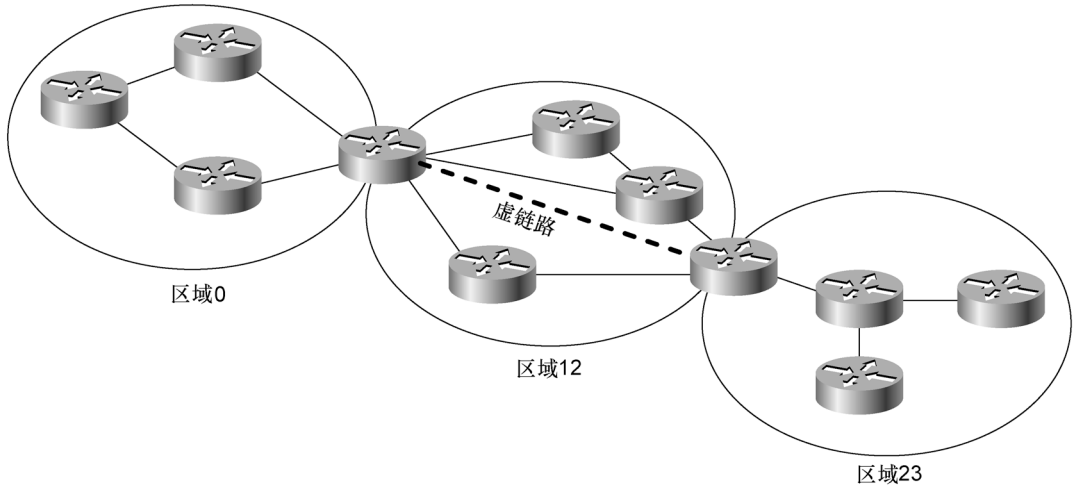


图 8-17　一条虚链路用来把区域 23 经由区域 12 连接到骨干区域

（2）通过一个非骨干区域连接一个分段的骨干区域两边的部分区域（如图 8-18 所示）。

在这两个例子中，虚链路和具体的物理链路没有关系。虚链路事实上是一个逻辑通道（tunnel），数据包可以通过选择最优的路径从一端到达另一端。

在配置虚链路的时候，有几条相关的规则，说明如下：

- 虚链路必须配置在两台 ABR 路由器之间；
- 配置了虚链路所经过的区域必须拥有全部的路由选择信息，这样的区域又被称为传送区域（transit area）；
- 传送区域不能是一个末梢区域。

正如前面所提及的，OSPF 协议也把虚链路归类为一个网络类型。更特别的是，虚链路可以看成是在两台 ABR 路由器之间的一个无编码的——也就是说是无编址地址——链路，

并且它是属于骨干区域的。这些 ABR 路由器之间虽然没有物理的数据链路相连，但是它们可以看作是通过它们之间的虚链路逻辑上虚拟连接的邻居。在每一个 ABR 路由器的路由表中，当发现有到达邻居的 ABR 路由器的路由时，虚链路将转换到完全可操作的点到点接口状态。这条虚链路的代价就是到达它的邻居路由器的路由代价。当接口状态变为点到点状态时，一个邻接关系将通过这条虚链路建立成功。

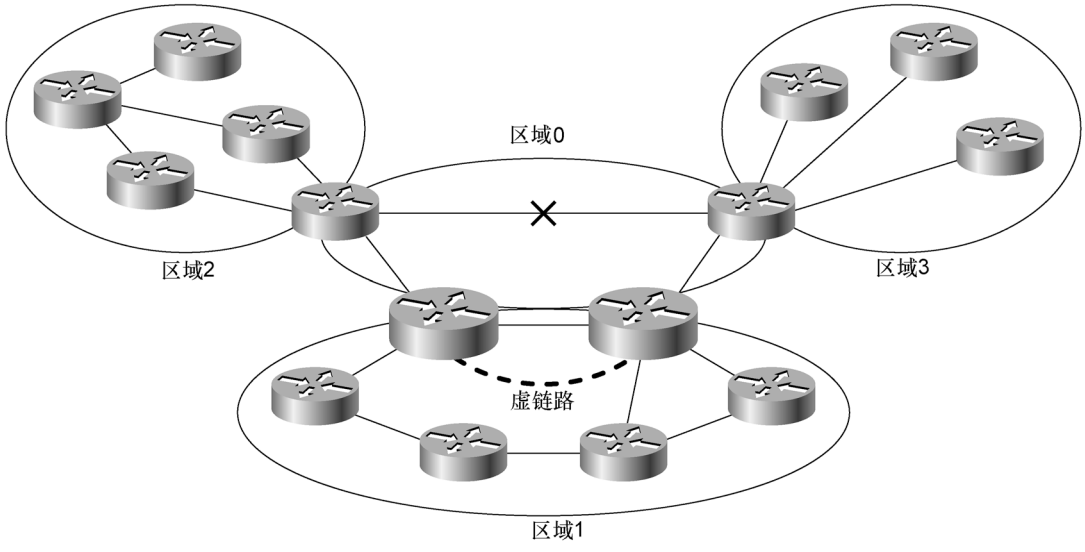


图 8-18 一条虚链路穿过一个非骨干区域重新连接一个分段的骨干区域

显然，虚链路的存在增加了网络的复杂程度，而且使故障的排除更加困难。因此，最好避免使用虚链路，而应该在区域上，特别是骨干区域上设计冗余链路来确保防止分段区域的产生。当有两个或多个网络要合并时，预先要制定好充分的计划，以便确保那些没有直连链路到达骨干区域的区域不被遗漏。

如果配置了一条虚链路，设计者应该仅仅把它用来作为修复无法避免的网络拓扑问题的一种临时手段。虚链路可以看作是一个标明网络的某个部分是否需要重新设计的标志。事实上，永久虚链路的存在总是一个设计比较糟糕的网络的标志。

8.1.3 链路状态数据库

一台路由器中所有有效的 LSA 都被存放在它的链路状态数据库中。正确的 LSA 将可以描述一个 OSPF 区域网络拓扑的结构。因为一个区域中的每一台路由器都要利用这个数据库的信息来计算它自己的最短路径树，因此，所有区域数据库的统一性对于正确的路由选择来说就变得十分重要。

参见示例 8-9 所示，要观察一个链路状态数据库中的所有 LSA 的列表可以通过命令 **show ip ospf database** 来实现。图中所显示的列表并不是数据库中存储的关于每个 LSA 的全部信息，而仅仅是这些 LSA 的头部信息。这里要注意，这个数据库如果是包含多个区域的 LSA 信息的，那么就表明这台路由器是 ABR 路由器。

示例 8-9 中的大多数条目出于简化的目的已经被删除，真实的链路状态数据库包含了

1 445 个 LSA 条目和 4 个区域，参见示例 8-10 所示。

示例 8-9　使用命令 show ip ospf database 来显示一个链路状态数据库中所有 LSA 的列表

```
Homer#show ip ospf database

OSPF Router with ID (192.168.30.50) (Process ID 1)

                  Router Link States (Area 0)

Link ID           ADV Router      Age   Seq#        Checksum   Link count
192.168.30.10     192.168.30.10   1010  0x80001416  0xA818     3
192.168.30.20     192.168.30.20   677   0x800013C9  0xDE18     3
192.168.30.70     192.168.30.70   857   0x80001448  0xFD79     3
192.168.30.80     192.168.30.80   1010  0x800014D1  0xEB5C     5

                  Net Link States (Area 0)

Link ID           ADV Router      Age   Seq#        Checksum
192.168.17.18     192.168.30.20   677   0x800001AD  0x849A
192.168.17.34     192.168.30.60   695   0x800003E2  0x4619
192.168.17.58     192.168.30.40   579   0x8000113C  0xF0D
192.168.17.73     192.168.30.70   857   0x8000044F  0xB0E7

                  Summary Net Link States (Area 0)

Link ID           ADV Router      Age   Seq#        Checksum
172.16.121.0      192.168.30.60   421   0x8000009F  0xD52
172.16.121.0      192.168.30.70   656   0x8000037F  0x86A
10.63.65.0        192.168.30.10   983   0x80000004  0x1EAA
10.63.65.0        192.168.30.80   962   0x80000004  0x780A

                  Summary ASB Link States (Area 0)

Link ID           ADV Router      Age   Seq#        Checksum
192.168.30.12     192.168.30.20   584   0x80000005  0xFC4C
192.168.30.12     192.168.30.30   56    0x80000004  0x45BA
172.20.57.254     192.168.30.70   664   0x800000CE  0xF2CF
172.20.57.254     192.168.30.80   963   0x80000295  0x23CC

                  Router Link States (Area 4)

ink ID            ADV Router      Age   Seq#        Checksum   Link count
192.168.30.14     192.168.30.14   311   0x80000EA5  0x93A0     7
192.168.30.24     192.168.30.24   685   0x80001333  0x6F56     6
192.168.30.50     192.168.30.50   116   0x80001056  0x42BF     2
192.168.30.54     192.168.30.54   1213  0x80000D1F  0x3385     2

                  Summary Net Link States (Area 4)

Link ID           ADV Router      Age   Seq#        Checksum
172.16.121.0      192.168.30.40   1231  0x80000D88  0x73BF
172.16.121.0      192.168.30.50   34    0x800003F4  0xF90D
10.63.65.0        192.168.30.40   1240  0x80000003  0x5110
10.63.65.0        192.168.30.50   42    0x80000005  0x1144
                  Summary ASB Link States (Area 4)

Link ID           ADV Router      Age   Seq#        Checksum
192.168.30.12     192.168.30.40   1240  0x80000006  0x6980
192.168.30.12     192.168.30.50   42    0x80000008  0xC423
```

（待续）

```
172.20.57.254       192.168.30.40   1241  0x8000029B  0xEED8
172.20.57.254       192.168.30.50   43    0x800002A8  0x9818

                    AS External Link States

Link ID             ADV Router      Age   Seq#        Checksum   Tag
10.83.10.0          192.168.30.60   459   0x80000D49  0x9C0B     0
10.1.27.0           192.168.30.62   785   0x800000EB  0xB5CE     0
10.22.85.0          192.168.30.70   902   0x8000037D  0x1EC0     65502
10.22.85.0          192.168.30.80   1056  0x800001F7  0x6B4B     65502
Homer#
```

示例 8-10　使用命令 show ip ospf database database-summary 来显示一个链路状态数据库当中基于区域和 LSA 类型分类的 LSA 通告的数量

```
Homer#show ip ospf database database-summary

            OSPF Router with ID (192.168.30.50) (Process ID 1)

Area ID         Router  Network  Sum-Net  Sum-ASBR  Subtotal  Delete  Maxage
0               8       4        185      27        224       0       0
4               7       0        216      26        249       0       0
5               7       0        107      13        127       0       0
56              2       1        236      26        265       0       0
AS External                                        580       0       0
Total           24      5        744      92        1445
Homer#
```

正如早前在“可靠的泛洪扩散：序列号、校验和、老化时间”部分所提及的，当 LSA 通告驻留在路由器的链路状态数据库中的时候，它们的老化时间是增大的。如果这些 LSA 通告达到了最大生存时间（1h），那么它们将从 OSPF 域中清除掉。这就意味着，在这里必须有一种机制来防止正常的 LSA 通告达到最大生存时间而被清除掉。这种机制就是链路状态重刷新（link state refresh）。每隔 30min（这个时间称为 LSRefreshTime）始发这条 LSA 通告的路由器就将泛洪扩散这条 LSA 的一个新拷贝，并将它的序列号增加 1，老化时间设置为 0。其他的 OSPF 路由器一旦收到这个新拷贝，就会用这个新拷贝替换该条 LSA 通告原来的拷贝，并且使这个新拷贝的老化时间开始增加。

虽然链路状态重刷新的机制是用来确保每条 LSA 通告的活动状态的，但是，它还带来一个额外的好处是，任何一个在路由器的链路状态数据库中可能已经被破坏的 LSA 通告，都可以被正常 LSA 通告刷新后的拷贝来替换。

由于每一个 LSA 通告都与一个独自的重刷新计时器相关联，这意味着每 30min，LSA 通告的 LSRefreshTime 将不会一下子都突然超时，从而重新泛洪扩散所有的 LSA 通告。作为替换做法，重新泛洪扩散将在一个半随机的模式（semi-random pattern）下传播出去。这种方法带来的问题是，每一个单独的 LSA 通告都会在它增加 LSRefreshTime 超时的时候被重新泛洪扩散，这使链路带宽的使用没有效率。更新数据包只能传送一些，甚至单个 LSA 通告。

在 IOS 软件 11.3 版本之前，Cisco 路由器只选用单个 LSRefreshTime 和整个链路状态数据库相关联。每隔 30min，每台路由器将重刷新它始发的所有 LSA 通告，而不管这些 LSA 通告实际的老化时间。虽然这种策略避免了链路带宽使用低效的问题，但是它再次引入了本应解决的独自重刷新计时器的问题。如果一个链路状态数据库很大，那么每隔 30min，网络上就会产生一个区域通信量和 CPU 利用率的高峰。

因此，一种称为 LSA 组步调（group pacing）的机制，作为 LSA 独自使用重刷新计时器

和单个统一计时器之间的一种折衷办法。每一个 LSA 通告都有属于自己的重刷新计时器，但是当它们独自使用的重刷新计时器超时的时候，会引入一个时延来延迟这些 LSA 通告的泛洪扩散。通过延迟重刷新时间，可以在泛洪扩散之前将更多的 LSA 通告共同编成一组，从而可以让更新数据包携带更大数量的 LSA 通告。缺省条件下，一个组步调的间隔时间是 240s（4min）。根据 IOS 软件版本不同，这个间隔时间可以通过命令 **timers lsa-group-pacing** 或 **timers pacing lsa-group** 来改变。[1]如果链路状态数据库非常大（10 000 或更多的 LSA），那么减小组步调的间隔时间是有好处的；而如果链路状态数据库很小，那么增加组步调的间隔时间会比较有用。在这里，组步调计时器的大小范围是 10～1800s。

1. LSA 的类型

由于 OSPF 协议定义了多种路由器的类型，因而定义多种 LSA 通告的类型也是必要的。例如，一台 DR 路由器必须通告多路访问链路和所有与这条链路相连的路由器，而其他类型的路由器将不需要通告这种类型的信息。在示例 8-9 和示例 8-10 中已经显示了多种类型的 LSA 通告。每一种 LSA 通告类型都描述了 OSPF 网络的一种不同情况。表 8-4 中列出了 LSA 通告的类型和标识这些 LSA 类型的代码。

表 8-4 LSA 类型

类型代码	描述
1	路由器 LSA
2	网络 LSA
3	网络汇总 LSA
4	ASBR 汇总 LSA
5	AS 外部 LSA
6	组成员 LSA
7	NSSA 外部 LSA
8	外部属性 LSA
9	Opaque LSA（链路本地范围）
10	Opaque LSA（本地区域范围）
11	Opaque LSA（AS 范围）

- **路由器 LSA（Router LSA）**——每一台路由器都会产生路由器 LSA 通告（如图 8-19 所示）。这个最基本的 LSA 通告列出了路由器所有的链路或接口，并指明了它们的状态和沿每条链路方向出站的代价，以及该链路上所有已知的 OSPF 邻居。这些 LSA 通告只会在始发它们的区域内部进行泛洪扩散。通过命令 **show ip ospf database router** 可以查看数据库中列出了所有路由器 LSA 通告。示例 8-11 显示了这条命令的一个变形，在命令后加了一个变量指定一个路由器 ID，从而观察到单台路由器 LSA 通告的详细信息。在该例及其后面的一些图示中，显示了记录在链路状态数据库中的完整的 LSA 信息。关于对所有 LSA 字段的介绍，请参考 8.1.7 小节中的讲述。

[1] 在 IOS11.3AA 版本中引入了组步调命令 **timers lsa-group-pacing**；而从 IOS12.2（4）T 版本开始命令改变为 **timers pacing lsa-group**。

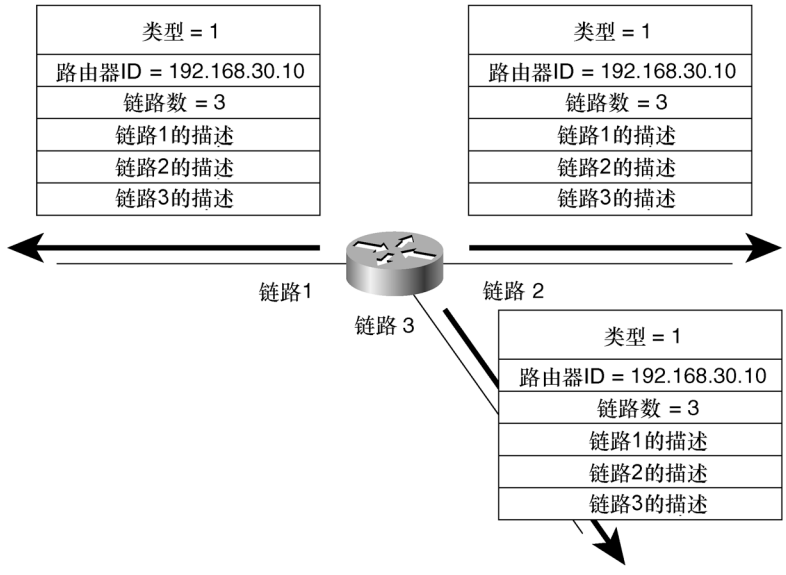


图 8-19 路由器 LSA 描述了所有的路由器接口

示例 8-11 通过命令 show ip ospf database router 可以显示出一个链路状态数据库中的路由器 LSA 通告

```
Homer#show ip ospf database router 192.168.30.10

       OSPF Router with ID (192.168.30.50) (Process ID 1)

                    Router Link States (Area 0)

Routing Bit Set on this LSA
LS age: 680
Options: (No TOS-capability)
LS Type: Router Links
Link State ID: 192.168.30.10
Advertising Router: 192.168.30.10
LS Seq Number: 80001428
Checksum: 0x842A
Length: 60
Area Border Router
 Number of Links: 3

  Link connected to: another Router (point-to-point)
   (Link ID) Neighboring Router ID: 192.168.30.80
   (Link Data) Router Interface address: 192.168.17.9
    Number of TOS metrics: 0
     TOS 0 Metrics: 64

  Link connected to: a Stub Network
   (Link ID) Network/subnet number: 192.168.17.8
   (Link Data) Network Mask: 255.255.255.248
    Number of TOS metrics: 0
     TOS 0 Metrics: 64

  Link connected to: a Transit Network
   (Link ID) Designated Router address: 192.168.17.18
   (Link Data) Router Interface address: 192.168.17.17
    Number of TOS metrics: 0
```

（待续）

```
      TOS 0 Metrics: 10

Homer#
```

请注意，在示例 8-11 和后面的几个 LSA 的显示中都有一行是“Routing Bit Set on this LSA”。路由选择位不是 LSA 本身的一部分；它是 IOS 软件中用来作内部网络维护的位，表示通过这个 LSA 通告到目的地的路由是有效的。因此，当你看到“Routing Bit Set on this LSA”时，表示到达这个目的地的路由在路由选择表中。

- **网络 LSA（Network LSA）**——每一个多路访问网络中的指定路由器（DR）将会产生网络 LSA 通告，如图 8-20 所示。正如前面讨论的，DR 路由器可以看作一个“伪”节点，或是一个虚拟路由器，用来描绘一个多路访问网络和与之相连的所有路由器。从这个角度来看，一条网络 LSA 通告也可以描绘一个逻辑上的“伪”节点，就像一条路由器 LSA 通告描绘一个物理上的单台路由器一样。网络 LSA 通告列出了所有与之相连的路由器，包括 DR 路由器本身。像路由器 LSA 一样，网络 LSA 也仅仅在产生这条网络 LSA 的区域内部进行泛洪扩散。示例 8-12 中使用命令 **show ip ospf database network** 可以查看一条网络 LSA 通告的信息。

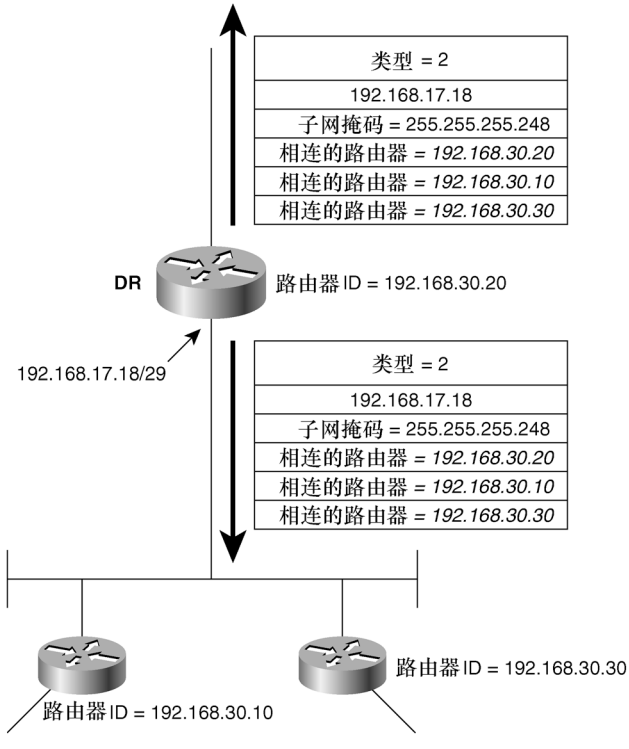


图 8-20 DR 路由器始发了一条可以表示一个多路访问网络和与之相连的所有路由器的网络 LSA 通告

示例 8-12 网络 LSA 通告可以通过命令 show ip ospf database network 来查看

```
Homer#show ip ospf database network 192.168.17.18

       OSPF Router with ID (192.168.30.50) (Process ID 1)

                Net Link States (Area 0)
```

（待续）

```
Routing Bit Set on this LSA
LS age: 244
Options: (No TOS-capability)
LS Type: Network Links
Link State ID: 192.168.17.18 (address of Designated Router)
Advertising Router: 192.168.30.20
LS Seq Number: 800001BF
Checksum: 0x60AC
Length: 32
Network Mask: /29
      Attached Router: 192.168.30.20
      Attached Router: 192.168.30.10
      Attached Router: 192.168.30.30

Homer#
```

请注意，和路由器 LSA 不同，网络 LSA 中没有度量字段。正如本章前面所讲述的，这是因为从 LSA 中表示的伪节点到任何相连的路由器的代价永远是 0。

- **网络汇总 LSA（Network Summary LSA）**——是由 ABR 路由器始发的。ABR 路由器将发送网络汇总 LSA 到一个区域，用来通告该区域外部的目的地址（如图 8-21 所示）。实际上，这些网络汇总 LSA 就是 ABR 路由器告诉在与之相连的区域内的内部路由器它所能到达的目的地址的一种方法。一台 ABR 路由器也可以通过网络汇总 LSA 向骨干区域通告与它相连的区域内部的目的地址。在一个区域外部，仍然在一个 OSPF 自主系统内部的缺省路由也可以通过这种 LSA 类型来通告。使用命令 **show ip ospf database summary** 可以显示链路状态数据库中的网络汇总 LSA 信息，参见示例 8-13 所示。

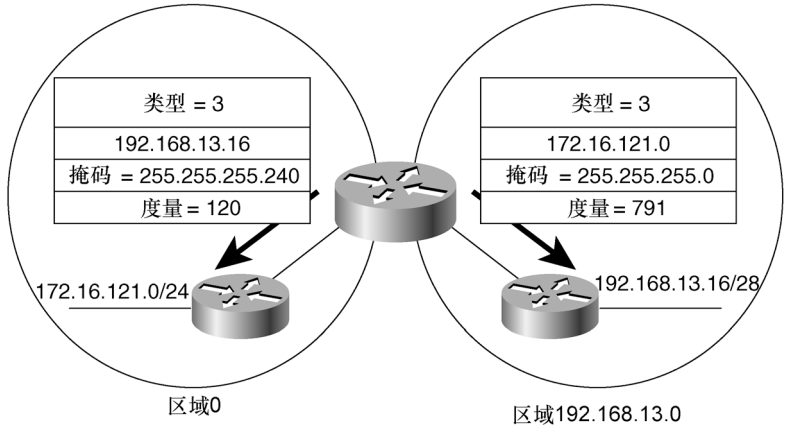


图 8-21 一台 ABR 路由器始发了一条网络汇总 LSA 来描述域间通信的目的地址

示例 8-13 可以通过命令 show ip ospf database summary 来查看网络汇总 LSA 通告的信息

```
Homer#show ip ospf database summary 172.16.121.0

       OSPF Router with ID (192.168.30.50) (Process ID 1)

                   Summary Net Link States (Area 0)

Routing Bit Set on this LSA
```

（待续）

```
LS age: 214
Options: (No TOS-capability)
LS Type: Summary Links(Network)
Link State ID: 172.16.121.0 (summary Network Number)
Advertising Router: 192.168.30.60
LS Seq Number: 800000B1
Checksum: 0xE864
Length: 28
Network Mask: /24
   TOS: 0 Metric: 791
```

当一台 ABR 路由器始发一条网络汇总 LSA 时，将包括从它本身到正在通告的这条 LSA 的目的地所耗费的代价。ABR 路由器即使知道它有多条路由可以到达目的地，它也只会为这个目的地始发单条网络汇总 LSA 通告。因此，如果一台 ABR 路由器在与它本身相连的区域内有多条路由可以到达目的地，那么它将只会始发单一的一条网络汇总 LSA 到骨干区域，而且这条网络汇总 LSA 是上述多条路由中代价最低的。同样地，如果一台 ABR 路由器经过骨干区域从其他的 ABR 路由器收到多条网络汇总 LSA，那么这台始发的 ABR 路由器将会选择这些 LSA 通告中代价最低的 LSA，并且将把这个 LSA 的最低代价通告给与它相连的非骨干区域。

当其他的路由器从一台 ABR 路由器收到一条网络汇总 LSA 通告时，它并不运行 SPF 算法。相反地，它只是简单地加上从它到那台 ABR 路由器之间路由的代价，并将这个代价包含在这个 LSA 通告当中。通过 ABR 路由器，到达所通告的目的地的路由连同所计算的代价一起被记录进了路由表。这个行为——依赖中间路由器代替确定到达目的地的全程路由（full route）的做法——其实是距离矢量协议的行为。因此，虽然在一个区域内部 OSPF 协议是一个链路状态协议，但是它却使用了距离矢量的算法来查找域间路由。[1]

- **ASBR 汇总 LSA（ASBR Summary LSA）**——也是由 ABR 路由器始发的。ASBR 汇总 LSA 除了所通告的目的地是一台 ASBR 路由器而不是一个网络外，其他的和网络汇总 LSA 都是一样的，如图 8-22 所示。使用命令 **show ip ospf database asbr-summary** 可以查看 ASBR 汇总 LSA 的信息，参见示例 8-14 所示。这里要注意，在这个图示中，目的地是一个主机地址，并且掩码是 0；通过 ASBR 汇总 LSA 通告的目的地将总是一个主机地址，因为它是一条到达一台路由器的路由。

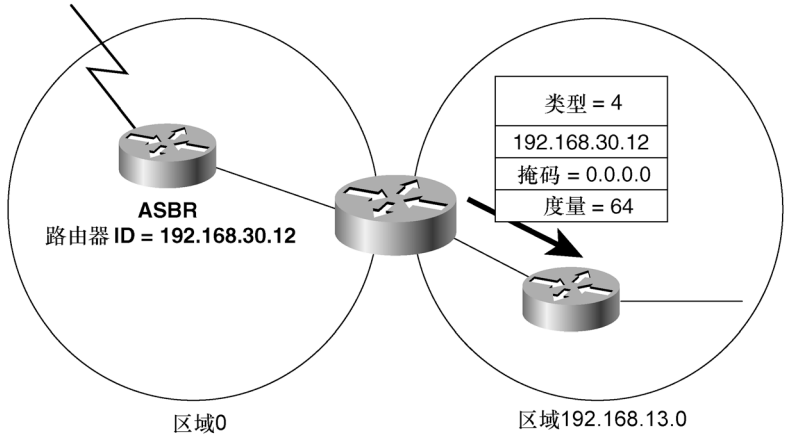


图 8-22 ASBR 汇总 LSA 通告到达 ASBR 路由器的路由

[1] 这个距离矢量的行为就是 OSPF 协议为什么需要一个骨干区域和为什么要求所有的域间通信量都必须通过骨干区域的原因。通过把这些区域构成一个本质上像星型一样的网络拓扑，可以避免距离矢量协议中易于出现的路由环路。

示例 8-14 可以通过命令 show ip ospf database asbr-summary 来查看 ASBR 汇总 LSA 通告的信息

```
Homer#show ip ospf database asbr-summary

       OSPF Router with ID (192.168.30.50) (Process ID 1)

                   Summary ASB Link States (Area 0)

Routing Bit Set on this LSA
LS age: 1640
Options: (No TOS-capability)
LS Type: Summary Links (AS Boundary Router)
Link State ID: 192.168.30.12 (AS Boundary Router address)
Advertising Router: 192.168.30.20
LS Seq Number: 80000009
Checksum: 0xF450
Length: 28
Network Mask: /0
   TOS: 0 Metric: 64
--More-
```

- **自主系统外部 LSA（Autonomous System External LSA）**——或者称为外部 LSA（External LSA），是始发于 ASBR 路由器的，用来通告到达 OSPF 自主系统外部的目的地或者 OSPF 自主系统外部的缺省路由[1]的 LSA，如图 8-23 所示。回顾一下示例 8-9，可以发现自主系统外部 LSA 是链路状态数据库中惟一不与具体的区域相关联的 LSA 通告。外部 LSA 通告将在整个自主系统中进行泛洪扩散。使用命令 **show ip ospf database external** 可以查看 AS 外部 LSA 的信息，参见示例 8-15 所示。

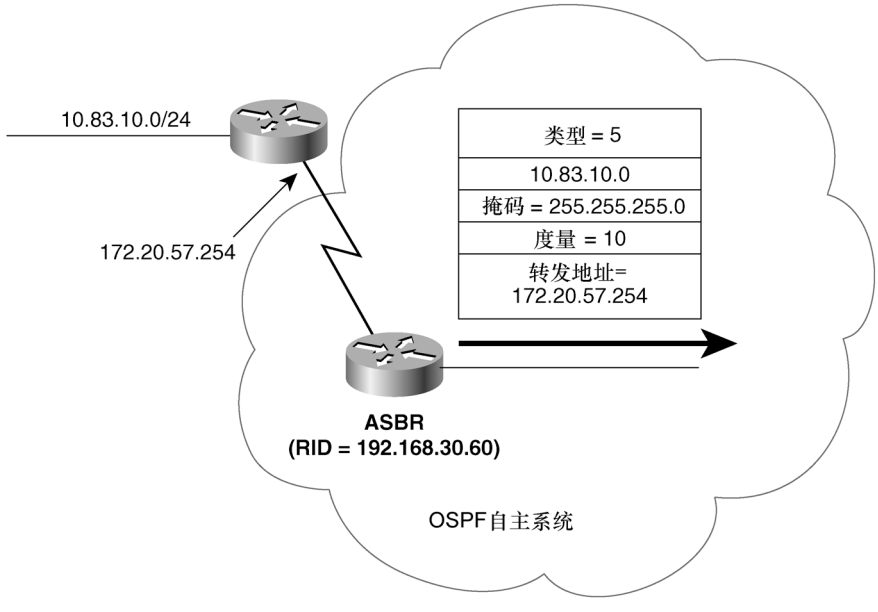


图 8-23 自主系统外部 LSA 用来通告到达 OSPF 自主系统外部的目的地

[1] 缺省路由是指在路由表中，如果没有更具体的路由存在就可以选用的路由。OSPF 协议和 RIP 协议使用 IP 地址 0.0.0.0 来表示一个缺省路由。更详细的信息请参考第 12 章的内容。

示例 8-15 可以通过命令 show ip ospf database external 来查看自主系统外部 LSA 通告的信息

```
Homer#show ip ospf database external 10.83.10.0

       OSPF Router with ID (192.168.30.50) (Process ID 1)

                    AS External Link States

Routing Bit Set on this LSA
LS age: 1680
Options: (No TOS-capability)
LS Type: AS External Link
Link State ID: 10.83.10.0 (External Network Number)
Advertising Router: 192.168.30.60
LS Seq Number: 80000D5A
Checksum: 0x7A1C
Length: 36
Network Mask: /24
      Metric Type: 1 (Comparable directly to link state metric)
      TOS: 0
      Metric: 10
      Forward Address: 172.20.57.254
      External Route Tag: 0
Homer#
```

- **组成员 LSA（Group Membership LSA）**——是用在 OSPF 协议的一个增强版本——组播 OSPF 协议（MOSPF 协议）中的。[1] MOSPF 协议将数据包从一个单一的源地址转发到多个目的地，或者是一组共享 D 类组播地址的成员。虽然 Cisco 路由器支持其他的组播路由选择协议，但是在编写本书时还不支持 MOSPF 协议。由于这个原因，本书将不介绍 MOSPF 协议，也不介绍组成员 LSA 通告。
- **NSSA 外部 LSA（NSSA External LSA）**——是指在非纯末梢区域（Not-So-Stubby Area，NSSA）内始发于 ASBR 路由器的 LSA 通告。NSSA 区域将在后面的章节介绍。正像 8.1.7 小节中所介绍的那样，NSSA 外部 LSA 通告几乎和自主系统外部 LSA 通告是相同的。只是不像自主系统外部 LSA 通告那样在整个 OSPF 自主系统内进行泛洪扩散，NSSA 外部 LSA 通告仅仅在始发这个 NSSA 外部 LSA 通告的非纯末梢区域内部进行泛洪扩散。可以通过命令 **show ip ospf database nssa-external** 来显示 NSSA 外部 LSA 通告的信息，参见示例 8-16 所示。

示例 8-16 可以通过命令 show ip ospf database nssa-external 来查看 NSSA 外部 LSA 通告的信息

```
Morisot#show ip ospf database nssa-external

         OSPF Router with ID (10.3.0.1) (Process ID 1)

                     Type-7 AS External Link States (Area 15)

LS age: 532
Options: (No TOS-capability, No Type 7/5 translation, DC)
LS Type: AS External Link
Link State ID: 10.0.0.0 (External Network Number)
```

（待续）

[1] John Moy, “Multicast Extensions to OSPF,” RFC 1584，1994 年 3 月。

```
Advertising Router: 10.3.0.1
LS Seq Number: 80000001
Checksum: 0x9493
Length: 36
Network Mask: /16
      Metric Type: 2 (Larger than any link state path)
      TOS: 0
      Metric: 100
      Forward Address: 10.3.0.1
      External Route Tag: 0

--More-
```

- **外部属性 LSA（External Attributes LSA）**——是被提议作为运行内部 BGP 协议（iBGP 协议）的另一种选择，以便用来传送 BGP 协议的信息穿过一个 OSPF 域。这个 LSA 从来没有在大范围部署过，IOS 软件也不支持该 LSA。
- **Opaque LSA**——是由标准的 LSA 头部后面跟随专用信息组成的一类 LSA。[1]这个信息字段可以直接由 OSPF 协议使用，或者由其他应用分发信息到整个 OSPF 域间接使用。Opaque LSA 类型现在用于对 OSPF 增加可变的扩展特性，例如在 MPLS 网络中应用的流量工程参数。

2. 末梢（Stub）区域

一个学习到外部目的地路由信息的 ASBR 路由器，将通过在整个 OSPF 自主系统中泛洪扩散自主系统外部 LSA 来通告那些外部的目的路由信息。在大多数的实际案例中，这些外部 LSA 通告可能会在每台路由器的链路状态数据库中构成较大百分比的 LSA 数量。例如，在示例 8-10 中显示的那个链路状态数据库中有 580 个 LSA（约 40%）是外部 LSA 通告。

如图 8-24 所示，并不是每一台路由器都需要了解所有外部目的地的信息。不管外部目的地在哪里，在区域 2 中的路由器都必须发送数据包到 ABR 路由器，以便到达那台 ASBR 路由器。在这种情况下，区域 2 可以被配置成为一个末梢区域。

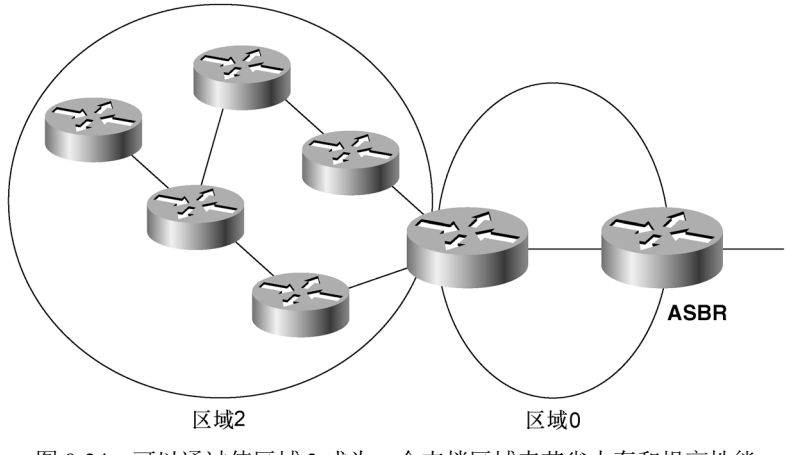


图 8-24 可以通过使区域 2 成为一个末梢区域来节省内存和提高性能

末梢区域是一个不允许 AS 外部 LSA 通告在其内部进行泛洪扩散的区域。如果在一个区域里没有学到类型 5 的 LSA 通告，那么类型 4 的 LSA 通告也是不必要的了，因此这些 LSA

[1] Rob Coltun, “The OSPF Opaque LSA Option,” RFC 2370，1998 年 7 月。

通告也将被阻塞。位于末梢区域边界的 ABR 路由器将使用网络汇总 LSA 向这个区域通告一个简单的缺省路由（目的地址是 0.0.0.0）。在区域内部路由器上，所有和域内或域间路由不能匹配的目的地址都将最终匹配这条缺省路由。由于缺省路由是由类型 3 的 LSA 通告传送的，因此它将不会被通告到这个区域的外部去。

由于在一个末梢区域里，路由器的链路状态数据库的大小被减小了，因此，这些路由器的性能将得到提高，并且内存也得到节省。当然，在一个含有大量类型 5 的 LSA 通告的 OSPF 域里，这种改进将更加显著。然而，在末梢区域中也有 4 个限制条件：

- 和所有的区域一样，一个末梢区域内部的所有路由器也必须拥有相同的链路状态数据库。为了确保满足这个条件，所有末梢区域内的路由器都会在它们的 Hello 数据包中设置一个标志——就是 E-bit 位，并将它设置为 0。这样，这些末梢区域路由器将不接受其他路由器发送的任何 E-bit 为 1 的 Hello 数据包。结果，没有配置成一个末梢区域路由器的任何路由器之间将无法成功建立邻接关系。
- 虚链路不能在一个末梢区域内进行配置，也不能穿过一个末梢区域。
- 末梢区域内的路由器不能是 ASBR 路由器。这个限制条件是很容易直观地理解的，因为 ASBR 路由器会产生类型 5 的 LSA 通告，而在一个末梢区域内不能存在类型 5 的 LSA 通告。
- 一个末梢区域可以拥有多台 ABR 路由器，但是因为缺省路由的原因，区域内部路由器将不能确定哪一台路由器才是到达 ASBR 路由器的最优网关。

（1）完全末梢区域

如果通过阻塞类型 5 和类型 4 的 LSA 传播到一个区域的方法来节省内存的话，那么要是能够把类型 3 的 LSA 也阻塞掉，不是可以节省更多的内存吗？对于这个问题，Cisco 借助于末梢区域的概念提出了称为完全末梢区域的概念。

完全末梢区域（totally stubby area）不仅使用缺省路由到达 OSPF 自主系统外部的目的地址，而且使用缺省路由到达这个区域外部的所有目的地址。一个完全末梢区域的 ABR 将不仅阻塞 AS 外部 LSA，而且阻塞所有的汇总 LSA——除了通告缺省路由的那一条类型 3 的 LSA。

（2）非纯末梢区域

在图 8-25 中，带有一些末梢网络的某台路由器必须通过区域 2 的其中一台路由器和 OSPF 网络相连。但是，该路由器仅支持 RIP 协议，因此，区域 2 的那台路由器将同时运行 RIP 协议和 OSPF 协议，并利用路由重新分配的方法把该路由器的那些末梢网络注入到 OSPF 域。不幸的是，这种配置将使区域 2 中的那台路由器成为一台 ASBR 路由器，因此，区域 2 也就不再是一个末梢区域了。

在这里，RIP 协议的宣告者并不需要学习 OSPF 域的路由，而只需要有一条缺省路由指向那台区域 2 的路由器就足够了。但是，OSPF 域内的路由器为了能够正确地将数据包转发到 RIP 路由器的那些目的地址，它们必须学习到和 RIP 路由器相连的目的网络。

非纯末梢区域（Not-So-Stubby-Area，NSSA）[1]允许外部路由通告到 OSPF 自主系统内部，而同时保留自主系统其余部分的末梢区域特征。为了做到这一点，在 NSSA 区域内的 ASBR 将始发类型 7 的 LSA 用来通告那些外部的目的网络。这些 NSSA 外部 LSA 将在整个 NSSA 区域中进行泛洪扩散，但是会在 ABR 路由器的地方被阻塞。

[1] Rob Coltun and Vince Fuller, “The OSPF NSSA Option,” RFC 1587，1994 年 3 月。

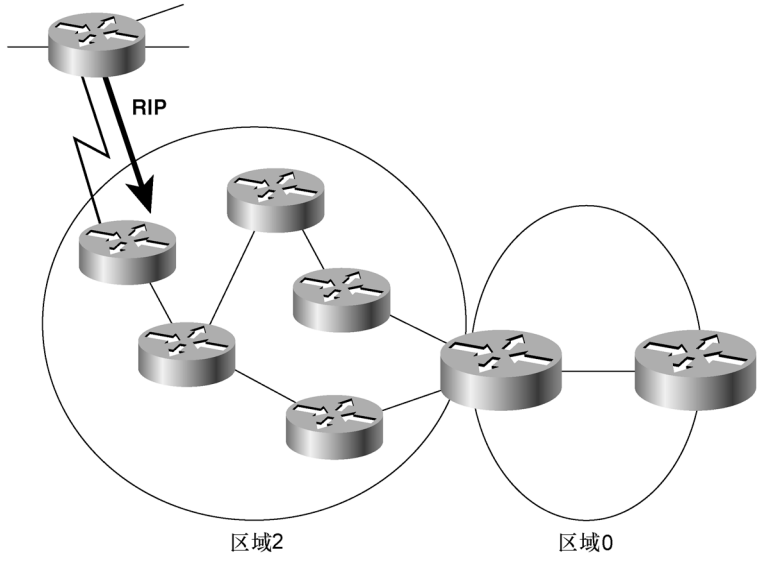


图 8-25 因为有一些 OSPF 域外部的目的网络必须在区域 2 的路由器上通过路由重新分配的方式注入到 OSPF 域中，因此区域 2 就不再满足末梢区域的条件了

NSSA 外部 LSA 在它的头部有一个称为 P-bit 位的标志。NSSA ASBR 路由器可以设置或清除这个 P-bit 位。如果一台 NSSA ABR 路由器收到一条 P-bit 设置为 1 的类型 7 的 LSA 数据包，那么它将把这条类型 7 的 LSA 转换成为类型 5 的 LSA，并且将这条 LSA 泛洪扩散到其他的区域中去（参见图 8-26）。如果这个 P-bit 位被设置为 0，那么将不会转换这条类型 7 的 LSA，而且这条类型 7 的 LSA 携带的目的地址也不能通告到这个 NSSA 区域的外部。这个选项允许我们设计一个 NSSA，使那个区域学到的外部目的地址仅仅被那个区域知晓。

NSSA 区域是在 IOS 软件 11.2 版及其以后的版本才被支持的。

表 8-5 总结了每一种区域内允许泛洪扩散的 LSA 类型。

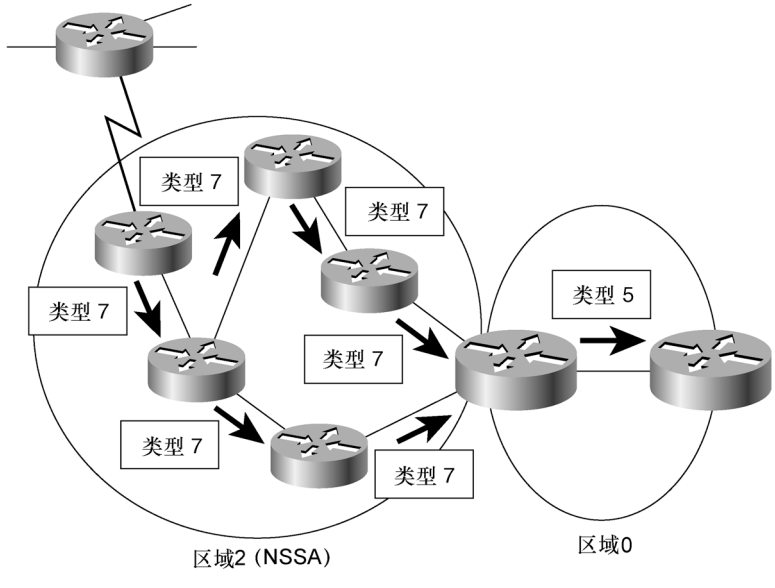


图 8-26 在 NSSA 区域内的 ASBR 路由器将会始发 NSSA 外部 LSA。如果一条 NSSA 外部 LSA 的 P-bit 位设置了，那么 ABR 路由器将会把这条 NSSA 外部 LSA 转换为一条 AS 外部 LSA

表 8-5 每一种区域内允许泛洪扩散的 LSA 类型

区域类型	1 和 2	3	4	5	7
骨干区域（区域 0）	允许	允许	允许	允许	不允许
非骨干区域，非末梢区域	允许	允许	允许	允许	不允许
末梢区域	允许	允许	不允许	不允许	不允许
完全末梢区域	允许	不允许*	不允许	不允许	不允许
NSSA	允许	允许	允许	不允许	允许

* 只有一个例外，就是在每一台 ABR 路由器上利用一个类型 3 的 LSA 来通告缺省路由。

8.1.4 路由表

根据链路状态数据库中的 LSA 信息，路由器使用 Dijkstra 算法来计算一棵最短路径树。第 4 章已经比较详细地讲述了 Dijkstra 算法，如果需要查看关于 OSPF 协议计算 SPF 树的完整描述，请参考 RFC 2328 中的第 16.1 节。

OSPF 协议是基于路由器的每一个接口指定的度量值来决定最短路径的，这里的度量值指的是接口指定的代价（cost）。一条路由的代价是指沿着到达目的网络的路由路径上所有出站接口的代价之和。RFC 2328 没有专门为代价指定任何值。Cisco 路由器使用 10^8/BW 作为缺省的 OSPF 代价，这里的 BW 是指在路由器接口上配置的带宽，10^8 是一个参考带宽。正如前面所讲述的，缺省的参考带宽可以通过命令 **auto-cost reference-bandwidth** 来更改。如果计算所得的结果是分数的话就采用四舍五入的方法取整数值。表 8-6 中显示了一些典型接口根据这种计算方式得出的缺省代价。

使用命令 **ip ospf cost** 可以替换缺省的自动进行的代价计算，这条命令可以给某个接口分配一个固定的代价。例如，在一个具有相同骨干链路速率的大型网络中，可以根据视距或线缆/光缆距离来分配链路代价。LSA 在 16 位的字段中记录代价，因此一个接口的总计代价范围可以是 1～65535。

表 8-6 Cisco 路由器的缺省接口代价

接口类型	代价（10^8/BW）
FDDI、快速以太网，任何接口>100Mbit/s	1
HSSI（45Mbit/s）	2
16Mbit/s 令牌环	6
以太网（10Mbit/s）	10
4Mbit/s 令牌环	25
T1（1.544Mbit/s）	64
DS0（64kbit/s）*	1562
56kbit/s*	1785
Tunnel（9kbit/s）	11111

* 假定串行接口的缺省带宽已被改变。

1. 目的类型（Destination Type）

每一个路由条目都可以被归类到目的类型中去。目的类型可以是网络也可以是路由器。网络条目（network entries）是数据包所要转发的目的网络地址。这些网络条目就是记录

到路由表中的目的网络地址，参见示例 8-17 所示。

示例 8-17 在路由表中的 OSPF 条目是网络目的类型

```
Homer#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route

Gateway of last resort is 192.168.32.2 to network 0.0.0.0

O E1 192.168.118.0/24 [110/94] via 192.168.17.74, 02:15:01, Ethernet0
O E1 10.0.0.0/8 [110/84] via 192.168.17.41, 02:15:01, Serial0.19
O E1 192.168.119.0/24 [110/94] via 192.168.17.74, 02:15:01, Ethernet0
O E2 172.19.0.0/16 [110/21] via 192.168.32.2, 02:15:01, Ethernet1
     172.21.0.0/16 is variably subnetted, 2 subnets, 2 masks
O E2   172.21.0.0/16 [110/801] via 192.168.21.6, 02:15:01, Serial1.724
O      172.21.121.0/24 [110/791] via 192.168.21.6, 04:18:30, Serial1.724
     172.16.0.0/16 is variably subnetted, 104 subnets, 7 masks
O      172.16.21.48/30 [110/844] via 192.168.21.10, 04:18:48, Serial1.725
O IA   172.16.30.61/32 [110/856] via 192.168.17.74, 02:15:19, Ethernet0
O IA   172.16.35.0/24 [110/865] via 192.168.17.74, 02:15:19, Ethernet0
C      172.16.32.0/24 is directly connected, Ethernet1
O      172.16.17.48/29 [110/74] via 192.168.17.74, 06:19:46, Ethernet0
O E1   172.16.46.0/24 [110/30] via 192.168.32.2, 02:15:19, Ethernet1
O      172.16.45.0/24 [110/20] via 192.168.32.2, 3d10h, Ethernet1
O IA   172.16.30.54/32 [110/1061] via 192.168.17.74, 02:15:21, Ethernet0
O      172.16.17.56/29 [110/84] via 192.168.17.74, 06:19:48, Ethernet0
O      172.16.54.0/24 [110/11] via 192.168.32.2, 3d10h, Ethernet1
O      172.16.55.0/24 [110/11] via 192.168.32.2, 3d10h, Ethernet1
O      172.16.52.0/24 [110/11] via 192.168.32.2, 3d10h, Ethernet1
O      172.16.53.0/24 [110/11] via 192.168.32.2, 3d10h, Ethernet1
C      172.16.25.28/30 is directly connected, Tunnel29
--More-
```

路由器条目（router entries）是到达 ABR 和 ASBR 路由器的路由。如果一台路由器需要发送数据包到一个区域外的目的地，那么它就必须知道如何到达一台 ABR 路由器；同样的，如果一台路由器需要发送数据包到一个 OSPF 域外部的目的地，那么它就必须知道如何到达一台 ASBR 路由器。路由器条目就包含了这个信息，并且路由器把路由器条目放在了一个单独的内部路由表中。这个路由表可以通过命令 **show ip ospf border-routers** 来查看，参见示例 8-18 所示。

正如示例 8-18 所显示的，这个内部的路由表看上去和其他普通路由表十分相似：也包含目的地、度量值、下一跳地址和出口接口。这里所不同的是，在这个内部的路由表中的所有目的地都是 ABR 和 ASBR 的路由器 ID。每一个条目都被打上区域内（i）或区域间（I）标志，用来表明这个条目的目的地是一台 ABR，或是一台 ASBR，或两者都是。所在的区域也被记录了，以便用来进行 SPF 算法的迭代查找。

示例 8-18 路由器条目放置在一个和网络条目相分开的内部表中，用来表示到达 ABR 和 ASBR 路由器的路由

```
Homer#show ip ospf border-routers

OSPF Process 1 internal Routing Table
```

（待续）

```
Codes: i - Intra-area route, I - Inter-area route
i 192.168.30.10 [74] via 192.168.17.74, Ethernet0, ABR, Area 0, SPF 391
I 192.168.30.12 [148] via 192.168.17.74, Ethernet0, ASBR, Area 0, SPF 391
I 192.168.30.18 [205] via 192.168.17.74, Ethernet0, ASBR, Area 0, SPF 391
i 192.168.30.20 [84] via 192.168.17.74, Ethernet0, ABR, Area 0, SPF 391
i 192.168.30.27 [781] via 192.168.21.6, Serial1.724, ASBR, Area 7, SPF 631
i 192.168.30.30 [74] via 192.168.17.74, Ethernet0, ABR/ASBR, Area 0, SPF 391
I 192.168.30.38 [269] via 192.168.17.74, Ethernet0, ASBR, Area 0, SPF 391
i 192.168.30.37 [390] via 192.168.21.10, Serial1.725, ASBR, Area 7, SPF 631
i 192.168.30.40 [84] via 192.168.17.74, Ethernet0, ABR/ASBR, Area 0, SPF 391
i 192.168.30.47 [400] via 192.168.21.10, Serial1.725, ASBR, Area 7, SPF 631
i 192.168.30.50 [74] via 192.168.17.41, Serial0.19, ABR/ASBR, Area 0, SPF 391
I 192.168.30.62 [94] via 192.168.17.74, Ethernet0, ASBR, Area 0, SPF 391
i 192.168.30.60 [64] via 192.168.17.41, Serial0.19, ABR/ASBR, Area 0, SPF 391
i 192.168.30.60 [790] via 192.168.21.10, Serial1.725, ABR/ASBR, Area 7, SPF 631
i 192.168.30.80 [10] via 192.168.32.5, Ethernet1, ABR/ASBR, Area 78, SPF 158
i 192.168.30.80 [10] via 192.168.17.74, Ethernet0, ABR/ASBR, Area 0, SPF 391
i 172.20.57.254 [10] via 192.168.32.2, Ethernet1, ASBR, Area 78, SPF 158
Homer#
```

2. 路径类型

每一条到达一个网络目的地的路由都可以被归类到 4 种路径类型中的一种。这些路径类型（path type）是区域内路径、区域间路径、类型 1 的外部路径和类型 2 的外部路径。

- **区域内路径（Intra-area path）**——是指在路由器所在的区域内就可以到达目的地的路径。
- **区域间路径（Inter-area path）**——是指目的地在其他区域但是还在 OSPF 自主系统内的路径。在示例 8-17 中，打上了 IA 标志的条目就是区域间路径，它总是至少通过一台 ABR 路由器。
- **类型 1 的外部路径（Type 1 external path，E1）**——是指目的地在 OSPF 自主系统外部的路径，在示例 8-17 中表示为 E1。当一条外部路由重新分配到任何一个自主系统时，它都必须指定一个对该自主系统中的路由选择协议有意义的度量值。在 OSPF 协议里，ASBR 路由器的责任是要给通告的外部路由指定一个代价值。对于类型 1 的外部路径来说，这个代价值是这条路由的外部代价加上到达 ASBR 路由器的路径代价之和。关于配置一台 ASBR 路由器使用 E1 类型的度量来通告一条外部路由（路由重新分配）的介绍将在第 11 章中讲述。
- **类型 2 的外部路径（Type 2 external path，E2）**——也是指目的地在 OSPF 自主系统外部的路径，但是在计算外部路由的度量时不再计入到达 ASBR 路由器的路径代价。

E1 和 E2 类型的路由给网络管理员提供了一个选项，这个选项是：选择计算到 ASBR 的内部代价重要，还是只选择外部路由的外部代价，而忽略到达 ASBR 的内部代价更重要。例如，“热土豆（hot potato）”路由选择（在网络最近的出口地点把到外部目的地址的数据包转发出该网络）通常要求 E1 度量，而如果你希望将数据包从到达它们的外部目的地址最近的地点转发出去，就应该使用 E2 度量。OSPF 外部路由在缺省条件下是 E2 类型路径。

在图 8-27 中，路由器 A 有两条到达外部目的网络 10.1.2.0 的路径。如果目的地址通过 E1 类型来通告，那么路径 A-B-D 的代价是 35（5+20+10），这条路径将比代价为 50（30+10+10）的路径 A-C-D 优先。如果目的地址通过 E2 类型来通告，那么到达 ASBR 路由器的那两条内部路径的代价将被忽略。在这个实例中，路径 A-B-D 的代价是 30（20+10），而路径 A-C-D

的代价为 20（10+10）。显然，后者是优先路径。

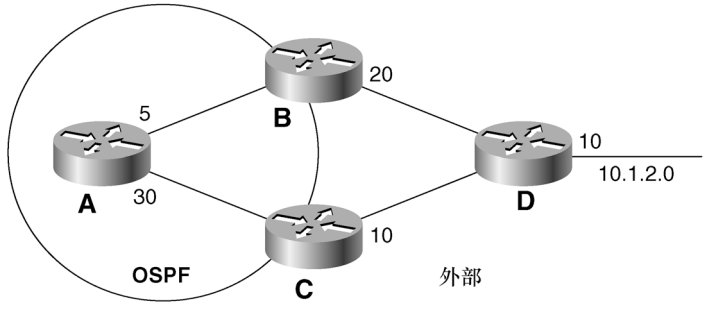


图 8-27 如果使用 E1 类型的度量来通告到达外部网络 10.1.2.0 的路由，那么路由器 A 将选择路由器 B 作为最近的 ASBR。如果使用 E2 类型的度量来通告外部目的网络，那么路由器 C 将被选为最近的 ASBR

3. 路由表的查找

当一台 OSPF 路由器检查一个数据包的目的地址时，它将通过下面的步骤来选择最优的路由：[1]

（1）选择可以和目的地址最精确匹配的路由。例如，如果路由表中存在路由条目 172.16.64.0/18、172.16.64.0/24 和 172.16.64.192/27，而目的地址是 172.16.64.205，那么最后一个路由条目将被选中。最精确的匹配应该总是最长匹配——拥有最长的地址掩码的路由。路由条目可以是主机地址、子网地址、网络地址、超网地址，或者缺省地址。如果路由器没有发现匹配的条目，那么它将发送一个 ICMP 目的不可达的消息给那个数据包的源地址，并且把这个数据包丢弃。

（2）通过排除次优的路径类型来剪除（prune）可选择条目的集合。路径类型根据下面的次序排列优先级，①表示最高的优先级，而④表示最低的优先级：

① 区域内路径；

② 区域间路径；

③ E1 外部路径；

④ E2 外部路径。

如果在最后的路由子集中还有多条等代价、等价路径类型的路由存在，那么 OSPF 协议将会利用它们。缺省条件下，Cisco 路由器可以在最多 16 条等代价的路径上实现负载均衡（老的 IOS 版本中是 4 条），这个数值可以通过命令 **maximum-paths** 来改变，改变的范围是 1～6。[2]

8.1.5 认证

OSPF 协议对邻居路由器之间交换的所有数据包都具有认证的能力。认证可以是简单的口令认证或 MD5 加密校验和认证。这些认证的方法在第 6 章中已经讲述过了，本章将在后面的配置一节中给出一个配置 OSPF 认证的例子。

---

[1] 这里描述的路由表查找过程是和 RFC 2328 一致的。早期的 OSPF RFC 规定首先要创建一个匹配路由的集合，然后再选择优先的路径类型，最后再进行最长匹配。

[2] 在写这个版本的书时，Cisco IOS 软件已经支持通过命令 **maximum-paths** 将最大值设为 6～16。

8.1.6 按需电路上的 OSPF

OSPF 协议每隔 10s 发送一次 Hello 数据包，并且每隔 30min 重新刷新一次它的 LSA。这些功能用来维护邻接关系，以便确保链路状态数据库的精确，而且它比像 RIP 这样传统的距离矢量协议使用的带宽要少得多。然而，即使是一个很小的通信量，也不希望在按需电路上存在——例如像在 X.25 SVC、ISDN 和拨号电路等即用即连（usage-sensitive connection）的电路。这些链路可能根据连接次数或通信量或两者皆有来循环计算费用，因而，网络管理员必须减少它们的上线时间。

一个在按需电路上实用的 OSPF 协议的增强特性是，使 OSPF 具有抑制 Hello 数据包和 LSA 重刷新的能力，以便链路不需要永久的有效。[1]虽然这个增强特性是为即用即连的链路设计的，但是它在任何带宽有限的链路上也可能是有用的。[2]

按需电路上的 OSPF 将会激活一条按需链路去执行最初的数据库同步，随后只会激活这条链路去泛洪扩散产生某些变化的 LSA。这些变化是：

- LSA 的可选字段发生了变化；
- 在老化时间达到 MaxAge 时收到了一个已经存在的 LSA 通告的新实例（instance）；
- LSA 头部的长度字段发生了变化；
- LSA 的内容发生了变化，但不包括 20 个八位组字节的头部、校验和或者序列号。

由于没有周期性的 Hello 数据包可以交换（Hello 数据包只有在链路激活时才能使用），OSPF 协议必须有一个可达性的假定。也就是说，OSPF 协议必须假定在需要链路连接的时候那条按需电路是可用的和有效的。但是在一些情况下，链路可能并不能立即可用和有效。例如，一个拨号链路可能正在使用，一个 BRI 链路的两个 B 信道可能也在使用，或者 X.25 所允许使用的最大的 SVC 电路数也都在使用。在这些情形下，链路并不是因为链路失效而变得不可用，而是因此链路太忙而变得不可用，这也是这种链路的正常特点，可以称为链路过忙（oversubscribed）。

OSPF 协议将不会把链路过忙的按需链路报告为失效，因而数据包转发到一条过忙的链路上将会被丢弃而不是放入缓冲队列排队。这样做是有道理的，因为 OSPF 没有办法预先知道那条繁忙的链路什么时候可以再次变得可用，一连串数据包转发到这个不可用的接口上就会导致它们的缓冲区溢出。

关于接口状态机和邻居状态机，以及泛洪扩散过程的处理必须有几处修改，以便支持 OSPF 协议运行在按需电路上（更详细的内容请参考 RFC1793）。在 LSA 通告的数据包格式里，也有两个地方需要修改。

首先，如果 LSA 通告没有通过按需电路进行周期性的重复刷新，那么经过 LSA 的最大生存时间（MaxAge）后，在这条链路的另一端将不会有路由器宣称这条 LSA 无效。OSPF 协议更改了 LSA 的 Age 字段，它将 Age 字段的更高一位指定为 DoNotAge 位，以便解决这种情况的发生。当一条 LSA 在按需电路上进行泛洪扩散时，传送的路由器将把 DoNotAge 设置为 1。这样，当这条 LSA 要泛洪扩散到这条链路另一端的所有路由器时，Age 字段通常会

[1] John Moy, “Extending OSPF to Support Demand Circuits,” RFC 1793，1995 年 4 月。

[2] 虽然按需电路上的 OSPF 可能被配置在任何接口上，但是，在多路访问类型的网络上不能抑制 Hello 数据包的发送——如果这么做会阻碍 DR 处理的一些相应功能。结果是，这个增强特性仅仅在点到点和点到多点类型的网络上才有实际用途。

增加一个 InfTransDelay 指定的秒数。[1] 但是，当一条 LSA 被安置到路由器的链路状态数据库中后，这条 LSA 将不再像其他 LSA 一样老化。

第二个修改来自于第一个修改。因为所有的路由器必须能够正确地识别这个更改的 DoNetAge 位，因此在所有的 LSA 中增加了一个新的标志，称为 Demand Circuit 位（DC-bit）。通过在所有 LSA 发起的时候设置这个标志位，路由器就可以通知其他路由器它是能够支持按需电路上的 OSPF 协议的。

按需电路上的 OSPF（即指定老化字段的高位为 DoNotAge 位）带来的一个另外的好处是，我们现在可以在一个稳定的网络拓扑中减少泛洪扩散。在某个接口上使用命令 **ip ospf flood-reduction**，DoNotAge 位就可以设置在这个接口通告出去的 LSA 中，因此这些 LSA 在它们发生变化前不会重新刷新。

8.1.7 OSPF 的数据包格式

OSPF 数据包是由多重封装构成的，解析一个 OSPF 数据包就像给洋葱剥皮一样。正如图 8-28 所示，这个“洋葱”的外面一层是 IP 包的头部。封装在 IP 头部内的是 5 种 OSPF 数据包类型中的一种。每一种数据包类型都是由一个 OSPF 数据包头开始的，这个 OSPF 数据包头对于所有的数据包类型都是相同的。紧跟 OSPF 数据包头之后的是 OSPF 数据包数据，并且根据数据包类型的不同会有所不同。每一种数据包类型都有许多的特有类型字段（type-specific fields），后跟更多的数据包数据。在 Hello 数据包中，这些数据包数据包含的是邻居路由器的列表。在链路状态请求数据包中包含的是一系列描述被请求的 LSA 的字段。在链路状态更新数据包中包含的是一个 LSA 的列表，如图 8-28 中所示的那样。这些 LSA 依次都含有它们自己的头部和特有类型数据字段。数据库描述和链路状态确认数据包将包含有一个 LSA 头部的列表。

这里要注意，在一个网络上，OSPF 数据包只能在邻居节点之间进行信息交换，它们从来都不会转发到始发它们的节点所在的网络。

图 8-29 显示了一台协议分析仪捕获到的传送 OSPF 数据的数据包的 IP 头部，表明 OSPF 的协议号是 89。从这里可以看出，当 OSPF 数据包多播发生时，它们的 TTL 设置为 1。既然一个 OSPF 数据包永远不应该跃过最近的邻居路由器转发，那么设置 TTL 为 1 可以帮助 OSPF 数据包确保自己的转发不会超过一跳。有一些路由器通过设置优先位（precedence bit）来运行一些具有优先次序的数据包进程。例如，这里的优先位可能是加权公平队列（Weighted Fair Queuing，WFQ）和加权随机预先检测（Weighted Random Early Detection，WRED）等。正如图 8-29 所显示的，OSPF 将设置优先位为互连网络控制（Internetwork Control，110b），以便这些具有优先次序的数据包进程给 OSPF 数据包一个高的优先级。

本小一节将从数据包头部开始，详细介绍这 5 种类型的 OSPF 数据包。随后的章节将详细介绍 LSA 数据包类型。在 Hello 数据包、数据库描述数据包和所有的 LSA 数据包中都含有一个可选（option）字段，这个字段的格式在所有的实例中都是一样的，因此将单独用一小节来详细介绍它。

1. 数据包头部

所有 OSPF 数据包都是由一个 24 个八位组字节的头部开始的，如图 8-30 所示。

[1] 注意，这意味着 MaxAge 实际上是 MaxAge + DoNotAge。

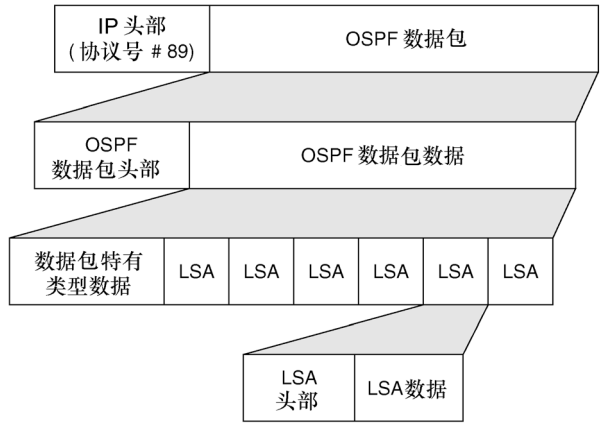


图 8-28　一个 OSPF 数据包由一系列封装组成

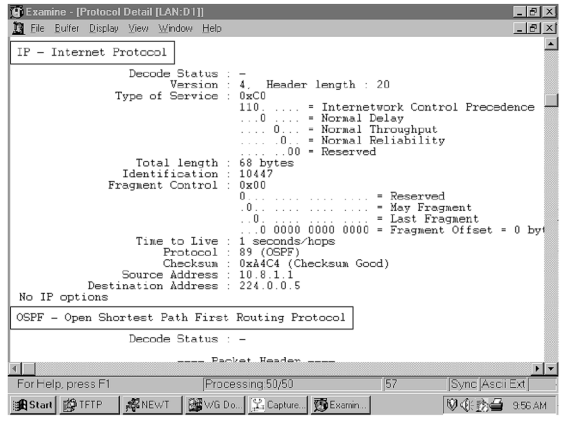
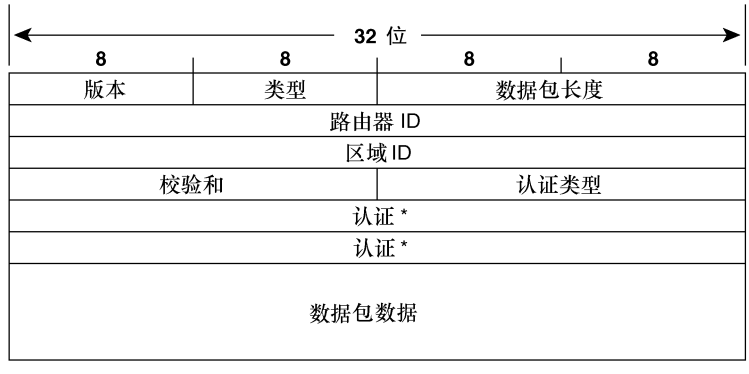


图 8-29　OSPF 使用的协议号是 89。OSPF 协议将 IP 头部的 TTL 值设置为 1，并且把优先位设置成互连网络控制



如果认证类型 = 2, 那么认证字段就是:

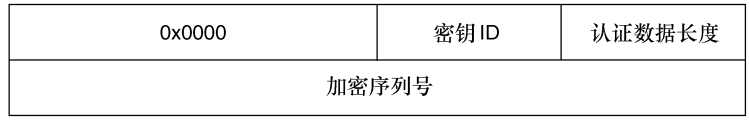


图 8-30　OSPF 包头

- **版本（Version）**——是指 OSPF 的版本号。OSPF 的版本号是 2。对于 IPv6 的路由

选择是 OSPF 版本 3，OSPFv3 将在下一章中讲述。

- **类型（Type）**——指出跟在头部后面的数据包类型。根据出现在类型字段的数字，表 8-7 列出了这 5 种数据包类型。

表 8-7 OSPF 数据包类型

类型代码	描述
1	Hello
2	数据库描述
3	链路状态请求
4	链路状态更新
5	链路状态确认

- **数据包长度（Packet Length）**——是指 OSPF 数据包的长度，包括数据包头部的长度，以八位组字节计。
- **路由器 ID（Router ID）**——是指始发路由器的 ID。
- **区域 ID（Area ID）**——是指始发数据包的路由器所在的区域。如果数据包是在一个虚链路上发送的，那么区域 ID 就为 0.0.0.0，也就是骨干区域的 ID，因为虚链路被认为是骨干的一部分。
- **校验和（Checksum）**——是指一个对整个数据包（包括包头）的标准 IP 校验和。
- **认证类型（AuType）**——是指正在使用的认证模式。

表 8-8 中列出了可能的认证模式。

表 8-8 OSPF 认证类型

认证类型代码（AuType）	认证类型
0	空（没有认证）
1	简单（明文）口令认证
2	加密校验和（MD5）

- **认证（Authentication）**——是指数据包认证的必要信息，认证可以是 AuType 字段中指定的任何一种认证模式。如果 AuType=0，将不检查这个认证字段，因此可以包含任何内容。如果 AuType=1，这个字段将包含一个最长为 64 位的口令。如果 AuType=2，这个认证字段将包含一个 Key ID、认证数据长度和一个不减小的加密序列号。这个消息摘要附加在 OSPF 数据包的尾部，不作为 OSPF 数据包本身的一部分。
- **密钥 ID（Key ID）**——标识认证算法和创建消息摘要使用的安全密钥。
- **认证数据长度（Authentication Data Length）**——指明附加在 OSPF 数据包尾部的消息摘要的长度，以八位组字节计。
- **加密序列号（Cryptographic Sequence Number）**——是一个不会减小的数字，用来防止重现攻击（replay attacks）。

2. Hello 数据包

如图 8-31 所示，Hello 数据包是用来建立和维护邻接关系的。为了形成一种邻接关系，Hello 数据包携带的参数必须和它的邻居保持一致。

- **网络掩码（Network Mask）**——是指发送数据包的接口的网络掩码。如果这个掩码

和接收该数据包的接口的网络掩码不匹配，那么该数据包将被丢弃。这一技术性的措施可以确保路由器之间只有在它们共享网络的地址精确匹配时才能互相成为邻居。



图 8-31　OSPF 协议 Hello 数据包

- **Hello 时间间隔（Hello Interval）**——与前面讲述的一样，是指接口上 Hello 数据包传送之间的时间间隔，也是一个周期性的时间段，并以秒来计。对于这个参数，如果发送和接收路由器没有相同的值，那么它们就不能建立一种邻居关系。
- **可选项（Option）**——这个字段将在 8.1.9 小节中讲述。Hello 数据包中包含的这个字段可以用来确保邻居之间的兼容性问题。一台路由器可以拒绝一台兼容性不匹配的邻居路由器。
- **路由器优先级（Router Priority）**——是用来做 DR 和 BDR 路由器的选举的。如果该字段设置为 0，那么始发路由器将没有资格被选成 DR 和 BDR 路由器。
- **路由器无效时间间隔（Router Dead Interval）**——是指始发路由器在宣告邻居路由器无效之前，将要等待从邻居路由器发出的 Hello 数据包的时长，以秒数计。如果路由器收到 Hello 数据包的时间和接收接口配置的 RouterDeadInterval 不匹配，那么这个 Hello 数据包将被丢弃。这种做法可以确保邻居之间的这个参数的一致性。
- **指定路由器（DR）**——是指网络上指定路由器接口的 IP 地址，注意，这里指的不是指定路由器的路由器 ID。在选取 DR 的过程中，这可能只是始发路由器所认为的 DR，而不是最终选举出来的 DR。如果没有 DR（因为 DR 可能还没有选出或者网络类型根本不需要 DR），那么这个字段就会被设置为 0.0.0.0。
- **备份指定路由器（BDR）**——是指网络上备份指定路由器接口的 IP 地址。同样的，在选举 BDR 的过程中，这可能只是始发路由器所认为的 BDR，而不是最终选举出来的 BDR。如果没有 BDR，那么这个字段就会被设置为 0.0.0.0。
- **邻居（Neighbor）**——是一个递归字段，如果始发路由器在过去的一个 Router DeadInterval 时间内，从网络上已经收到来自它的某些邻居的有效 Hello 数据包，那么将会在这个字段中列出所有这些邻居的 RID。

3. 数据库描述数据包

如图 8-32 所示，数据库描述数据包用于正在建立的邻接关系（请参考本章前面“建立一个邻接关系”部分的内容）。数据库描述数据包的一个主要目的是描述始发路由器数据库中的一些或者全部 LSA 信息，以便接收路由器能够确定所收到的 LSA 在其数据库中是否已经有一个匹配的 LSA。这个操作只需要列出 LSA 的头部就可以完成。LSA 头部包括了足够多的信息，不仅可以标识一个特定的 LSA，还可以标识该 LSA 的最近实例。由于在数据库描述（DD）处理过程中，可能需要交换多个数据库描述数据包，因此数据库描述数据包中包含了一个主/从控制关系的标志，用来管理这些数据包的交换。

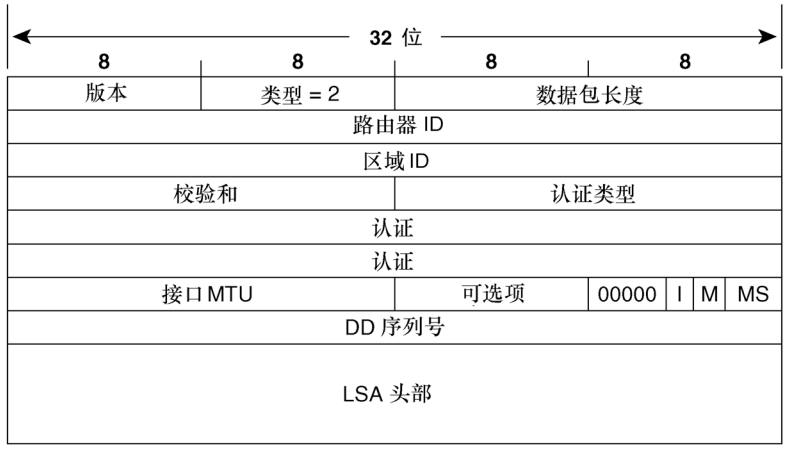


图 8-32 OSPF 数据库描述数据包

- **接口 MTU（Interface MTU）**——是指在数据包不分段的情况下，始发路由器接口可以发送的最大 IP 数据包大小，以八位组字节计。当数据包在虚链路上传送时，这个字段的值设置为 0x0000。
- **可选项（Option）**——这个字段将在 8.1.9 小节中讲述。该字段包含在数据库描述数据包中，使路由器可以选择不转发某些 LSA 到那些没有必要的支持能力的邻居路由器。

报文下一个八位组字节的前 5 位没有被使用而总是设置为 00000b。

- **I 位，或称为初始位（Initial bit）**——当发送的是一系列数据库描述数据包中的最初一个数据包时，该位设置为 1。后续的数据库描述数据包将把该位设置为 0，即 I-bit=0。
- **M 位，或称为后继位（More bit）**——当发送的数据包还不是一系列数据库描述数据包中的最后一个数据包时，将该位设置为 1。最后的一个数据库描述数据包将把该位设置为 0，即 M-bit=0。
- **MS 位，或称为主/从位（Master/Slave bit）**——在数据库同步过程中，该位设置为 1，用来指明始发数据库描述数据包的路由器是一台“主”路由器（也就是说，是主从关系协商过程的控制者）。“从”路由器将该位设置为 0，即 MS-bit=0。
- **数据库描述序列号（DD Sequence Number）**——在数据库的同步过程中，用来确保路由器能够收到完整的数据库描述数据包序列。这个序列号将由“主”路由器在最初发送的数据库描述数据包中设置一些惟一的数值，而后续数据包的序列号将依次增加。

- **LSA 头部（LSA Header）**——列出了始发路由器的链路状态数据库中部分或全部 LSA 头部。参见“链路状态头部”，那里有一个关于 LSA 头部的完整描述。在 LSA 头部里包含有足够的信息可以惟一地标识一个 LSA 和一个 LSA 的具体实例。

4. 链路状态请求数据包

在数据库同步过程中如果收到了数据库描述数据包，路由器将会查看数据库描述数据包里有哪些 LSA 不在自己的数据库中，或者有哪些 LSA 比自己数据库中的 LSA 更新。然后，将把这些 LSA 记录在链路状态请求列表中。接着，路由器会发送一个或多个链路状态请求数据包去向它的邻居请求发送在链路状态请求列表中的这些 LSA 的副本，如图 8-33 所示。这里要注意，一个数据包可以根据一个 LSA 头部的类型、ID 和通告路由器进行惟一的标识，但是它不能请求这个 LSA 的具体实例（LSA 的具体实例由 LSA 头部的序列号、校验和以及老化时间标识）。因此，不论请求路由器是否知道是 LSA 的哪个具体实例，它所请求的都是 LSA 的最新实例。这个过程避免了这样一种情形——在路由器最新描述 LSA 到其副本被请求的时间之间，邻居可能获得或发起一个更新的 LSA 副本。

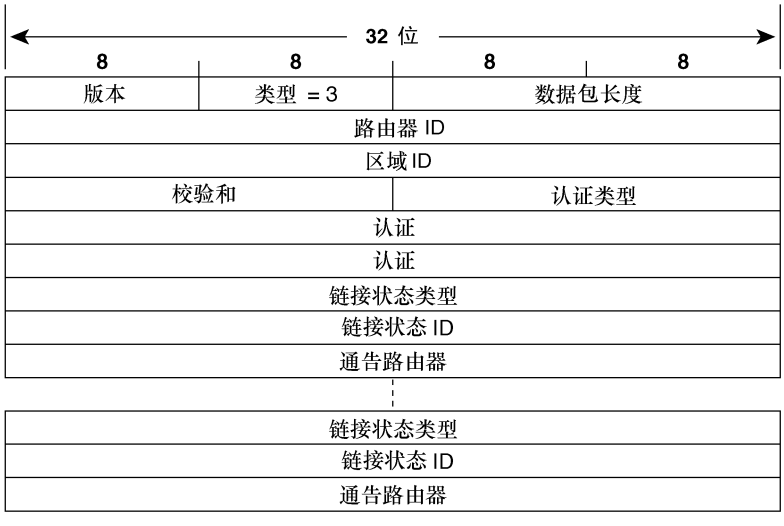


图 8-33　OSPF 链路状态请求数据包

- **链路状态类型（Link State Type）**——是一个链路状态类型号，用来指明 LSA 标识是一个路由器 LSA、一个网络 LSA 还是其他类型的 LSA，等等。表 8-4 中列出了这些类型号。
- **链路状态 ID（Link State ID）**——是 LSA 头部中和类型无关的字段。请参见“链路状态头部”和具体介绍 LSA 的章节可以得到关于不同类型的 LSA 如何使用该字段的完整描述。
- **通告路由器（Advertising Router）**——是指始发 LSA 通告的路由器的路由器 ID。

5. 链路状态更新数据包

如图 8-34 所示，链路状态（LS）更新数据包是用于 LSA 的泛洪扩散和发送 LSA 去响应链路状态请求数据包的。请记住，OSPF 数据包是不能离开发起它们的网络的。因此，一个链路状态数据包可以携带一个或多个 LSA，但是这些 LSA 只能传送到始发它们的路由器的

直连邻居。接收 LSA 的邻居路由器将负责在新的 LS 更新数据包中重新封装相关的 LSA，从而进一步泛洪扩散到它自己的邻居。

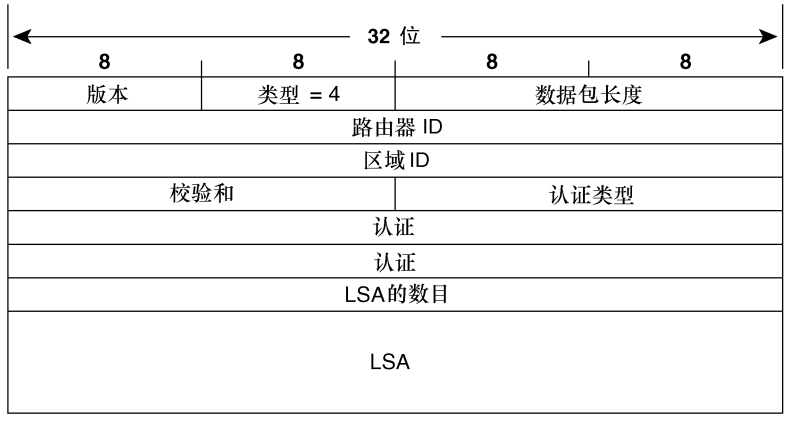


图 8-34 OSPF 链路状态更新数据包

- **LSA 数量（Number of LSA）**——指出这个数据包中包含的 LSA 的数量。
- **链路状态通告（LSA）**——是指在 OSPF 协议的 LSA 数据包格式中描述的全部 LSA。每一个更新数据包都可以携带多个 LSA，它的大小可以达到传送该数据包的链路所允许的最大数据包尺寸。

6. 链路状态确认数据包

链路状态确认数据包是用来进行 LSA 可靠的泛洪扩散的。一台路由器从它的邻居路由器收到的每一个 LSA 都必须在链路状态确认数据包中进行明确的确认。被确认的 LSA 是根据在链路状态确认数据包里包含它的头部来辨别的，并且多个 LSA 可以通过单个数据包来确认。正如图 8-35 所显示的，一个链路状态确认数据包的组成除了 OSPF 包头和一个 LSA 头部的列表之外，就没有其他多余的内容了。

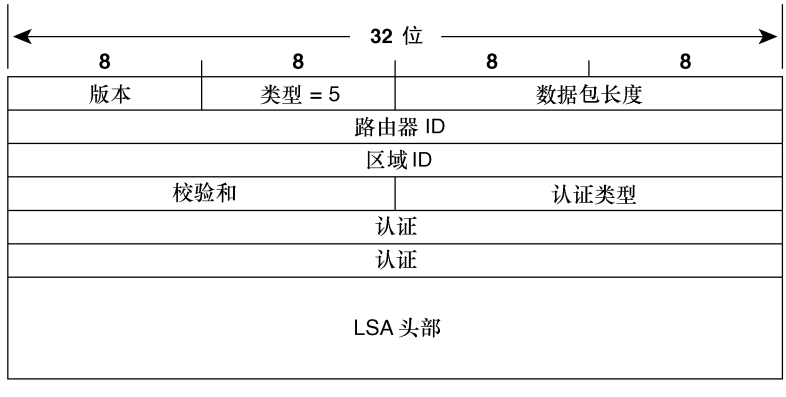


图 8-35 OSPF 链路状态确认数据包

8.1.8 OSPF 的 LSA 格式

本小节将详细描述类型 1～类型 5 和类型 7 的 LSA 字段的含义，不讨论组成员 LSA（类

型 6）是因为 MOSPF 协议不在本书的讲述范围，同样地，类型 8～类型 11 的 LSA 也不讨论，因为它们或者还没有部署（类型 8），或者它们支持的一些特性已经超出了本书的范围。

1. LSA 的头部

LSA 头部在所有 LSA 的开始处，如图 8-36 所示。在数据库描述数据包和链路状态确认数据包里也使用了 LSA 的头部本身。在 LSA 头部中有 3 个字段可以惟一地识别每个 LSA：类型、链路状态 ID 和通告路由器。另外，还有其他 3 个字段可以惟一地识别一个 LSA 的最新实例：老化时间、序列号和校验和。



图 8-36　OSPF 协议 LSA 头部

- **老化时间（Age）**——是指自从发出 LSA 后所经历的时间，以秒数计。当泛洪扩散 LSA 时，在从每一台路由器接口转发出去时，LSA 的老化时间都会增加一个 InfTransDelay 的秒数。当然，当 LSA 驻留在链路状态数据库内时，这个老化时间也会增大。
- **可选项（Option）**——这个字段将在 8.1.9 小节中讲述。在 LSA 的头部中，该字段指出了在部分 OSPF 域中 LSA 能够支持的可选性能。
- **类型（Type）**——就是 LSA 的类型。一些类型的代码可以参见表 8-4。
- **链路状态 ID（Link State ID）**——用来指定 LSA 所描述的部分 OSPF 域。这个字段的特殊用法根据 LSA 的类型而会有所不同。每一个 LSA 的描述都包含了一个怎样使用这个字段的描述。
- **通告路由器（Advertising Router）**——是指始发 LSA 的路由器的 ID。
- **序列号（Sequence Number）**——当 LSA 每次有新的实例产生时，这个序列号就会增加。这个更新可以帮助其他路由器识别最新的 LSA 实例。
- **校验和（Checksum）**——这是一个除了 Age 字段之外，关于 LSA 的全部信息的校验和。因为如果包含了 Age 字段，那么这个校验和将会随着老化时间的增大而每次都需要进行重新计算。
- **长度（Length）**——是一个包含 LSA 头部在内的 LSA 的长度，用八位组字节表示。

2. 路由器 LSA

如图 8-37 所示，路由器 LSA 是由每一台路由器产生的。它列出了一台路由器的链路或接口，同时也列出了这些接口的状态和每一条链路的出站代价。这些路由器 LSA 只能在始发它们的 OSPF 区域内进行泛洪扩散。使用命令 **show ip ospf database router** 可以列出链路状态数据库里的路由器 LSA（请参见示例 8-11）。这里要注意，路由器 LSA 是把主机路由作为末梢网络来通告的，它的链路 ID 字段携带的是主机的 IP 地址，而链路数据字段携带的是主机地址的掩码——255.255.255.255。

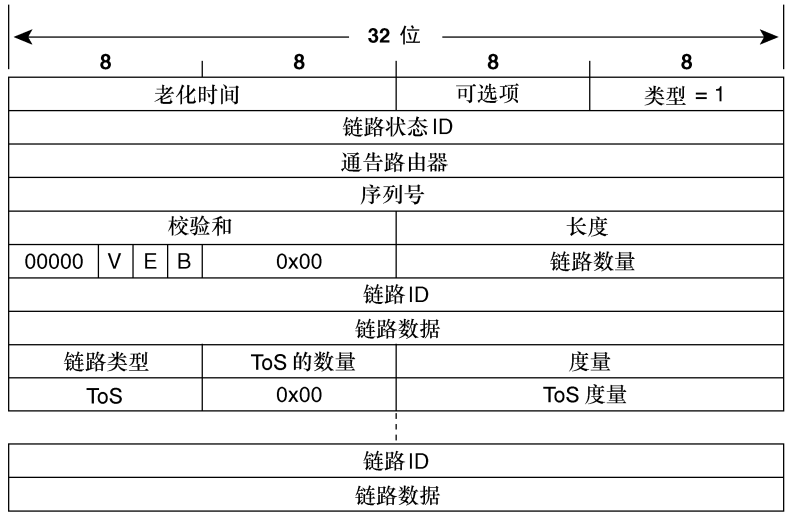


图 8-37 OSPF 的路由器 LSA

- **链路状态 ID（Link State ID）**——路由器 LSA 的链路状态 ID 是指始发路由器的路由器 ID。
- **V，或虚链路端点位（Virtual Link Endpoint bit）**——设置为 1 时，说明始发路由器是一条或多条具有完全邻接关系的虚链路的一个端点，这里被描述的区域是传送区域。
- **E，或外部位（External bit）**——当始发路由器是一个 ASBR 路由器时，设置该位为 1。
- **B，或边界位（Border bit）**——当始发路由器是一个 ABR 路由器时，设置该位为 1。
- **链路数量（Number of Links）**——标明一个 LSA 所描述的路由器链路数量。对于 LSA 进行泛洪扩散的区域，路由器 LSA 必须描述始发路由器的所有链路或接口。

在路由器 LSA 后续的字段里描述了每一条链路，并且出现的次数和前面链路数量字段中的数量是一致的。虽然这个字段是出现在链路数据字段之后的，但是在这里将首先讲述链路类型字段。这是因为链路 ID 和链路数据字段的描述会根据链路类型字段的值而有所变化，因此首先理解链路类型是必要的。

- **链路类型（Link Type）**——描述了链路所提供连接的一般类型。表 8-9 列出了这个字段可能的值和相关的连接类型。

表 8-9 链路类型的值

链路类型	连接
1	点到点连接到另一台路由器
2	连接到一个传送网络
3	连接到一个末梢网络
4	虚链路

- **链路 ID（Link ID）**——用来标识链路连接的对象。这个字段依赖于表 8-10 中的链路类型字段。注意，当连接的对象是另一台路由器时，链路 ID 和在邻居路由器的 LSA 头部的链路状态 ID 是相同的。在计算路由表的期间，这个值可以用来发现链

路状态数据库中邻居的 LSA。

表 8-10 链路 ID 的值

链路类型	链路 ID 字段的值
1	邻居路由器的路由器 ID
2	DR 路由器的接口的 IP 地址
3	IP 网络或子网地址
4	邻居路由器的路由器 ID

- **链路数据（Link Data）**——也是依赖于链路类型字段值的字段，如表 8-11 所示。

表 8-11 链路数据的值

链路类型	链路数据字段的值
1	和网络相连的始发路由器接口的 IP 地址*
2	和网络相连的始发路由器接口的 IP 地址
3	网络的 IP 地址或子网掩码
4	始发路由器接口的 MIB-II ifIndex 值

* 如果点到点链路是无编号的，那么这个字段将替代携带接口的 MIB-II ifIndex 值。

- **ToS 号（Number of ToS）**——为列出的这条链路指定服务类型度量的编号。虽然 RFC 2328 已经不再支持 ToS，但是为了向前兼容早期部署的 OSPF，仍旧保留这个字段。如果没有 ToS 度量和一条链路相关联，那么这个字段就设置为 0x00。
- **度量（Metric）**——是指一条链路（接口）的代价。

接下来的两个与链路相关联的字段是和 ToS 号（#）字段一致的。如果 ToS 的#=3，那么将有 3 个 32 位字包含这些字段的 3 个实例。如果 ToS 的#=0，那么将没有这些字段的实例。

注意，Cisco 路由器只支持 ToS=0。

- **ToS**——指定了后面提及的度量涉及到的服务类型。[1] 表 8-12 列出了 ToS 的值（在 RFC 1349 中指定的）、在 IP 头部中相应的比特值和在 OSPF ToS 字段中使用的相应值。

表 8-12 OSPF ToS 的值

RFC ToS 的值	IP 头部 ToS 字段	OSPF 的 ToS 编码
正常的服务	0000	0
最小的成本代价	0001	2
最大的可靠性	0010	4
最大的吞吐量	0100	8
最小的时延	1000	16

- **ToS 度量（ToS Metric）**——和指定的 ToS 值相关联的度量。

[1] Philip Almquist, “Type of Service in the Internet Protocol Suite,” RFC 1349，1992 年 7 月。

3. 网络 LSA

如图 8-38 所示，网络 LSA 是始发于指定路由器（DR）的。这些网络 LSA 将通告一个多路访问网络和与这个网络相连的所有路由器（包括 DR）。像路由器 LSA 一样，网络 LSA 也只能在始发这条网络 LSA 的区域内进行泛洪扩散。可以使用 **show ip ospf database network** 来查看一条网络 LSA，参见示例 8-12。

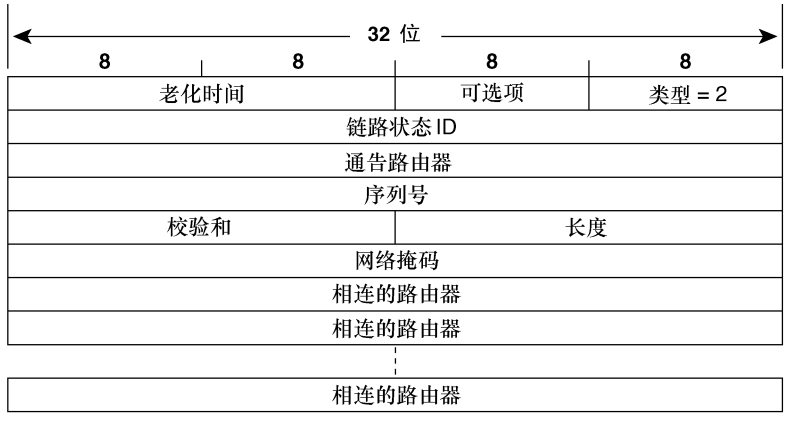


图 8-38 OSPF 的网络 LSA

- **链路状态 ID（Link State ID）**——网络 LSA 的链路状态 ID 是指网络中 DR 路由器接口上的 IP 地址。
- **网络掩码（Network Mask）**——指定这个网络上使用的地址或子网的掩码。
- **相连的路由器（Attached Router）**——列出了多路访问网络上所有与 DR 形成完全邻接关系的路由器的路由器 ID，以及 DR 路由器本身的路由器 ID。这个字段的实例数量（也是列出的路由器的数量）可以由 LSA 头部的长度字段推断出来。

4. 网络汇总 LSA 和 ASBR 汇总 LSA

网络汇总 LSA（类型 3）和 ASBR 汇总 LSA（类型 4）具有同样的格式，如图 8-39 所示。在它们的字段内容里，惟一的不同之处是它们所指的类型和链路状态 ID。ABR 路由器将产生这两种类型的汇总 LSA。网络汇总 LSA 通告的是一个区域外部的网络（包括缺省路由），而 ASBR 汇总 LSA 通告的是一个区域外部的 ASBR 路由器。这两种类型的 LSA 都只能泛洪扩散到单个区域。使用命令 **show ip ospf database summary** 可以查看一台路由器的链路状态数据库中的网络汇总 LSA，参见示例 8-13 所示。而使用命令 **show ip ospf database asbr-summary** 可以查看链路状态数据库中的 ASBR 汇总 LSA，参见示例 8-14。

- **链路状态 ID（Link State ID）**——对于类型 3 的 LSA 来说，它是所通告的网络或子网的 IP 地址。对于类型 4 的 LSA 来说，链路状态 ID 是所通告的 ASBR 路由器的路由器 ID。
- **网络掩码（Network Mask）**——在类型 3 的 LSA 中，是指所通告的网络的子网掩码或地址。在类型 4 的 LSA 中，这个字段没有什么实际意义，并被设置为 0.0.0.0。

如果一条类型 3 的 LSA 通告的是一条缺省路由，那么链路状态 ID 和网络掩码字段都将是 0.0.0.0。

- **度量（Metric）**——是指到达目的地的路由的代价。

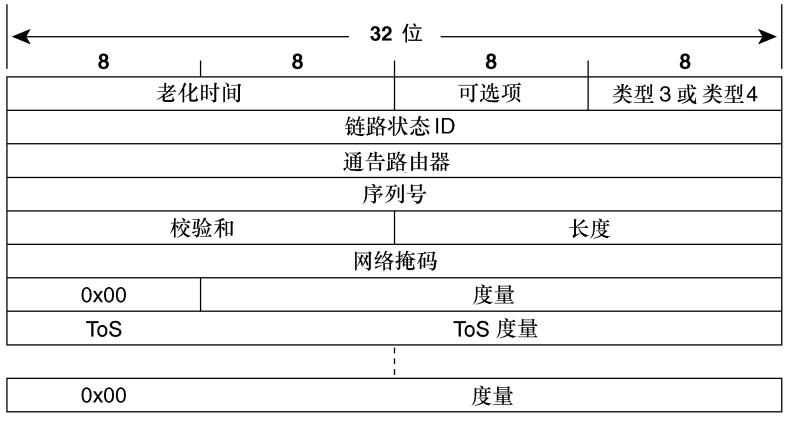


图 8-39　OSPF 的汇总 LSA。类型 3 和类型 4 的汇总 LSA 具有同样的格式

ToS 字段和 ToS 度量字段都是可选字段，并且已经在“路由器 LSA”部分描述过了。另外提醒一下，Cisco 路由器只支持 ToS=0。

5. 自主系统外部 LSA

如图 8-40 所示，自主系统外部 LSA 是由 ASBR 路由器始发的。这些自主系统外部 LSA 是用来通告 OSPF 自主系统外部的目的网络的，这里也包括到达外部目的网络的缺省路由。自主系统外部 LSA 可以泛洪扩散到 OSPF 域中所有非末梢区域中去。可以使用命令 **show ip ospf database external** 来查看 AS 外部 LSA，参见示例 8-15。

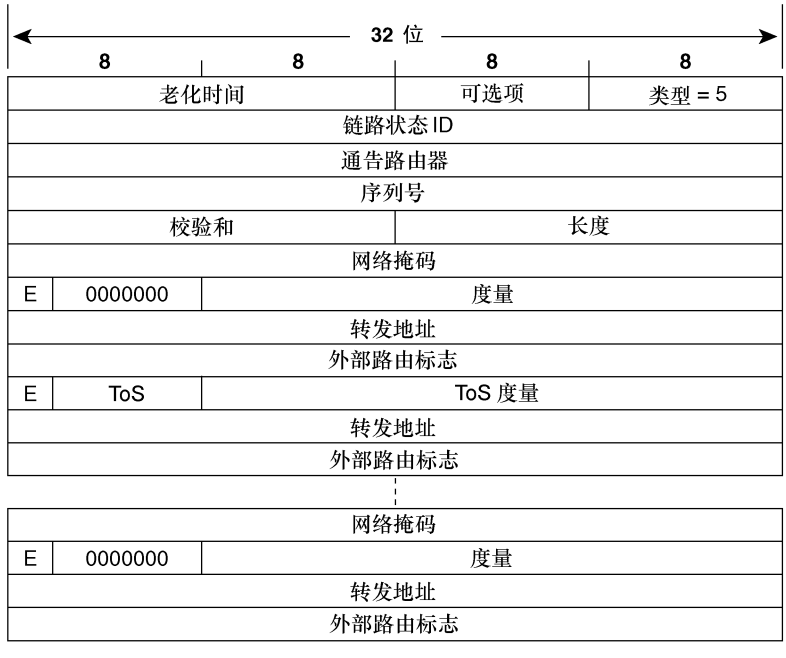


图 8-40　OSPF 的自主系统外部 LSA

- **链路状态 ID**——自主系统外部 LSA 的链路状态 ID 是指目的地的 IP 地址。
- **网络掩码**——是指所通告的目的地的子网掩码或地址。

如果类型 5 的 LSA 正在通告的是一条缺省路由，那么链路状态 ID 和网络掩码字段都将被设置为 0.0.0.0。

- **E，或称外部度量位（External Metric bit）**——用来指定这条路由使用的外部度量的类型。如果该 E-bit 设置为 1，那么度量类型就是 E2；如果该 E-bit 设置为 0，那么度量类型就是 E1。请参考本章前面讲述的“路径类型”部分的内容，那里可以找到关于 E1 和 E2 外部度量类型的更多信息。
- **度量**——是指路由的代价，由 ASBR 路由器设定。
- **转发地址（Forwarding Address）**——是指到达所通告的目的地的数据包应该被转发到的地址。如果转发地址是 0.0.0.0，那么数据包将被转发到始发 ASBR 上。
- **外部路由标志（External Route Tag）**——是一个应用于外部路由的任意标志。OSPF 协议本身并不使用这个字段，而是由外部路由来管理和控制。该标志的设定和用法将在第 14 章中介绍。

可选地，ToS 字段也可以和某个目的地相关联。这些字段和前面讲述的是相同的，只是每一个 ToS 度量也都有自己的 E-bit、转发地址和外部路由标志。

6. NSSA 外部 LSA

NSSA 外部 LSA 是由一个 NSSA 区域内的 ASBR 路由器始发的。如图 8-41 所示，除了转发地址字段外，NSSA 外部 LSA 的所有字段都和一个 AS 外部 LSA 的字段相同。不像 AS 外部 LSA 那样是在整个 OSPF 自主系统中进行泛洪扩散的，NSSA 外部 LSA 仅仅在始发它们的一个非纯末梢区域中进行泛洪扩散。使用命令 **show ip ospf database nssa-external** 可以显示出 NSSA 外部 LSA 的信息，参见示例 8-16 所示。

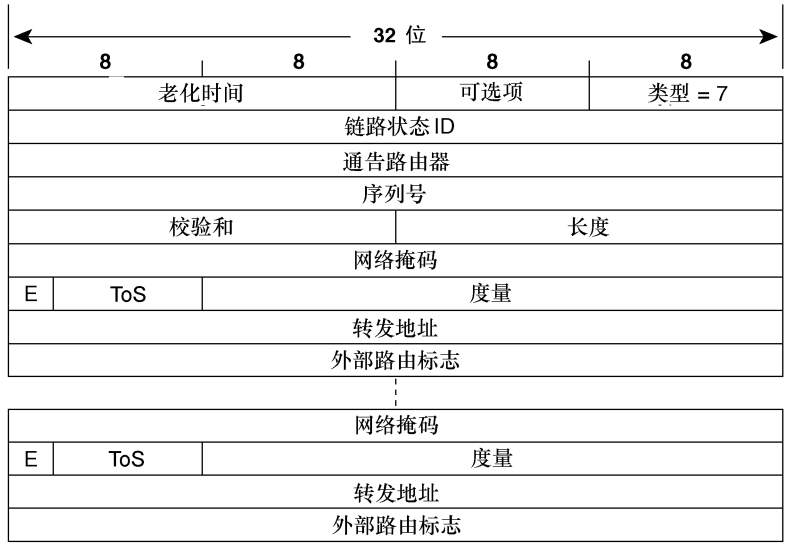


图 8-41 OSPF 的 NSSA 外部 LSA

- **转发地址**——如果网络在一台 NSSA ASBR 路由器和邻接的自主系统之间是作为一条内部路由通告的，那么这个转发地址就是指这个网络的下一跳地址。如果网络不是作为一条内部路由通告，那么这个转发地址将是 NSSA ASBR 路由器的路由器 ID。

8.1.9 可选项字段

如图 8-42 所示，可选字段是出现在每一个 Hello 数据包、数据库描述数据包和每一个 LSA 中的。可选项字段允许路由器和其他路由器进行一些可选性能的通信。

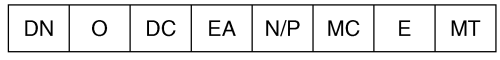


图 8-42 OSPF 的可选项字段

- **DN**——用于基于 MPLS 的第 3 层虚拟专用网（VPN），在 RFC 规定 VPN 之后通常称为 RFC 2547 VPN。当一条路由通过 OSPF 从某个客户网络学到后，它就会穿过使用多协议 BGP（Multiprotocol BGP）的 RFC 2547 VPN 被通告到网络对端，接着再通过 OSPF 被通告回客户网络。通告回的 OSPF 路由在 BGP 中会被重新分配到 VPN 运营商的网络，这样就会产生一个环路。DN 位用来避免这个环路。当在类型 3、类型 5 或类型 7 的 LSA 中设置了 DN 位后，接收路由器就不能在它的 OSPF 路由计算中使用该 LSA。
- **O**——设置用来表明始发路由器支持 Opaque LSA（类型 9，类型 10 和类型 11）。
- **DC 位**——当始发路由器具有支持按需电路上的 OSPF 的能力时，该位将被设置。
- **EA 位**——当始发路由器具有接收和转发外部属性 LSA 的能力时，该位将被设置。这些 LSA 还没有一般的用法，因此本书将不介绍它们。
- **N 位**——只用在 Hello 数据包中。一台路由器设置 N-bit=1 表明它支持 NSSA 外部 LSA。如果设置 N-bit=0，那么路由器将不接受和发送 NSSA 外部 LSA。邻居路由器如果错误配置了 N-bit 将不会形成邻接关系，这个限制可以确保一个区域内的所有路由器都同样地具有支持 NSSA 的能力。如果 N-bit=1，那么 E-bit 必须设置为 0。
- **P 位**——只用在 NSSA 外部 LSA 的头部（由于这种情况，N-bit 和 P-bit 可以使用在同一位置）。该位将告诉一个非纯末梢区域中的 ABR 路由器将类型 7 的 LSA 转换为类型 5 的 LSA。
- **MC 位**——当始发路由器具有转发 IP 组播数据包的能力时，该位将被设置。这一位使用在 MOSPF 协议当中。
- **E 位**——当始发路由器具有接受 AS 外部 LSA 的能力时，该位将被设置。在所有的 AS 外部 LSA 和所有始发于骨干区域以及非末梢区域的 LSA 中，该位将设置为 1。而在所有始发于末梢区域的 LSA 当中，该位设置为 0。另外，可以在 Hello 数据包中使用该位来表明一个接口具有接收和发送类型 5 的 LSA 的能力。E-bit 配置错误的邻居路由器将不能形成邻接关系，这个限制可以确保一个区域的所有路由器都同样地具有支持末梢区域的能力。
- **MT 位**——设置了该位表示始发路由器支持多拓扑 OSPF（MT-OSPF）。在本书编写期间，MT-OSPF 还仅仅是一个提议，并未被广泛采用。

旧的 OSPF 标准规定，目前被 MT 位占用的可选位是 T 位。设置了 T 位表示始发路由器具有支持 ToS 的能力。但是，由于 ToS 特性从来没有部署过，所以 T 位也就从来没有使用过。

8.2 配置 OSPF

在一个大型的 IP 网络中，有很多可用的 OSPF 选项和配置变量经常用于 IGP。然而，偶尔会听到这样一种观点，认为在一个小型的网络中使用 OSPF 协议不是一种好的选择，因为 OSPF 协议的配置显得“太复杂”了。这纯粹是无稽之谈。正如下面所介绍的第一个配置案例中显示的，完成一个基本的 OSPF 配置并使 OSPF 协议可以正常地运行，只需要在 **network** 命令中额外地敲几下键盘而已。如果对 OSPF 的操作已经有了相当地理解，那么所输入的这些额外的内容也显得十分直观和自然了。

8.2.1 案例研究：一个基本的 OSPF 配置

配置一个基本的 OSPF 的过程含有以下 3 个必要的步骤：

步骤 1：确定和每一个路由器接口相连的区域。

步骤 2：使用 **router ospf** *process-id* 命令来启动一个 OSPF 进程。

步骤 3：使用 **network area** 命令来指定运行 OSPF 协议的接口和它们所在的区域。

与 IGRP 和 EIGRP 协议中相关的进程 ID 不同，OSPF 协议的进程 ID 不是一个自主系统号。OSPF 的这个进程 ID 可以是任何正整数，并且仅在配置它的路由器内有意义。Cisco IOS 软件允许同一台路由器中运行多个 OSPF 进程，[1] 进程 ID 不过是在同一台设备中用来区分一个进程与另一个不同的进程而已。

在 RIP 协议中，命令 **network** 只允许用来指定一个主网络地址。如果在这个网络中的一些接口不能运行该路由选择协议的话，那么就在这些协议中使用 **passive-interface** 命令来抑制这些接口。正如在 EIGRP 协议中使用带通配符掩码的 **network** 命令，使用命令 **network area** 比 RIP 与 EIGRP 在引入掩码选项之前使用的 **network** 命令更加灵活，它可以完全反映 OSPF 协议的无类别特性。任何一个地址范围都能够使用一个（地址，反向掩码）对来指定。这里的反向掩码（inverse mask）和在访问列表中使用反向掩码是一样的。[2] 对于区域的指定，既可以使用一个十进制数字表示，也可以使用一个点分十进制数来表示。

图 8-43 显示了一个 OSPF 的网络。注意，在这里每一个区域都具有一个指定的 IP 地址，这个地址源于它的子网。限制一个区域为一个地址或子网是不必要的，但是这样做会带来一些有效的好处，这在后面的 8.2.8 小节中将会看到。注意，这个例子是设计用来演示多个区域的配置的。在现实的网络环境中，将图中这样的一个小型网络设计成单个区域应该是一种更聪明的做法。更进一步来说，这里的单个区域也不必一定是区域 0。OSPF 的规则规定所有的区域必须连接到骨干区域；因此，只有当有多于一个的区域时才需要骨干区域。

在图 8-43 中的 4 台路由器上，每一台路由器的配置都不同，这样做是为了更好地说明 **network area** 命令的灵活性。相关的配置参见示例 8-19～示例 8-22。

---

[1] 虽然可以在同一台路由器上配置多个 OSPF 进程，但是并不十分鼓励这么做，因为，运行多个数据库将要占用较多的路由器资源。

[2] 请参考附录 B 的内容，查找有关于反向掩码的用法说明。

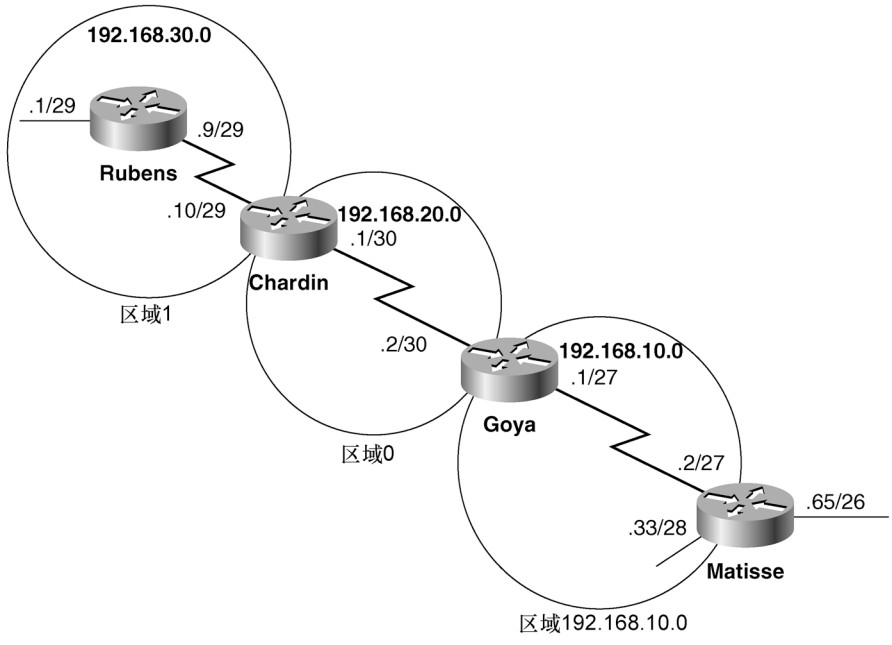


图 8-43 路由器 Chardin 和 Goya 都是 ABR 路由器，而路由器 Rubens 和 Matisse 是内部路由器

示例 8-19 路由器 Rubens 的 OSPF 网络区域配置

```
router ospf 10
 network 0.0.0.0 255.255.255.255 area 1
```

示例 8-20 路由器 Chardin 的 OSPF 网络区域配置

```
router ospf 20
 network 192.168.30.0 0.0.0.255 area 1
 network 192.168.20.0 0.0.0.255 area 0
```

示例 8-21 路由器 Goya 的 OSPF 网络区域配置

```
router ospf 30
 network 192.168.20.0 0.0.0.3 area 0.0.0.0
 network 192.168.10.0 0.0.0.31 area 192.168.10.0
```

示例 8-22 路由器 Matisse 的 OSPF 网络区域配置

```
router ospf 40
 network 192.168.10.2 0.0.0.0 area 192.168.10.0
 network 192.168.10.33 0.0.0.0 area 192.168.10.0
```

这里要注意的第一件事情是，在每一台路由器上配置的进程 ID 都是不同的。一般情况下，为了保持在一个网络中配置的一致性，这些数字是相同的。而在这里，配置不同的进程 ID 号只不过是为了演示这样一个事实——它们在本地路由器之外是没有意义的。当然，这 4 个不同编号的进程是可以相互通信的。

要注意的第二件事情就是 **network area** 命令的格式。紧跟在 **network** 之后的部分是一个 IP 地址和一个反向掩码。当 OSPF 进程第一次启动时，它将根据第一个网络语句（即第一个 **network** 语句）的（地址，反向掩码）对来“启动”所有有效接口的 IP 地址。所有匹配的接口将被分配到根据 **network** 命令的 **area** 部分指定的区域。接着，这个过程根据第二条 **network** 语句继续匹配启动所有不匹配第一条 **network** 语句的接口。这样，根据一条条 **network** 语句

运行 IP 地址的过程一直持续到所有的接口都匹配，或者所有的 **network** 语句都被使用为止。这里要注意有一点非常重要，就是这个过程从第一条 **network** 语句开始是有顺序处理的。正如在后面故障排除一节中所描述的，**network** 语句的顺序变得很重要。

路由器 Rubens 的 **network** 语句将匹配路由器上所有的接口。地址 0.0.0.0 实际上仅仅是一个占位符。在这里，反向掩码 255.255.255.255 才是所有操作的核心。当“可以忽略”的位占据了所有 4 个八位组字节时，掩码将认为是和任何地址相匹配的，并且把相应的接口指定到区域 1 中。这种方法提供了一种最粗略地控制接口运行 OSPF 的手段。

路由器 Chardin 是区域 1 和区域 0 之间的 ABR 路由器。这个事实是由在它们上面配置的 **network** 语句看出的。在这里，（地址，反向掩码）对将把与主网络 192.168.30.0 的任何子网相连的所有接口放置到区域 1 中，并把与主网络 192.168.20.0 的任何子网相连的所有接口放到骨干区域。

路由器 Goya 也是一台 ABR 路由器。在这里，（地址，反向掩码）对将只匹配两个接口上配置的具体子网。这里也要注意，骨干区域是用点分十进制来指定的。这种格式和路由器 Chardin 上配置的十进制格式是兼容的，它们都会使 OSPF 数据包格式中相应的区域字段设置为 0x00000000。

路由器 Matisse 有一个接口 192.168.10.65/26，但这个接口并不运行 OSPF 协议。这台路由器上的 **network** 语句配置的是每个单独的接口地址，因而它的反向掩码指明所有 32 位都必须精确地匹配。这种方法提供了一种最精确地控制接口运行 OSPF 的手段。

最后，请注意虽然路由器 Matisse 的接口 192.168.10.65/26 没有运行 OSPF 协议，但是这个地址在数值上是该路由器上最高的 IP 地址。路由器 Matisse 的路由器 ID 就是 192.168.10.65，参见示例 8-23 所示。

示例 8-23 使用命令 show ip ospf process-id 显示了指定进程的信息

```
Matisse#show ip ospf 40
 Routing Process "ospf 40" with ID 192.168.10.65
 Supports only single TOS(TOS0) routes
 Supports opaque LSA
 Supports Link-local Signaling (LLS)
 Initial SPF schedule delay 5000 msecs
 Minimum hold time between two consecutive SPFs 10000 msecs
 Maximum wait time between two consecutive SPFs 10000 msecs
 Minimum LSA interval 5 secs. Minimum LSA arrival 1 secs
 LSA group pacing timer 240 secs
 Interface flood pacing timer 33 msecs
 Retransmission pacing timer 66 msecs
 Number of external LSA 0. Checksum Sum 0x000000
 Number of opaque AS LSA 0. Checksum Sum 0x000000
 Number of DCbitless external and opaque AS LSA 0
 Number of DoNotAge external and opaque AS LSA 0
 Number of areas in this router is 1. 1 normal 0 stub 0 nssa
 External flood list length 0
    Area 192.168.10.0
        Number of interfaces in this area is 2
        Area has no authentication
        SPF algorithm last executed 00:47:42.792 ago
        SPF algorithm executed 4 times
        Area ranges are
        Number of LSA 5. Checksum Sum 0x02A444
        Number of opaque link LSA 0. Checksum Sum 0x000000
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```

8.2.2 案例研究：使用 Loopback 接口设置路由器的 ID

假设图 8-43 中的路由器 Matisse 已经在设备配置中心配置好了，并且被发送到了要安装的地点。在这台路由器启动的过程中，将报告它无法定位一个路由器 ID，并且看上去好像在报告 **network area** 命令出现配置错误，参见示例 8-24。更麻烦的是，OSPF 命令也不再是可运行的配置了。

示例 8-24 如果找不到一个有效的 IP 地址作为它的路由器 ID，OSPF 将不会启动

```
Cisco Internetwork Operating System Software
IOS (tm) C2600 Software (C2600-J1S3-M), Version 12.3(6), RELEASE SOFTWARE (fc3)
Copyright (c) 1986-2004 by Cisco Systems, Inc.
Compiled Wed 11-Feb-04 19:24 by kellythw
Image text-base: 0x80008098, data-base: 0x8199F778

cisco 2621 (MPC860) processor (revision 0x200) with 61440K/4096K bytes of memory


Processor board ID JAD05090PW2 (1141326406)
M860 processor: part number 0, mask 49
Bridging software.
X.25 software, Version 3.0.0.
TN3270 Emulation software.
2 FastEthernet/IEEE 802.3 interface(s)
1 Serial network interface(s)
32K bytes of non-volatile configuration memory.
16384K bytes of processor board System flash (Read/Write)


 network 192.168.10.2 0.0.0.0 area 192.168.10.0
    ^
% Invalid input detected at '^' marker.
 network 192.168.10.33 0.0.0.0 area 192.168.10.0
    ^
% Invalid input detected at '^' marker.



Press RETURN to get started!


%OSPF-4-NORTRID: OSPF process 40 cannot start. There must
be at least one "up" IP interface, for OSPF to use as router ID
```

这里出现的问题是，在路由器启动期间，其上所有的接口都是管理关闭（administratively shutdown）的。如果 OSPF 不能发现一个有效的 IP 地址作为它的路由器 ID，那么 OSPF 将不会启动。再进一步，如果 OSPF 进程没有启动，那么随后的 **network area** 命令也将是无效的。

解决这个问题的方法（在这里假定关闭所有的物理接口是有合理的原因的）是使用一个 loopback 接口（环回接口）。loopback 接口是一个仅在软件上有意义的、虚拟的接口，并且它总是有效（up）的。因此，loopback 接口的 IP 地址也总是有效的。

在 OSPF 路由器上使用 loopback 接口还有一个更普遍的原因是，这些接口允许网络管理员对路由器 ID 进行控制。当 OSPF 进程查找一个路由器 ID 时，OSPF 将越过所有物理接口的 IP 地址，优先选用 loopback 接口的 IP 地址，而且不管 IP 地址在数值上的高低顺序如何。如果路由器具有多个带 IP 地址的 loopback 接口，那么 OSPF 将选用在数值上最高的 loopback

地址。

控制路由器 ID 使单个 OSPF 路由器更加容易识别，从而使网络的管理和故障排除更加容易。路由器 ID 的管理通常使用以下两种方法之一：

- 单独使用合法的网络或子网地址作为路由器 ID；
- 使用一段“伪造”的 IP 地址段。

第一种方法的缺点是需要使用合法的网络地址空间。第二种方法可以节省合法的地址，但是有一点要记住，在一个网络里伪造的地址在另一个网络里可能是合法的地址。只要你记得这些地址不是合法的地址，那么使用一些简单、易于识别的地址，例如 1.1.1.1、2.2.1.1 等将是比较好的做法。必须要小心的是，伪造的地址千万不能泄漏到公共的 Internet 网络上去。

在前面一节的配置中使用 loopback 地址进行修改参见示例 8-25～示例 8-28。

示例 8-25 路由器 Rubens 的配置中增加了一个 loopback 接口

```
interface Loopback0
 ip address 192.168.50.1 255.255.255.255
!
router ospf 10
 network 192.168.30.0 0.0.0.255 area 1
```

示例 8-26 路由器 Chardin 的配置中增加了一个 loopback 接口

```
interface Loopback0
 ip address 192.168.50.2 255.255.255.255
!
router ospf 20
 network 192.168.30.0 0.0.0.255 area 1
 network 192.168.20.0 0.0.0.255 area 0
```

示例 8-27 路由器 Goya 的配置中增加了一个 loopback 接口

```
interface Loopback0
 ip address 192.168.50.3 255.255.255.255
!
router ospf 30
 network 192.168.20.0 0.0.0.3 area 0.0.0.0
 network 192.168.10.0 0.0.0.31 area 192.168.10.0
```

示例 8-28 路由器 Matisse 的配置中增加了一个 loopback 接口

```
interface Loopback0
 ip address 192.168.50.4 255.255.255.255
!
router ospf 40
 network 192.168.10.2 0.0.0.0 area 192.168.10.0
 network 192.168.10.33 0.0.0.0 area 192.168.10.0
```

对于这个例子，网络地址 192.168.50.0 独自使用来作为路由器 ID。因此，在这个网络中，路由器 ID 可以容易地和其他 IP 地址区分开来。

这里要注意的第一件事情是，在这个配置中 loopback 地址所使用的地址掩码：每一个掩码都配置成一个主机地址。这一步其实不是必要的，因为 OSPF 会把一个 loopback 接口作为一个末梢网络来看待。无论（地址，掩码）对配置成什么，loopback 接口的地址都将被作为一条主机路由来通告。主机掩码仅仅用来保持一种整齐的格式，并且用来反映所通告的地址的一种方式。

然而，第二个需要引起注意的地方和第一个有点不相关。请记住，OSPF 协议虽然使用一个接口的 IP 地址作为它的路由器 ID，但是并不一定需要在这个接口上运行 OSPF。事实上，

OSPF 所通告的 loopback 地址不过是创建了一个不必要的 LSA 而已。在上面一个例子的显示中要注意，那里的 **network area** 语句并没有涉及到 loopback 地址。事实上，路由器 Rubens 上的配置不得不更改。路由器 Rubens 在前面例子中的命令 **network 0.0.0.0 255.255.255.255 area 1** 应该已含有 loopback 地址。

另外，为了对网络的管理和故障排除有所帮助，使用 loopback 接口也可以使一个 OSPF 网络更加稳定。在一些早期的 IOS 软件版本中，如果一个作为路由器 ID 的物理接口出现了硬件故障，[1] 或者这个接口被管理关闭（administratively shutdown），或者这个 IP 地址被无意删除，那么 OSPF 进程将必须获取一个新的路由器 ID。因此，路由器必须过早地老化和泛洪扩散它原来的 LSA，并且要泛洪扩散包含新的路由器 ID 的 LSA。loopback 接口却没有硬件上的故障问题。目前的 IOS 软件版本是获取路由器 ID，如果路由器 ID 相关联的接口失效或它的 IP 地址被删除，那么这台路由器只有在它被重新启动或重新运行 OSPF 进程后才能获取路由器 ID。

另一种可行的做法是，使用 OSPF 命令 **router-id** 手工为该路由器指定一个路由器 ID。当使用 loopback 接口作为路由器 ID 时，你可以任意选择一个 IP 地址作为 **router-id** 命令的值。如果使用该命令配置一个路由器 ID，这个命令的值将在重启 OSPF 进程或重启该路由器时成为路由器 ID。然而，在路由器被重新启动的时候，OSPF 进程还没运行之前，必须要有一个在线的带有 IP 地址的接口。在 OSPF 进程运行后，使用 **router-id** 命令指定的地址将成为路由器 ID。

8.2.3 案例研究：域名服务查询

loopback 地址由于可以提供预先设计好的路由器 ID，从而使一个 OSPF 网络的管理和故障排除变得很简单。为了进一步得到简化，可以把路由器 ID 记录到一个域名服务（DNS）数据库中。在路由器上可以配置域名服务，使路由器向一台域名服务器请求相关“地址到名称”的映射，或称为反向 DNS 查找，从而可以通过显示路由器的名称来代替路由器的 ID，参见示例 8-29 所示。

示例 8-29 OSPF 可以配置成利用 DNS 来实现某些 show 命令中路由器 ID 到名称的映射

```
Goya#show ip ospf neighbor

Neighbor ID     Pri   State       Dead Time   Address       Interface
chardin           0   FULL/  -    00:00:36    192.168.20.1  Serial0/0.2
matisse           0   FULL/  -    00:00:34    192.168.10.2  Serial0/0.1
Goya#show ip ospf database

            OSPF Router with ID (192.168.50.3) (Process ID 30)

                Router Link States (Area 0.0.0.0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
192.168.50.2    chardin         78          0x80000007 0x005A70 2
192.168.50.3    goya            78          0x80000008 0x004C7B 2

                Summary Net Link States (Area 0.0.0.0)
```

（待续）

[1] 如果仅仅是断开接口连接并不会引起路由器 ID 的改变。

```
Link ID         ADV Router      Age         Seq#       Checksum
192.168.10.0    goya            74          0x80000001 0x00B356
192.168.10.32   goya            54          0x80000001 0x00DCFB
192.168.30.0    chardin         85          0x80000001 0x007766
192.168.30.8    chardin         100         0x80000001 0x001DB9

                Router Link States (Area 192.168.10.0)


Link ID         ADV Router      Age         Seq#       Checksum Link count
192.168.50.3    goya            68          0x80000007 0x0024D3 2
192.168.50.4    matisse         67          0x80000007 0x00E080 3
--More-
```

路由器 Goya 可以使用示例 8-30 中的配置来执行 DNS 的查找。

示例 8-30 配置路由器 Goya 执行 DNS 查找

```
ip name-server 172.19.35.2
!
ip ospf name-lookup
```

第一个命令用来指定一个 DNS 服务器的地址，第二个命令用来启动 OSPF 进程执行 DNS 查找。在一些实际案例中，一台路由器是通过一个接口地址来识别的，而不是路由器 ID。这种情况下，可以为路由器的这个接口增加一个条目到 DNS 数据库中，例如 rubens-e0。这样做可以使路由器通过这个接口的名字来识别，这与路由器 ID 不同。

在这个例子中使用的域名服务器的地址是不属于图 8-43 中显示的某一个子网的。到达这个网络的方法将是下一个案例研究的主题。

8.2.4 案例研究：OSPF 和辅助地址

在一个 OSPF 的环境中，辅助地址的用法有以下两个相关的规则：

- 只有在主网络或子网也运行 OSPF 协议的时候，OSPF 才会通告一个辅助的网络或子网。
- OSPF 将把辅助地址看作是末梢网络（这些网络上没有 OSPF 邻居），从而不会在这些网络上发送 Hello 数据包。因此，在辅助网络上也就无法建立邻接关系。

图 8-44 中显示了一台 DNS 服务器，并另外增添了一台路由器连接到路由器 Matisse 的 FA0/0 接口上。这台 DNS 服务器和新加的路由器都放在子网 172.19.35.0/25 中，并给路由器 Matisse 的 FA0/0 接口分配一个辅助地址 172.19.35.15/25（参见示例 8-31）。

示例 8-31 路由器 Matisse 配置了一个辅助地址

```
interface FastEthernet0/0
 ip address 172.19.35.15 255.255.255.128 secondary
 ip address 192.168.10.33 255.255.255.240
!
router ospf 40
 network 192.168.10.2 0.0.0.0 area 192.168.10.0
 network 192.168.10.33 0.0.0.0 area 192.168.10.0
 network 172.19.35.15 0.0.0.0 area 192.168.10.0
```

根据这个配置，路由器 Matisse 将向它的邻居路由器通告子网 172.19.35.0/25。但是，如果关于主地址 192.168.10.33 的 **network area** 语句被删除了的话，那么子网 172.19.35.0/25 也将不再被通告。

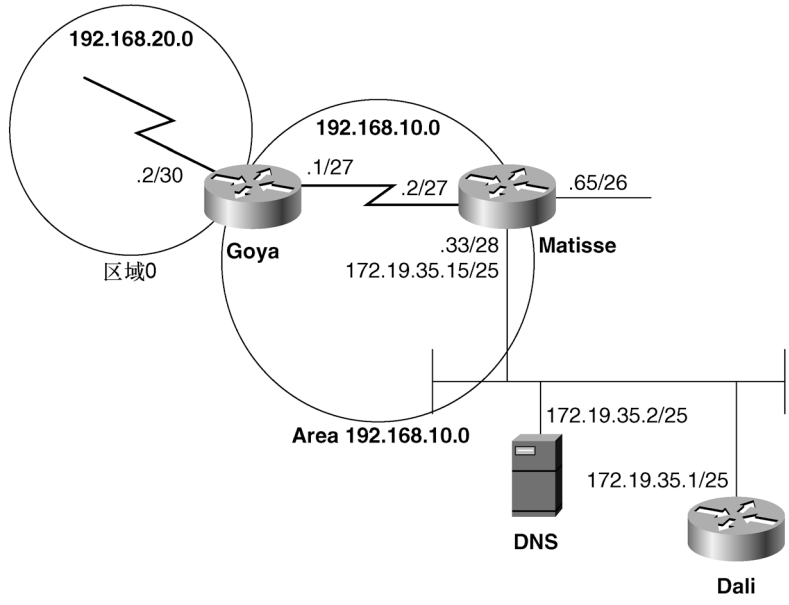


图 8-44　路由器 Dali 和 DNS 服务器都不是 OSPF 域的一部分，
并且它们是通过一个辅助地址连接到路由器 Matisse 上的

由于路由器 Matisse 是通过一个辅助地址和子网 172.19.35.0/25 相连的，因此它不能和这个子网上的任何路由器建立邻接关系，如图 8-45 所示。但是，DNS 服务器使用了路由器 Dali 作为它的缺省网关。因此，路由器 Matisse 和路由器 Dali 之间必须能够互相转发数据包。

到目前为止，对于上述的网络可以得出以下的结论：

- 子网 172.19.35.0/25 正在被通告到 OSPF 域中；一个目的地址是 172.19.35.2 的数据包将被转发到路由器 Matisse 的 FA0/0 接口，并且从那里直接转发到 DNS 服务器，参见示例 8-32 所示。
- 由于 DNS 服务器必须发一些回复数据包到与它不在同一网段的网络，因而它会根据路由选择发送这些回复到路由器 Dali。
- 路由器 Dali 和路由器 Matisse 之间并不交换路由选择信息，因此它将不知道怎样到达 OSPF 域内的网络。

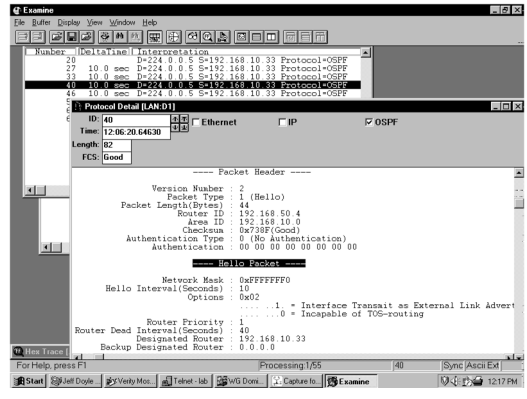


图 8-45　这台协议分析仪显示的信息是从与路由器 Matisse、Dali 和 DNS 服务器相连的网络上捕获得。
小一点的窗口中显示出 Hello 数据包只能由路由器 Matisse 的主地址 192.168.10.33 发送。
大一点的窗口显示了一个 Hello 数据包的解码信息

示例 8-32 在路由器 Matisse 的 ARP 缓存中记录了 DNS 服务器的 MAC 地址标识，这表明该 DNS 服务器是可以直接到达的。但是，如果到达 DNS 服务器的数据包必须通过路由器 Dali 路由转发才能到达的话，那么在这个缓存中，这台 DNS 服务器和路由器 Dali 的 MAC 地址将应该都是 0000.0c0a.2aa9

```
Matisse#show arp
Protocol  Address      Age (min)  Hardware Addr   Type   Interface
Internet  192.168.10.65       -   0005.5e6b.50a1  ARPA   FastEthernet0/1
Internet  192.168.10.33       -   0005.5e6b.50a0  ARPA   FastEthernet0/0
Internet  172.19.35.15        -   0005.5e6b.50a0  ARPA   FastEthernet0/0
Internet  172.19.35.1         1   0010.7b3c.6bd3  ARPA   FastEthernet0/0
Internet  172.19.35.2        22   0002.6779.0f4c  ARPA   FastEthernet0/0
```

因此，这里需要一步去“连通一条电路”，来告诉路由器 Dali 怎样才能到达 OSPF 域内的网络。这一步可以通过配置一条静态路由很容易地完成：

```
Dali(config)#ip route 192.168.0.0 255.255.0.0 172.19.35.15
```

这里需要注意的是，静态路由是无类别的路由，因而它可以使用超网的条目来匹配 OSPF 自主系统内的所有地址。

在这个例子当中，路由器 Matisse 并不是一台 ASBR 路由器。这是因为，虽然它将数据包发送到 OSPF 自主系统外部的目的地，但是它并没有接受外部目的地的任何信息，因此，也没有始发任何类型 5 的 LSA 通告。

图 8-46 中显示了通过路由器 Dali 到达的一组新的目的地址。路由器 Matisse 现在就必须变成一台 ASBR 路由器，并且要将这些路由通告到 OSPF 域内。但是，它首先必须学习到这些路由。这个工作可以通过配置一系列静态路由或者运行一个在辅助网络上通信的路由选择协议来完成。无论在上述哪种情况下，路由器都必须重新分配这些路由到 OSPF 域中。

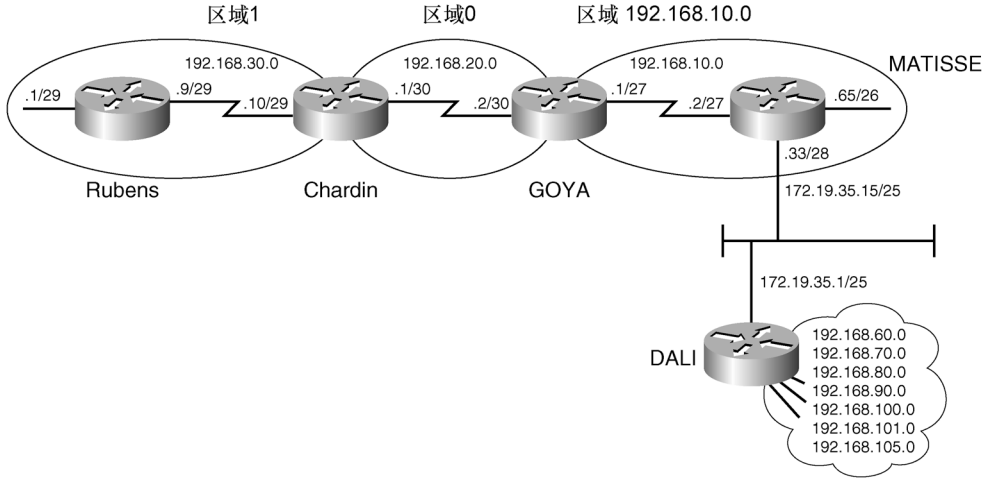


图 8-46 OSPF 自主系统必须学习到这些通过路由器 Dali 可达的目的地址，但是路由器 Matisse 到达路由器 Dali 的辅助地址妨碍了这两台路由器共享 OSPF 信息

RIP 协议在辅助地址上使用没有什么困难。因此，在路由器 Matisse 上可以选用 RIP 协议作为与路由器 Dali 的通信。路由器 Matisse 的配置参见示例 8-33。

示例 8-33 在路由器 Matisse 的配置中增加了 RIP

```
interface FastEthernet0/0
 ip address 172.19.35.15 255.255.255.128 secondary
 ip address 192.168.10.33 255.255.255.240
!
router ospf 40
 redistribute rip metric 10 subnets
 network 192.168.10.2 0.0.0.0 area 192.168.10.0
 network 192.168.10.33 0.0.0.0 area 192.168.10.0
!
router rip
 network 172.19.0.0
```

上述配置可以在 FA0/0 接口的辅助网络上启用 RIP 协议，它可以让路由器 Matisse 从路由器 Dali 那里学习到路由，参见示例 8-34 所示。这些路由重新分配到 OSPF 域中（此时这些路由已不再运行在辅助地址上了），并且给它们指定一个 OSPF 的度量代价为 10，这是通过命令 **redistribute rip metric 10 subnets** 来实现的。关于路由重新分配的更加详细的介绍请参考第 11 章。在示例 8-35 中，通告到 OSPF 域内的路由是使用缺省的外部类型度量的，即外部类型 2（E2）。注意观察在路由器 Rubens 上，这些路由的代价仍然是 10。路由器 Matisse 将使用类型 5 的 LSA 通告这些外部的目的地址，并使自己成为一台 ASBR 路由器（参见示例 8-36 所示）。

示例 8-34 路由器 Dali 通过 RIP 协议将它的路由选择信息传递给路由器 Matisse

```
Matisse#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

R    192.168.90.0/24 [120/1] via 172.19.35.1, 00:00:09, FastEthernet0/0
R    192.168.105.0/24 [120/1] via 172.19.35.1, 00:00:09, FastEthernet0/0
     192.168.30.0/29 is subnetted, 2 subnets
O IA    192.168.30.0 [110/193] via 192.168.10.1, 02:12:53, Serial0/0.1
O IA    192.168.30.8 [110/192] via 192.168.10.1, 02:12:53, Serial0/0.1
R    192.168.60.0/24 [120/1] via 172.19.35.1, 00:00:09, FastEthernet0/0
     192.168.10.0/24 is variably subnetted, 3 subnets, 3 masks
C       192.168.10.64/26 is directly connected, FastEthernet0/1
C       192.168.10.32/28 is directly connected, FastEthernet0/0
C       192.168.10.0/27 is directly connected, Serial0/0.1
     172.19.0.0/25 is subnetted, 1 subnets
C       172.19.35.0 is directly connected, FastEthernet0/0
R    192.168.80.0/24 [120/1] via 172.19.35.1, 00:00:10, FastEthernet0/0

     192.168.20.0/30 is subnetted, 1 subnets
O IA    192.168.20.0 [110/128] via 192.168.10.1, 02:13:06, Serial0/0.1
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.4 is directly connected, Loopback0
R    192.168.70.0/24 [120/1] via 172.19.35.1, 00:00:13, FastEthernet0/0
R    192.168.100.0/24 [120/1] via 172.19.35.1, 00:00:13, FastEthernet0/0
R    192.168.101.0/24 [120/1] via 172.19.35.1, 00:00:13, FastEthernet0/0
```

示例 8-35 通过 RIP 协议学习到的路由将作为路径类型 E2 的路由重新分配到 OSPF 自主系统当中

```
Rubens#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

O E2 192.168.90.0/24 [110/10] via 192.168.30.10, 02:25:59, Serial0/0.1
O E2 192.168.105.0/24 [110/10] via 192.168.30.10, 02:25:59, Serial0/0.1
     192.168.30.0/29 is subnetted, 2 subnets
C       192.168.30.0 is directly connected, FastEthernet0/0
C       192.168.30.8 is directly connected, Serial0/0.1
O E2 192.168.60.0/24 [110/10] via 192.168.30.10, 02:25:59, Serial0/0.1
     192.168.10.0/24 is variably subnetted, 2 subnets, 2 masks
O IA    192.168.10.32/28 [110/193] via 192.168.30.10, 02:26:04, Serial0/0.1
O IA    192.168.10.0/27 [110/192] via 192.168.30.10, 02:26:11, Serial0/0.1
     172.19.0.0/25 is subnetted, 1 subnets
O E2    172.19.35.0 [110/10] via 192.168.30.10, 00:00:27, Serial0/0.1

O E2 192.168.80.0/24 [110/10] via 192.168.30.10, 02:26:00, Serial0/0.1
     192.168.20.0/30 is subnetted, 1 subnets
O IA    192.168.20.0 [110/128] via 192.168.30.10, 02:26:17, Serial0/0.1
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.1 is directly connected, Loopback0O E2 192.168.70.0/24 [110/10]
via 192.168.30.10, 02:26:03, Serial0/0.1
O E2 192.168.100.0/24 [110/10] via 192.168.30.10, 02:26:03, Serial0/0.1
O E2 192.168.101.0/24 [110/10] via 192.168.30.10, 02:26:03, Serial0/0.
```

示例 8-36 路由器 Matisse（RID=192.168.50.4）现在变成了一台 ASBR 路由器，这是因为它正在始发自主系统外部 LSA 来通告外部路由

```
Rubens#show ip ospf border-routers

OSPF Process 10 internal Routing Table

Codes: i - Intra-area route, I - Inter-area route

i 192.168.50.2 [64] via 192.168.30.10, Serial0/0.1, ABR, Area 1, SPF 7
I 192.168.50.4 [192] via 192.168.30.10, Serial0/0.1, ASBR, Area 1, SPF 7
```

8.2.5 案例研究：末梢区域

由于在图 8-46 中的区域 1 里面没有始发类型 5 的 LSA，因而它可以配置成一个末梢区域。注意，当一个相连的区域被配置成一个末梢区域时，路由器始发的 Hello 数据包进入那个区域后，它的可选字段中的 E 位将会设置为 0，即 E=0。其他没有同样配置的所有路由器收到这些 Hello 数据包后将丢弃这些数据包，并且不能在这些路由器之间建立邻接关系。即使已经存在一个邻接关系，它也会被阻断。因此，如果需要把一个正在运行的区域重新配置成一个末梢区域的话，应该计划好路由阻断的时间；因此在这期间路由将被阻断，一直到所有的路由器都重新配置后才能恢复。

一个末梢区域的配置可以通过在 OSPF 进程中增加命令 **area stub** 来完成，具体配置参见示例 8-37 和示例 8-38。

示例 8-37　在路由器 Rubens 上把区域 1 配置为末梢区域

```
router ospf 10
 network 0.0.0.0 255.255.255.255 area 1
 area 1 stub
```

示例 8-38　在路由器 Chardin 上把区域 1 配置为末梢区域

```
router ospf 20
 network 192.168.30.0 0.0.0.255 area 1
 network 192.168.20.0 0.0.0.255 area 0
 area 1 stub
```

比较路由器 Rubens 在配置成末梢区域前（参见示例 8-39）、后（参见示例 8-40）的链路状态数据库，可以显示出所有的自主系统外部 LSA 和 ASBR 汇总 LSA 都已经从这个数据库中清除。在这种情况下，数据库的大小也缩减了 50%。

示例 8-39　在区域 1 配置成末梢区域前，路由器 Rubens 的数据库总共包括 14 条 LSA

```
Rubens#show ip ospf database database-summary

            OSPF Router with ID (192.168.50.1) (Process ID 10)

Area 1 database summary
  LSA Type      Count    Delete   Maxage
  Router        2        0        0
  Network       0        0        0
  Summary Net   3        0        0
  Summary ASBR  1        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      6        0        0

Process 10 database summary
  LSA Type      Count    Delete   Maxage
  Router        2        0        0
  Network       0        0        0
  Summary Net   3        0        0
  Summary ASBR  1        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Type-5 Ext    8        0        0

  Opaque AS     0        0        0
  Total         14       0        0
```

示例 8-40　在配置成末梢区域后，使路由器 Rubens 从它的数据库中清除了 7 条类型 5 的 LSA 通告和 1 条类型 4 的 LSA，而只增加了一条通告缺省路由的类型 3 的 LSA

```
Rubens#show ip ospf database database-summary

            OSPF Router with ID (192.168.50.1) (Process ID 10)

Area 1 database summary
  LSA Type      Count    Delete   Maxage
```

（待续）

```
  Router          2         0         0
  Network         0         0         0
  Summary Net     4         0         0
  Summary ASBR    0         0         0
  Type-7 Ext      0         0         0
  Opaque Link     0         0         0
  Opaque Area     0         0         0
  Subtotal        6         0         0

Process 10 database summary
  LSA Type        Count     Delete    Maxage
  Router          2         0         0
  Network         0         0         0
  Summary Net     4         0         0
  Summary ASBR    0         0         0
  Type-7 Ext      0         0         0
  Opaque Link     0         0         0
  Opaque Area     0         0         0

  Type-5 Ext      0         0         0
  Opaque AS       0         0         0
  Total           6         0         0
```

当一个末梢区域和一台 ABR 路由器相连时，路由器将会通过一条网络汇总 LSA 自动地通告一个缺省路由（目的地址是 0.0.0.0）到这个区域。在示例 8-40 中所示的数据库汇总信息表明了这个额外的类型 3 的 LSA。路由器 Rubens 的路由表（示例 8-41）中的最后一个条目显示了这条缺省路由是由路由器 Chardin 通告的。

示例 8-41 路由器 Rubens 的路由表显示，所有的外部路由都被清除（请将这里所显示的和示例 8-35 显示的相比较），但增加了一条缺省路由

```
Rubens#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 192.168.30.10 to network 0.0.0.0

     192.168.30.0/29 is subnetted, 2 subnets
C       192.168.30.0 is directly connected, FastEthernet0/0
C       192.168.30.8 is directly connected, Serial0/0.1
     192.168.10.0/24 is variably subnetted, 2 subnets, 2 masks
O IA    192.168.10.32/28 [110/193] via 192.168.30.10, 00:05:24, Serial0/0.1
O IA    192.168.10.0/27 [110/192] via 192.168.30.10, 00:05:24, Serial0/0.1
     192.168.20.0/30 is subnetted, 1 subnets
O IA    192.168.20.0 [110/128] via 192.168.30.10, 00:05:24, Serial0/0.1
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.1 is directly connected, Loopback0
O*IA 0.0.0.0/0 [110/65] via 192.168.30.10, 00:05:25, Serial0/0.1
```

ABR 路由器将通告代价为 1 的缺省路由。在路由器 Rubens 和路由器 Chardin 之间的串行链路的代价是 64，在示例 8-41 中显示了路由器 Rubens 到达缺省路由的总代价将是 64+1=65。这里的缺省代价可以通过命令 **area default-cost** 来改变。例如，路由器 Chardin 可以配置一条代价为 20 的缺省路由，配置参见示例 8-42 所示。

示例 8-42 路由器 Chardin 的配置设置了缺省路由的代价

```
router ospf 20
 network 192.168.30.0 0.0.0.255 area 1
 network 192.168.20.0 0.0.0.255 area 0
 area 1 stub
 area 1 default-cost 20
```

结果，从示例 8-43 中可以看出，这条缺省路由的代价增大了——64+20=84。在这里改变缺省路由的代价没有什么实际意义，但是在多于一台 ABR 路由器的末梢区域里将可能会比较有用。通常情况下，每一台内部路由器都只会选择代价最低的缺省路由。通过操纵和控制所通告的代价，网络管理员可以促使所有的内部路由器都使用同一台 ABR 路由器。关于第二台 ABR 路由器，可以通告比较高的代价，而使它仅仅在第一台 ABR 路由器失效时才被使用。

示例 8-43 路由器 Rubens 的路由表反映了更改缺省路由的代价后的结果

```
Rubens#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 192.168.30.10 to network 0.0.0.0

     192.168.30.0/29 is subnetted, 2 subnets
C       192.168.30.0 is directly connected, FastEthernet0/0
C       192.168.30.8 is directly connected, Serial0/0.1
     192.168.10.0/24 is variably subnetted, 2 subnets, 2 masks
O IA    192.168.10.32/28 [110/193] via 192.168.30.10, 00:18:49, Serial0/0.1
O IA    192.168.10.0/27 [110/192] via 192.168.30.10, 00:18:49, Serial0/0.1
     192.168.20.0/30 is subnetted, 1 subnets
O IA    192.168.20.0 [110/128] via 192.168.30.10, 00:18:49, Serial0/0.1
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.1 is directly connected, Loopback0
O*IA 0.0.0.0/0 [110/84] via 192.168.30.10, 00:10:36, Serial0/0.1
```

8.2.6 案例研究：完全末梢区域

完全末梢区域的配置可以通过在命令 **area stub** 的末端增加关键字 **no-summary** 来实现。这一步的配置操作只有在 ABR 路由器上才是必需的，在内部路由器上使用标准的末梢区域配置就可以了。把图 8-46 的区域 1 配置成一个完全末梢区域，路由器 Chardin 上的配置参见示例 8-44。

示例 8-44 在路由器 Chardin 上把区域 1 配置成一个完全末梢区域

```
router ospf 20
 network 192.168.30.0 0.0.0.255 area 1
 network 192.168.20.0 0.0.0.255 area 0
 area 1 stub no-summary
```

在示例 8-45 中，可以看出路由器 Rubens 的数据库中 LSA 的数量已经减少到 3 条了。在示例 8-46 中显示了它的路由表。

示例 8-45 改变区域 1 成为一个完全末梢区域，消除了类型 3 的 LSA 之外的所有 LSA（缺省路由）

```
Rubens#show ip ospf database database-summary

            OSPF Router with ID (192.168.50.1) (Process ID 10)

Area 1 database summary
  LSA Type      Count    Delete   Maxage
  Router        2        0        0
  Network       0        0        0
  Summary Net   1        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      3        0        0

Process 10 database summary
  LSA Type      Count    Delete   Maxage
  Router        2        0        0
  Network       0        0        0
  Summary Net   1        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0

  Type-5 Ext    0        0        0
  Opaque AS     0        0        0
  Total         3        0        0
```

示例 8-46 在完全末梢区域中的路由表将只包含区域内的路由和缺省路由

```
Rubens#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 192.168.30.10 to network 0.0.0.0

     192.168.30.0/29 is subnetted, 2 subnets
C       192.168.30.0 is directly connected, FastEthernet0/0
C       192.168.30.8 is directly connected, Serial0/0.1
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.1 is directly connected, Loopback0
O*IA 0.0.0.0/0 [110/84] via 192.168.30.10, 00:15:52, Serial0/0.1
```

8.2.7 案例研究：NSSA 区域

在前面的 8.2.4 小节中，讨论到路由器 Matisse 接受了通过 RIP 从路由器 Dali 学习到的路由，并把这些路由重新分配到 OSPF 域中（请参见图 8-46）。这一步操作使路由器 Matisse 成为一台 ASBR 路由器，更进一步来说，也使区域 192.168.10.0 无法满足成为一个末梢区域或完全末梢区域的条件。然而，这里并不需要 AS 外部 LSA 从骨干区域通告到这个区域。因此，可以把区域 192.168.10.0 配置成一个 NSSA，在路由器 Matisse 上的

配置见示例 8-47。

示例 8-47　路由器 Matisse 把区域 192.168.10.0 配置成一个 NSSA

```
router ospf 40
 redistribute rip metric 10
 network 192.168.10.2 0.0.0.0 area 192.168.10.0
 network 192.168.10.33 0.0.0.0 area 192.168.10.0
 area 192.168.10.0 nssa
!
router rip
 network 172.19.0.0
```

这里显示的 **area nssa** 语句可以同样地配置在路由器 Goya 上。由于路由器 Goya 是一台 ABR 路由器，因此它将把与 NSSA 区域相连的接口收到的类型 7 的 LSA 转换成类型 5 的 LSA。这些转换过的 LSA 将泛洪扩散到整个骨干区域中去，从而也泛洪扩散到其他的区域中去。比较路由器 Goya 和路由器 Chardin 的路由表，可以看出路由器 Goya 把外部路由标记成 NSSA[1]（如示例 8-48 所示）。而路由器 Chardin 把外部路由标记成 E2（如示例 8-49 所示），这表明它们是从类型 5 的 LSA 学到的。

示例 8-48　从路由器 Matisse 学到的外部路由在路由器 Goya 上被标记成 NSSA 路由

```
Goya#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

O N2 192.168.90.0/24 [110/10] via 192.168.10.2, 00:00:41, Serial0/0.1
O N2 192.168.105.0/24 [110/10] via 192.168.10.2, 00:00:41, Serial0/0.1
     192.168.30.0/29 is subnetted, 2 subnets
O IA    192.168.30.0 [110/129] via 192.168.20.1, 00:00:41, Serial0/0.2
O IA    192.168.30.8 [110/128] via 192.168.20.1, 00:00:41, Serial0/0.2
O N2 192.168.60.0/24 [110/10] via 192.168.10.2, 00:00:41, Serial0/0.1
     192.168.10.0/24 is variably subnetted, 2 subnets, 2 masks
O       192.168.10.32/28 [110/65] via 192.168.10.2, 00:00:43, Serial0/0.1
C       192.168.10.0/27 is directly connected, Serial0/0.1
     172.19.0.0/25 is subnetted, 1 subnets
O N2    172.19.35.0 [110/10] via 192.168.10.2, 00:00:43, Serial0/0.1
O N2 192.168.80.0/24 [110/10] via 192.168.10.2, 00:00:43, Serial0/0.1
     192.168.20.0/30 is subnetted, 1 subnets
C       192.168.20.0 is directly connected, Serial0/0.2
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.3 is directly connected, Loopback0
O N2 192.168.70.0/24 [110/10] via 192.168.10.2, 00:00:55, Serial0/0.1
O N2 192.168.100.0/24 [110/10] via 192.168.10.2, 00:00:56, Serial0/0.1
O N2 192.168.101.0/24 [110/10] via 192.168.10.2, 00:00:56, Serial0/0.1
```

[1] N2 表明的是和 E2 相同的度量计算——也就是说，只使用外部的代价。随后的一个例子将演示 E1 和 N1 的度量类型。

示例 8-49 路由器 Chardin 把同样的路由标记成 E2 类型，表明它们是从自主系统外部 LSA 学习到的

```
Chardin#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

O E2 192.168.90.0/24 [110/10] via 192.168.20.2, 00:02:49, Serial0/0.2
O E2 192.168.105.0/24 [110/10] via 192.168.20.2, 00:02:49, Serial0/0.2
     192.168.30.0/29 is subnetted, 2 subnets
O       192.168.30.0 [110/65] via 192.168.30.9, 00:08:41, Serial0/0.1
C       192.168.30.8 is directly connected, Serial0/0.1
O E2 192.168.60.0/24 [110/10] via 192.168.20.2, 00:02:49, Serial0/0.2
     192.168.10.0/24 is variably subnetted, 2 subnets, 2 masks
O IA    192.168.10.32/28 [110/129] via 192.168.20.2, 00:02:55, Serial0/0.2
O IA    192.168.10.0/27 [110/128] via 192.168.20.2, 00:03:16, Serial0/0.2
     172.19.0.0/25 is subnetted, 1 subnets
O E2    172.19.35.0 [110/10] via 192.168.20.2, 00:02:50, Serial0/0.2
O E2 192.168.80.0/24 [110/10] via 192.168.20.2, 00:02:50, Serial0/0.2
     192.168.20.0/30 is subnetted, 1 subnets
C       192.168.20.0 is directly connected, Serial0/0.2
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.2 is directly connected, Loopback0
O E2 192.168.70.0/24 [110/10] via 192.168.20.2, 00:02:52, Serial0/0.2
O E2 192.168.100.0/24 [110/10] via 192.168.20.2, 00:02:52, Serial0/0.2
O E2 192.168.101.0/24 [110/10] via 192.168.20.2, 00:02:52, Serial0/0.2
```

这个转换也可以通过检查路由器 Goya 的链路状态数据库来观察。在示例 8-50 中，数据库中同时包含了相同外部路由的类型 7 和类型 5 的 LSA。只不过类型 7 的 LSA 是始发于路由器 Matisse 的，而类型 5 的 LSA 是始发于路由器 Goya 的。

示例 8-50 路由器 Goya 的链路状态数据库表明了来自路由器 Matisse（192.168.50.4）的类型 7 LSA 已经被 Goya（192.168.50.3）转换成类型 5 的 LSA 了

```
                Type-7 AS External Link States (Area 192.168.10.0)

Link ID         ADV Router      Age         Seq#       Checksum Tag
172.19.35.0     192.168.50.4    371         0x80000001 0x00C444 0
192.168.60.0    192.168.50.4    371         0x80000001 0x00A521 0
192.168.70.0    192.168.50.4    371         0x80000001 0x003785 0
192.168.80.0    192.168.50.4    371         0x80000001 0x00C8E9 0
192.168.90.0    192.168.50.4    371         0x80000001 0x005A4E 0
192.168.100.0   192.168.50.4    371         0x80000001 0x00EBB2 0
192.168.101.0   192.168.50.4    371         0x80000001 0x00E0BC 0
192.168.105.0   192.168.50.4    371         0x80000001 0x00B4E4 0

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
172.19.35.0     192.168.50.3    305         0x80000001 0x005FB4 0
192.168.60.0    192.168.50.3    306         0x80000001 0x004091 0
192.168.70.0    192.168.50.3    306         0x80000001 0x00D1F5 0
```

（待续）

```
192.168.80.0    192.168.50.3    306         0x80000001 0x00635A 0
192.168.90.0    192.168.50.3    390         0x80000001 0x00F4BE 0
192.168.100.0   192.168.50.3    390         0x80000001 0x008623 0
192.168.101.0   192.168.50.3    390         0x80000001 0x007B2D 0
192.168.105.0   192.168.50.3    390         0x80000001 0x004F55 0
```

对于 ABR 路由器来说还有几个可用的配置选项。第一个是 **no-summary** 选项，可以和命令 **area nssa** 一起用来阻塞类型 3 和类型 4 的 LSA 泛洪扩散到 NSSA 中。这个配置可以使区域 192.168.10.0 变成一个名字有点怪异的区域——“完全非纯末梢区域”（totally stubby not-so-stubby area）。这时，路由器 Goya 上的配置应该如示例 8-51 所显示的。

示例 8-51 路由器 Goya 把区域 192.168.10.0 配置成一个完全非纯末梢区域

```
router ospf 30
 network 192.168.20.0 0.0.0.3 area 0
 network 192.168.10.0 0.0.0.31 area 192.168.10.0
 area 192.168.10.0 nssa no-summary
```

路由器 Matisse 的路由表参见示例 8-52 所示，可以看出所有的区域间路由都被消除了，而另外增加了一条由路由器 Goya 通告的缺省路由。

示例 8-52 所有的区域间路由都被一条到达 ABR 的缺省路由替代

```
Matisse#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 192.168.10.1 to network 0.0.0.0

R    192.168.90.0/24 [120/1] via 172.19.35.1, 00:00:20, FastEthernet0/0
R    192.168.105.0/24 [120/1] via 172.19.35.1, 00:00:20, FastEthernet0/0
R    192.168.60.0/24 [120/1] via 172.19.35.1, 00:00:20, FastEthernet0/0
     192.168.10.0/24 is variably subnetted, 3 subnets, 3 masks
C       192.168.10.64/26 is directly connected, FastEthernet0/1
C       192.168.10.32/28 is directly connected, FastEthernet0/0
C       192.168.10.0/27 is directly connected, Serial0/0.1
     172.19.0.0/25 is subnetted, 1 subnets
C       172.19.35.0 is directly connected, FastEthernet0/0
R    192.168.80.0/24 [120/1] via 172.19.35.1, 00:00:21, FastEthernet0/0
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.4 is directly connected, Loopback0
R    192.168.70.0/24 [120/1] via 172.19.35.1, 00:00:21, FastEthernet0/0

R    192.168.100.0/24 [120/1] via 172.19.35.1, 00:00:22, FastEthernet0/0
R    192.168.101.0/24 [120/1] via 172.19.35.1, 00:00:22, FastEthernet0/0
O*IA 0.0.0.0/0 [110/65] via 192.168.10.1, 00:11:40, Serial0/0.1
```

在图 8-47 中，路由器 Dali 的链路从路由器 Matisse 移到 Goya 的接口上，相关的 IP 地址也作相应的变动。路由器 Goya 现在成为了一台 ASBR 路由器，并把 RIP 学习到的路由重新分配到 OSPF 当中。

当一台 ABR 路由器同时也是一台 ASBR 路由器，并且和一个非纯末梢区域相连时，它的缺省行为是通告重新分配的路由到这个 NSSA 中去，参见示例 8-53。

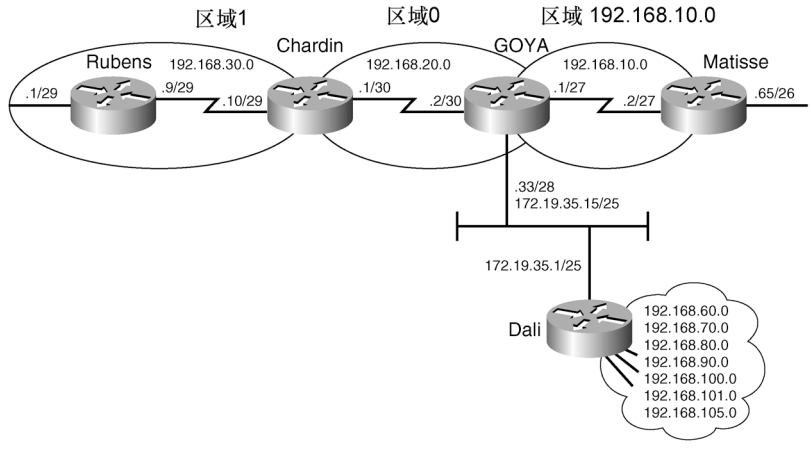


图 8-47 路由器 Dali 的链路移到了路由器 Goya 上，现在变成路由器 Goya 来宣告 RIP 到路由器 Dali，并且重新分配学到的路由到 OSPF

示例 8-53 一台 ABR 路由器同时也是一台 ASBR 路由器时，将会利用类型 7 的 LSA 通告外部路由到 NSSA 区域。在这个例子中，路由器 Goya 正在使用 N1 的度量类型通告外部路由

```
Matisse#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

O N1 192.168.90.0/24 [110/74] via 192.168.10.1, 00:00:07, Serial0/0.1
O N1 192.168.105.0/24 [110/74] via 192.168.10.1, 00:00:07, Serial0/0.1
     192.168.30.0/29 is subnetted, 2 subnets
O IA    192.168.30.0 [110/193] via 192.168.10.1, 00:03:11, Serial0/0.1
O IA    192.168.30.8 [110/192] via 192.168.10.1, 00:03:11, Serial0/0.1
O N1 192.168.60.0/24 [110/74] via 192.168.10.1, 00:00:07, Serial0/0.1
     192.168.10.0/24 is variably subnetted, 3 subnets, 3 masks
C       192.168.10.64/26 is directly connected, FastEthernet0/1
C       192.168.10.32/28 is directly connected, FastEthernet0/0
C       192.168.10.0/27 is directly connected, Serial0/0.1
     172.19.0.0/25 is subnetted, 1 subnets
O N1    172.19.35.0 [110/74] via 192.168.10.1, 00:03:07, Serial0/0.1
O N1 192.168.80.0/24 [110/74] via 192.168.10.1, 00:00:08, Serial0/0.1
     192.168.20.0/30 is subnetted, 1 subnets

O IA    192.168.20.0 [110/128] via 192.168.10.1, 00:03:14, Serial0/0.1
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.4 is directly connected, Loopback0
O N1 192.168.70.0/24 [110/74] via 192.168.10.1, 00:00:10, Serial0/0.1
O N1 192.168.100.0/24 [110/74] via 192.168.10.1, 00:00:10, Serial0/0.1
O N1 192.168.101.0/24 [110/74] via 192.168.10.1, 00:00:10, Serial0/0.1
```

这个在 ABR/ASBR 路由器上缺省的路由重新分配行为可以通过在命令 **area nssa** 之后增加参数 **no-redistribution** 来关闭。这样，在所述例子的网络中，就不应该有类型 3、类型 4、类型 5 或类型 7 的 LSA 从 ABR 路由器发送到区域 192.168.10.0。所需要的路由重新分配可以

通过示例 8-54 的配置在路由器 Goya 上完成。[1]

示例 8-54 路由器 Goya 的配置不把 RIP 路由重新分配到 NSSA 192.168.10.0

```
interface Ethernet0/0
 ip address 172.19.35.15 255.255.255.128
!
router ospf 30
 redistribute rip metric 10 metric-type 1 subnets
 network 192.168.20.0 0.0.0.3 area 0
 network 192.168.10.0 0.0.0.31 area 192.168.10.0
 area 192.168.10.0 nssa no-redistribution no-summary
!
router rip
 network 172.19.0.0
```

在这里，**area nssa** 命令阻塞了类型 5 的 LSA 通过路由器 Goya 进入到该区域，而 **no-redistribution** 阻塞了类型 7 的 LSA，命令 **no-summary** 阻塞了类型 3 和类型 4 的 LSA。和前面一样，命令 **no-summary** 使路由器 Goya 发送一条类型 3 的 LSA 来向该区域通告一个缺省路由。在示例 8-55 中，显示了在路由器 Goya 禁止了类型 7 的路由重新分配之后路由器 Matisse 的路由表。注意，即使外部网络不在这个路由表中，它们依然是可达的，这是因为路由表中含有缺省路由的缘故。

示例 8-55 在路由器 Goya 的 area nssa 命令中增加了 no-redistribution 参数后，示例 8-53 中的路由表将不再包括从类型 7 的 LSA 中学到的路由

```
Matisse#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 192.168.10.1 to network 0.0.0.0

     192.168.10.0/24 is variably subnetted, 3 subnets, 3 masks
C       192.168.10.64/26 is directly connected, FastEthernet0/1
C       192.168.10.32/28 is directly connected, FastEthernet0/0
C       192.168.10.0/27 is directly connected, Serial0/0.1
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.4 is directly connected, Loopback0
O*IA 0.0.0.0/0 [110/65] via 192.168.10.1, 00:00:51, Serial0/0.1
```

在最后的这个例子中，假设需要路由器 Goya 允许类型 3 和类型 4 的 LSA 泛洪扩散到 NSSA 区域，但是不允许类型 5 和类型 7 的 LSA 泛洪扩散到该区域。这里的问题是当在路由器上去除了 **no-summary** 参数后，ABR 路由器将不再始发一条类型 3 的 LSA 通告缺省路由了。没有缺省路由，外部网络也就无法从 NSSA 区域内部到达。要解决这个问题，可以在命令 **area nssa** 后面增加一个 **default-information-originate** 参数，这就可以使用 ABR 路由器来通告一条缺省路由到这个 NSSA 区域。这时，它是使用类型 7 的 LSA 来通告这条缺省路由的。要使用这个参数，路由器 Goya 上的 OSPF 配置参见示例 8-56。

[1] 注意，在 **redistribution** 命令中使用了 *metric-type 1* 参数。这个参数的作用是使外部的目的路由利用 E1 度量来通告。在 NSSA 中，度量类型变成了 N1，参见示例 8-53 所示。

示例 8-56 路由器 Goya 使用 **default-information-originate** 命令发起缺省路由

```
router ospf 30
 redistribute rip metric 10 metric-type 1 subnets
 network 192.168.20.0 0.0.0.3 area 0
 network 192.168.10.0 0.0.0.31 area 192.168.10.0
 area 192.168.10.0 nssa no-redistribution default-information-originate
```

在示例 8-57 中，显示了做过重新配置后的路由器 Matisse 的路由表。在这里，路由表中包括了区域间路由和一条缺省路由，这条缺省路由带有 N2 标志，表明这条路由是从类型 7 的 LSA 学习到的。

示例 8-57 在命令 area nssa 中增加 default-information-originate 参数后，可以使 ABR 路由器通告一条缺省路由到这个 NSSA 区域

```
Matisse#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 192.168.10.1 to network 0.0.0.0

     192.168.30.0/29 is subnetted, 2 subnets
O IA    192.168.30.0 [110/193] via 192.168.10.1, 00:00:26, Serial0/0.1
O IA    192.168.30.8 [110/192] via 192.168.10.1, 00:00:26, Serial0/0.1
     192.168.10.0/24 is variably subnetted, 3 subnets, 3 masks
C       192.168.10.64/26 is directly connected, FastEthernet0/1
C       192.168.10.32/28 is directly connected, FastEthernet0/0
C       192.168.10.0/27 is directly connected, Serial0/0.1
     192.168.20.0/30 is subnetted, 1 subnets
O IA    192.168.20.0 [110/128] via 192.168.10.1, 00:00:27, Serial0/0.1
     192.168.50.0/32 is subnetted, 1 subnets
C       192.168.50.4 is directly connected, Loopback0
O*N2 0.0.0.0/0 [110/1] via 192.168.10.1, 00:00:22, Serial0/0.1
```

8.2.8 案例研究：地址汇总

虽然末梢区域可以通过防止某些 LSA 进入该区域，从而达到在一个非骨干的域中节省资源的目的，但是从骨干区域上来看，这些区域除了节省资源外并没有做其他任何事情。一个区域内所有的地址仍然会通告到骨干区域中。这种情形可以通过地址汇总来帮助解决。像末梢区域一样，地址汇总也是通过减少泛洪扩散的 LSA 数量来达到节省资源的目的。另外，它还可以通过屏蔽一些网络不稳定的细节来节省资源。例如，一个忽好忽坏的不稳定的子网在它每一次状态发生转变的时候，都会引起 LSA 在整个网络中进行泛洪扩散。但是，如果这个子网地址被汇总后包含在一个汇总地址中，那么单独的子网和它的稳定性就不再被通告出去了。

在 Cisco 路由器上可以执行两种类型的地址汇总：区域间路由汇总和外部路由汇总。区域间路由汇总（inter-area summarization），顾名思义，是指在区域之间的地址汇总。这种类型的汇总通常是配置在 ABR 路由器上的。外部路由汇总（external route summarization）允许一组外部地址汇总为一条汇总地址并通过重新分配注入到一个 OSPF 域中，这种类型的汇总通常是配置在

ASBR 路由器上的。区域间路由汇总将在本小节介绍，而外部路由汇总将在第 11 章讲述。

在图 8-48 中，区域 15 包含了 8 个子网：10.0.0.0/16～10.7.0.0/16。图 8-49 显示了这些子网地址可以使用单个汇总地址 10.0.0.0/13 来表示。

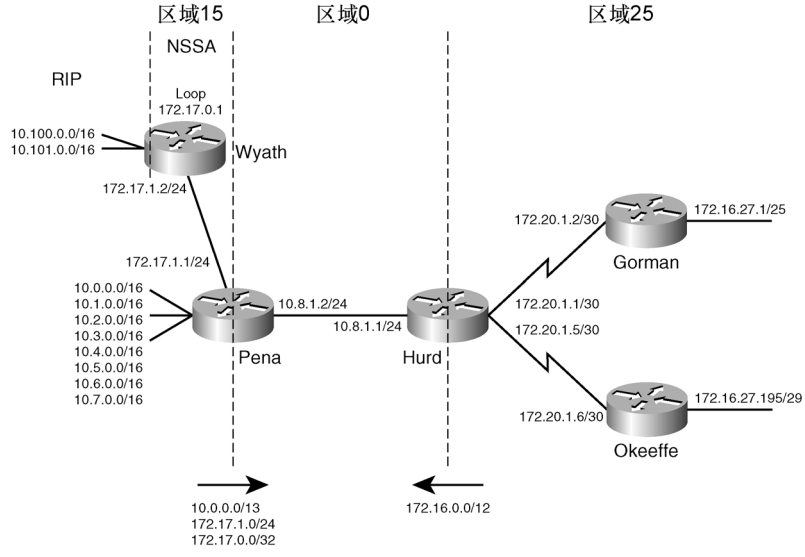


图 8-48　在区域 15 和区域 25 中的地址能够汇总到骨干区域里

```
11111111111111110000000000000000 = 16位掩码
00001010000000000000000000000000 = 10.0.0.0/16
00001010000000010000000000000000 = 10.1.0.0/16
00001010000000100000000000000000 = 10.2.0.0/16
00001010000000110000000000000000 = 10.3.0.0/16
00001010000001000000000000000000 = 10.4.0.0/16
00001010000001010000000000000000 = 10.5.0.0/16
00001010000001100000000000000000 = 10.6.0.0/16
00001010000001110000000000000000 = 10.7.0.0/16
00001010000000000000000000000000 = 10.0.0.0/13
```

图 8-49　可以使用汇总地址 10.0.0.0/13 来表示地址 10.0.0.0/16～10.7.0.0/16 的地址范围

在一台 ABR 路由器上配置一个汇总地址，既可以通告到骨干区域，也可以通告到一个非骨干的区域。最好的方法是，一个非骨干区域的地址应该通过它自己的 ABR 路由器汇总到骨干区域。而与之相对的是使其他所有的 ABR 路由器将这个区域汇总到它们各自的区域内。然后，在骨干区域中，被汇总的地址将穿越骨干区域并通告到其他区域中去。这两种方法都简化了路由器的配置并减小了骨干区域内链路状态数据库的大小。

area range 命令指定了汇总地址所属的区域、汇总地址和地址掩码。回忆第 7 章的内容，当为 EIGRP 协议配置一条汇总路由时，会有一条指向空接口（null Interface）的路由被自动地加入路由表中，以便用来避免路由黑洞和路由环路。[1] OSPF 也不会自动地加入这条路由。但是，在 IOS 软件早于 12.1 版的主要版本中，不会自动地加入 null 接口的路由。在能够创建这条路由的 IOS 软件版本中，路由器也可以使用命令 **no discard-route** 不在它的路由表中加载这条路由。因此，无论何时，在一个 OSPF 域内配置汇总路由时，都应该确认是否为这条汇总地址增加了指向 null 接口的静态路由。如果它没有被自动创建，那么必须增加这条路由

[1] 增加这条路由的理由将在第 11 章和第 12 章中结合实例，作更详细的讲述。

或使用 **discard-route** 命令。

示例 8-58 显示了路由器 Pena 的 OSPF 配置。

示例 8-58 路由器 Pena 的 OSPF 配置

```
router ospf 1
 network 10.0.0.0 0.7.255.255 area 15
 network 10.8.0.0 0.7.255.255 area 0
 area 15 range 10.0.0.0 255.248.0.0
!
ip route 10.0.0.0 255.248.0.0 Null0
```

图 8-49 中显示出 10.0.0.0/13 表示的地址范围是连续的，也就是说，被汇总的 3 个二进制位构成了每一个 000～111 的组合。而在区域 25 中的地址不同，它们不能形成一个连续的地址范围。但是，它们仍然可以通过示例 8-59 中的配置在路由器 Hurd 上进行汇总。

示例 8-59 路由器 Hurd 的 OSPF 配置

```
router ospf 1
 network 10.8.0.0 0.0.255.255 area 0
 network 172.20.0.0 0.0.255.255 area 25
 area 25 range 172.16.0.0 255.240.0.0
!
ip route 172.16.0.0 255.240.0.0 Null0
```

即使在这个汇总地址范围内的地址出现在这个网络以外，该汇总路由也是可以正常工作的。在图 8-48 中，网络 172.17.0.0/16 在区域 15 内，尽管这个地址是属于来自区域 25 汇总的那一段地址的。路由器 Pena 将通告这个地址到骨干区域，而路由器 Hurd 将学到它并通告到区域 25 中去。跟随这个地址的网络掩码比汇总地址 172.16.0.0/12 的掩码更具体（也就是说，是更长）；又因为 OSPF 协议是无类别的路由选择协议，因此，OSPF 将能够转发属于网络 172.17.0.0 的目的地址到正确的目的地。

尽管图 8-48 中的地址配置方法可以工作，但是这并不是一个值得推荐的设计方法。汇总地址的地方是节省了资源，但对网络 172.17.1.0/24 的通告还是必须独立于网络 172.16.0.0/12，并穿过骨干区域。在像这样的网络设计中，如果使用了缺省路由，还可能会产生路由环路的隐患。这个问题会在第 12 章中进行详细讲述。

这里也要注意图 8-48 中，子网 172.16.27.0/25（和路由器 Gorman 相连）和子网 172.16.27.192/29（和路由器 Okeeffe 相连）是不连续的。又因为 OSPF 协议是一个无类别路由选择协议，因而路由器 Gorman 和 Okeeffe 不会扮演网络边界路由器的角色。这些子网和它们的掩码将会通告到网络 172.20.0.0 中去，并且不会有路由选择不确定的情况发生。

area range 命令的缺省行为是通告指定的范围。它也可以用来抑制一个地址范围的通告。带有关键字 **not-advertise** 的命令 **area range** 将使指定范围的前缀被抑制。它们将不会在 LSA 中通告。

配置路由器 Pena 来抑制范围 172.17.0.0/16。路由器 Pena 的配置参见示例 8-60 所示。

示例 8-60 路由器 Pena 的配置抑制了指定地址范围的通告

```
router ospf 1
 area 15 range 10.0.0.0 255.248.0.0
 area 15 range 172.17.0.0 255.255.0.0 not-advertise
 network 10.0.0.0 0.7.255.255 area 15
 network 10.8.0.0 0.7.255.255 area 0
 network 172.17.0.0 0.0.255.255 area 15
```

这条命令达到了抑制 172.17.0.0/16 地址范围的预期影响，但是也产生了一个不希望出现的结果。请查看下面路由器 Hurd 在配置这条命令之前（参见示例 8-61）和之后（参见示例 8-62）的路由表。

示例 8-61 在路由器 Pena 抑制一个地址范围之前，路由器 Hurd 的路由表中显示了两条外部路由

```
Hurd#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

     172.17.0.0/16 is variably subnetted, 2 subnets, 2 masks
O IA    172.17.1.0/24 [110/11] via 10.8.1.2, 00:03:29, Ethernet0/0
O IA    172.17.0.1/32 [110/12] via 10.8.1.2, 00:03:29, Ethernet0/0
     172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       172.16.27.192/29 [110/65] via 172.20.1.6, 00:03:29, Serial0/0.2
O       172.16.27.0/25 [110/65] via 172.20.1.2, 00:03:29, Serial0/0.1
     172.20.0.0/30 is subnetted, 2 subnets
C       172.20.1.0 is directly connected, Serial0/0.1
C       172.20.1.4 is directly connected, Serial0/0.2
     10.0.0.0/8 is variably subnetted, 4 subnets, 3 masks
C       10.8.1.0/24 is directly connected, Ethernet0/0
O IA    10.0.0.0/13 [110/11] via 10.8.1.2, 00:03:30, Ethernet0/0
O E2    10.100.0.0/16 [110/10] via 10.8.1.2, 00:03:30, Ethernet0/0
O E2    10.101.0.0/16 [110/10] via 10.8.1.2, 00:03:32, Ethernet0/0
S    172.16.0.0/12 is directly connected, Null0
```

示例 8-62 在路由器 Pena 抑制一个地址范围之后，路由器 Hurd 的路由表中显示这两条外部路由已经在路由表中了

```
Hurd#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

     172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       172.16.27.192/29 [110/65] via 172.20.1.6, 00:06:11, Serial0/0.2
O       172.16.27.0/25 [110/65] via 172.20.1.2, 00:06:11, Serial0/0.1
     172.20.0.0/30 is subnetted, 2 subnets
C       172.20.1.0 is directly connected, Serial0/0.1
C       172.20.1.4 is directly connected, Serial0/0.2
     10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.8.1.0/24 is directly connected, Ethernet0/0
O IA    10.0.0.0/13 [110/11] via 10.8.1.2, 00:06:12, Ethernet0/0
S    172.16.0.0/12 is directly connected, Null0
```

在路由器 Wyeth 上通过 RIP 学习到的外部路由 10.100.0.0 和 10.101.0.0 已经不在路由器

Hurd 的路由表中了。示例 8-63 显示了 Hurd 的 OSPF 数据库中的 10.100.0.0。请注意指定了的转发地址。

示例 8-63 类型 5 的外部链路状态显示转发地址是始发这条外部路由的路由器的地址

```
Hurd#show ip ospf data external

            OSPF Router with ID (172.20.1.5) (Process ID 1)

                Type-5 AS External Link States

  Routing Bit Set on this LSA
  LS age: 421
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 10.100.0.0 (External Network Number )
  Advertising Router: 172.17.0.1
  LS Seq Number: 80000001
  Checksum: 0xF1CC
  Length: 36
  Network Mask: /16
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 10
        Forward Address: 172.17.0.1
        External Route Tag: 0
```

这里的转发地址是路由器 Wyeth 的 loopback 接口。如果指定了外部 LSA 的转发地址，而这个地址是不可达的，那么这个 LSA 中包含的地址就不会插入到路由表中。当路由器 Pena 将类型 7 的 NSSA LSA 转换为类型 5 的 LSA 时，缺省情况下转发地址将从类型 7 转换为类型 5。在转换期间，可以配置 ABR 路由器来抑制它的转发地址，这可以使用地址 0.0.0.0 代替指定的地址。当另一台路由器接收这个抑制了转发地址的类型 5 外部 LSA 时，接收路由器将会把要到达外部地址的流量引导到转换类型 7 到类型 5 的 ABR 路由器上，而不是引导到转发地址。

示例 8-64 显示了路由器 Pena 的新配置。

示例 8-64 路由器 Pena 的配置抑制了包括在转换类型 5 LSA 中的转发地址

```
router ospf 1
 area 15 range 10.0.0.0 255.248.0.0
 area 15 range 172.17.0.0 255.255.0.0 not-advertise
 network 10.0.0.0 0.7.255.255 area 15
 network 10.8.0.0 0.7.255.255 area 0
 network 172.17.0.0 0.0.255.255 area 15
 area 15 nssa translate type7 suppress-fa
```

示例 8-65 显示了路由器 Hurd 的外部数据库中有关 10.100.0.0 的条目，示例 8-66 显示了路由器 Hurd 的新路由表。

示例 8-65 外部 OSPF 数据库的条目显示了一个被抑制的转发地址

```
Hurd#show ip ospf data external

            OSPF Router with ID (172.20.1.5) (Process ID 1)

                Type-5 AS External Link States
```

（待续）

```
  Routing Bit Set on this LSA
  LS age: 13
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 10.100.0.0 (External Network Number )
  Advertising Router: 172.17.0.1
  LS Seq Number: 80000002
  Checksum: 0xA9D2
  Length: 36
  Network Mask: /16
        Metric Type: 2 (Larger than any link state path)
        ToS: 0
        Metric: 10
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

示例 8-66　在抑制了转发地址后，外部路由 10.100.0.0 和 10.101.0.0 被插入到路由表中

```
Hurd#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

     172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
O       172.16.27.192/29 [110/65] via 172.20.1.6, 00:09:32, Serial0/0.2
O       172.16.27.0/25 [110/65] via 172.20.1.2, 00:09:32, Serial0/0.1
     172.20.0.0/30 is subnetted, 2 subnets
C       172.20.1.0 is directly connected, Serial0/0.1
C       172.20.1.4 is directly connected, Serial0/0.2
     10.0.0.0/8 is variably subnetted, 4 subnets, 3 masks
C       10.8.1.0/24 is directly connected, Ethernet0/0
O IA    10.0.0.0/13 [110/11] via 10.8.1.2, 00:09:33, Ethernet0/0
O E2    10.100.0.0/16 [110/10] via 10.8.1.2, 00:00:46, Ethernet0/0
O E2    10.101.0.0/16 [110/10] via 10.8.1.2, 00:00:46, Ethernet0/0
S    172.16.0.0/12 is directly connected, Null0
```

在路由器 Hurd 的数据库中显示的转发地址现在是 0.0.0.0，原来两条外部路由回到了路由表中。

8.2.9　案例研究：在区域之间进行过滤

在路由器 Okeeffe 上增加另外一个 LAN。LAN 上的设备可以被区域 25 内的其他设备访问，但不能被这个网络中其余的设备访问。在区域之间限制和控制地址交换的另一种方法是在 ABR 上配置类型 3 的 LSA 过滤。ABR 可以过滤由类型 3 的 LSA 通告的网络地址进出某个区域。

在区域 25 的路由器 Okeeffe 增加 LAN 的地址 192.168.1.0/24。这个在 LSA 中通告的地址虽然只能被区域 25 内的设备访问，但可以通过整个网络。为了避免这个地址被通告到区域 25 的外部，可以在路由器 Hurd 上配置类型 3 的 LSA 的过滤，参见示例 8-67。

示例 8-67 路由器 Hurd 应用了类型 3 的 LSA 的过滤

```
router ospf 1
 area 25 filter-list prefix area25outbound out
!
ip prefix-list area25outbound seq 10 deny 192.168.1.0/24
ip prefix-list area25outbound seq 20 permit 0.0.0.0/0 le 32
```

OSPF 命令 **area** *PID* **filter-list prefix** 指定了应用于入站或出站 LSA 的过滤列表的名称。出站列表过滤了发送到由该命令指定的区域之外的其他区域的 LSA。在我们的例子中，该列表过滤了由区域 25 发起的，并发送到非区域 25 的区域的地址的 LSA，例如区域 0。入站列表过滤了发送到区域 25 的 LSA。

前缀列表第一行的作用是清楚的。这条语句拒绝地址 192.168.1.0/24 由类型 3 的 LSA 通告出去。第二行语句允许所有其他地址通过：从掩码长度为 0 位到 32 位的所有地址。这里第二行的语句是必须的，因为在前缀列表的末尾缺省隐含的语句是“deny all”。地址 192.168.1.0/24 防止在类型 3 的 LSA 中发送到区域 25 外部，而其他所有的地址都被允许通过。

8.2.10 案例研究：认证

OSPF 数据包可以通过认证来防止有意或无意地引入有害路由信息的情况发生。表 8-8 中列出了有效的认证类型。Null 认证（类型 0）是路由器缺省使用的类型，表示在数据包头部没有包含认证信息，也就是说，不需要认证。在路由器上可以使用简单的明文口令（类型 1）或者 MD5 加密校验和（类型 2）来配置认证。如果一个区域内某处配置了认证，那么就必须在整个区域内都配置认证。

如果网络管理者是以增加网络的安全性为目标的，那么只有在 OSPF 区域内的设备不能支持更安全的类型 2 的认证时才考虑使用类型 1 的认证。明文认证会在网络上给网络攻击者留下一个安全漏洞，因为网络上传送的数据包能够被协议分析仪捕获，并读出所设置的口令（请参考第 6 章，并请注意图 6-8）。但是，类型 1 的认证在进行 OSPF 的重新配置时会变得比较有用。例如，不同的口令可以用在进行重新配置时“旧”OSPF 路由器和“新”OSPF 路由器上，从而避免它们在共享一个公共广播网络的情况下相互通信。

在一个区域上配置类型 1 的认证方式，可以使用命令 **ip ospf authentication-key** 来为和该区域相连的每一个接口分配一个口令，这个口令最长为 8 个八位组字节长。所分配的口令不必在整个区域上都相同，但是在一对邻居路由器之间必须相同。在 OSPF 协议的配置下添加 **area authentication** 命令来使类型 1 的认证方式有效。

参考图 8-48，在区域 0 和区域 25 上启用类型 1 的认证。路由器 Hurd 的配置参见示例 8-68。

示例 8-68 路由器 Hurd 配置了明文认证

```
interface Ethernet 0/0
 ip address 10.8.1.1 255.255.255.0
 ip ospf authentication-key santafe

interface Serial 0/0.1
 ip address 172.20.1.1 255.255.255.252
 ip ospf authentication-key taos

interface Serial 0/0.2
```

（待续）

```
 ip address 172.20.1.5 255.255.255.252
 ip ospf authentication-key abiquiu

router ospf 1
 network 10.8.0.0 0.0.255.255 area 0
 network 172.20.0.0 0.0.255.255 area 25
 area 25 range 172.16.0.0 255.240.0.0
 area 0 authentication
 area 25 authentication

ip route 172.16.0.0 255.240.0.0 null0
```

在这里，口令“santafe”用在路由器 Hurd 和 Pena 之间；口令“taos”用在路由器 Hurd 和 Gorman 之间；而口令“abiquiu”用在路由器 Hurd 和 Okeeffe 之间。

MD5 算法用在类型 2 的认证方式中。它用来为 OSPF 数据包内容和一个口令（或密钥）计算一个散列值（hash value）。这个散列值将和一个密钥 ID，以及一个不变小的序列号一起在数据包中传送。拥有相同口令的接收路由器将会计算它自己的散列值。如果传送的消息中什么内容都没改变，那么接收路由器的散列值应该和发送路由器在消息数据包中传送的散列值相匹配。密钥 ID 允许路由器指定多个口令，这样可以使口令的改变比较容易，并具有更好的安全性。在这个案例研究中包含了一个口令迁移的例子。序列号用来防止“重现攻击（replay attacks）”，以避免 OSPF 数据包被捕获、更改和重新传送给一台路由器。

在一个区域上配置类型 2 的认证方式，可以使用命令 **ip ospf message-digest-key md5** 来为和该区域相连的每一个接口分配一个口令和一个密钥 ID（Key ID），这里的口令最长为 16 个八位组字节，而密钥 ID 在 1～255 之间。和类型 1 的认证方式相同，所分配的口令不必在整个区域上都相同，但是在一对邻居路由器之间的密钥 ID 和口令都必须相同。在 OSPF 协议的配置下添加 **area authentication message-digest** 命令来使类型 2 的认证方式有效。

在路由器 Hurd 上启用类型 2 的认证方式，有关配置见示例 8-69。

示例 8-69 路由器 Hurd 配置了 MD5 认证

```
interface Ethernet 0/0
 ip address 10.8.1.1 255.255.255.0
 ip ospf message-digest-key 5 md5 santafe

interface Serial 0/0.1
 ip address 172.20.1.1 255.255.255.252
 ip ospf message-digest-key 10 md5 taos

interface Serial 0/0.2
 ip address 172.20.1.5 255.255.255.252
 ip ospf message-digest-key 15 md5 abiquiu

router ospf 1
 network 10.8.0.0 0.0.255.255 area 0
 network 172.20.0.0 0.0.255.255 area 25
 area 25 range 172.16.0.0 255.240.0.0
 area 0 authentication message-digest
 area 25 authentication message-digest

ip route 172.16.0.0 255.240.0.0 null0
```

密钥允许路由器在不需要使认证无效的情况下更改它的口令。例如，为了在路由器 Hurd 和 Okeeffe 之间更改口令，新的口令可以和一个不同的密钥一起配置。示例 8-70 显示了路由器 Hurd 的配置。

示例 8-70 路由器 Hurd 配置了一个新的 MD5 钥匙

```
interface Serial0/0.2
 ip address 172.20.1.5 255.255.255.252
 ip ospf message-digest-key 15 md5 abiquiu
 ip ospf message-digest-key 20 md5 steiglitz
```

路由器 Hurd 现在将会在 S0/0.2 接口为每一个 OSPF 数据包发送两个重复的拷贝，一个使用密钥 15 认证，另一个使用密钥 20 认证。当路由器 Hurd 开始从路由器 Okeeffe 那里收到使用密钥 20 认证的 OSPF 数据包时，它就会停止发送密钥为 15 的认证数据包。一旦新的密钥可以使用了，原来的密钥就可以使用命令 **no ip ospf message-digest-key 15 md5 abiquiu** 从这两台路由器上移掉。

一个在运行的网络中的口令从来不应该向这些例子中的口令那样可以预先得知。在所有使用认证的路由器的配置文件里增加命令 **service password-encryption** 是比较明智的选择。这样做可以使路由器对配置文件中任何显示的口令进行加密，因此可以避免口令不被别人通过简单地查看路由器配置的文本拷贝就可以得知的隐患。如果已经给路由器 Hurd 的配置口令加密，那么，它的配置参见示例 8-71。

示例 8-71 在路由器 Hurd 上配置了口令加密，用来加密配置文件中显示的口令

```
service password-encryption
!
interface Ethernet0/0
 ip address 10.8.1.1 255.255.255.0
 ip ospf message-digest-key 5 md5 7 001712008105A0D03
!
interface Serial0/0.1
 ip address 172.20.1.1 255.255.255.252
 ip ospf message-digest-key 10 md5 7 03105A0415
!
interface Serial0/0.2
 ip address 172.20.1.5 255.255.255.252
 ip ospf message-digest-key 20 md5 7 070E23455F1C1010
```

在本书讲述的其他协议的认证配置中，钥匙链用于配置口令，或称为钥匙串。在编写本书的时候，OSPF 不支持钥匙链的配置。

8.2.11 案例研究：虚链路

如图 8-50 所示，显示了一个骨干区域设计的比较糟糕的网络。如果路由器 Hokusai 和 Hiroshige 之间的链路失效了，那么这个网络的骨干区域将被分割成两部分。结果是，路由器 Sesshiu 和 Okyo 不能相互通信。即使这两台路由器是分离区域的 ABR，区域间的通信量也将会在这些区域之间被阻塞。

在这个实例中，最有效的解决办法是在路由器 Sesshiu 和 Okyo 之间为骨干区域增加另外一条链路。在这个骨干区域得到改进之前，作为一种过渡方案，可以在路由器 Hokusai 和 Hiroshige 之间建立一条穿过区域 100 的虚链路。

虚链路总是建立在 ABR 路由器之间的，至少它们之中有一台 ABR 路由器是必须和区域 0 相连的。[1] 在每一台 ABR 路由器的 OSPF 配置中，通过添加 **area virtual-link** 命令来配置一

[1] 当一条虚链路穿过一个非骨干区域连接某个区域到骨干区域时，其中一台 ABR 路由器将位于这两个非骨干区域之间。

条虚链路，并指定了这条虚链路要穿过的区域以及这条链路远端的 ABR 的路由器 ID。在路由器 Hokusai 和 Hiroshige 之间建立一条虚链路的配置参见示例 8-72 和示例 8-73。

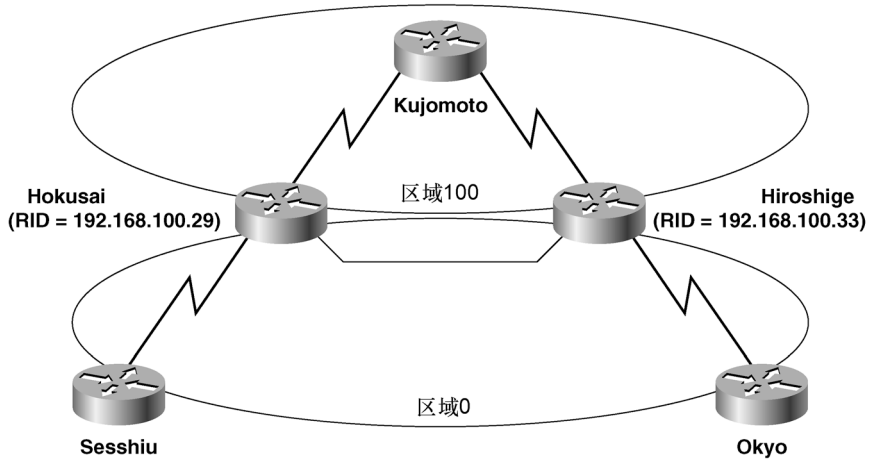


图 8-50 路由器 Hokusai 和 Hiroshige 之间的链路失效了将会使它们的骨干区域变成分段区域

示例 8-72 路由器 Hokusai 的虚链路配置

```
router ospf 10
 network 192.168.100.1 0.0.0.0 area 0
 network 192.168.100.29 0.0.0.0 area 0
 network 192.168.100.21 0.0.0.0 area 100
 area 100 virtual-link 192.168.100.33
```

示例 8-73 路由器 Hiroshige 的虚链路配置

```
router ospf 10
 network 192.168.100.2 0.0.0.0 area 0
 network 192.168.100.33 0.0.0.0 area 0
 network 192.168.100.25 0.0.0.0 area 100
 area 100 virtual-link 192.168.100.29
```

完成以上的配置后，在正常情况下，路由器 Sesshiu 和 Okyo 之间的数据包可以通过路由器 Hokusai 和 Hiroshige 之间的骨干区域上的链路进行转发。但是，如果那条链路失效，将会利用虚链路进行数据包的转发。在示例 8-74 中，虽然每一台路由器都把这条链路看作是一条无编号的点到点网络，但实际上数据包的转发是通过路由器 Kujomoto 的。[1]

示例 8-74 使用命令 show ip ospf virtual-link 可以查看一条虚链路的状态

```
Hokusai#show ip ospf virtual-link
Virtual Link OSPF_VL1 to router 192.168.100.33 is up
   Run as demand circuit
   DoNotAge LSA not allowed (Number of DCbitless LSA is 2).
   Transit area 100, via interface Serial0, Cost of using 128
   Transmit Delay is 1 sec, State POINT_TO_POINT,
   Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
      Hello due in 00:00:00
      Adjacency State FULL (Hello suppressed)
Hokusai#
```

[1] 根据你使用的 IOS 版本的不同，命令 **show ip ospf virtual-link** 的输出可能有点不同。

8.2.12 案例研究：运行在 NBMA 网络上的 OSPF

在一些非广播多路访问（multi-access）网络上，例如 X.25、帧中继和 ATM 等，运行 OSPF 协议会产生一个问题。“多路访问”意味着一个 NBMA 的网络“云”是多台设备共同相连的单个网络，和以太网或者令牌环网络一样（如图 8-51 所示）。但是它又和以太网、令牌环网等广播型网络不同，它是非广播的。“非广播”意味着发送到这个网络上的数据包不一定会被和该网络相连的其他所有的路由器看到。这样，由于 NBMA 网络是多路访问的，OSPF 协议将需要选取一台 DR 路由器和 BDR 路由器。但是由于 NBMA 网络又是非广播的，它不能保证所有相连的路由器都能收到其他所有路由器发送的 Hello 数据包。因此，所有的路由器不一定能够自动地了解它的所有邻居，因而不一定能够正确地进行 DR 的选取。

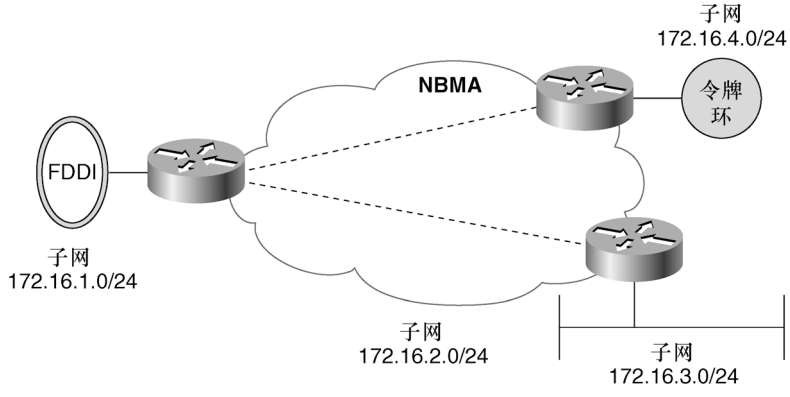


图 8-51 路由选择协议会把 NBMA 网络看作是有多台设备相连的一个子网。但是当一个 NBMA 网络是部分网状连接时，正如这里所示，不是所有相连的路由器都和其他所有路由器直接连接

本小节内容将阐述几个解决 NBMA 网络问题的方案。具体方案的选择依赖于实现该解决方案的网络的特征。

最早的解决方案出现在 Cisco IOS 10.0 版之前的有关版本中。它是使用 **neighbor** 命令以手工方式指定每一台路由器的邻居并创建 DR 的。如图 8-52 所显示了一个与 4 台路由器相连的帧中继网络。

在图 8-52 中，由于 PVC 电路的配置是部分网状连接的星型（hub-and-spoke）结构，因此，路由器 Rembrandt 必须成为一台 DR 路由器。作为中心（hub），它是惟一的和其他所有的路由器直接相连的路由器。这 4 台路由器的配置参见示例 8-75～示例 8-78。

示例 8-75 路由器 Rembrandt 的配置

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.1 255.255.255.0
 frame-relay map ip 172.16.2.2 100
 frame-realy map ip 172.16.2.3 300
 frame-relay map ip 172.16.2.4 500
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 neighbor 172.16.2.2
 neighbor 172.16.2.3
 neighbor 172.16.2.4
```

示例 8-76 路由器 Hals 的配置指定了一个邻居的优先级

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.2 255.255.255.0
 frame-relay map ip 172.16.2.1 600
 frame-relay map ip 172.16.2.3 600
 frame-relay map ip 172.16.2.4 600
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 neighbor 172.16.2.1 priority 10
```

示例 8-77 路由器 Vandyck 的配置指定了一个邻居的优先级

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.3 255.255.255.0
 frame-relay map ip 172.16.2.1 400
 frame-relay map ip 172.16.2.2 400
 frame-relay map ip 172.16.2.4 400
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 neighbor 172.16.2.1 priority 10
```

示例 8-78 路由器 Brueghel 的配置指定了一个邻居的优先级

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.4 255.255.255.0
 frame-relay map ip 172.16.2.1 200
 frame-relay map ip 172.16.2.2 200
 frame-relay map ip 172.16.2.3 200
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 neighbor 172.16.2.1 priority 10
```

在路由器 Rembrandt 上，使用 **neighbor** 命令配置了它的 3 个邻居的接口 IP 地址。缺省的优先级是 0，在路由器 Rembrandt 上不改变这个缺省值，没有邻居有资格成为一台 DR 或者 BDR。

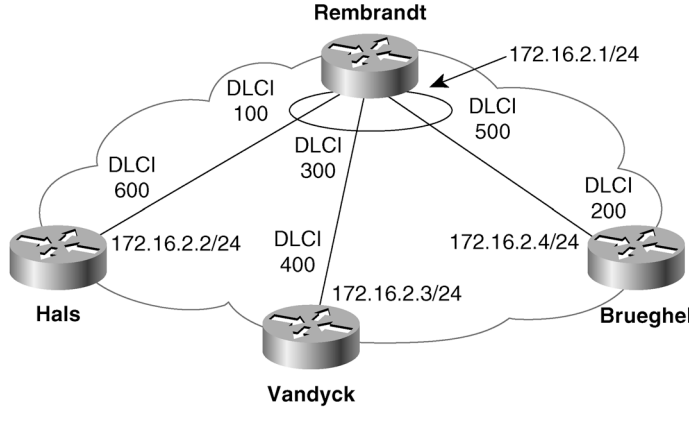


图 8-52 在这个 NBMA 网络中配置 OSPF 存在的几种选择

另外 3 台路由器仅把路由器 Rembrandt 作为它们各自的邻居来配置，并把优先级设置为 10，这意味着路由器 Rembrandt 将成为 DR 路由器。通过使路由器 Rembrandt 成为 DR 路由器，这些 PVC 电路可以精确地仿效假设这 4 台路由器连接到一个广播型多路访问网络上时所形成的邻接关系。现在，OSPF 数据包将以单播的方式转发到所配置的邻居地址上。

再次重申，**neighbor** 命令只在老的 IOS 版本（10.0 版本以前）上才是必要的。而新的解决方案中，使用 **ip ospf network** 命令可以改变缺省的 OSPF 网络类型。这条命令的一个选项是改变网络类型为广播型，这可以在每一个帧中继接口上使用 **ip ospf network broadcast** 来实现。这个网络类型的更改将会让 OSPF 把这个 NBMA 视为一个广播型网络；在这种方案里，4 台路由器的配置见示例 8-79～示例 8-82。

示例 8-79　路由器 Rembrandt 的帧中继接口配置为一个 OSPF 广播型网络

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.1 255.255.255.0
 ip ospf network broadcast
 ip ospf priority 10
 frame-relay map ip 172.16.2.2 100 broadcast
 frame-realy map ip 172.16.2.3 300 broadcast
 frame-relay map ip 172.16.2.4 500 broadcast
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

示例 8-80　路由器 Hals 的帧中继接口配置为一个 OSPF 广播型网络

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.2 255.255.255.0
 ip ospf network broadcast
 ip ospf priority 0
 frame-relay map ip 172.16.2.1 600 broadcast
 frame-relay map ip 172.16.2.3 600 broadcast
 frame-relay map ip 172.16.2.4 600 broadcast
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

示例 8-81　路由器 Vandyck 的帧中继接口配置为一个 OSPF 广播型网络

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.3 255.255.255.0
 ip ospf network broadcast
 ip ospf priority 0
 frame-relay map ip 172.16.2.1 400 broadcast
 frame-relay map ip 172.16.2.2 400 broadcast
 frame-relay map ip 172.16.2.4 400 broadcast
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

示例 8-82　路由器 Brueghel 的帧中继接口配置为一个 OSPF 广播型网络

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.4 255.255.255.0
```

（待续）

```
 ip ospf network broadcast
 ip ospf priority 0
 frame-relay map ip 172.16.2.1 200 broadcast
 frame-relay map ip 172.16.2.2 200 broadcast
 frame-relay map ip 172.16.2.3 200 broadcast
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

这里注意，在这个例子中，路由器 Rembrandt 的接口的优先级设置为 10，而其他接口的优先级设置为 0。这将可以再次确保路由器 Rembrandt 能够成为一台 DR 路由器。注意，静态的帧中继映射命令也设置为转发到广播和组播地址上了。

影响 DR 选取的另一种方案是实现一个全网状连接的拓扑结构，即每一台路由器都有一个 PVC 电路和其他所有的路由器相连。从路由器的角度来看，这种方案实际上是所有 NBMA 网络实现中最有效的方案。但是这种方案的一个显而易见的缺点就是开销相当昂贵。假设有 n 台路由器，那么为了创建一个全网状连接的拓扑结构将必须要有 $n(n\text{-}1)/2$ 条 PVC 电路才能实现。例如，图 8-52 中的 4 台路由器要创建一个全网状连接的拓扑应该需要 6 条 PVC 电路，而 16 台路由器则需要 120 条 PVC 电路。

另外一种方法，就是将网络类型改为点到多点网络，这样就可以避免 DR/BDR 选取的处理。点到多点网络把 PVC 当作一个点到点链路的集合，因此就没有 DR/BDR 的选取发生。在多厂商的网络环境中，点到多点类型可能是广播型网络之外惟一的一种选择。

在示例 8-83～示例 8-86 的配置中，把和每一个接口相关的 OSPF 网络类型改变成了点到多点类型。

示例 8-83 路由器 Rembrandt 的帧中继接口配置为一个 OSPF 点到多点型网络

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.1 255.255.255.0
 ip ospf network point-to-multipoint
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

示例 8-84 路由器 Hals 的帧中继接口配置为一个 OSPF 点到多点型网络

```
interface Serial0
encapsulation frame-relay
ip address 172.16.2.2 255.255.255.0
ip ospf network point-to-multipoint
!
router ospf 1
network 172.16.0.0 0.0.255.255 area 0
```

示例 8-85 路由器 Vandyck 的帧中继接口配置为一个 OSPF 点到多点型网络

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.3 255.255.255.0
 ip ospf network point-to-multipoint
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

示例 8-86 路由器 Brueghel 的帧中继接口配置为一个 OSPF 点到多点型网络

```
interface Serial0
 encapsulation frame-relay
 ip address 172.16.2.4 255.255.255.0
 ip ospf network point-to-multipoint
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

在这些配置中，利用帧中继网络逆向 ARP 解析功能动态地把网络层的地址映射到 DLCI 上，这种方法替代了上面例子中使用的静态映射命令。当然，如果想用，静态映射依然可以使用。

OSPF 点到多点的网络类型把下层的网络看作一组点到点链路的集合，而不是一个多路访问网络，并且 OSPF 数据包以组播的方式发送到它的邻居。这种解决方案在那些动态连接的网络上会产生问题，像帧中继 SVC 或者 ATM SVC 等。自 IOS 11.3AA 开始，这个问题可以通过同时将一个网络声明为点到多点（point-to-multipoint）和非广播（non-broadcast）的方式而得到解决，配置参见示例 8-87～示例 8-90。

示例 8-87 路由器 Rembrandt 的帧中继接口配置为一个 OSPF 点到多点非广播网络

```
interface Serial0
 ip address 172.16.2.1 255.255.255.0
 encapsulation frame-relay
 ip ospf network point-to-multipoint non-broadcast
 map-group Leiden
 frame-relay lmi-type q933a
 frame-relay svc
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 neighbor 172.16.2.2 cost 30
 neighbor 172.16.2.3 cost 20
 neighbor 172.16.2.4 cost 50
```

示例 8-88 路由器 Hals 的帧中继接口配置为一个 OSPF 点到多点非广播网络

```
interface Serial0
 ip address 172.16.2.2 255.255.255.0
 encapsulation frame-relay
 ip ospf network point-to-multipoint non-broadcast
 map-group Haarlem
 frame-relay lmi-type q933a
 frame-relay svc
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 neighbor 172.16.2.1 priority 10
```

示例 8-89 路由器 Vandyck 的帧中继接口配置为一个 OSPF 点到多点非广播网络

```
interface Serial0
 ip address 172.16.2.3 255.255.255.0
 encapsulation frame-relay
 ip ospf network point-to-multipoint non-broadcast
 map-group Antwerp
 frame-relay lmi-type q933a
 frame-relay svc
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 neighbor 172.16.2.1 priority 10
```

示例 8-90 路由器 Brueghel 的帧中继接口配置为一个 OSPF 点到多点非广播网络

```
interface Serial0
 ip address 172.16.2.4 255.255.255.0
 encapsulation frame-relay
 ip ospf network point-to-multipoint non-broadcast
 map-group Brussels
 frame-relay lmi-type q933a
 frame-relay svc
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 neighbor 172.16.2.1 priority 10
```

由于网络是非广播的，邻居将不会被自动地发现，因此必须要手工地配置。另一个在 IOS 11.3AA 中引入的特性可以从路由器 Rembrandt 的配置中看出来：即可以利用 **neighbor** 命令基于每一个 VC 来指定它们的代价。

最后一种解决方案就是，使用它们各自本身的子网把每一个 PVC 连接作为一个单独的点到点网络，如图 8-53 所示。这种解决方案可以通过子接口来完成，配置见示例 8-91～示例 8-94。

示例 8-91 路由器 Rembrandt 配置了点到点子接口

```
interface Serial0
 no ip address
 encapsulation frame-relay
 interface Serial0.100 point-to-point
  description ------------------------- to Hals
  ip address 172.16.2.1 255.255.255.252
  frame-relay interface-dlci 100
 interface Serial0.300 point-to-point
  description ------------------------- to Vandyck
  ip address 172.16.2.5 255.255.255.252
  frame-relay interface-dlci 300
 interface Serial0.500 point-to-point
  description ------------------------- to Brueghels
  ip address 172.16.2.9 255.255.255.252
  frame-relay interface-dlci 500
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

示例 8-92 路由器 Hals 配置了点到点子接口

```
interface Serial0
 no ip address
 encapsulation frame-relay
 interface Serial0.600
  description ------------------------- to Rembrandt
  ip address 172.16.2.2 255.255.255.252
  frame-relay interface-dlci 600
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

示例 8-93 路由器 Vandyck 配置了点到点子接口

```
interface Serial0
 no ip address
 encapsulation frame-relay
```

（待续）

```
 interface Serial0.400
  description --------------------------- to Rembrandt
  ip address 172.16.2.6 255.255.255.252
  frame-relay interface-dlci 400
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

示例 8-94 路由器 Brueghel 配置了点到点子接口

```
interface Serial0
 no ip address
 encapsulation frame-relay
 interface Serial0.200
  description --------------------------- to Rembrandt
  ip address 172.16.2.10 255.255.255.252
  frame-relay interface-dlci 200
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
```

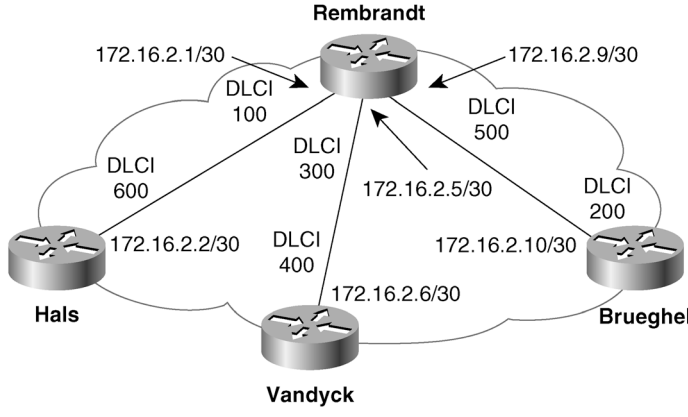


图 8-53 点到点网络的子接口允许每一条 PVC 配置成一个单独的子网，并在 NBMA 网络上排除了 DR/BDR 的选取问题

对于所有运行在 NBMA 网络上 OSPF 配置来说，这种配置是最容易管理的。这种配置代码的一些优点是显而易见的，例如可以使用一个接口号对应于帧中继的 DLCI，并且包括一个解释行。然而，还有一个主要的优点就是可以在路由器之间建立简单的一对一的对应关系。

使用子接口有一个不太经常碰到的缺点是，每一个 PVC 电路都必须拥有自己的子网地址。在大多数情况下，这个需求不应该会产生问题，因此 OSPF 协议是支持 VLSM 的。正如这里的例子所显示的，可以从一个子网地址中创建更小的子网来分配给这个网络“云”是一件容易的事情。而且，由于 PVC 电路现在是点到点的链路，也可以使用无 IP 地址编号的方法作为子网地址需求的另一种选择。另一个重要的好处是路由器故障的检测。在点到点子接口上，点到点网络两端的路由器能够检测失效的虚电路（如果帧中继云提供足够的信号），如果存在另一条路由，路由选择协议也能够收敛到该路由上。

更需谨慎的是子接口会占用更多的内存。在一些内存有限的小型路由器上可能会给路由器带来较大的负担。

8.2.13 案例研究：运行在按需电路上的 OSPF

配置运行在按需电路上的 OSPF 是很简单的，这可以在与按需电路相连的接口上，通过增加 **ip ospf demand-circuit** 命令来实现。而且只需要在点到点电路的一端，或者点到多点电路的多点端宣告为按需电路就可以了。在一般情况下，运行在按需电路上的 OSPF 不应该在一个广播型介质上实现。这是因为，这样的网络不能抑制 Hello 数据包的发送，从而使链路一直保持是活动（up）的。

如果图 8-52 中的虚电路是帧中继 SVC 电路，路由器 Rembrandt 的配置见示例 8-95。

示例 8-95 路由器 Rembrandt 配置为按需电路的 OSPF

```
interface Serial0/0
 ip address 172.16.2.1 255.255.255.0
 encapsulation frame-relay
 ip ospf network point-to-multipoint non-broadcast
 ip ospf demand-circuit
 map-group Leiden
 frame-relay lmi-type q933a
 frame-relay svc
!
router ospf 1
 network 172.16.0.0 0.0.255.255 area 0
 neighbor 172.16.2.2 cost 30
 neighbor 172.16.2.3 cost 20
 neighbor 172.16.2.4 cost 50
```

在按需电路上实现 OSPF，需要记住以下几点：

- 只有在一个区域的链路状态数据库中的所有 LSA 都设置了 DC-bit 位以后，设置 DoNotAge 位的 LSA 才被允许进入该区域。这种设置方法可以确保该区域的所有路由器都具有识别 DoNotAge 位的能力。
- 实现按需电路上的 OSPF 的区域内的所有路由器都必须具有支持这种配置方式的能力。
- 如果运行在按需电路上的 OSPF 是在一个非末梢区域里实现的，那么在所有的非末梢区域内的路由器都必须支持这种方式。原因是 DC-bit 位的设置是在类型 5 的 LSA 中实现的，而这种类型的 LSA 是泛洪扩散到所有的非末梢区域中的。
- 应该尽量限制在末梢区域、完全末梢区域或 NSSA 区域上实现按需电路的 OSPF。这种实现方式可以取消要求 OSPF 域内的所有路由器支持按需电路上的 OSPF 的需要。这种方式也将因其他区域拓扑改变而引起改变的 LSA 的数量减到最小，并可以防止过多地使按需电路处于活动状态。
- 如果配置了按需电路上的 OSPF 并且配置了一条虚链路要穿过这条按需电路，那么这条虚链路也将被看作是一条按需电路。另外，这条虚链路的通信流量将会保持按需电路为活动（up）。
- 每隔 30min，OSPF 协议就会重新刷新一次它的 LSA，以防止这些 LSA 驻留在链路状态数据库时变得无效。由于设置 DoNotAge 位的 LSA 在穿过按需电路的时候不会重新刷新，因此也就丧失了 OSPF 的这个稳定特性。
- 重新刷新过程可以在一条按需电路两端外的其他所有接口上发生，但是 LSA 在通过这条按需电路时不能进行重新刷新。结果是，这条链路两端的同一个 LSA 的各自序

列号可能是不同的。网络管理工作站可以使用某些 MIB 变量[1]去验证数据库的同步；如果序列号在数据库中不匹配，那么将会错误地报告一个错误。

8.3 OSPF 故障诊断

OSPF 协议的故障排除有时是令人恐怖的，这在一个大型网络上显得尤其明显。但是，OSPF 产生的路由选择问题和其他任何路由选择协议的路由选择问题没有什么不同，故障可能是下列某种原因之一引起的：

- 路由信息丢失；
- 错误的或不精确的路由信息。

对路由器的检查仍然是获取故障排除信息的主要来源。使用命令 **show ip ospf database** 查看不同的 LSA 也会得到重要的信息。例如，如果一条链路是不稳定的，那么通告它的 LSA 将会频繁变化。这种情况反映在它的序列号会比其他 LSA 的序列号明显的偏高。网络不稳定的另外一个迹象是 LSA 的老化时间从来不会变得很大。

请记住，一个区域内所有路由器的链路状态数据库都是相同的。因此，除非你怀疑某些路由器的链路状态数据库本身变得有问题，否则就可以通过检查某一台路由器的链路状态数据库来检查整个区域的链路状态数据库。另外一个好的经验是，为每一个区域的链路状态数据库做一个拷贝（软拷贝或硬拷贝）。

当检查单独一台路由器的配置时，需要考虑以下几点：

- 所有接口配置的地址和掩码是否正确？
- **network area** 语句使用的反向掩码是否正确？是否匹配正确的接口？
- **network area** 语句是否把所有的接口都指定到正确的区域中了？
- **network area** 语句的使用顺序正确吗？

当检查邻接关系时（或者缺少邻接关系时），请考虑下面这些问题：

- 从这两台邻居路由器中有 Hello 数据包正在发送吗？
- 这两个邻居路由器之间的计时器设置相同吗？
- 这两个邻居路由器之间的可选性能字段设置相同吗？
- 接口是配置在同一个子网上的吗（也就是说，它们的地址/掩码对是否属于同一个子网）？
- 邻居路由器的接口是否是同一种网络类型？
- 一台路由器是否正在试图和它的邻居路由器的辅助地址形成一种邻接关系？
- 如果使用了认证，那么在邻居路由器之间的认证类型是否相同？口令和密钥（在 MD5 的实例中）是否相同？该区域内的所有路由器是否都启用了认证？
- 在所有的访问列表中，是否有正在阻塞 OSPF 的访问列表？
- 如果邻居关系穿过一条虚链路，那么这条链路是否配置在一个末梢区域内？

如果怀疑某个邻居或者某个邻接关系变得不稳定了，那么可以通过 **debug ip ospf adj** 命令来监控这些邻接关系。然而，这条命令经常会产生比所需要的信息多得多的内容，参

[1] 具体来说，就是 ospfExternLSACksumSum 和 ospfAreaLSACksumSum。这些是单独的 LSA 校验和字段的总和。由于校验和的计算包括序列号，并且序列号可能不同，因此，校验和也可能不同。

见示例 8-96。邻居的状态变化被非常详细的记录下来。如果记录了普通的 Hello 数据包处理。如果在一个扩展期间执行了监控，这些正常的信息将会溢出路由器的内部缓冲区。从 IOS11.2 版本开始，路由器可以在它的 OSPF 配置里增加命令 **log-adjacency-changes** [**detail**] 来监控邻接关系。这条命令将会记录邻接关系变化的一个简单日志，参见示例 8-97 和示例 8-98。

示例 8-96　使用命令 debug ip ospf adj 输出的调试信息显示了，当一台邻居路由器的以太网接口暂时断开随后又重新连接上的结果

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0/0, changed state to down
OSPF: Interface Ethernet0/0 going Down
OSPF: 172.16.2.2 address 10.8.1.1 on Ethernet0/0 is dead, state DOWN
OSPF: Neighbor change Event on interface Ethernet0/0
OSPF: DR/BDR election on Ethernet0/0
OSPF: Elect BDR 0.0.0.0
OSPF: Elect DR 172.16.2.3
OSPF: Elect BDR 0.0.0.0
OSPF: Elect DR 172.16.2.3
        DR: 172.16.2.3 (Id)   BDR: none
OSPF: 172.16.2.3 address 10.8.1.2 on Ethernet0/0 is dead, state DOWN
OSPF: Neighbor change Event on interface Ethernet0/0
OSPF: DR/BDR election on Ethernet0/0
OSPF: Elect BDR 0.0.0.0
OSPF: Elect DR 0.0.0.0
        DR: none    BDR: none
OSPF: Remember old DR 172.16.2.3 (id)
OSPF: Build router LSA for area 0, router ID 172.16.2.2, seq 0x80000035
OSPF: Build router LSA for area 25, router ID 172.16.2.2,
seq 0x80000005
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0/0, changed state to up
OSPF: Interface Ethernet0/0 going Up
OSPF: Send with youngest Key 5
OSPF: Build router LSA for area 0, router ID 172.16.2.2, seq 0x80000036
OSPF: Build router LSA for area 25, router ID 172.16.2.2, seq 0x80000006
OSPF: Send with youngest Key 5
OSPF: Send with youngest Key 5
OSPF: Send with youngest Key 5
OSPF: Rcv DBD from 172.16.2.3 on Ethernet0/0 seq 0x728 opt 0x52 flag 0x7 len 32  mtu
1500 state INIT
OSPF: 2 Way Communication to 172.16.2.3 on Ethernet0/0, st ate 2WAY
OSPF: Nbr state is 2WAY
OSPF: Rcv DBD from 172.16.2.3 on Ethernet0/0 seq 0x728 opt 0x52 flag 0x7 len 32  mtu
1500 state 2WAY
OSPF: Nbr state is 2WAY
OSPF: end of Wait on interface Ethernet0/0
OSPF: DR/BDR election on Ethernet0/0
OSPF: Elect BDR 172.16.2.2
OSPF: Elect DR 172.16.2.3
OSPF: Elect BDR 172.16.2.2
OSPF: Elect DR 172.16.2.3
        DR: 172.16.2.3 (Id)   BDR: 172.16.2.2 (Id)
OSPF: Send DBD to 172.16.2.3 on Ethernet0/0 seq 0x1B85 opt 0x52 flag 0x7 len 32
OSPF: Send with youngest Key 5
OSPF: Send with youngest Key 5
OSPF: Rcv DBD from 172.16.2.3 on Ethernet0/0 seq 0x728 opt 0x52 flag 0x7 len 32  mtu
1500 state EXSTART
OSPF: NBR Negotiation Done. We are the SLAVE
OSPF: Send DBD to 172.16.2.3 on Ethernet0/0 seq 0x728 opt 0x52 flag 0x2 len 292
OSPF: Send with youngest Key 5
```

（待续）

```
OSPF: Rcv DBD from 172.16.2.3 on Ethernet0/0 seq 0x729 opt 0x52 flag 0x3 len 272
mtu 1500 state EXCHANGE
OSPF: Send DBD to 172.16.2.3 on Ethernet0/0 seq 0x729 opt 0x52 flag 0x0 len 32
OSPF: Send with youngest Key 5
OSPF: Send with youngest Key 5
OSPF: Database request to 172.16.2.3
OSPF: sent LS REQ packet to 10.8.1.2, length 12
OSPF: Send with youngest Key 5
OSPF: Rcv DBD from 172.16.2.3 on Ethernet0/0 seq 0x72A opt 0x52 flag 0x1 len 32  mtu
1500 state EXCHANGE
OSPF: Exchange Done with 172.16.2.3 on Ethernet0/0
OSPF: Send DBD to 172.16.2.3 on Ethernet0/0 seq 0x72A opt 0x52 flag 0x0 len 32
OSPF: Send with youngest Key 5
OSPF: Synchronized with 172.16.2.3 on Ethernet0/0, state FULL
OSPF: Build router LSA for area 0, router ID 172.16.2.2, seq 0x80000037
```

示例 8-97　这些日志信息的记录结果来自命令 log-adjacency-changes，显示了示例 8-96 中相同的邻居失效的信息，但是没有上面的调试信息那么详细

```
Hurd#show logging
Syslog logging: enabled (0 messages dropped, 1 messages rate-limited, 0 flushes,

 0 overruns, xml disabled)
    Console logging: level debugging, 248 messages logged, xml disabled
    Monitor logging: level debugging, 0 messages logged, xml disabled
    Buffer logging: level debugging, 5 messages logged, xml disabled
    Logging Exception size (4096 bytes)
    Count and timestamp logging messages: disabled
    Trap logging: level informational, 99 message lines logged

Log Buffer (4096 bytes):

%OSPF-5-ADJCHG: Process 1, Nbr 172.16.2.3 on Ethernet0/0 from FULL to DOWN,
 Neighbor Down: Interface down or detached
%OSPF-5-ADJCHG: Process 1, Nbr 172.16.2.3 on Ethernet0/0 from LOADING to FULL,
 Loading Done
```

示例 8-98　这些日志信息的记录结果来自 OSPF 配置命令 log-adjacency- changes，显示了示例 8-96 中相同的邻居失效的信息，但比示例 8-97 中的记录更加详细

```
Hurd#show logging
Syslog logging: enabled (0 messages dropped, 1 messages rate-limited, 0 flushes,

 0 overruns, xml disabled)
    Console logging: level debugging, 248 messages logged, xml disabled
    Monitor logging: level debugging, 0 messages logged, xml disabled
    Buffer logging: level debugging, 5 messages logged, xml disabled
    Logging Exception size (4096 bytes)
    Count and timestamp logging messages: disabled
    Trap logging: level informational, 99 message lines logged

Log Buffer (4096 bytes):

%OSPF-5-ADJCHG: Process 1, Nbr 172.16.2.3 on Ethernet0/0 from FULL to DOWN,
 Neighbor Down: Interface down or detached
%OSPF-5-ADJCHG: Process 1, Nbr 172.16.2.3 on Ethernet0/0 from DOWN to INIT,
 Received Hello
%OSPF-5-ADJCHG: Process 1, Nbr 172.16.2.3 on Ethernet0/0 from INIT to 2WAY,
 2-Way Received
%OSPF-5-ADJCHG: Process 1, Nbr 172.16.2.3 on Ethernet0/0 from 2WAY to EXSTART,
 AdjOK?
```

（待续）

```
%OSPF-5-ADJCHG: Process 1, Nbr 172.16.2.3 on Ethernet0/0 from EXSTART to
 EXCHANGE, Negotiation Done
%OSPF-5-ADJCHG: Process 1, Nbr 172.16.2.3 on Ethernet0/0 from EXCHANGE to
 LOADING, Exchange Done
%OSPF-5-ADJCHG: Process 1, Nbr 172.16.2.3 on Ethernet0/0 from LOADING to FULL,
 Loading Done
```

如果怀疑一个链路状态数据库出现问题，或者这两个数据库不同步，那么可以使用命令 **show ip ospf database database-summary** 来观察每一台路由器中数据库的 LSA 数量。对于给定的一个区域，在所有的路由器上，每一种 LSA 类型的数量都应该是相同的。下一条命令 **show ip ospf database** 将显示一台路由器数据库的每一条 LSA 的校验和。在一个给定的区域内，在每一台路由器的数据库中的每一条 LSA 的校验和都应该相同。除非是在一个非常小的数据库里，否则校验这个情况的正确与否将非常乏味和痛苦。不过幸运的是，还有 MIB[1]，MIB 可以在一个 SNMP 网络管理平台上报告一个数据库校验和的总和。如果在一个区域的所有数据库都同步，那么每一个数据库中的总和都应该是相同的。

当检查一个区域层面上的问题时，请记住以下几个问题：

- ABR 路由器是否配置正确？
- 对于相同区域的类型是否所有的路由器都配置了？例如，如果一个区域是末梢区域，那么所有的路由器都必须使用 **area stub** 命令。
- 如果配置了地址汇总，那么配置的正确吗？

如果是路由器的性能出现了问题，那么可以在路由器上检查它们的 CPU 和内存的使用情况。如果内存的使用率在 70%以上，那么可能是链路状态数据库太大了；如果 CPU 的利用率一直保持在 60%以上，那么网络的拓扑可能存在不稳定的情况。如果内存或 CPU 的利用率超过了 50%的警戒线，网络管理员就应该开始分析网络性能加重的原因，并基于得出的分析结果，来制定改善网络的升级计划。

末梢区域和地址汇总能够帮助减小链路状态数据库的大小并能容忍网络的一些不稳定性。最能加重一台 OSPF 路由器负担的是对 LSA 的处理，而不是 SPF 算法的计算。在个别情况下，类型 1 和类型 2 的 LSA 对处理器的影响比汇总 LSA 更大。但是，类型 1 和类型 2 的 LSA 可以编成组发送，而汇总 LSA 却只能在单个数据包中发送。结果是，汇总 LSA 实际上对处理器的影响更大。

下面的案例研究演示了进行 OPSF 协议的故障排除时，使用最频繁的技巧和工具。

8.3.1 案例研究：孤立的区域

区域内的数据包可以在图 8-54 中的区域 1 内进行路由转发，但是所有的区域间通信的尝试都失败了。这种情况下，首先应该马上怀疑是区域 1 的 ABR 路由器出现了故障。而且由于内部路由器没有关于 ABR 路由器的入口，这使得更加坚信可能是 ABR 路由器出现了问题（参见示例 8-99）。

[1] 也就是 ospfExternLsaCksumSum 和 ospfAreaLsaCksumSum。

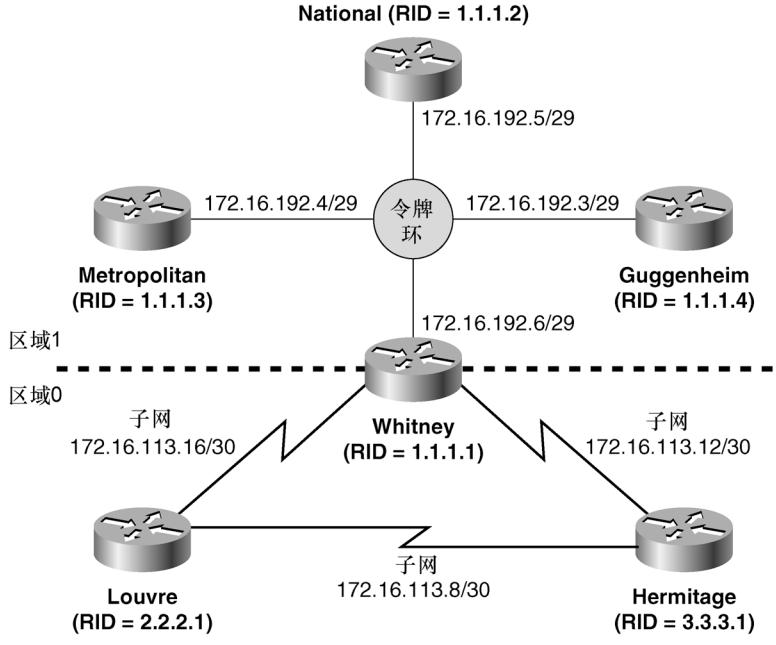


图 8-54 区域 1 内的系统和路由器通信正常，但是没有通信流量可以通过或者来自区域 0

示例 8-99 使用命令 show ip ospf border-router 可以检查内部路由器的内部路由表信息。没有显示关于 ABR 路由器的路由器条目

```
National#show ip ospf border-routers

OSPF Process 8 internal Routing Table

Codes: i - Intra-area route, I - Inter-area route

National#
```

下一步是验证连接到 ABR 路由器的物理链路是否正常以及 OSPF 协议是否在正常工作。参见示例 8-100 所示，在上述的那一台内部路由器的邻居表中，显示了关于 ABR 路由器的邻居状态都是完全邻接的，这表明邻接关系是存在的。事实上，这台 ABR 路由器是这里令牌环网络的 DR 路由器。邻接关系的存在证实链路是正常的，而且也可以交换含有正确参数的 OSPF Hello 数据包。

示例 8-100 路由器 National 的邻居表表明与 ABR 路由器（1.1.1.1）的邻接关系是完全邻接的

```
National#show ip ospf neighbor
Neighbor ID  Pri  State      Dead Time   Address         Interface
1.1.1.1        1  FULL/DR    00:00:33    172.16.192.6    TokenRing0
1.1.1.3        1  FULL/BDR   00:00:34    172.16.192.4    TokenRing0
1.1.1.4        1  FULL/ -    00:00:30    172.16.192.3    TokenRing0
National#
```

在路由器 National 的链路状态数据库和它的路由表中可以发现一些其他的和这个故

障有关的迹象。在示例 8-101 中，路由器 National 的数据库中只包含了路由器 LSA（类型 1）和网络 LSA（类型 2），但是没有记录通告区域外部目的地址的网络汇总 LSA（类型 3）。同时，出现了由路由器 Whitney（1.1.1.1）始发的 LSA。这个信息再次表明路由器 Whitney 与路由器 National 是有邻接关系的，但是却没有信息从区域 0 通过到达区域 1。

示例 8-101 路由器 National 的链路状态数据库显示，和路由器 Whitney 之间是有邻接的，但是却没有通告区域间的目的地址

```
National#show ip ospf database

        OSPF Router with ID (1.1.1.2) (Process ID 8)

                    Router Link States (Area 1)

Link ID         ADV Router  Age   Seq#         Checksum  Link count
172.16.192.6    1.1.1.1     132   0x80000034   0xAC4D    3
172.16.219.120  1.1.1.2 1   458   0x8000002B   0x6B46    2

                    Net Link States (Area 1)

Link ID         ADV Router  Age   Seq#         Checksum
172.16.192.6    1.1.1.1     132   0x8000002E   0x2078
National#
```

参见示例 8-102，在路由器 National 的路由表中，区域 1 外部惟一的目的地址是与路由器 Whitney 相连的串行链路地址。然而，在这里还显示了另外一条线索：这些路由条目被标记成区域内路由（O）。依照图 9-54，这些路由应该是在区域 0 内的，因此，它们应该是被标记成区域间路由（O IA）。现在，问题显然出在 ABR 路由器的区域 0 的一端。

示例 8-102 路由器 Whitney 正在通告它的串行接口的子网，但是它们正在被作为区域内的目的地址通告

```
National#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

     172.16.0.0/16 is variably subnetted, 4 subnets, 3 masks
C       172.16.219.112/28 is directly connected, Serial0
C       172.16.192.0/29 is directly connected, TokenRing0
O       172.16.113.12/30 [110/70] via 172.16.192.6, 09:32:15, TokenRing0

O       172.16.113.16/30 [110/70] via 172.16.192.6, 09:32:15, TokenRing0
National#
```

参见示例 8-103，虽然还没有发现问题产生的最终原因，但是通过对路由器 Whitney 的串行链路的检查已经发现了问题所在。这两个串行接口应该在区域 0 内，但都被区域 1 替代了。它们都和网络逻辑拓扑上的邻居（路由器 Louvre 和 Hermitage）相连，但是却没有记录 OSPF 邻居。有规律显示的错误信息表明路由器 Whitney 正在接收来自路由器 Louvre 和 Hermitage 的

Hello 数据包，而这些数据包的区域字段设置为 0，因而引起不匹配的情况发生。

示例 8-103 路由器 Whitney 的串行接口被配置成区域 1，从而替代了区域 0；这个配置在接收区域 0 的 Hello 数据包时会引起错误信息

```
Whitney#show ip ospf interface serial 0
Serial0 is up, line protocol is up
  Internet Address 172.16.113.18/30, Area 1

  Process ID 8, Router ID 1.1.1.1, 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
    oob-resync timeout 40
    Hello due in 00:00:03
  Index 1/3, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 2, maximum is 2
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 0, Adjacent neighbor count is 0
  Suppress hello for 0 neighbor(s)

Whitney#show ip ospf interface serial 1
Serial0 is up, line protocol is up
  Internet Address 172.16.113.14/30, Area 1

  Process ID 8, Router ID 1.1.1.1, 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
    oob-resync timeout 40
    Hello due in 00:00:06
  Index 1/3, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 2, maximum is 2
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 0, Adjacent neighbor count is 0
  Suppress hello for 0 neighbor(s)
Whitney#

%OSPF-4-ERRRCV: Received invalid packet: mismatch area ID,

 from backbone area must be virtual-link but not found from 172.16.113.13, Serial1

%OSPF-4-ERRRCV: Received invalid packet: mismatch area ID,

 from backbone area must be virtual-link but not found from 172.16.113.17, Serial0
```

路由器 Whitney 的 OSPF 配置参见示例 8-104。

示例 8-104 路由器 Whitney 的 OSPF 配置

```
router ospf 8
 network 172.16.0.0 0.0.255.255 area 1
 network 172.16.113.0 0.0.0.255 area 0
```

乍一看，这个配置好像是没有问题的。但是，回忆一下第一个案例研究中提到的，**network area** 命令是连续执行的。第二条 **network area** 命令只可能影响到那些和第一条 **network area** 命令不匹配的接口。在这样的配置下，所有匹配第一条 **network area** 命令语句的接口都被设置到区域 1 了。而第二条命令并没有被应用。

正确的配置显示在示例 8-105 中。

示例 8-105 路由器 Whitney 正确的 OSPF 配置

```
router ospf 8
 network 172.16.192.0 0.0.0.255 area 1
 network 172.16.113.0 0.0.0.255 area 0
```

当然，这里可以有多种有效的正确配置。但重要的一点是第一条 **network area** 命令必须足够精确地只去匹配区域 1 的地址，而不含有区域 0 接口的地址。

8.3.2 案例研究：路由汇总配置错误

如图 8-55 所示，显示了一个骨干区域和与之相连的 3 个区域。为了减小链路状态数据库的大小和增强网络的稳定性，在区域之间使用了路由汇总。

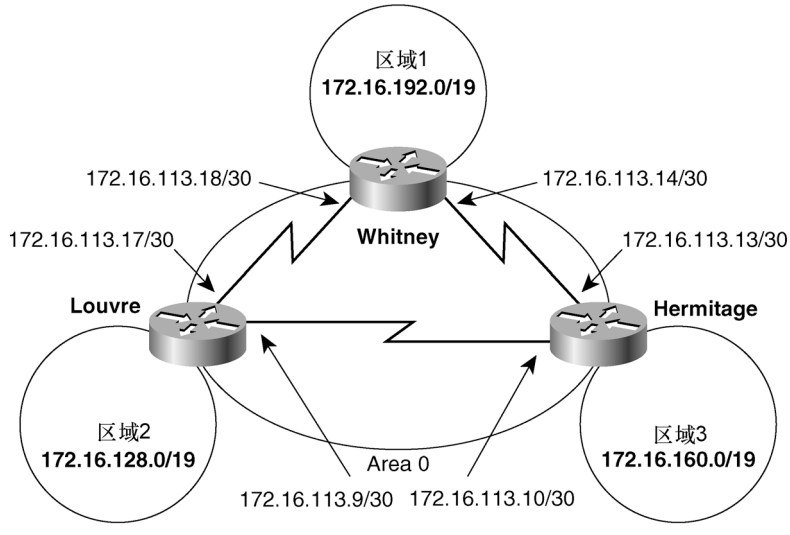


图 8-55 这里显示了被通告到区域 0 内的每一个区域的汇总地址。区域 0 的子网地址也被汇总到其他的区域中

图 8-55 中显示的每个区域的地址汇总了这 3 个非骨干区域内的个别子网。例如，区域 1 内的一些子网可能是：

172.16.192.0/29
172.16.192.160/29
172.16.192.248/30
172.16.217.0/24
172.16.199.160/29
172.16.210.248/30

图 8-56 中显示了这些子网地址能够被汇总成地址 172.16.192.0/19。

10101100000100001100000000000000 = 172.16.192.0/29
10101100000100001100000011111000 = 172.16.192.248/30
10101100000100001101100100000000 = 172.16.217.0/24
10101100000100001100011110100000 = 172.16.199.160/29
10101100000100001101001011111000 = 172.16.210.248/30
10101100000100001100000000000000 = 172.16.192.0/19

图 8-56 一些子网地址能够被汇总成地址 172.16.192.0/19。粗体字部分表明了每一个地址的网络位

路由器 Whitney 的配置参见示例 8-106。

示例 8-106 路由器 Whitney 带有地址汇总的 OSPF 配置

```
router ospf 8
 network 172.16.192.0 0.0.0.255 area 1
 network 172.16.113.0 0.0.0.255 area 0
 area 1 range 172.16.192.0 255.255.224.0
 area 0 range 172.16.113.0 255.255.224.0
```

在其他 3 台 ABR 路由器上也做类似地配置。每台 ABR 路由器将向区域 0 通告与之相连的非骨干区域的汇总地址，并且也把区域 0 的子网地址汇总到了非骨干区域中去。

示例 8-107 中显示出了一个问题。在查看区域 1 的某台内部路由器的路由表时，发现区域 0 的子网地址没有被正确地汇总（讲得再清楚一点，区域 1 的内部子网也没有显示出来）。虽然关于区域 2 和区域 3 的汇总地址被显示出来，但是在路由表中区域 0 的汇总地址却被它内部单独的子网替代了。

示例 8-107 在区域 1 内的某台内部路由器的路由表中记录了区域 0 的个别子网，而替代了应该出现的汇总地址

```
National#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

     172.16.0.0/16 is variably subnetted, 7 subnets, 4 masks
O IA    172.16.160.0/19 [110/80] via 172.16.192.6, 09:32:15, TokenRing0
O IA    172.16.128.0/19 [110/80] via 172.16.192.6, 09:32:15, TokenRing0

C       172.16.192.0/29 is directly connected, TokenRing0
O IA    172.16.113.12/30 [110/70] via 172.16.192.6, 09:32:15, TokenRing0
O IA    172.16.113.8/30 [110/134] via 172.16.192.6, 09:32:15, TokenRing0
O IA    172.16.113.16/30 [110/70] via 172.16.192.6, 09:32:15, TokenRing0
National#
```

当网络管理员以二进制的表示方式检查区域 0 的 3 个子网时，就会发现关于区域 0 的 **area range** 命令可能有问题（如图 8-57 所示）。

10101100000100000111000100001000 = 172.16.113.8/30
10101100000100000111000100001100 = 172.16.113.12/30
10101100000100000111000100010000 = 172.16.113.16/30
11111111111111111111100000000000 = 255.255.224.0
10101100000100000110000000000000 = 172.16.96.0

图 8-57 区域 0 的子网、所配置的汇总掩码和正确的汇总地址

从上面可以看出，这里的问题是 **area range** 命令指定汇总地址（172.16.113.0）比它携带的掩码（255.255.224.0）更具体了。对于这个 19 位的掩码，正确的地址应该是 172.16.96.0（参见示例 8-108）。

示例 8-108 路由器 Whitney 带有正确的地址汇总的 OSPF 配置

```
router ospf 8
 network 172.16.192.0 0.0.0.255 area 1
 network 172.16.113.0 0.0.0.255 area 0
 area 1 range 172.16.192.0 255.255.224.0
 area 0 range 172.16.96.0 255.255.224.0
```

示例 8-109 中显示了更改配置后的路由表的结果。当然，对于汇总区域 0 内的地址也有可以选择的其他地址。例如，172.16.113.0/24 和 172.16.113.0/27 也都是可用的。最恰当的汇总地址是依赖于网络设计的优先性选择的。例如在示例 8-101 中显示的网络里，选用 172.16.96.0/19 是为了保持配置的一致性——所有的汇总地址都使用了 19 位的掩码。另一方面，为了网络更好的扩展性，应该选用 172.16.113.0/27 作为汇总地址。在这个汇总地址里，还可以增加 5 个子网用于骨干区域，而剩余的更为广泛的地址范围可以用于网络上的其他地方。

示例 8-109 区域 0 现在可以被正确地汇总了

```
National#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

     172.16.0.0/16 is variably subnetted, 5 subnets, 3 masks
O IA    172.16.160.0/19 [110/80] via 172.16.192.6, 00:25:09, TokenRing0
O IA    172.16.128.0/19 [110/80] via 172.16.192.6, 09:32:15, TokenRing0

C       172.16.192.0/29 is directly connected, TokenRing0
O IA    172.16.96.0/19 [110/70] via 172.16.192.6, 00:00:10, TokenRing0
National#
```

8.4 展　　望

当提到链路状态路由选择协议的时候，大多数人们首先想到的是 OSPF 协议。但是，OSPF 协议并不是 IP 网络上惟一的链路状态协议。ISO 的中间系统-中间系统（IS-IS）虽然是设计用来为其他网络协议进行路由选择的，但是它也可以为 IP 网络进行路由选择。第 10 章将讲述这个鲜为人知的链路状态路由选择协议。

8.5 总结表：第 8 章命令总结

命令	描述
area *area-id* **authentication[message-digest]**	使一个区域的类型 1 或者类型 2 的认证有效
area *area-id* **default-cost** *cost*	为 ABR 路由器发送到一个末梢区域的缺省路由指定一个代价值

续表

命令	描述
area *area-id* **filter-list prefix** *prefix_list_name* [**out**|**in**]	定义一个类型 3 LSA 的过滤列表
area *area-id* **nssa [no-redistribution][default-information-originate][no-summary][translate type7 suppress-fa]**	配置一个区域为非纯末梢区域（NSSA）
area *area-id* **range** *address mask* [**advertise**|**not-advertise**] [**cost**]	汇总地址进入或离开一个区域，并可以指定汇总地址的代价
area *area-id* *stub***[no-summary]**	配置一个区域作为末梢区域或者完全末梢区域
area *area-id* **virtual-link** *router-id*	在 ABR 路由器之间定义一条虚链路
debug ip ospf adj	显示有关一个 OSPF 邻接关系的创建或中断的事件
[**no**] **discard-route** {**internal**|**external**}	**No discard-route** 去除了自动创建静态路由到 Null 接口
ip ospf authentication-key *password*	使用类型 1 的认证方式分配一个口令给一个 OSPF 接口
ip ospf cost *cost*	在一个 OSPF 接口上指定出站接口的代价大小
ip ospf dead-interval *seconds*	在一个接口上指定 OSPF 的 RouterDeadInterval 大小
ip ospf demand-circuit	配置一个接口作为 OSPF 按需链路
ip ospf hello-interval *seconds*	为一个接口指定 OSPF 的 HelloInterval 的值
ip ospf message-digest-key *key-id* **md5** *key*	使用类型 2 的认证方式指定一个接口的密钥 ID 和密钥（口令）
ip ospf name-lookup	使域名的反向 DNS 域名查找有效，以便在某些 **show** 命令匹配路由器 ID
ip ospf network [broadcast] [nonbroadcast][point-to-multipoint]	配置 OSPF 网络类型
ip ospf priority *number*	设定一个接口的路由器优先级，以便用来选取 DR 和 BDR 路由器
ip ospf retransmit-interval *seconds*	设置一个接口的 OSPF RxmtInterval 值
ip ospf transmit-delay *seconds*	设置一个接口的 OSPF InfTransDelay 值
ip prefix-list *prefix_list_name* [**seq** *num*] {**deny**|**permit**} *address/length*	定义一个前缀列表，指定哪些地址被允许或拒绝
log-adjacency-changes [detail]	记录邻居状态的变化
maximum-paths	设置 OSPF 执行负载均衡的最大路径数量
neighbor *ip-address***[priority** *number***] [poll-interval** *seconds***] [cost** *cost***]**	在一个非广播网络上手工配置邻居路由器
network *address inverse-mask* **area** *area-id*	指定运行 OSPF 协议的接口，并指定这些接口相连的 OSPF 区域
ospf auto-cost reference-bandwidth *reference-bandwidth*	为计算链路的代价，改变缺省的 OSPF 参考带宽
router ospf *process-id*	启动一个 OSPF 路由选择进程
show ip ospf [*process-id*]	显示有关 OSPF 路由选择进程的一般信息
show ip ospf border-routers	显示一台路由器的内部 OSPF 路由表
show ip ospf [*process-id area-id*] **database**	显示 OSPF 链路状态数据库中的所有条目
show ip ospf [*process-id area-id*] **database router** [*link state-id*]	显示 OSPF 链路状态数据库中类型 1 的 LSA
show ip ospf [*process-id area-id*] **database network** [*link state-id*]	显示 OSPF 链路状态数据库中类型 2 的 LSA
show ip ospf [*process-id area-id*] **database summary** [*link state-id*]	显示 OSPF 链路状态数据库中类型 3 的 LSA
show ip ospf [*process-id area-id*] **database asbr-summary** [*link state-id*]	显示 OSPF 链路状态数据库中类型 4 的 LSA
show ip ospf [*process-id area-id*] **database nssa-external** [*link state-id*]	显示 OSPF 链路状态数据库中类型 7 的 LSA
show ip ospf [*process-id*] **database external** [*link state-id*]	显示 OSPF 链路状态数据库中类型 5 的 LSA

续表

命令	描述
show ip ospf [*process-id area-id*] **database database-summary**	根据类型和区域 ID 显示 OSPF 链路状态数据库中 LSA 的数量
show ip ospf interface [*type number*]	显示一个接口具体的 OSPF 信息
show ip ospf neighbor [*type number*][*neighbor-id*][**detail**]	显示 OSPF 邻居表的信息
show ip ospf virtual-links	显示有关 OSPF 虚链路的信息
timer lsa-group-pacing *seconds* 或 **timer pacing lsa-group** *seconds*	在重刷新计时器超时的两组 LSA 之间设定的最小步调时间

8.6 推荐读物

John Moy，“OSPF Version 2”，RFC 2328，1998 年 4 月。

John Moy，*OSPF：Anatomy of an Internet Routing Protocol*。Reading，Massachusetts：Addison-Wesley，1998。

由 OSPF 的最初设计者和 RFC 的作者之一编写，这本书是非常值得阅读，它不仅全面阐述了 OSPF 协议，还给出了重要观点。第 3 章特别有趣，反映了在路由选择协议的设计、测试和标准化方面的内行观点。

8.7 复 习 题

1．什么是 OSPF 邻居？
2．什么是 OSPF 邻接关系？
3．OSPF 数据包的 5 种类型是什么？每一种类型的用途是什么？
4．什么是 LSA？怎样区分一个 LSA 和一个 OSPF 更新数据包的不同？
5．LSA 的类型 1 到类型 5，以及类型 7 分别是什么？每一种类型的用途是什么？
6．什么是链路状态数据库？链路状态数据库的同步是什么意思？
7．什么是缺省的 HelloInterval？
8．什么是缺省的 RouterDeadInterval？
9．什么是路由器 ID？怎样确定一个路由器 ID？
10．什么是区域？
11．区域 0 的含义是什么？
12．什么是最大生存时间（MaxAge）？
13．OSPF 协议的 4 种路由器类型是什么？
14．OSPF 协议的 4 种路径类型是什么？
15．OSPF 协议的 5 种网络类型是什么？
16．什么是指定路由器 DR？
17．在 Cisco 路由器上是怎样计算一个接口的出站代价的？
18．什么是分段的区域？

19．什么是虚链路？

20．末梢区域、完全末梢区域和非纯末梢区域之间有什么不同？

21．OSPF 网络条目和 OSPF 路由器条目之间有什么不同之处？

22．为什么类型 2 的认证方式比类型 1 的认证方式更好？

23．在 LSA 头部中哪 3 个字段是用来区分不同的 LSA 的？另外，在 LSA 头部中哪 3 个字段是用来区分相同 LSA 的不同实例的？

8.8 配置练习

1．表 8-13 显示了 14 台路由器的接口和地址，其中也表明了每一个接口相连的 OSPF 区域。根据表中提供的信息，假定下面的事实：

- 每一台路由器的所有接口都显示在表中。
- 如果没有区域显示（-），就表示在相关的接口上不运行 OSPF 协议。
- 子网地址的第二个八位组字节和区域 ID 相同。
- 每一个 OSPF 接口地址的前 16 位指定一个区域。例如，前缀是 10.30.x.x 的地址将只能在区域 30 中出现。

请写出表 8-13 中的路由器关于 OSPF 的配置（提示：可以首先画出一台路由器和子网的拓扑图）。

表 8-13 关于配置练习 1～6 的路由器信息

路由器	接口	地址/掩码	区域 ID
A	L0	10.100.100.1/32	-
	E0	10.0.1.1/24	0
	E1	10.0.2.1/24	0
	E2	10.0.3.1/24	0
	E3	10.0.4.1/24	0
B	L0	10.100.100.2/32	-
	E0	10.0.1.2/24	0
	E1	10.5.1.1/24	5
	S0	10.5.255.13/30	5
	S1	10.5.255.129/30	5
C	L0	10.100.100.3/32	-
	E0	10.0.2.2/24	0
	E1	10.10.1.1/24	10
	S0	10.30.255.249/30	30
D	L0	10.100.100.4/32	-
	E0	10.0.3.2/24	0
	E1	10.20.1.1/24	20
E	L0	10.100.100.5/32	-
	E0	10.0.4.2/24	0
	S0	10.15.255.1/30	15
F	L0	10.100.100.6/32	-
	E0	10.5.5.1/24	5
	S0	10.5.255.130/30	5
	S1	10.5.255.65/30	5

续表

路由器	接口	地址/掩码	区域 ID
G	L0 E0 S0	10.100.100.7/32 10.10.1.58/24 10.10.255.5/30	- 10 -
H	L0 E0 E1 S0	10.100.100.8/32 10.20.1.2/24 10.20.100.100/27 10.20.255.225/30	- 20 20 -
I	L0 E0 S0	10.100.100.9/32 10.35.1.1/24 10.5.255.66/30	- 35 5
J	L0 E0 S0	10.100.100.10/32 10.15.227.50/24 10.15.225.2	- 15 15
K	L0 E0 S0*	10.100.100.11/32 10.30.1.1/24 10.30.254.193/26	- 30 30
L	L0 E0 S0*	10.100.100.12/32 10.30.2.1/24 10.30.254.194/26	- 30 30
M	L0 E0 S0* S1	10.100.100.13/32 10.30.3.1/24 10.30.254.195/26 10.30.255.250/30	- 30 30 30
N	L0 E0 S0*	10.100.100.14/32 10.30.4.1/24 10.30.254.196/26	- 30 30

* 表示帧中继封装。

2．在表 8-13 中的所有 ABR 路由器上配置路由汇总。

3．更改配置，使区域 15 成为一个末梢区域。

4．更改配置，使区域 30 成为一个完全末梢区域。

5．路由器 H 的 S0 接口和一个运行其他路由选择协议的路由器相连，并将从该路由选择协议学习到的路由重新分配到 OSPF 域中。更改必要的配置，以便使这些重新分配的路由可以在整个 OSPF 域内通告，但是不允许任何类型 5 的 LSA 通告到区域 20 中。

6．在路由器 C 和路由器 M 之间的串行链路是一条带宽很低的链路。更改配置，以便使 OSPF 协议把这条链路作为一个按需链路看待。

8.9 故障排除练习

1．在两台路由器上的 OSPF 不能工作。当打开 **debug** 命令进行调试后，发现每 10s 就会收到示例 8-110 中显示的信息。请问网络发生了什么故障？

示例 8-110 故障排除练习 1 的调试信息

```
RTR_EX1#debug ip ospf adj
OSPF adjacency events debugging is on
RTR_EX1#
OSPF: Rcv pkt from 172.16.27.1, TokenRing0, area 0.0.0.25 : src not on the same
  network
OSPF: Rcv pkt from 172.16.27.1, TokenRing0, area 0.0.0.25 : src not on the same
  network
OSPF: Rcv pkt from 172.16.27.1, TokenRing0, area 0.0.0.25 : src not on the same
  network
OSPF: Rcv pkt from 172.16.27.1, TokenRing0, area 0.0.0.25 : src not on the same
  network
OSPF: Rcv pkt from 172.16.27.1, TokenRing0, area 0.0.0.25 : src not on the same
  network
```

2．根据示例 8-111 中显示的调试信息，查明网络出现的故障。

示例 8-111 故障排除练习 2 的调试信息

```
RTR_EX2#debug ip ospf adj
OSPF adjacency events debugging is on
RTR_EX2#
OSPF: Hello from 172.16.27.195 with mismatched Stub/Transit area option bit
OSPF: Hello from 172.20.1.1 with mismatched Stub/Transit area option bit
OSPF: Hello from 172.16.27.195 with mismatched Stub/Transit area option bit
OSPF: Hello from 172.20.1.1 with mismatched Stub/Transit area option bit
OSPF: Hello from 172.16.27.195 with mismatched Stub/Transit area option bit
OSPF: Hello from 172.20.1.1 with mismatched Stub/Transit area option bit
OSPF: Hello from 172.16.27.195 with mismatched Stub/Transit area option bit
OSPF: Hello from 172.20.1.1 with mismatched Stub/Transit area option bit
```

3．根据示例 8-112 中显示的错误信息，查明网络出现的故障。

示例 8-112 故障排除练习 3 的调试信息

```
RTR_EX3#
OSPF: Send with youngest Key 10
OSPF: Rcv pkt from 10.8.1.1, Ethernet0 : Mismatch Authentication type. Input
packet specified type 0, we use type 2
OSPF: Send with youngest Key 10
OSPF: Rcv pkt from 10.8.1.1, Ethernet0 : Mismatch Authentication type. Input
packet specified type 0, we use type 2
RTR_EX3#
```

4．根据示例 8-113 中显示的错误信息，查明网络出现的故障。

示例 8-113 故障排除练习 4 的调试信息

```
RTR_EX4#
OSPF: Send with youngest Key 10
OSPF: Rcv pkt from 10.8.1.1, Ethernet0 : Mismatch Authentication Key - Message D
igest Key 10
OSPF: Send with youngest Key 10
OSPF: Rcv pkt from 10.8.1.1, Ethernet0 : Mismatch Authentication Key - Message D
igest Key 10
RTR_EX4#
```

5．根据示例 8-114 中显示的错误信息，查明网络出现的故障。

示例 8-114　故障排除练习 5 的调试信息

```
RTR_EX5#
%OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must
be virtual-link
but not found from 10.8.1.1, Ethernet0
%OSPF-4-ERRRCV: Received invalid packet: mismatch area ID, from backbone area must
be virtual-link
but not found from 10.8.1.1, Ethernet0
RTR_EX5#
```

6．在图 8-58 中所示的路由器上做如下配置。

路由器 A：

```
router ospf 15
network 192.168.50.224 0.0.0.31 area 192.168.50.0
network 192.168.50.240 0.0.0.15 area 0.0.0.0
area 192.168.50.0 authentication message-digest
```

路由器 B：

```
router ospf 51
network 192.168.50.0 0.0.0.255 area 0
```

在这里，路由器 A 和 B 不能形成一个邻接关系。请问发生了什么问题？

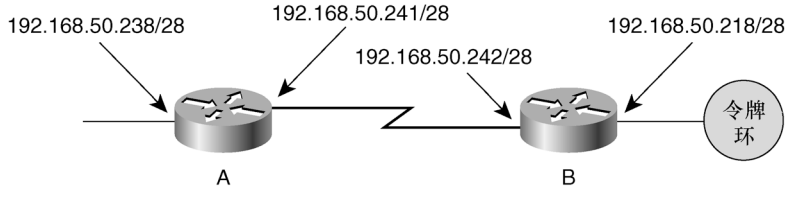


图 8-58　故障排除练习 6 的调试信息

7．示例 8-115 显示了某个区域里的一个链路状态数据库，其中这个区域里存在一条不稳定的链路。基于所显示的信息，哪一条链路看起来像是有问题的链路？

示例 8-115　故障排除练习 7 的链路状态数据库

```
RTR_EX7#show ip ospf database

OSPF Router with ID (10.8.20.1) (Process ID 1)

Router Link States (Area 0)

Link ID    ADV Router   Age   Seq#         Checksum   Link count
10.3.0.1   10.3.0.1     18    0x8000001B   0x6AF8     5
10.8.5.1   10.8.5.1     15    0x80000267   0xFDA0     6
10.8.20.1  10.8.20.1    478   0x8000001E  0xD451     4

Net Link States (Area 0)

Link ID    ADV Router   Age   Seq#         Checksum
10.8.1.2   10.3.0.1     18    0x80000013   0xA747
RTR_EX2#
```

本章包括以下主题：

- OSPFv3 的基本原理与实现；
- OSPFv3 的配置；
- OSPFv3 故障诊断。

第 9 章

OSPFv3

正如读者在前面章节中所看到的，IPv6 的路由选择需要对协议进行一些改动。首先最主要的改动就是，每个协议消息都必须改为能够传送长度为 IPv4 地址 4 倍的地址。理论上来说，这些可以通过开放最短路径优先（OSPF）协议来实现，要么通过更改现有的链路状态通告（LSA），要么定义新的 LSA。但是，OSPF 协议的发展早在 20 世纪 80 年代末就开始了，当时路由器的性能很低，时延很大，而且内存昂贵。现在这些问题都不存在了，OSPFv2 的一些用来适应和调节那些早期连网实际情况的特性现在已不实用了。更进一步来说，OSPFv2 协议的大量实际应用经验表明，它在某些领域显得没有效率。

因此，在起初考虑扩展 OSPF 支持 IPv6 的时候，就意识到这是一个改进优化 OSPF 协议本身的机会。结果是，不仅仅为 IPv6 对 OSPFv2 进行了扩展，还创建了一个新的 OSPF 的改进版本——OSPF 第 3 版。

9.1 OSPFv3 的基本原理与实现

OSPFv3 在 RFC 2740 中有详细描述。OSPFv3 与 OSPFv2 的关系，非常类似于 RIPng 与 RIPv2 的关系。最重要的是，OSPFv3 使用了与 OSPFv2 相同的基本实现机制——SPF 算法、泛洪扩散、DR 选举、区域等。还有一些像计时器与度量等常量和变量也是相同的。

另外一个和 RIPng 与 RIPv2 的关系的类似之处是，OSPFv3 也不向后兼容 OSPFv2。因此，如果读者希望在 IPv4 和 IPv6 环境中同时使用 OSPF 协议，就必须同时运行 OSPFv2 和 OSPFv3 协议。在撰写本章的时候，有一个有关增加 IPv4 协议支持 OSPFv3 协议的讨论，但是还没有完成相关

的细节和规范。作为个人来讲，我希望增加这个支持，因为 OSPFv3 协议是一个重要的改进协议。

在这里，假定读者已经阅读了前面的章节，并掌握了 OSPFv2 的有关基本原理与实现。本章将不再重复讲述那些内容，而仅仅阐明 OSPFv3 的重要不同之处——主要的基本原理与 LSA 格式。

9.1.1 OSPFv3 与 OSPFv2 的不同之处

除了在下面章节讲述的有关 LSA 的变化外，还有一些 OSPF 机制本身的变化。在本小节中将讲述其中一些最重要的变化。

- **每个链路上的协议处理**——读者在前面的章节中已经看到，一条链路的接口可以拥有多个 IPv6 的地址。事实上，单条链路可以属于多个子网，与同一条链路相连但属于不同的 IPv6 子网的两个接口仍然可以通信。在 OSPFv3 中将 OSPFv2 的“子网”概念改变为了“链路”概念，而且允许在同一条链路上但属于不同 IPv6 子网的两个邻居交换数据包。
- **取消了寻址概念**——正如读者在随后的章节中所读到的，OSPFv3 的路由器 LSA 和网络 LSA 都不再携带 IP 地址。这里会定义一个新的 LSA 来实现这项功能，这具有一些扩展性方面的好处。但是，32 位的 RID、AID，以及 LSA ID 都保留在 IPv6 中。记住，虽然这些 ID 是用点分十进制编写的，经常来自工作在 IPv4 地址环境中的 OSPFv2 网络，但是它们不是 IPv4 地址。这些 OSPFv3 的 ID 仍然用点分十进制表示，允许在现有的 OSPFv2 网络上简单地叠加一个 OSPFv3 网络。
- **邻居总是通过路由器 ID 来标识**——在广播网络和 NBMA 网络的链路上，OSPFv2 邻居是通过它们的接口地址来标识的，而其他类型链路上的邻居是通过 RID 来标识的。OSPFv3 取消了这种不一致性，在所有类型的链路上的所有邻居都通过 RID 来标识。
- **增加了链路本地泛洪扩散范围**——OSPFv3 保留了 OSPFv2 中域（或 AS）和区域（area）泛洪扩散的范围，但增加了一个链路本地泛洪扩散的范围。读者从第 2 章的介绍中已经了解到，IPv6 使链路本地范围具有很多用途。正如读者将在后面章节所看到的，增加新的 LSA——链路 LSA（Link LSA）用来携带仅仅与单个链路上的邻居相关联的信息。这种 LSA 具有链路本地泛洪扩散的范围，这就意味着它不能超出任何相连的路由器的范围而进行扩散。
- **链路本地地址的使用**——读者已经知道 OSPFv2 的数据包具有链路本地的范围，它们不能被任何路由器转发。OSPFv3 则使用路由器的链路本地 IPv6 地址（回忆在第 2 章中，这些地址总是以 FF80::/10 开头的）作为源地址和下一跳地址。
- **对每个链路上多个实例的支持**——这种支持应用在多台 OSPF 路由器连接到单个广播链路上，但它们又不属于单个邻接关系的情况。这种情况的一个例子就是共享的网络接入点（Network Access Point，NAP）。例如，假定有 4 台路由器连接到同一条以太网链路。路由器 1 和路由器 2 属于一个 OSPF 域，路由器 3 和路由器 4 属于另外一个不同的 OSPF 域。那么，它们应该分别在路由器 1 和路由器 2 之间以及路由器 3 和路由器 4 之间建立相应的邻接关系，而不是建立在路由器 1 和路由器 3 之间。读者可以利用认证操作来完成这种 OSPFv2 邻接关系的隔离，但这不是一种理想的方法。在另外一些情况，属于一个邻接关系的路由器将会不断地记录被其他邻接拒绝的 Hello 包的认证失败记录。OSPFv3 允许每条链路上具

有多个不同的实例，这是通过在 OSPF 包头里增加一个实例标识（Instance ID）区别不同的实例来实现的。一个分配了给定实例标识的实例将会丢弃那些与该实例标识不匹配的 OSPF 数据包。

- **取消了 OSPF 特有的认证**——IPv6 协议使用认证扩展报头，这是一个标准的认证过程。由于这个原因，OSPFv3 不需要 OSPFv3 数据包自己的认证，它只要使用 IPv6 的认证就可以了。
- **更灵活地处理未知 LSA 类型**——在 OSPFv2 中总是丢弃未知的 LSA 类型，而 OSPFv3 可以把它们当作链路本地泛洪扩散范围，或者像它们被识别一样保存和泛洪扩散，但在它们自己的 SPF 算法中将被忽略。结果是，在 OSPFv3 中处理网络变化和继承新的特性比在 OSPFv2 中更容易。

9.1.2 OSPFv3 的消息

OSPFv3 和 OSPFv2 使用相同的协议号——89，当然 OSPFv3 作为一个 IPv6 的协议，更准确地说是使用设置为 89 的下一报头值。和 OSPFv2 一样，OSPFv3 尽可能的使用多播。IPv6 的 AllSPFRouters 多播地址为 FF02::5，AllDRouters 多播地址为 FF02::6。它们都是链路本地的范围。读者在这里可以很容易地发现它们和 OSPFv2 的相应多播地址 224.0.0.5 和 224.0.0.6 的最后一些位是相似的。

OSPFv3 同样使用相同的 5 种消息类型——Hello、DD、LS 数据库请求、LS 数据库更新和 LS 确认，这和 OSPFv2 中的一样，包括它们的编号也相同。如图 9-1 所示显示了 OSPFv3 的消息头部，这与 OSPFv2 消息头部稍微有些不同。当然，它的版本号不是 2，而是 3。但更重要的是，在它的报头里没有认证字段。正如前面所讨论的，OSPFv3 使用 IPv6 数据包本身的扩展报头来进行认证，而不是利用它自己的认证进程。在这里，我们可以看到有一个实例标识，用来允许在同一个链路上运行多个 OSPFv3 实例。请注意，接口标识只在本地链路上具有意义，因为 OSPFv3 消息不能转发到始发它的链路之外。

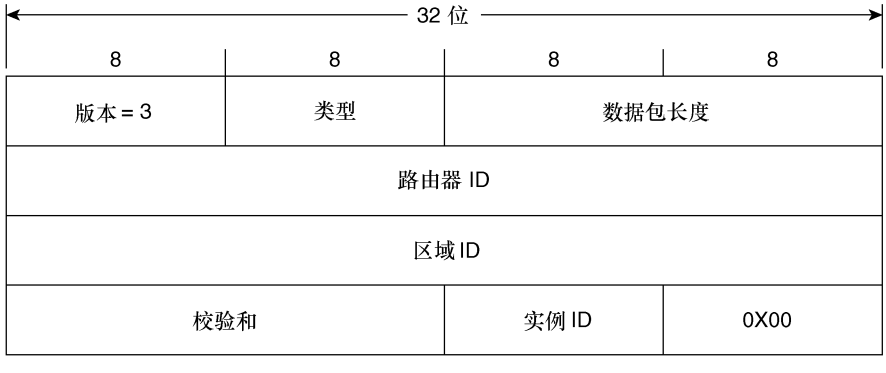


图 9-1 OSPFv3 的包头格式

除了消息报头之外，只有两种 OSPFv3 消息的格式和 OSPFv2 对应的消息不同：Hellos 和数据库描述消息。

如图 9-2 所示，图中显示了 OSPFv3 的 Hello 消息格式。和 OSPFv2 不同，由于 IPv6 不需要网络掩码，这里的消息格式中没有网络掩码字段。除此以外，两个版本的数据包都具有

相同的字段（报头以下）。可选项字段的大小增大到了 24 位，路由器无效时间间隔则从 32 位减少到了 16 位。这个字段意味着路由器无效时间间隔的理论最大值从 43 亿秒减少为 65 535 秒。这个变化对于所运行的网络是比较小的，影响很小，甚至没有。在大多数普通的 OSPF 实现中所配置的最大路由器无效时间间隔总是 65 535 秒，好的 OSPF 设计是不会接近于这个最大值的。因此，这个字段大小的变化仅仅是节约了无用的空间。



图 9-2 OSPFv3 的 Hello 消息格式

在图 9-3 中显示了 OSPFv3 数据库描述数据包的格式。与 OSPFv2 的对应部分所不同的仅仅是更大一点的可选项字段。

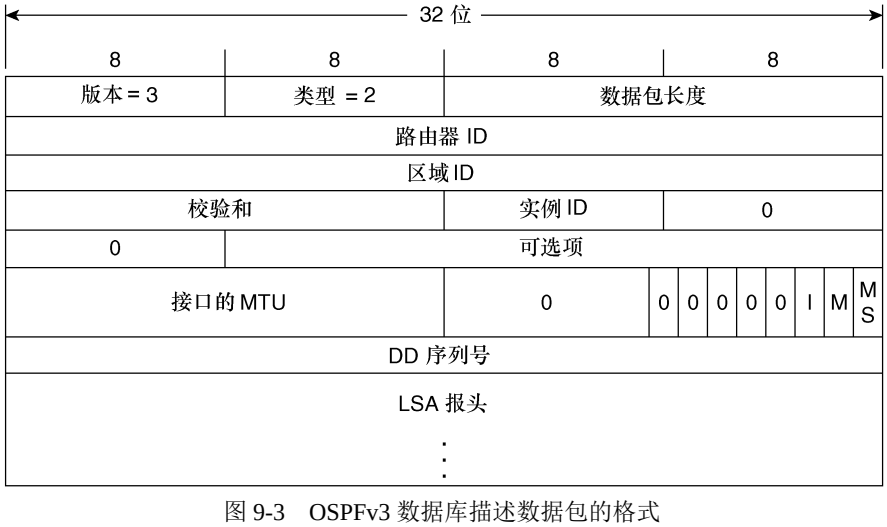


图 9-3 OSPFv3 数据库描述数据包的格式

其他 3 种类型的消息——链路状态请求、链路状态更新和链路状态确认——的格式都和 OSPFv2 中对应的消息类型格式相同，因此在这一章将不再赘述。

9.1.3 OSPFv3 的 LSA 概述

如图 9-4 所示，图中显示了 OSPFv3 的 LSA 报头。与图 8-54 中显示的 OSPFv2 的 LSA 报头对比，读者可以看出除了两处不同外，它们几乎是相同的。一是没有了可选项字段，二是链路状态类型字段的大小是 16 位，而不是 OSPFv2 中的 8 位类型字段。

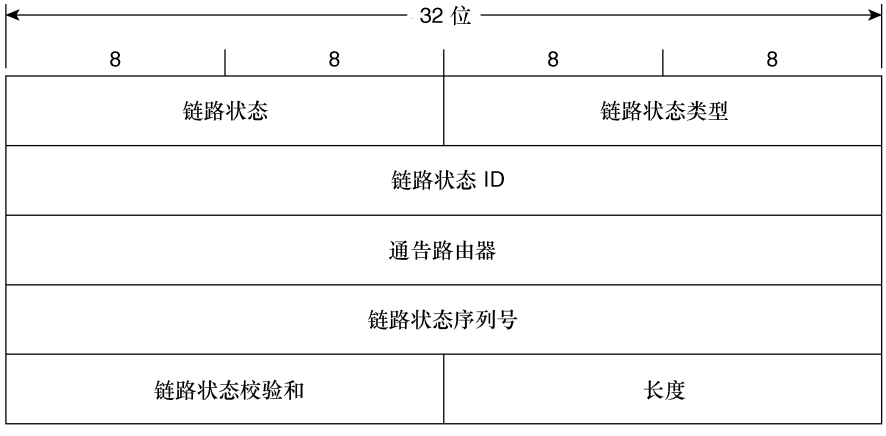


图 9-4 OSPFv3 的 LSA 报头格式

OSPFv3 的 LSA 报头中链路状态类型字段加长的原因是因为它包含了 3 个前置位，如图 9-5 所示。



图 9-5 OSPFv3 的 LSA 报头的链路状态类型字段

U 位——指出了一台路由器如果不能识别 LSA 的功能代码（LSA Function Code）时，应该如何处理该 LSA。如果该位没有设置，未知的 LSA 将被作为链路本地泛洪扩散的范围处理。如果该位被设置，未知的 LSA 将会像被识别的 LSA 一样被保存和扩散。

S2 和 S1 位——指出了 LSA 的泛洪扩散范围。在表 9-1 中列出了这两位可能的值和相对应的泛洪扩散范围。

表 9-1 OSPFv3 的 LSA 链路状态类型字段中的 S 位和它们对应的泛洪扩散范围

S2	S1	泛洪扩散范围
0	0	链路本地
0	1	区域
1	0	自主系统（AS）（路由选择域）
1	1	保留

LSA 功能代码（LSA Function Code）是 LS 类型字段的最后 13 位，这和 OSPFv2 的类型

字段相一致。表 9-2 中显示了 OSPFv3 常用的 LSA 类型和它们对应的 LS 类型的值。如果读者翻译这些十六进制的值，将可以看到它们所有的 U 位缺省都是 0。除了两种类型之外，其他所有的 LSA 的 S 位都标识为区域范围。两个例外是，AS 外部 LSA 具有 AS 泛洪扩散范围，而链路 LSA 具有链路本地的泛洪扩散范围。大多数 OSPFv3 的 LSA 和 OSPFv2 相应的 LSA 具有相同的功能，这些 OSPFv2 的 LSA 和它们的类型也显示在表 9-2 中。

表 9-2 OSPFv3 的 LSA 类型和它们在 OSPFv2 中对应部分

OSPFv3 的 LSA		OSPFv2 的 LSA	
LS 类型	名称	类型	名称
0x2001	路由器 LSA	1	路由器 LSA
0x2002	网络 LSA	2	网络 LSA
0x2003	区域间前缀 LSA	3	网络汇总 LSA
0x2004	区域间路由器 LSA	4	ASBR 汇总 LSA
0x4005	AS 外部 LSA	5	AS 外部 LSA
0x2006	组成员 LSA	6	组成员 LSA
0x2007	类型 7 LSA	7	NSSA 外部 LSA
0x0008	链路 LSA		没有对应的 LSA
0x2009	区域内前缀 LSA		没有对应的 LSA

虽然 OSPFv3 和 OSPFv2 的路由器 LSA 与网络 LSA 具有相同的名字，但它们的功能在两个版本里有很大的区别。特别地，OSPFv3 的路由器与网络 LSA 并不通告前缀。在协议的扩展性方面，这是一个重要的改进。正如读者在第 8 章所了解的，这些 LSA 把路由器首先看作 SPF 树上的一个节点。因此，当路由器 LSA 或网络 LSA 扩散时， 就假定网络拓扑已经发生了变化，区域内的所有路由器在收到 LSA 后就要重新运行 SPF。但是，由于 OSPFv2 路由器也使用这些 LSA 通告它们所连接的子网，如果子网发生变化，相应的 LSA 也必须进行扩散以便通告这种变化。即使是像一些并不影响 SPF 拓扑的地址变化，收到路由器 LSA 或网络 LSA 也总会触发一个 SPF 算法的运行。这在那些具有很多定期变化的末梢链路的边界或接入路由器上特别容易出现问题。

OSPFv3 在路由器 LSA 和网络 LSA 中取消了通告前缀的功能，而是把这项功能放入新的区域内前缀 LSA 当中。这样路由器 LSA 和网络 LSA 对于 SPF 来说就只代表路由器的节点信息，并且只有当与 SPF 算法相关的信息发生变化时它们才会进行扩散。如果前缀发生变化，或者末梢链路的状态发生改变，这些信息将在区域内前缀 LSA 中进行扩散，而区域内前缀 LSA 不会触发 SPF 的运行。

路由器与网络 LSA 在两个 OSPF 版本之间的另外一个不同之处是，一些只与直连邻居相关的信息交换。OSPFv2 把这个信息放入路由器与网络 LSA 里，虽然只有直连邻居关心这个信息，但它却伴随着路由器与网络 LSA 在整个区域里进行扩散。OSPFv3 则把这个邻居特有的信息放入新的链路 LSA 中，链路 LSA 只具有链路本地的扩散范围。链路 LSA 的引入确实提高了 OSPFv2 的效率，尽管这种提高并不明显。

区域间前缀、区域间路由器和类型 7 LSA 的名字虽然变化了，但它们分别与 OSPFv2 中对应的网络汇总、ASBR 汇总和 NSSA LSA 具有相同的功能。AS 外部和组成员 LSA 的名字和功能在两个 OSPF 版本中都是一样的。

9.1.4 OSPFv3 的 LSA 格式

这一小节将详细讲述表 9-2 中前面 5 种 OSPFv3 的 LSA 类型和两种新的 LSA 类型的格式。虽然有一个在多播 OSPF（MOSPF）中专用的组成员 LSA，但多播协议并不在本书的讲述范围内，因此也不详细讨论这个 LSA。在大多数情况下，这里仅仅讨论与 OSPFv2 的对应部分有所不同的 LSA 字段特性。建议读者将具有对应的 OSPFv2 LSA 的 LSA 与第 8 章中的内容比较。

1. 路由器 LSA

如图 9-6 所示，图中显示了 OSPFv3 路由器 LSA 的格式。正如前面章节所讨论的，在路由器 LSA 中没有包含前缀信息，而在 OSPFv2 版本对应的路由器 LSA 中是包含的。OSPFv3 的路由器 LSA 仅仅描述始发路由器和与连接到邻居的链路，用于 SPF 的计算。前缀信息则在区域内前缀 LSA 中携带。

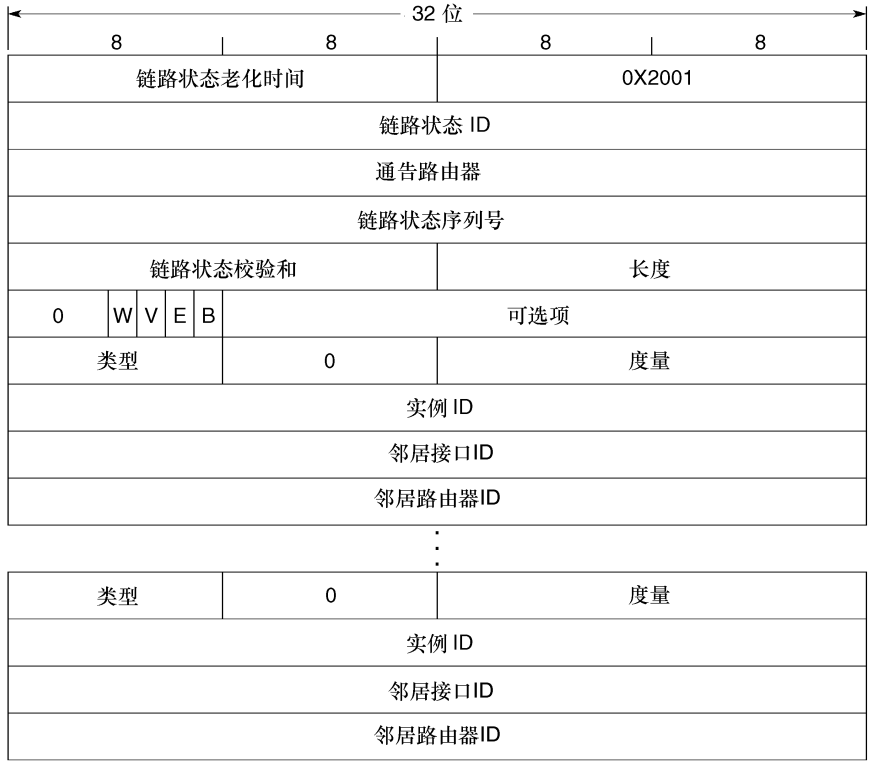


图 9-6 OSPFv3 的路由器 LSA 格式

这里也要注意，在 OSPFv2 中已经淘汰的 ToS 度量字段也没有包含在这个 LSA 中。

- **可选项字段**——OSPFv3 与 OSPFv2 的可选项字段相比，它在 LSA 格式中的位置不同，长度也加长了（24 位），但它们承担的作用是相同的，都是标识一些可选的性能。在后面的章节中将会分别讲述各种数据包和 LSA 中携带的可选项字段。

跟在可选项字段之后的是一组多次出现用来描述每一种相关联的接口的字段集合。

- **类型（Type）**——是指接口的类型。表 9-3 中列出了可能的接口类型的值。

表 9-3 路由器 LSA 的类型字段中指定的接口类型

类型	描述
1	点到点连接到另一台路由器
2	连接到一个传送网络（transit network）
3	保留
4	虚链路

- **度量（Metric）**——是指接口的出站代价（outbound cost）。
- **接口 ID**——是一个 32 位的值，用来把该接口与始发路由器上其他的接口区分开来。
- **邻居接口 ID（Neighbor Interface ID）**——是在 Hello 数据包中链路上邻居通告的接口 ID，或者是类型 2 的链路上链路的指定路由器（DR）的接口 ID。
- **邻居路由器 ID（Neighbor Router ID）**——是指邻居的 RID，或者是类型 2 的链路上指定路由器（DR）的 RID。

2. 网络 LSA

如图 9-7 所示，图中显示了 OSPFv3 的网络 LSA 格式。就其功能而言，它与 OSPFv2 中的网络 LSA 是相同的（始发于 DR，代表一个伪节点，假定到与它直接相连的邻居的代价为 0，等等）。它们之间惟一的重要不同之处是可选项字段的位置不同，OSPFv3 中取消了在 IPv6 协议中没有意义的网络掩码字段。

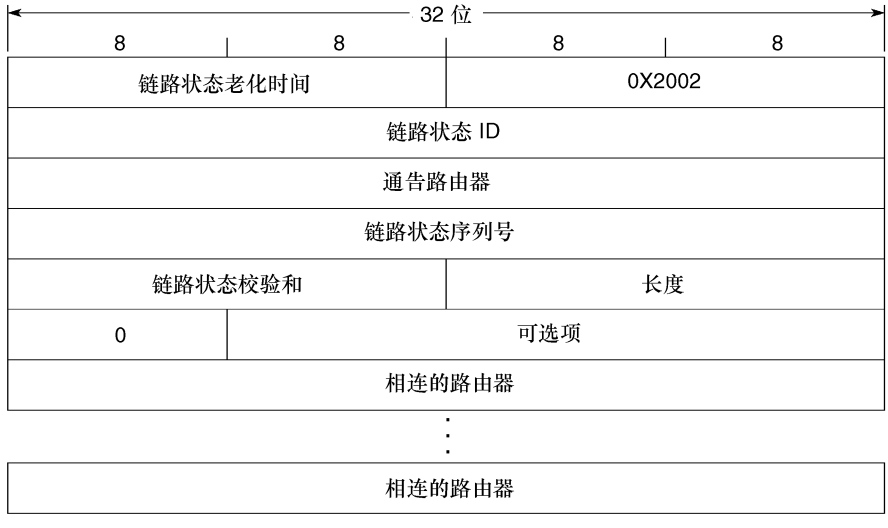


图 9-7 OSPFv3 的网络 LSA 格式

3. 区域间前缀 LSA

如图 9-8 所示，图中显示了 OSPFv3 区域间前缀 LSA 的格式。它的作用和 OSPFv2 中类型 3 汇总 LSA 相同——由 ABR 路由器始发这种 LSA 到一个区域，通告该区域外部但仍然属于它的 OSPF 域内的网络。在 OSPFv3 中只是名字变得更具有描述性。ABR 路由器必须为每一个在区域内被通告的 IPv6 前缀发起一个单独的区域间前缀 LSA。ABR 路由器也能够始发一个区域间前缀 LSA 向一个末梢区域通告一条缺省路由。

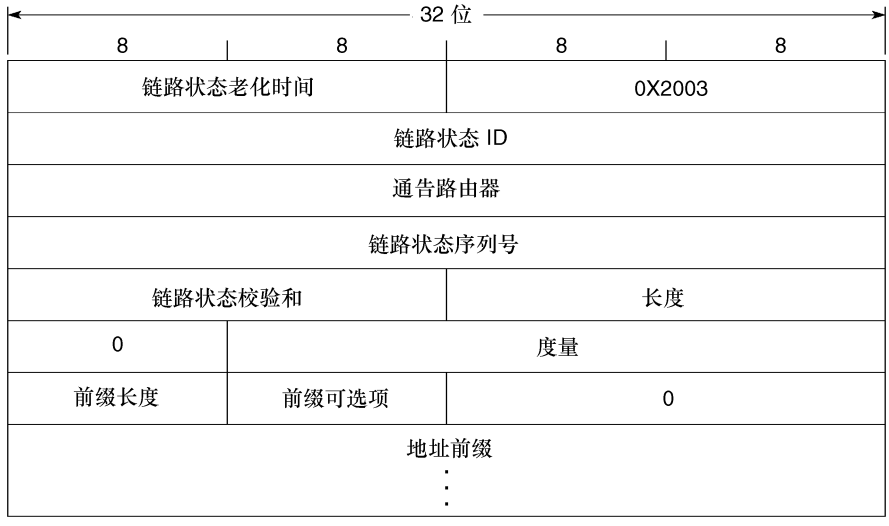


图 9-8 OSPFv3 区域间前缀 LSA 的格式

4. 区域间路由器 LSA

在 OSPFv3 中区域间路由器 LSA 的功能和 OSPFv2 中类型 4 汇总 LSA 所承担的功能是相同的。ABR 向一个区域内始发一条区域间路由器 LSA，用来通告一个在该区域外的 ASBR 路由器。对于所通告的每一个 ASBR，ABR 都需要始发单独的区域间路由器 LSA。

如图 9-9 所示，图中显示了 OSPFv3 区域间路由器 LSA 的格式。即使 OSPFv2 类型 3 和类型 4 汇总 LSA 具有相同的格式，读者通过比较图 9-8 和图 9-9 能够看到区域间网络 LSA 和区域间路由器 LSA 是不同的。字段是自描述的。

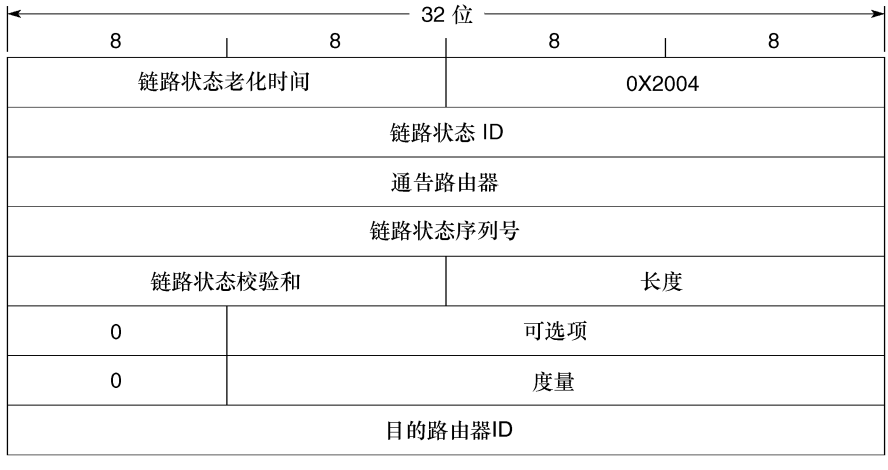


图 9-9 OSPFv3 区域间路由器 LSA 的格式

- **可选项**——表示 ASBR 的可选性能。
- **度量**——表示 ASBR 的代价。
- **目的路由器 ID（Destination Router ID）**——是指 ASBR 的 RID。

5. AS 外部 LSA

和 OSPFv2 中一样，AS 外部 LSA 通告处于 OSPF 路由选择域外部的前缀。对于每一条

需要通告的外部前缀都需要一条这样的 LSA。但是，OSPFv3 中 AS 外部 LSA 的格式（参见图 9-10）不同于 OSPFv2 中的对应部分。

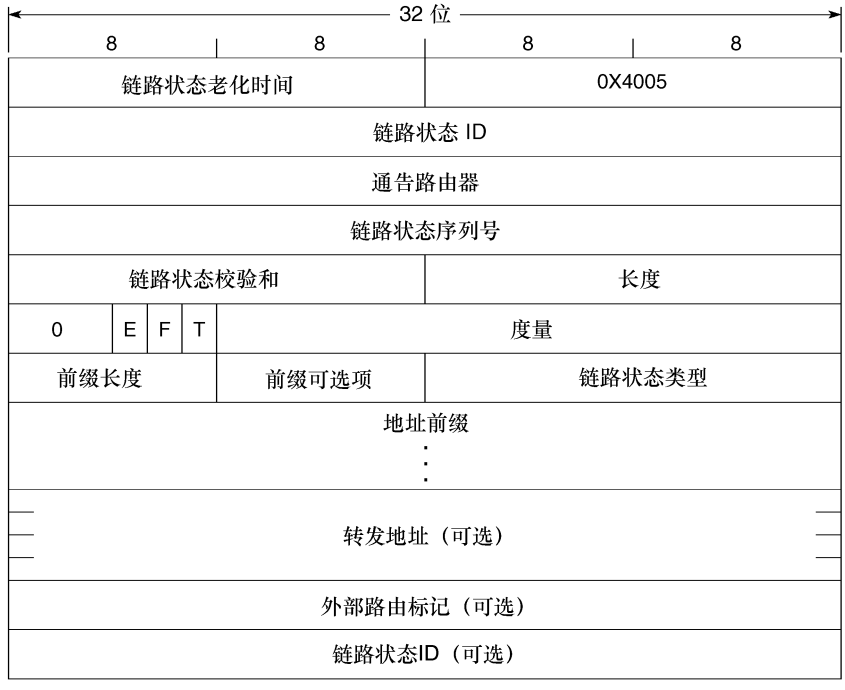


图 9-10 OSPFv3 AS 外部 LSA 的格式

- **E 标记**——在 OSPFv3 版的 LSA 中它的功能与它在 OSPFv2 版的 LSA 中的功能相同。如果设置该位，表示度量是类型 2 的外部度量。如果没有设置该位，表示度量是类型 1 的外部度量。
- **F 标记**——当设置该位时，表示该 LSA 中包含一个转发地址。
- **T 标记**——当设置该位时，表示该 LSA 中包含一个外部路由标记。
- **度量**——顾名思义，表示路由的代价。无论它是类型 1 还是类型 2，都依赖于 E 标记的值。
- **前缀长度、前缀可选项和地址前缀**——完整地描述了嵌套的前缀。
- **转发地址**——如果包含这一项，它是一个完整的 128 位 IPv6 地址，用来表示目的前缀的下一跳地址。只有设置了 F 标记才会包含这一项。
- **外部路由标记（External Route Tag）**——如果包含这一项，它所承担的作用和 OSPFv2 中 AS 外部 LSA 的外部路由标记字段的作用是相同的。只有设置了 T 标记才会包含这一项。
- **引用链路状态 ID（Referenced Link State ID）**和引用链路状态类型**（Referenced Link State Type）**——如果包含这两项，将允许该前缀的附加信息包含在另一个 LSA 中。如果使用了这两个字段，它们将描述携带附加信息的 LSA 的链路状态 ID 和类型。所引用的 LSA 的通告路由器字段也必须与该 AS 外部 LSA 中通告路由器字段的值相匹配。如果带有外部路由标记，这些附加信息就和 OSPFv3 本身无关，它们可以在边界路由器之间穿越整个 OSPF 域。如果这个功能项没有使用，那么引用链路状

态类型字段就设置为全 0。

6. 链路 LSA

链路 LSA 只有用于两个直接相连的邻居之间的信息通信才是有意义的。如图 9-11 所示，图中显示了链路 LSA 的格式。每一个与路由器相连并属于同一个 OSPFv3 域的链路都要始发一个单独的链路 LSA，而且由于它属于链路本地扩散的范围，收到它的路由器从来不会把它转发到其他链路上去。

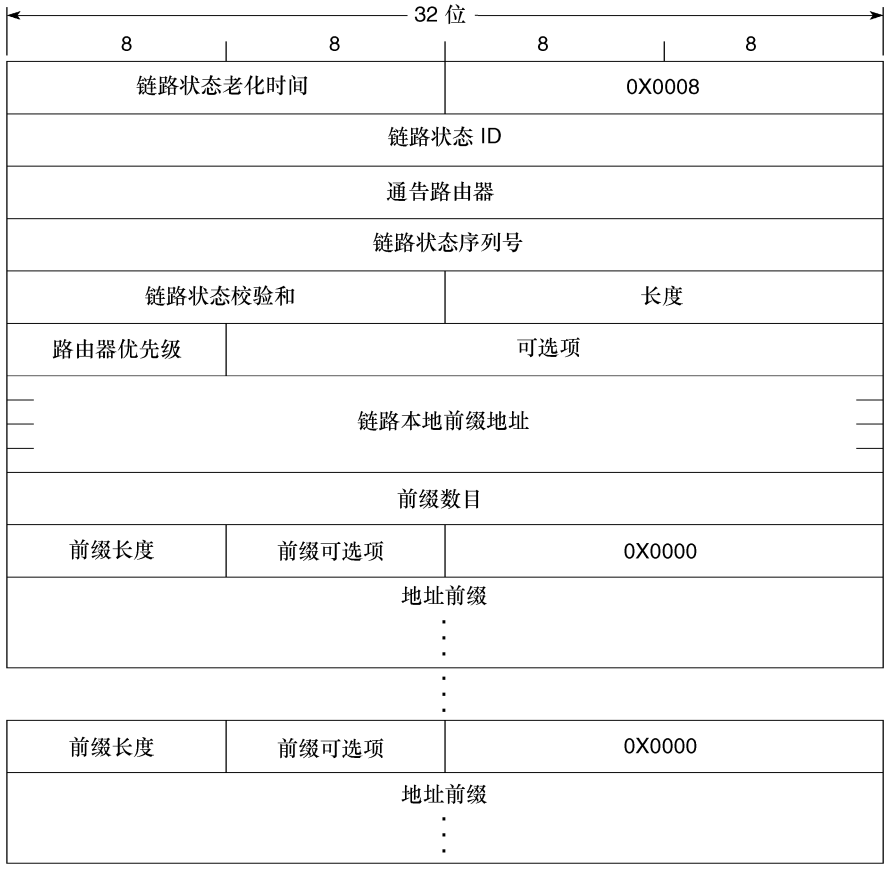


图 9-11　OSPFv3 链路 LSA 的格式

链路 LSA 具有以下 3 个功能：

（1）链路 LSA 向与这条链路相连的其他所有路由器提供始发路由器的链路本地地址。

（2）链路 LSA 提供了与这条链路有关的 IPv6 前缀列表。

（3）链路 LSA 提供了这条链路始发的网络 LSA 有关的一组可选位的集合。

- 路由器优先级（Router Priority）——标识了分配给始发路由器接口的路由器优先级。
- **可选项**——表示始发路由器应该在为该链路发起的网络 LSA 中设置的可选项位。这同样是一个 24 位的可选项字段，由 OSPFv3 Hello 和 DD 数据包，以及许多 OSPFv3 LSA 携带。可选项字段将在后面的章节中详细讲述。
- **链路本地前缀地址**——用来标识始发路由器与该链路相连的接口的 128 位链路本

地前缀。

- **前缀数目**——是指在 LSA 中包含的 IPv6 前缀的数量，这些前缀通过下面讲述的前缀长度、前缀可选项和地址前缀字段来描述。
- **前缀长度、前缀可选项和地址前缀**——用来描述一台或多台始发路由器上与链路相连的 IPv6 前缀。这个字段集合不仅用于链路 LSA，还用于区域内前缀、区域间前缀和 AS 外部 LSA。所通告的前缀的长度可以是 0~128 之间的任意值。当前缀不是一个 32 位的偶数倍时，它就会被填充 0 来达到地址前缀字段的 32 位边界。前缀长度字段指定了未填充的前缀的长度，以位数为单位。前缀可选项字段如图 9-12 所示，它指定了在路由选择计算期间可选的操作。

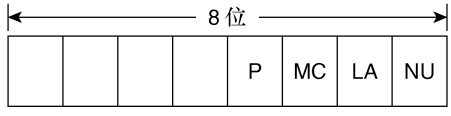


图 9-12 前缀可选项字段

- **传播位（Propagate，P）**——传播位设置在 NSSA 区域边界重新通告的 NSSA 区域前缀上。
- **多播位（MC）**——如果设置该位，表示应该在多播路由选择计算中包含该前缀。
- **本地地址位（Local Address，LA）**——如果设置该位，表示该前缀是一个通告路由器的接口地址。
- **无单播位（No Unicast，NU）**——如果设置该位，表示该前缀应该从单播路由选择计算中排除。

7. 区域内前缀 LSA

如图 9-13 所示，图中显示了区域内前缀 LSA 的格式。回忆前面章节有关这个 LSA 的讨论，在 OSPFv2 版本中由路由器与网络 LSA 携带的前缀信息在 OSPFv3 版本中则由区域内前缀 LSA 携带。前缀的变化——包括增加和删除——都不会以任何方式影响 SPF 树的分支变化。但是 OSPFv2 在路由器与网络 LSA 中通告前缀，无论前缀是否发生变化，都会引起区域内所有路由器进行 SPF 的计算。在 OSPFv3 中，当一条链路或它的前缀发生变化时，相连的路由器只会始发一台区域内前缀 LSA 在整个区域内扩散这个信息。这个 LSA 并不会触发 SPF 计算，收到该 LSA 的路由器只是简单地把这个新的前缀信息与始发路由器关联起来。在 OSPFv3 中，路由器与网络 LSA 只用来提供拓扑的信息服务。结果是，这个新的 LSA 将使 OSPFv3 在处理存在大量频繁变化前缀的网络方面具有更好的扩展性。

- **前缀数目**——表示该 LSA 中包含的前缀的数量。
- **引用链路状态 ID（Referenced Link State ID）**、引用链路状态类型（**Referenced Link State Type**）和引用通告路由器（**Reference Advertising Router**）——用来标识与所包含的前缀相关联的路由器或网络 LSA。

如果一个前缀与一个路由器 LSA 关联，那么引用链路状态类型为 1，引用链路状态 ID 为 0，而引用通告路由器的值则为始发路由器的 RID。如果一个前缀应该与一个网络 LSA 关

联，那么引用链路状态类型为 2，引用链路状态 ID 为该链路的 DR 的接口 ID[1]，而引用通告路由器的值则为 DR 路由器的 RID。



图 9-13 区域内前缀 LSA 的格式

正如前面讲述的链路 LSA 的前缀一样，这里的每一个前缀也都通过一组前缀长度、前缀可选项和地址前缀字段来描述。除了这 3 个字段还增加了一个度量字段，用来表示前缀的代价。

8. 可选项字段

可选项字段的长度为 24 位，它用来指定始发路由器的可选性能；在路由器、网络、区域间路由器，以及链路 LSA 中都可以携带这个字段。这个字段还可以在 Hello 与数据库描述数据包中携带。如图 9-14 所示，图中显示了可选项字段的格式。在撰写本书的时候，只有最右边的 6 位已经定义作为可选标记，并且大多数的标记是读者在学习 OSPFv2 时已经熟悉的。OSPFv3 路由器忽略未被识别的可选项标记。

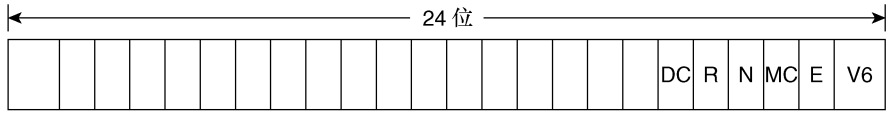


图 9-14 可选项字段的格式

- **DC 位**——表示具有支持按需电路的能力。
- **R 位**——指明始发路由器是否是一台有效的路由器。当该位清 0 时，传送该始发节

[1] 在这里和其他 OSPFv3 LSA 中使用的接口 ID 不应该与 IPv6 地址的 64 位接口 ID 部分产生混淆。这个字段用的是 32 位的值，使这个接口区别于始发路由器上其他的接口。在 RFC 2740 中建议使用 MIB－II If Index 作为接口 ID 的值。

点的路由将不被计算。所以 R 标记增加了一个类似于 IS-IS 协议中超载位（OL 位）的功能，超载位将在第 10 章中讲述。

- **N 位**——表示支持 NSSA LSA。
- **MC 位**——表示支持 MOSPF 协议。
- **E 位**——表示对于末梢区域的形成，AS 外部 LSA 是怎样进行扩散的。
- **V6 位**——如果该位清 0，表示应该从 IPv6 路由选择计算中排除该路由器或链路。

9.2 OSPFv3 的配置

OSPFv3 的配置选项和 OSPFv2 中的相关选项基本相同。在 OSPFv3 中也需要指定进程 ID 和区域。在进程中需要包括接口和地址。区域可以被定义为末梢、完全末梢或者 NSSA 区域。在区域之间可以对前缀进行汇总。OSPFv3 能够在按需电路和 NBMA 网络中配置。OSPFv3 的大多数配置都和 OSPFv2 中的配置一样；在某些情况下需要增加 IPv6 关键字，而且 IPv6 的前缀或地址代替 IPv4 中的子网或地址。本节讲述的案例研究将阐述 OSPFv3 与 OSPFv2 不同的配置，以及一些配置非常相似但需要着重强调的地方。

9.2.1 案例研究：OSPFv3 的基本配置

除了两点例外，OSPFv3 的配置和 OSPFv2 的配置基本是相同的。回忆一下第 8 章中在路由器和接口上进行 OSPFv2 配置的步骤。首先，要使用 **router ospf** 命令创建一个 OSPF 的进程。接着，要使用 **network area** 命令指定一个覆盖需要运行 OSPF 协议的接口地址的地址范围。如果一个接口配置了属于由 **network area** 命令指定的地址范围内的 IPv4 地址，那么这个接口将会运行 OSPF 协议。如果一个接口配置的 IPv4 地址不属于指定的地址范围，那么这个接口或这个地址（接口上配置有多个 IPv4 地址的情况）都将不会参与到 OSPFv2 进程的运行当中。支持 IPv6 的 OSPFv3 协议是通过在路由器的接口配置模式下指定一个 OSPF 进程 ID 和一个区域激活的。如果一个 OSPFv3 进程还没有创建，那么它会自动地创建。配置在路由器接口上的所有 IPv6 地址都被包含在这个指定的 OSPF 进程中运行。如图 9-15 所示，图中显示了一个运行 OSPFv3 协议的网络。

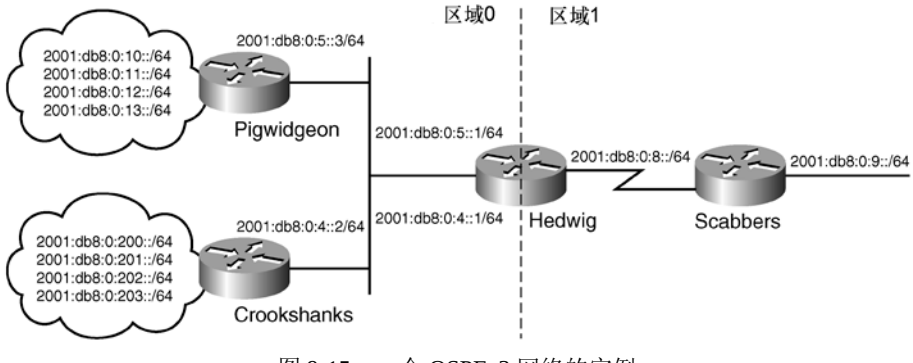


图 9-15 一个 OSPFv3 网络的实例

路由器 Hedwig 与 Pigwidgeon 的配置参见示例 9-1 与示例 9-2。

示例 9-1 使用接口配置模式的命令 ipv6 ospf 1 area 1 在路由器 Hedwig 上配置 OSPFv3

```
interface Serial 0/0
 ipv6 address 2001:db8:0:8::1/64
 ipv6 ospf 1 area 1
interface Ethernet0/0
 ipv6 address 2001:db8:0:4::1/64
 ipv6 ospf 1 area 0
```

示例 9-2 路由器 Pigwidgeon 上的 OSPFv3 配置

```
interface Ethernet 0/0
 ipv6 address 2001:db8:0:5::3/64
 ipv6 ospf 1 area 0
interface Serial 0/0
 ipv6 address 2001:db8:0:10::1/64
 ipv6 ospf 1 area 0
```

参见示例 9-3，使用命令 **ipv6 ospf area** 在路由器 Hedwig 的串行接口 Serial0/0 与以太网接口 Ethernet0/0 上启用了 OSPFv3 协议，并把路由器 Hedwig 的串行接口配置到区域 1，把以太网接口配置到区域 0 中，同时在该路由器上创建了一个 ID 为“1”的 OSPFv3 进程。在 OSPFv2 中，需要两个命令才能完成相同的任务：使用 **router ospf 1** 命令创建一个 OSPF 进程，然后在接口上使用 **network area** 命令启用 OSPFv2 协议。这里有一点要注意，虽然在多个接口上启用 OSPFv2 可以使用单条 **network area** 命令，但是在 OSPFv3 中必须在每一个需要运行 OSPFv3 协议的接口上配置 **ipv6 ospf area** 命令。

示例 9-3 命令 show ipv6 protocol 显示了该路由器上 IPv6 的 OSPF 进程 ID 和配置到每一个区域中的接口

```
Hedwig#show ipv6 protocol
IPv6 Routing Protocol is "connected"
IPv6 Routing Protocol is "static"
IPv6 Routing Protocol is "ospf 1"
  Interfaces (Area 0):
    Ethernet0/0
  Interfaces (Area 1):
    Serial0/0
  Redistribution:
    None
Hedwig#
```

一个接口上所有的 IPv6 地址都属于该接口上创建的 OSPF 进程。例如，在示例 9-4 中，路由器 Hedwig 的以太网接口增加了一个辅助的 IPv6 地址。

示例 9-4 路由器 Hedwig 的以太网接口 Ethernet0/0 上增加了一个辅助的 IPv6 地址。由于在这个接口上配置了 OSPFv3 协议，因此该接口上的这两个 IPv6 地址都会包含在 OSPFv3 进程中

```
interface Ethernet0/0
 ipv6 address 2001:db8:0:4::1/64
 ipv6 address 2001:db8:0:5::1/64
 ipv6 ospf 1 area 0
```

由于这个接口配置了 OSPFv3 协议，所以它的辅助地址将会自动地加入到 OSPFv3 的进

程中去，而不需要额外的 OSPFv3 命令来为路由选择进程增加新的地址块。接口上的 IPv6 地址不能有选择地加入到 OSPFv3 进程中。它要么在接口上配置 OSPFv3 使接口的所有地址都加入到 OSPFv3 进程；要么在接口上不配置 OSPFv3，从而没有任何地址加入到 OSPFv3 进程。

OSPFv3 路由器使用它们的链路本地地址作为 Hello 数据包的源地址。在 Hello 数据包中没有包含 IPv6 前缀的信息。在一条链路上能够配置多个 IPv6 地址。在单条链路上配置多个地址与 IPv4 中的配置不同，没有地址需要定义成辅助地址。在邻居之间除了链路本地地址外，即使它们没有共同的 IPv6 前缀，它们也可以建立邻接关系。这是 OSPFv3 与 IPv4 协议中的 OSPFv2 之间的不同之处。OSPFv2 邻居之间只有属于相同的 IP 子网，并且这个共同的子网都配置作为邻居接口的主 IP 地址时，它们之间才能形成邻接关系。如图 9-15 所示，路由器 Hedwig 在它的以太网接口上配置了 2001:db8:0:4::/64 和 2001:db8:0:5::/64。路由器 Pigwidgeon 的以太网接口配置了 2001:db8:0:5::/64，路由器 Crookshanks 的以太网接口配置了 2001:db8:0:4::/64。在示例 9-5 中，显示了 Crookshanks 同时与 Hedwig（10.1.1.1）和 Pigwidgeon（10.1.3.1）建立了邻接关系。其中，路由器 Pigwidgeon 是备份指定路由器。

示例 9-5 使用 show ipv6 ospf neighbor 显示了路由器 Crookshanks 的邻居是 FULL 状态（即完全邻接状态）的，尽管除了链路本地地址之外它们并没有共同的 IPv6 前缀

```
Crookshanks#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
10.1.1.1          1   FULL/DROTHER    00:00:30    3               Ethernet0/0
10.1.3.1          1   FULL/BDR        00:00:37    3               Ethernet0/0
Crookshanks#
```

两个邻居之间在建立起邻接关系之前，其他的一些参数都必须匹配。这些参数和 IPv4 中的参数是相同的：邻居之间必须使用相同的区域 ID，它们必须具有相同的 Hello 时间间隔和无效时间（dead time），并且它们必须具有相同的 E 位数值用来标识区域是否是一个末梢区域。一个 OSPFv3 数据包也必须具有与接收接口相同的接口 ID，否则这个 OSPFv3 数据包将被丢弃。在本章后面的 9.2.3 小节中将会讨论接口 ID。

OSPFv3 使用 32 位的数字作为路由器 ID。如果路由器配置了 IPv4，缺省情况下，选择 RID 的方式与 OSPFv2 中的方式相同。路由器 loopback 接口配置的最高 IPv4 地址将作为该路由器的 RID。如果没有配置 loopback 接口，则选择所有其他接口上配置的最高地址作为它的 RID。

IPv6 邻居总是通过它们的 RID 进行告知。这与 IPv4 不同，在 IPv4 中，点到点网络的邻居是通过 RID 告知的，而广播网络、NBMA 和点到多点网络的邻居都是通过它们的接口地址告知的。邻居的 ID 参见示例 9-5，它显示了路由器的 ID 是通过配置的 IPv4 地址获得的。

如果在网络中没有配置 IPv4 协议地址，你也不希望仅仅因为得到一个路由器 ID 而配置 IPv4 地址，那么在启动 OSPF 进程之前，就必须使用 IPv6 的 OSPF 路由选择进程命令 **router-id** 配置路由器 ID。

当在接口上配置了 OSPFv3 后，路由选择进程就创建了。像链路的成本代价等接口参数

可以在接口配置模式下更改，而全局参数则需要在 OSPF 进程模式下更改配置。

9.2.2 案例研究：末梢区域

命令 **ipv6 router ospf** 可以让路由器切换到全局的进程配置模式，这和 IPv4 中的 **router ospf** 命令是一样的。IPv6 中的配置方式与 IPv4 中的配置也是一样的。在 IPv6 中，可以用与 IPv4 中 OSPFv2 完全相同的方式利用 **area stub**、**area nssa** 和 **area stub no-summary** 命令来支持和配置末梢、NSSA 和完全末梢等区域。在路由器 Hedwig 和 Scabbers 上利用示例 9-6 和示例 9-7 中的配置可以把图 9-15 中的区域 1 配置成一个完全末梢区域。

示例 9-6 在路由器 Hedwig 上，区域 1 被配置成一个完全末梢区域，no-summary 只能配置在 ABR 上

```
ipv6 router ospf 1
 area 1 stub no-summary
```

示例 9-7 路由器 Scabbers 上的区域 1 也被配置成一个末梢区域，因为末梢区域中的所有路由器都必须被配置为一个末梢

```
ipv6 router ospf 1
 area 1 stub
```

在示例 9-8 中，显示了路由器 Scabbers 的区域 1 变成完全末梢之前时它的 IPv6 路由表，而在示例 9-9 中显示了 Scabbers 作为完全末梢区域的节点后的路由表。

示例 9-8 路由器 Scabbers 的 IPv6 路由表，它包含了 10 条 OSPF 条目，都是通过路由器 Hedwig 学习到的

```
Scabbers#show ipv6 route
IPv6 Routing Table - 16 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
OI  2001:DB8:0:4::/64 [110/74]
     via FE80::202:FDFF:FE5A:E40, Serial0/0
OI  2001:DB8:0:5::/64 [110/74]
     via FE80::202:FDFF:FE5A:E40, Serial0/0
C   2001:DB8:0:8::/64 [0/0]
     via ::, Serial0/0
L   2001:DB8:0:8::2/128 [0/0]
     via ::, Serial0/0
C   2001:DB8:0:9::/64 [0/0]
     via ::, Ethernet0/0
L   2001:DB8:0:9::2/128 [0/0]
     via ::, Ethernet0/0
OI  2001:DB8:0:10::/64 [110/138]
     via FE80::202:FDFF:FE5A:E40, Serial0/0
OI  2001:DB8:0:11::/64 [110/138]
     via FE80::202:FDFF:FE5A:E40, Serial0/0
OI  2001:DB8:0:12::/64 [110/138]
     via FE80::202:FDFF:FE5A:E40, Serial0/0
```

（待续）

```
OI   2001:DB8:0:13::/64 [110/138]
      via FE80::202:FDFF:FE5A:E40, Serial0/0
OI   2001:DB8:0:200::/64 [110/138]
      via FE80::202:FDFF:FE5A:E40, Serial0/0
OI   2001:DB8:0:201::/64 [110/138]
      via FE80::202:FDFF:FE5A:E40, Serial0/0
OI   2001:DB8:0:202::/64 [110/138]
      via FE80::202:FDFF:FE5A:E40, Serial0/0
OI   2001:DB8:0:203::/64 [110/138]
      via FE80::202:FDFF:FE5A:E40, Serial0/0
L    FE80::/10 [0/0]
      via ::, Null0
L    FF00::/8 [0/0]
      via ::, Null0
Scabbers#
```

示例 9-9 作为完全末梢区域的一个节点，路由器 Scabbers 现在只有一条 OSPF 条目了，即缺省路由

```
Scabbers#show ipv6 route
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
OI   ::/0 [110/65]
      via FE80::202:FDFF:FE5A:E40, Serial0/0
C    2001:DB8:0:8::/64 [0/0]
      via ::, Serial0/0
L    2001:DB8:0:8::2/128 [0/0]
      via ::, Serial0/0
C    2001:DB8:0:9::/64 [0/0]
      via ::, Ethernet0/0
L    2001:DB8:0:9::2/128 [0/0]
      via ::, Ethernet0/0
L    FE80::/10 [0/0]
      via ::, Null0
L    FF00::/8 [0/0]
      via ::, Null0
Scabbers#
```

9.2.3 案例研究：一条链路上的多个实例

在图 9-16 中，在以太网的网段上增加了一台新的路由器 Hermes。假设我们希望实现的设计是把路由器 Pigwidgeon 与 Hermes 之间的 OSPFv3 通信量，和路由器 Hedwig 与 Crookshanks 之间的通信量隔离开来。根据现有的配置，网络将会在 4 台路由器之间选择 DR 和 BDR，并且每一台路由器都和 DR 与 BDR 建立邻接关系。OPSFv3 的 Hello 数据包包含了一个实例 ID。这个实例 ID 可以用来把运行在同一个 LAN 上的两个 OSPF 进程分开。所收到的 Hello 数据包中的实例 ID 必须与接收该数据包的路由器接口上配置的实例 ID 相匹配，否则该数据包将被丢弃。如果实例 ID 的值没有另外指定，那么缺省地使用实例 ID 0。在路由器 Pigwidgeon 和 Hermes 配置与路由器 Hedwig 和 Crookshanks 不同的实例 ID，就可以建立预期设计的邻接关系。

路由器 Pigwidgeon 的配置更改如示例 9-10 所显示的那样。

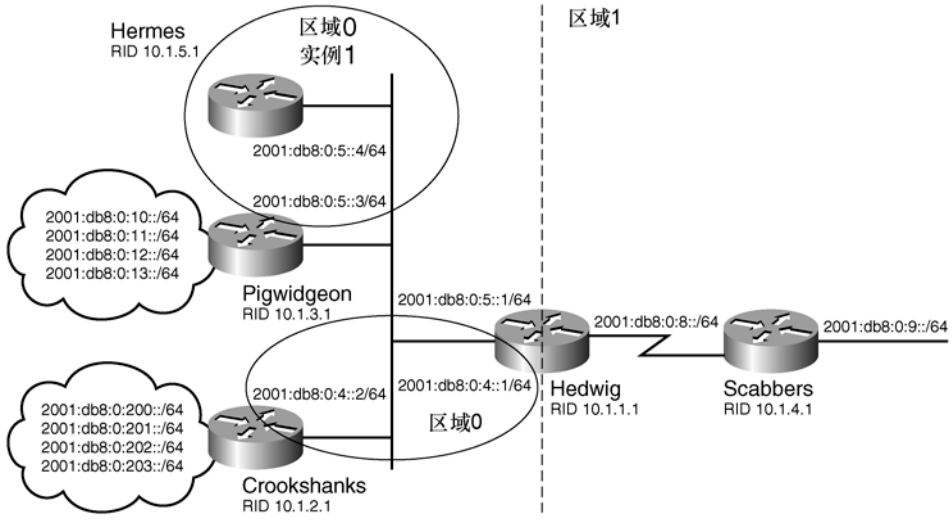


图 9-16 在图 9-15 的网络中增加一台新的路由器

实例 9-10 路由器 Pigwidgeon 的实例 ID 由缺省的实例 ID 0 更改为 1，这样在以太网上创建了一个明确的 OSPF 进程

```
interface Ethernet 0/0
 ipv6 address 2001:db8:0:5::3/64
 ipv6 ospf 1 area 0 instance 1
```

路由器 Pigwidgeon 的实例 ID 由缺省的 0 更改为 1。路由器 Hedwig 和 Crookshanks 继续使用缺省值 0 作为它们的实例 ID。可以使用与路由器 Pigwidgeon 相似的配置将路由器 Hermes 的实例 ID 改为 1。参见示例 9-11，可以看到路由器 Hermes 的 Ethernet0/0 接口运行的 IPv6 OSPF 配置，其中只有路由器 Pigwidgeon（10.1.3.1）是建立邻接的。

虽然在一台路由器上可以运行多个 OSPFv3 进程，但是在一个接口上只能运行单个进程或实例。

实例 9-11 路由器 Hermes 的 IPv6 OSPF 以太网接口的配置显示了 Pigwidgeon 是建立邻接的，因为路由器 Pigwidgeon 是这个以太网段上除 Hermes 外惟一使用实例 ID 为 1 的路由器

```
Hermes#show ipv6 ospf interface ethernet 0/0
Ethernet0/0 is up, line protocol is up
  Link Local Address FE80::206:28FF:FEB6:5BC0, Interface ID 3
  Area 0, Process ID 1, Instance ID 1, Router ID 10.1.5.1
  Network Type BROADCAST, Cost: 10
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 10.1.5.1, local address FE80::206:28FF:FEB6:5BC0
  Backup Designated router (ID) 10.1.3.1, local address FE80::201:42FF:FE79:E500
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:09
  Index 1/1/1, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 4
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 10.1.3.1  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
Hermes#
```

9.2.4 案例研究：运行在 NBMA 网络上的 OSPFv3 配置

对于 NBMA 网络，OSPFv3 和 OSPFv2 的配置选项是相同的。也就是说，从 OSPF 的角度来看，NBMA 网络可以保留 NBMA 的配置，或者当作广播网络或点到多点的网络配置，或者也可以当作使用子接口配置的点到点网络配置。点到点链路可以直接进行配置，其配置方法与通过串行链路直接相连的路由器的配置相同。在 9.2.1 小节中的路由器 Hedwig 和 Scabbers 就是配置成直接相连的串行链路。如果这两台路由器是通过点到点的帧中继子接口相连的，那么相应的 OSPFv3 的配置应该是一样的。在本小节中将会讲述 NBMA、广播网和点到多点网络。

在 OSPFv3 能够学习到相关的邻居之前，OSPF 用到的邻居地址必须先与穿过 NBMA 网络的特定虚电路相关联。建立帧中继映射来标识远程的 IPv6 地址映射到与本地路由器相连的某条 PVC 电路上[1]。由于 IPv6 可以在单个接口上配置多个地址，那么对于每一条 PVC 电路都需要多个 **map** 语句。所有的 OSPFv3 数据包都使用链路本地地址作为源地址。对于单播的 OSPFv3 数据包来说，也使用链路本地地址作为目的地址和在路由选择转发时的下一跳地址。因此，至少链路本地地址必须被映射。如图 9-17 所示，图中显示了一个由 4 台相连的路由器组成的帧中继网络。

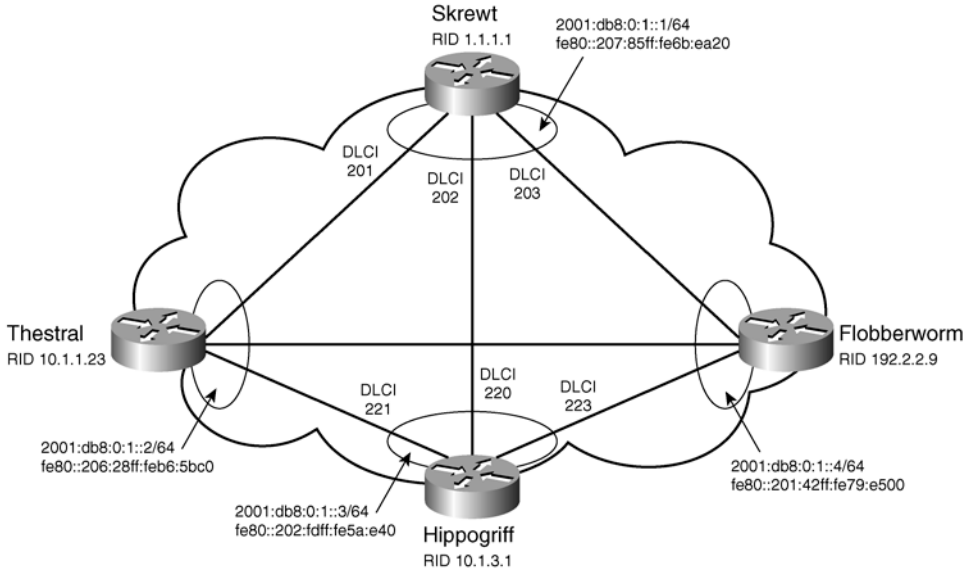


图 9-17 在这个 IPv6 的帧中继网络上，配置 OSPFv3 的几个选项

配置 OSPFv3 路由器的第一种方法是手工配置 OSPFv3 邻居。首先远程路由器的链路本地地址映射到正确的 DLCI 上，接着定义 IPv6 OSPFv3 的邻居，这两个步骤都是在接口配置模式下完成的。示例 9-12 中显示了路由器 Skrewt 的配置。

[1] IPv4 使用帧中继反向 ARP 来动态地映射地址。在撰写本书时，IOS 软件还不支持动态地映射地址到帧中继 PVC 电路的 IPv6 功能。

示例 9-12 在路由器 Skrewt 上手工配置 OSPFv3 邻居

```
interface Serial 0/0
 encapsulation frame-relay
 ipv6 address 2001:DB8:0:1::1/64
 ipv6 ospf 1 area 1
 frame-relay map ipv6 fe80::206:28ff:feb6:5bc0 201
 frame-relay map ipv6 fe80::202:fdff:fe5a:e40 202
 frame-relay map ipv6 fe80::201:42ff:fe79:e500 203
 ipv6 ospf neighbor fe80::206:28ff:feb6:5bc0 priority 5
 ipv6 ospf neighbor fe80::202:fdff:fe5a:e40 priority 10
 ipv6 ospf neighbor fe80::201:42ff:fe79:e500
```

这里请注意，在 **frame-relay map** 语句和 **ospf neighbor** 语句中使用的是本地链路地址。通过在命令 **ipv6 ospf neighbor** 中使用可选的关键字可以指定优先级，以便指出哪一台路由器将成为 DR 和 BDR。

配置 OSPFv3 路由器的另一种方法是，OSPFv3 动态地发现它的邻居。这种配置方法需要两个步骤：首先使用 **ipv6 ospf network broadcast** 命令将一个 OSPF 网络定义成广播网络；然后在 **frame-relay map** 语句使用关键字 **broadcast** 指出在 PVC 电路上进行广播转发。路由器 Skrewt 的配置更改参见示例 9-13。

示例 9-13 路由器 Skrewt 被配置为广播型的 PVC 电路，运行在帧中继链路上的 OSPFv3 网络也被配置成为一个广播型网络

```
interface Serial 0/0
 encapsulation frame-relay
 ipv6 address 2001:DB8:0:1::1/64
 ipv6 ospf network broadcast
 ipv6 ospf 1 area 1
 ipv6 ospf priority 20
 frame-relay map ipv6 fe80::206:28ff:feb6:5bc0 201 broadcast
 frame-relay map ipv6 fe80::202:fdff:fe5a:e40 202 broadcast
 frame-relay map ipv6 fe80::201:42ff:fe79:e500 203 broadcast
```

在上面多路访问网络中其他路由器的配置也是类似的。但注意不再有任何需要定义 IPv6 OSPF 邻居了。如果像下面的 NBMA 网络那样并不是全连接的网络结构（每一台路由器与其他的任何一台路由器之间都有 PVC 电路相连），那么 DR 和 BDR 的选择必须谨慎。DR 和 BDR 路由器和其他的任何一台路由器都必须具有完整的虚电路连接，只有这样才能建立起正确的邻接关系。利用接口配置模式命令 **ipv6 ospf priority** 可以在你希望作为 DR 的路由器上指定一个较高的优先级。指定 DR 的方法我们已经在 8.2.12 小节中讲述。参见示例 9-14，显示了路由器 Skrewt 的串行接口 Serial 0/0 的 IPv6 OSPF 配置。

示例 9-14 使用 **show ipv6 ospf interface** 命令可以显示路由器接口的 IPv6 OSPFv3 配置

```
Skrewt#show ipv6 ospf interface serial 0/0
Serial0/0 is up, line protocol is up
  Link Local Address FE80::207:85FF:FE6B:EA20, Interface ID 4
  Area 1, Process ID 1, Instance ID 0, Router ID 1.1.1.1
  Network Type BROADCAST, Cost: 64
  Transmit Delay is 1 sec, State DR, Priority 20
  Designated Router (ID) 1.1.1.1, local address FE80::207:85FF:FE6B:EA20
  Backup Designated router (ID) 10.1.3.1, local address FE80::202:FDFF:FE5A:E40
```

（待续）

```
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:08
  Index 1/1/1, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 1, maximum is 4
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 3, Adjacent neighbor count is 3
    Adjacent with neighbor 10.1.1.23
    Adjacent with neighbor 10.1.3.1  (Backup Designated Router)
    Adjacent with neighbor 192.2.2.9
  Suppress hello for 0 neighbor(s)
Skrewt#
```

请注意，这里的网络类型是 BROADCAST（广播型）。读者也可以注意到，它的邻居数目是 3 个，并存在 3 个邻接关系。这台路由器是 DR 路由器，因此它和网络上其他所有的路由器都建立了邻接关系。那些不是 DR 或 BDR 的邻居也具有 3 个邻居，但只有两个邻接关系，这是因为广播型网络上的路由器只需要与 DR 和 BDR 之间建立邻接关系。例如路由器 Flobberworm，参见示例 9-15。

示例 9-15 在广播型网络上的 4 台路由器中，不是 DR 或 BDR 的路由器将拥有 3 个邻居，但只具有两个完全的邻接关系

```
Flobberworm#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
1.1.1.1          20   FULL/DR         00:00:36    3               Ethernet0/0
10.1.1.23         1   2WAY/DROTHER    00:00:36    3               Ethernet0/0
10.1.3.1         15   FULL/BDR        00:00:37    3               Ethernet0/0
Flobberworm#
```

为了避免进行 DR/BDR 的选举，可以将 OSPF 网络的类型改为点到多点的网络类型。参见示例 9-16，显示了路由器 Skrewt 更改后的配置。

示例 9-16 路由器 Skrewt 的配置显示 PVC 电路仍然为广播型的，但 OSPFv3 的网络类型已经改变为点到多点类型

```
Interface Serial 0/0
 encapsulation frame-relay
 ipv6 address 2001:DB8:0:1::1/64
 ipv6 ospf network point-to-multipoint
 ipv6 ospf 1 area 1
 ipv6 ospf priority 20
 frame-relay map ipv6 fe80::206:28ff:feb6:5bc0 201 broadcast
 frame-relay map ipv6 fe80::202:fdff:fe5a:e40 202 broadcast
 frame-relay map ipv6 fe80::201:42ff:fe79:e500 203 broadcast
```

参见示例 9-17，路由器 Skrewt 的接口配置显示了点到多点网络的缺省计时器与广播型网络的缺省计时器有所不同。点到多点网络上的 Hello 数据包是每 30s 发送一次，而广播型网络上的 Hello 数据包是每 10s 发送一次。另外，OSPFv3 接口配置也没有任何 DR，路由器和所有的 3 台邻居路由器都建立了邻接关系。

示例 9-17 使用 show ipv6 ospf interface 命令可以显示一个 OSPF 点到多点网络上 IPv6 OSPF 接口的配置

```
Skrewt#show ipv6 ospf interface serial 0/0
Serial0/0 is up, line protocol is up
```

（待续）

```
  Link Local Address FE80::207:85FF:FE6B:EA20, Interface ID 4
  Area 1, Process ID 1, Instance ID 0, Router ID 1.1.1.1
  Network Type POINT_TO_MULTIPOINT, Cost: 64
  Transmit Delay is 1 sec, State POINT_TO_MULTIPOINT,
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
    Hello due in 00:00:13
  Index 1/1/1, flood queue length 0
  Next 0x0(0)/0x0(0)/0x0(0)
  Last flood scan length is 2, maximum is 4
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 3, Adjacent neighbor count is 3
    Adjacent with neighbor 10.1.1.23
    Adjacent with neighbor 10.1.3.1
    Adjacent with neighbor 192.2.2.9
  Suppress hello for 0 neighbor(s)
Skrewt#
```

将一个 NBMA 网络配置成 OSPF 点到多点网络不需要 PVC 电路具有全连接。如图 9-18 所示，图中显示了一些 PVC 电路已经从图 9-17 的网络中去掉。在示例 9-18 中，配置 DLCI 202 的 **map** 语句也从路由器 Skrewt 上去掉了。

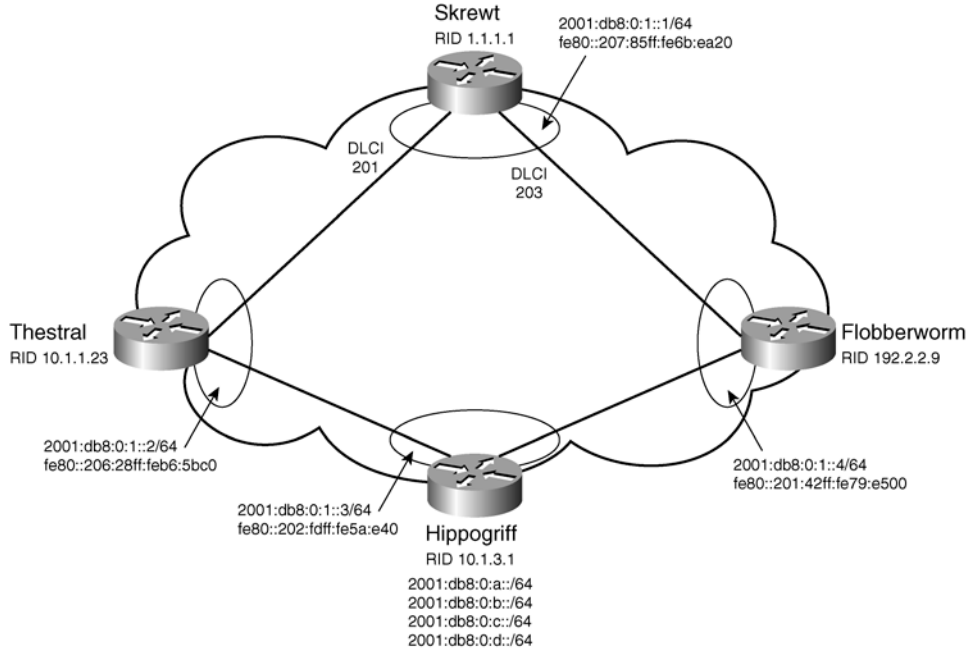


图 9-18 去掉一些 PVC 电路的图 9-17 的网络

示例 9-18 路由器 Skrewt 的帧中继配置映射 IPv6 地址到剩下的两条 PVC 电路上

```
interface Serial0/0
 no ip address
 encapsulation frame-relay
 ipv6 address 2001:DB8:0:1::1/64
 ipv6 ospf network point-to-multipoint
 ipv6 ospf priority 20
 ipv6 ospf 1 area 1
 frame-relay map ipv6 FE80::201:42FF:FE79:E500 203 broadcast
 frame-relay map ipv6 FE80::206:28FF:FEB6:5BC0 201 broadcast
```

正如示例 9-19 中所显示的，在路由器 Skrewt 的 IPv6 路由表中可以看到，对于 Skrewt 来说，路由器 Hippogriff 和它所学到的 IPv6 前缀都是可访问的。路由器 Skrewt 和 Hippogriff 都与路由器 Thestral 和 Flobberworm 形成了邻接关系。除了由 Hippogriff 通告的 IPv6 前缀，Hippogriff 的串行接口 Serial0/0 的 IPv6 地址（2001:db8:0:1::3）是通过路由器 Skrewt 的串行接口 Serial0/0 学习到的，它的下一跳链路本地地址是 FE80::206:28FF:FEB6:5BC0 和 FE80::201:42FF:FE79:E500。

示例 9-19 路由器 Skrewt 的路由表显示，通过在点到多点网络上与路由器 Skrewt 有 PVC 电路连接，路由器 Skrewt 仍然可以访问那些与它没有 PVC 连接的路由器

```
Skrewt#show ipv6 route
IPv6 Routing Table - 16 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
C   2001:DB8:0:1::/64 [0/0]
     via ::, Serial0/0
L   2001:DB8:0:1::1/128 [0/0]
     via ::, Serial0/0
O   2001:DB8:0:1::2/128 [110/64]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
O   2001:DB8:0:1::3/128 [110/128]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
     via FE80::201:42FF:FE79:E500, Serial0/0
O   2001:DB8:0:1::4/128 [110/64]
     via FE80::201:42FF:FE79:E500, Serial0/0
O   2001:DB8:0:5::/64 [110/74]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
O   2001:DB8:0:A::/64 [110/138]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
     via FE80::201:42FF:FE79:E500, Serial0/0
O   2001:DB8:0:B::/64 [110/138]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
     via FE80::201:42FF:FE79:E500, Serial0/0
O   2001:DB8:0:C::/64 [110/138]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
     via FE80::201:42FF:FE79:E500, Serial0/0
O   2001:DB8:0:D::/64 [110/138]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
     via FE80::201:42FF:FE79:E500, Serial0/0
O   2001:DB8:0:50::/64 [110/74]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
O   2001:DB8:0:51::/64 [110/74]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
O   2001:DB8:0:52::/64 [110/74]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
O   2001:DB8:0:53::/64 [110/74]
     via FE80::206:28FF:FEB6:5BC0, Serial0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
Skrewt#
```

配置在帧中继点到多点接口上的 IPv6 地址 2001:db8:0:1::2/64、2001:db8:0:1::3/64 和 2001:db8:0:1::4/64，都是通过带有 128 位前缀长度的 OSPF 通告的。这些地址与路由器 Hippogriff 通告到 OSPF 里的其他 IPv6 前缀地址，都可以通过路由器 Skrewt 的两台邻接路由

器的链路本地地址到达。

9.3 OSPFv3 的故障诊断

IPv6 协议的 OSPFv3 的故障诊断所使用的方法实际上与 IPv4 协议中的 OSPFv2 是相同的。主要的不同之处是寻址：直接发送数据包到一个邻居时，OSPFv3 使用链路本地地址作为数据包的源地址和目的地址。

案例研究：帧中继的映射

回到上面图 9-17 的最初配置，路由器 Skrewt 和 Hippogriff 配置参见示例 9-20 和示例 9-21。

示例 9-20 路由器 Skrewt 最初帧中继映射的配置

```
interface Serial 0/0
 encapsulation frame-relay
 ipv6 address 2001:DB8:0:1::1/64
 ipv6 ospf network broadcast
 ipv6 ospf 1 area 1
 frame-relay map ipv6 2001:db8:0:1::2 201 broadcast
 frame-relay map ipv6 2001:db8:0:1::3  202 broadcast
 frame-relay map ipv6 2001:db8:0:1::4 203 broadcast
 ipv6 ospf 1 area 1
```

示例 9-21 路由器 Hippogriff 最初帧中继映射的配置

```
interface Serial0/0
 no ip address
 encapsulation frame-relay
 ipv6 address 2001:DB8:0:1::3/64
 ipv6 ospf network broadcast
 ipv6 ospf 1 area 1
 frame-relay map ipv6 2001:DB8:0:1::1 220 broadcast
 frame-relay map ipv6 2001:DB8:0:1::2 221 broadcast
 frame-relay map ipv6 2001:DB8:0:1::4 223 broadcast
```

其他路由器的配置也是相似的。

参见示例 9-22，显示出路由器 Skrewt 的 OSPFv3 邻居表中路由器之间没有形成邻接关系。

示例 9-22 路由器 Skrewt 没有和 DR 与 BDR 建立邻接关系

```
Skrewt#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
10.1.3.1          1   EXSTART/DR      00:00:34    4               Serial0/0
10.1.1.23         1   EXSTART/BDR     00:00:31    4               Serial0/0
192.2.2.9         1   2WAY/DROTHER    00:00:36    4               Serial0/0
Skrewt#
```

参见示例 9-23 所示，对 IPv6 OSPF 的 Hello 数据包与邻接关系进行调试，显示出可以收到 Hello 数据包，并建立了双向通信状态。同时可以看到也选举出了 DR 和 BDR 路由器。数据库同步尝试完成与 DR 和 BDR 之间的邻接。虽然路由器 Skrewt 能够连续地收到来自可能的 DR 路由器的 Hello 数据包，但是它尝试发送数据库描述数据包（DBD）到可能的 DR 路

由器时却没有收到任何确认。与 DR 和 BDR 路由器之间的邻居状态是 EXSTART，这表明正在建立主/从关联关系，并已经发送了一个初始的 DBD 数据包。

示例 9-23 使用命令 **debug ipv6 hello** 和 **debug ipv6 ospf adj** 显示了 Hello 数据包、双向通信连接建立、DR/BDR 选举，以及正在发送 DBD 数据包

```
Skrewt#debug ipv6 ospf hello
Skrewt#debug ipv6 ospf adj
OSPFv3: Rcv hello from 10.1.1.23 area 1 from Serial0/0 FE80::202:FDFF:FE5A:E40
interface ID 4
OSPFv3: 2 Way Communication to 10.1.1.23 on Serial0/0, state 2WAY
OSPFv3: Neighbor change Event on interface Serial0/0
OSPFv3: DR/BDR election on Serial0/0
OSPFv3: Elect BDR 10.1.1.23
OSPFv3: Elect DR 10.1.1.23
        DR: 10.1.1.23 (Id)   BDR: 10.1.1.23 (Id)
OSPFv3: Send DBD to 10.1.1.23 on Serial0/0 seq 0xF78 opt 0x0013 flag 0x7 len 28
OSPFv3: Rcv hello from 10.1.3.1 area 1 from Serial0/0 FE80::201:42FF:FE79:E500
interface ID 4
OSPFv3: 2 Way Communication to 10.1.3.1 on Serial0/0, state 2WAY
OSPFv3: Neighbor change Event on interface Serial0/0
OSPFv3: DR/BDR election on Serial0/0
OSPFv3: Elect BDR 10.1.1.23
OSPFv3: Elect DR 10.1.3.1
        DR: 10.1.3.1 (Id)   BDR: 10.1.1.23 (Id)
OSPFv3: Send DBD to 10.1.3.1 on Serial0/0 seq 0x1C93 opt 0x0013 flag 0x7 len 28
OSPFv3: Remember old DR 10.1.1.23 (id)
OSPFv3: End of hello processing
OSPFv3: Send DBD to 10.1.1.23 on Serial0/0 seq 0xF78 opt 0x0013 flag 0x7 len 28
OSPFv3: Retransmitting DBD to 10.1.1.23 on Serial0/0 [1]
OSPFv3: Send DBD to 10.1.3.1 on Serial0/0 seq 0x1C93 opt 0x0013 flag 0x7 len 28
OSPFv3: Retransmitting DBD to 10.1.3.1 on Serial0/0 [1]
OSPFv3: Send DBD to 10.1.1.23 on Serial0/0 seq 0xF78 opt 0x0013 flag 0x7 len 28
OSPFv3: Retransmitting DBD to 10.1.1.23 on Serial0/0 [2]
OSPFv3: Rcv hello from 10.1.1.23 area 1 from Serial0/0 FE80::202:FDFF:FE5A:E40
interface ID 4
OSPFv3: End of hello processing
Skrewt#
```

读者在这里注意到并没有收到确认数据包。进一步使用有关 IPv6 OSPF 数据包的调试命令，来显示有关正在发送的 DBD 数据包更多的信息[1]，参见示例 9-24 所示。

示例 9-24 进一步使用 **debug ipv6 ospf packet** 调试命令显示出数据包封装失败

```
Skrewt#debug ipv6 ospf packet
OSPFv3: Send DBD to 10.1.1.23 on Serial0/0 seq 0x2422 opt 0x0013 flag 0x7 len 28
IPV6: source FE80::207:85FF:FE6B:EA20 (local)
      dest FE80::202:FDFF:FE5A:E40 (Serial0/0)
      traffic class 224, flow 0x0, len 68+0, prot 89, hops 1, originating
 IPv6: Encapsulation failed
OSPFv3: Retransmitting DBD to 10.1.1.23 on Serial0/0 [4]
```

DBD 数据包和 Hello 数据包不同，它不是多播传送的。DBD 数据包是被发送到邻居路由器的 IPv6 地址的。回忆 OSPFv3 是使用链路本地地址进行包交换的，这可以通过 IPv6 数据包的调试命令输出看到。路由器 Skrewt 在接口 Serial0/0 上没有帧中继电路映射到地址 FE80::202:FDFF:FE5A:E40 上。这就是为什么会出现封装失败的原因。配置帧中继映射必须将邻居路由器的链路本地地址配置到本地 DLCI 上。

[1] 不建议在一个实际运行的网络中调试 IPv6 数据包的信息。

为了容易地获取 IPv6 接口的链路本地地址，可以使用命令 **show ipv6 interface serial0/0** 来查看，参见示例 9-25 所示。

示例 9-25 使用命令 **show ipv6 interface** 可以看到有关接口的 IPv6 信息

```
Skrewt#show ipv6 interface serial0/0
Serial0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::207:85FF:FE6B:EA20
  Global unicast address(es):
    2001:DB8:0:1::1, subnet is 2001:DB8:0:1::/64
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::5
    FF02::1:FF00:1
    FF02::1:FF6B:EA20
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
  ND DAD is not supported
  ND reachable time is 30000 milliseconds
  Hosts use stateless autoconfig for addresses
Skrewt#
```

路由器 Skrewt 和 Hippogriff 更改后的配置显示在示例 9-26 和示例 9-27 中。

示例 9-26 路由器 Skrewt 的帧中继映射配置更改为 IPv6 链路本地地址

```
interface Serial0/0
 encapsulation frame-relay
 ipv6 address 2001:DB8:0:1::1/64
 ipv6 ospf network broadcast
 ipv6 ospf 1 area 1
 frame-relay map ipv6 fe80::206:28ff:feb6:5bc0 201 broadcast
 frame-relay map ipv6 fe80::202:fdff:fe5a:e40 202 broadcast
 frame-relay map ipv6 fe80::201:42ff:fe79:e500 203 broadcast
```

示例 9-27 路由器 Hippogriff 的帧中继映射配置更改为 IPv6 链路本地地址

```
interface Serial0/0
 no ip address
 encapsulation frame-relay
 ipv6 address 2001:DB8:0:1::3/64
 ipv6 ospf network broadcast
 ipv6 ospf 1 area 1
frame-relay map ipv6 fe80::207:85ff:fe6b:ea20 220 broadcast
frame-relay map ipv6 fe80::206:28ff:feb6:5bc0 221 broadcast
frame-relay map ipv6 fe80::201:42ff:fe79:e500 223 broadcast
```

必须映射到 DLCI 上的 IPv6 地址是邻居的链路本地地址。参见示例 9-28 显示了路由器 Skrewt 正确的 IPv6 OSPFv3 邻居表。

示例 9-28 在将邻居的链路本地地址映射到 DLCI 后，路由器 Skrewt 与 DR 和 BDR 路由器建立了完整的邻接关系

```
Skrewt#show ipv6 ospf neighbor

Neighbor ID     Pri   State           Dead Time   Interface ID    Interface
192.2.2.9         1   FULL/DR         00:00:32    4               Serial0/0
10.1.1.23         1   2WAY/DROTHER    00:00:37    4               Serial0/0
10.1.3.1          1   FULL/BDR        00:00:38    4               Serial0/0
Skrewt#
```

9.4 展 望

我们用了两章的篇幅——其中一章篇幅还很长——描述了 OSPF 的主要内容。OSPF 协议不仅是众所周知的链路状态路由选择协议，可能还是在 IP 路由选择协议中使用最为广泛的协议了。在下面的一章中，我们将讲述一个不太知名的链路状态协议：IS-IS 协议。一些人认为它是一个“比较另类”的协议，但是实际上读者将会看到它相对于 OSPF 协议来说简单多了。

9.5 总结表：第 9 章命令总结

命令	描述
area *area-id* **nssa** [**no-redistribution**] [**default-information-originate**[**metric**][**metric-type**]][**no-summary**]	定义一个非纯末梢区域（NSSA 区域），并提供一些相关参数的配置选项
area *area-id* **range** {*ipv6-prefix/prefix-length*} [**advertise**|**not-advertise**] [**cost** *cost*]	在指定的区域中创建一些具体路由的汇总前缀通告，并扩散到（或者并不扩散，这依赖于选项 **advertise**|**not-advertise**）其他区域中去
area *area-id* **stub** [**no-summary**]	定义一个末梢区域或完全末梢区域
debug ipv6 ospf packet	提供路由器收到的每一个 IPv6 OSPF 数据包的调试信息
debug ipv6 ospf [**adj**|**ipsec**|**database-timer**|**flood** |**hello**|**lsa-generation**|**retransmission**]	提供具体类型的 OSPF 数据包的相关调试信息
ipv6 ospf *process-id* **area** *area-id* [**instance** *instance-id*]	创建一个 OSPFv3 进程（如果还没有创建的话），并在相应接口上运行 OSPF
ipv6 ospf neighbor *ipv6-address*[**priority** *number*] [**poll-interval** *seconds*][**cost** *number*][**database-filter all out**]	手工配置 OSPFv3 的邻居
ipv6 ospf network {**broadcast**|**non-broadcast**| **{point-to-multipoint**[**non-broadcast**]|**point-to-point**}}	配置 OSPF 的网络类型不同于给定传输介质支持的缺省类型
ipv6 ospf priority *number-value*	配置给定路由器的 OSPFv3 的优先级，用于 DR 和 BDR 路由器的选举
ipv6 router ospf *process-id*	进入 OSPFv3 路由器配置模式
show ipv6 ospf [*process-id*][*area-id*]	显示有关 OSPFv3 进程的一般信息
show ipv6 ospf [*process-id*][*area-id*] **neighbor** [*interface-type interface-number*][*neighbor-id*][**detail**]	显示有关 OSPFv3 邻居的一般信息
show ipv6 ospf [*process-id*][*area-id*] **interface** [*interface-type interface-number*]	显示有关某个具体接口的 OSPFv3 信息
summary-prefix *prefix* [**not-advertise**|**tag** *tag-value*]	汇总重新分配到 OSPFv3 当中的前缀

9.6 推荐读物

Coltun R.，D.Ferguson 和 J.Moy 于 1999 年 12 月编写的“OSPF for IPv6”，即 RFC 2740。

9.7 复 习 题

1．OSPFv3 能够支持 IPv4 吗？

2．在 OSPFv3 中，能够在每条链路上支持多个实例是什么意思？ OSPFv3 消息头部的什么字段可以实现这个目的？

3．OSPFv3 数据包是怎么进行认证的？

4．OSPFv3 的下一报头号是什么？

5．两个保留的 OSPFv3 多播地址是什么？

6．OSPFv3 是否使用了与 OSPFv2 不同的消息类型？

7．在 OSPFv3 LSA 报头的链路状态类型字段中的起始 3 位的用途是什么？

8．什么泛洪扩散范围是 OSPFv3 支持的，但 OSPFv2 不支持？这个泛洪扩散范围使用什么 LSA？

9. 比较 OSPFv3 与 OSPFv2 中对应的路由器与网络 LSA，它们最显著的不同之处是什么？

10．区域内前缀 LSA 的用途是什么？

11．链路 LSA 的用途是什么？

9.8 配置练习

1．哪一个 OSPFv3 命令需要包含在 OSPFv3 进程的接口上已经配置的辅助 IPv6 前缀？

2．如果一台路由器配置的 IPv6 地址为 2001:db8:0:1::1/64，另一台路由器配置的 IPv6 地址为 2001:db8:0:100::1/126，那么这两台路由器能够形成邻接关系吗？

3．如图 9-19 所示，写出每一台路由器的 IPv6 OSPFv3 配置。

4．在图 9-19 中，配置区域 1 成为一个完全末梢区域，并将地址汇总到区域 0 中。

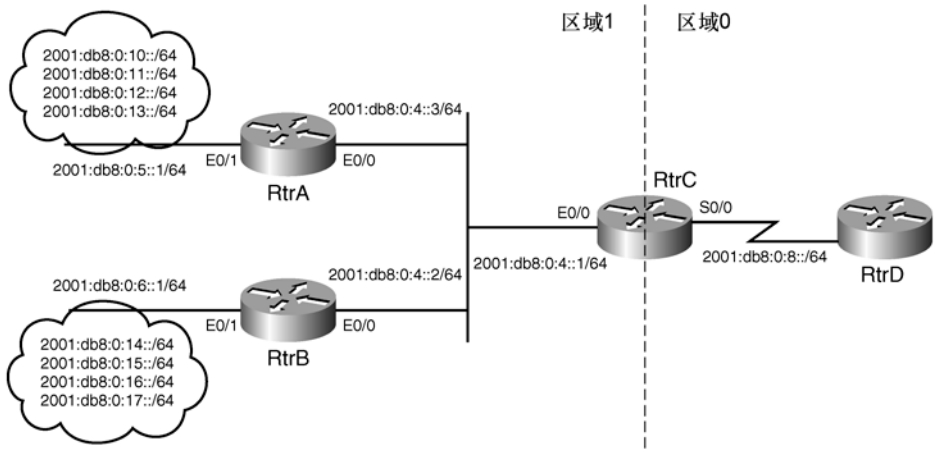


图 9-19 IPv6 OSPFv3 网络

本章包括以下主题：

- 集成 IS-IS 协议的基本原理与实现；
- 集成 IS-IS 协议的配置；
- 集成 IS-IS 协议的故障诊断。

第 10 章

集成 IS-IS 协议

每当人们提到链路状态协议和 IP 协议的术语时，大多数人会立即想到 OSPF 协议。一些人会说：“哦，是的，也有 IS-IS 协议，但是用得不多。”只有少数人会认真地考虑使用集成的 IS-IS 协议替代 OSPF 协议。但是，IS-IS 协议的用户虽然为数不多但毕竟还是存在，在一些网络——主要是一些 ISP 和运营商网络——运行 IS-IS 协议进行 IP 路由选择。

IS-IS 的意思是表示中间系统到中间系统，并且是为 ISO 无连接网络协议（ISO's Connectionless Network Protocol，CLNP）设计的路由选择协议。IS-IS 协议是由 ISO10589 定义和解释的。[1] 这个协议是由数字设备公司（DEC）的 DECnet PhaseV 第一次作为产品开发的。

ISO 发展 IS-IS 协议的时间和 IAB（Internet Architecture Board，Internet 体系结构委员会）发展 OSPF 协议的时间基本是同一时期，只是稍早或稍迟一点而已。并且提议采用 IS-IS 协议替代 OSPF 协议作为 TCP/IP 协议的路由选择协议。驱动该提议的观点是，即 TCP/IP 协议只是一个过渡的协议簇，并且最终会被 OSI 协议簇代替。向着 OSI 发展的推动力来自一些技术规范，例如 GOSIP 和 EPHOS 等。GOSIP（United States' Government Open Systems Interconnection Profile）是指美国政府开放系统互连框架文件，EPHOS（European Procurement Handbook for Open Systems）是指欧洲国家关于开放系统的采购手册。

[1] 国际标准化组织，“中间系统到中间系统域内路由选择信息交换协议，提供无连接模型网络服务（ISO 8473）”ISO/IEC 10589，1992 年。

为了支持从 TCP/IP 协议向 OSI 协议可预见的转换，又提出了一个扩展的 IS-IS 协议，[1] 称为集成 IS-IS 协议。提出集成 IS-IS 协议的目的是为了把它作为一个具有双重功能的 IS-IS 协议，即利用单个路由选择协议同时为 CLNS 协议 [2] 和 IP 协议提供路由选择的能力。这个协议可以设计用来在一个单纯的 CLNS 环境，一个单纯的 IP 环境，或者一个 CLNS/IP 的混和环境中运行。

说是画了一条战线可能有点过分夸张，但是至少是形成了两个鲜明的派别——ISO 的支持者和 OSPF 的支持者。读者应该阅读和对比一些经典的书籍，其中关于 OSPF 和 IS-IS 的讲述是很有启发作用的。这些书籍是由 Christian Huitema[3]——IAB 的前任主席和 Radia Perlman[4]——IS-IS 的首席设计师编写的。最后，IETF（Internet 工程任务组）采用了 OSPF 协议作为建议使用的 IGP 协议。技术上的不同的确会影响到决心，但是，有时也会存在行政上的因素。ISO 标准化是一个缓慢的处理过程，它一般需要 4 个步骤，并且依赖多个委员会最终投票表决同意。而 IETF 的处理过程却快捷得多。可以看出，通过 RFC 进程，发展 OSPF 协议要比采纳拘泥于形式化的 IS-IS 协议更有意义。

另一方面，非常小但却非常复杂的 IS-IS 用户群体证明 IS-IS 协议对于它的实施者与用户是有好处的。支持该协议的扩展特性很快就达成一致，因为 IS-IS 的用户大多数是高端的 ISP 和运营商，他们对他们的路由器厂商具有很高的要求。对于任何一个希望进入高性能网络市场的路由器厂商来说，它们都必须升级自己的产品以支持 IS-IS 协议。

不考虑行政上的争议，OSPF 工作组实际上学习和利用了很多 IS-IS 设计中的基本机制。从表面看来，OSPF 协议和 IS-IS 协议有很多共同的特性：

- 它们都维护一个链路状态数据库，并且这个数据库都是来自一个基于 Dijkstra 的 SPF 算法计算的一棵最短路径树；
- 它们都利用 Hello 数据包来形成和维护邻接关系；
- 它们都使用区域的概念来构成一个两级层次化的拓扑结构；
- 它们都具有在区域之间提供地址汇总的能力；
- 它们都是无类别路由选择协议；
- 它们都通过选取一个指定路由器来描述广播型网络；
- 它们都具有认证的能力。

除了这些类似之处外，它们也有明显的不同。本章将通过检查它们的这些不同之处开始讲述。本章只把集成 IS-IS 协议（以下简称为 IS-IS 协议）作为一个 IP 路由选择协议来讲述，只有在使用 IS-IS 协议为 IP 协议路由以及与 CLNS 协议有关的地方时才讲述 CLNS 协议。正如前面已经提到的，IS-IS 协议几乎是专门用在服务提供商网络中的，这样的网络往往对可靠性与可扩展性具体很高的要求；因此，本章另一个重点就是研究 IS-IS 协议在特大型网络中满足某些特殊要求的特性。

---

1 Ross Callon, “Use of OSI IS-IS for Routing in TCP/IP and Dual Environments,” RFC 1195, 1990 年 12 月。
2 无连接模型的网络服务——CLNP 的网络层协议。
3 Christian Huitema，Routing in the Internet，Prentice Hall PTR，Englewood Cliffs，NJ，1995 年。
4 Radia Perlman，Interconnections：Bridges and Routers，Addison-Wesley，Reading，MA，1992 年。

10.1 集成 IS-IS 协议的基本原理与实现

ISO 经常使用不同的术语来描述 IETF 所描述的相同概念实体，这种情况有时会引起混淆。ISO 的术语将在本节介绍和定义，但是在一般情况下，本章将使用本书其余章节使用的更类似于 IETF 的术语。[1] 有一些 ISO 的术语是非常基本的，因此，在具体讲述 IS-IS 协议的所有术语之前先介绍一下这些术语。

一台路由器就是一个中间系统（Intermediate System，IS），而一台主机就是一台端系统（End System，ES）。因此，提供主机与路由器之间通信的协议称为 ES-IS 协议，而被路由器用来进行相互宣告的协议（路由选择协议）称为 IS-IS 协议（如图 10-1 所示）。虽然 IP 协议使用路由器发现机制，例如 Proxy ARP、IRDP，或 IPv6 NDP，或者在主机上配置简单的缺省网关，但是 CLNP 协议还使用 ES-IS 协议来形成端系统和中间系统之间的邻接关系。对于 IP 协议来说，ES-IS 协议和 IS-IS 协议没有什么相关之处，因此本书将不包括有关 ES-IS 协议的讨论。

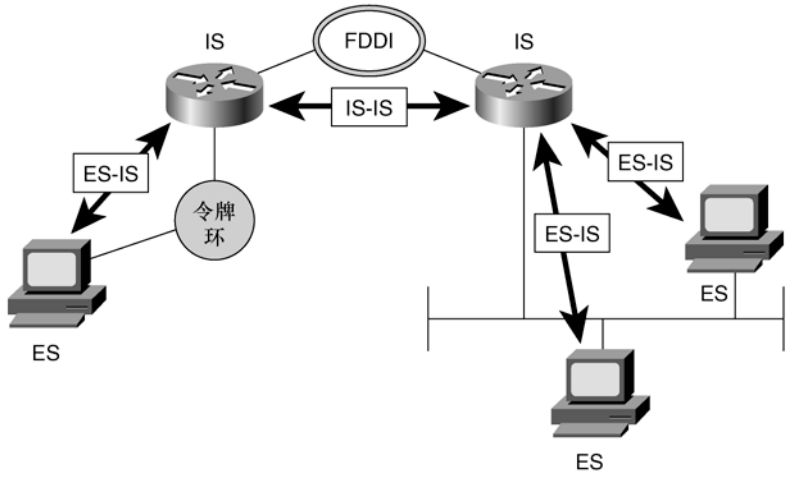


图 10-1 在 ISO 的术语中，主机是端系统，而路由器是中间系统

与一个子网相连的接口称为子网连接点（Subnetwork Point of Attachment，SNPA）。SNPA 有一些概念化，因为它实际上是定义了一个提供子网服务的“点”，而不是一个实际的物理接口。SNPA 的基本概念特性和子网本身的基本概念特性是相符合的，它可以由数据链路交换机相连的多个数据链路组成。

从一个节点的 OSI 层到另一个节点对等的 OSI 层的数据单元称为协议数据单元（Protocol Data Unit，PDU）。因此，一个帧就是一个数据链路 PDU（DLPDU），而一个数据包（或者分组）就是一个网络层协议数据单元（NPDU）。执行与 OSPF 协议中的 LSA 等价功能的数据单元称为链路状态 PDU（LSP）[2]。但与 LSA 不同，LSA 是封装在 OSPF 头部之后的，并且都

[1] 对于某些共同术语的 ISO/欧洲拼法，如“routeing”与“neighbour”是可以不用的。
[2] 在某些文档中，例如 RFC 1195，把 LSP 定义为一个链路状态数据包（Link State Packet）。

被封装在一个 IP 数据包内，而一个 LSP 本身就是一个数据包。

10.1.1 IS-IS 区域

虽然 IS-IS 协议和 OSPF 协议都使用区域的概念来创建两级的层次化网络拓扑结构，但是它们存在一个基本的不同之处，就是这两种协议在定义区域的方法上不一样。如图 10-2 所示，OSPF 协议的区域边界是通过路由器来划分的。某些接口属于一个区域，而另一些接口属于其他区域。如果一台 OSPF 路由器具有的接口分布在多于一个的区域里，那么这台路由器就是一台区域边界路由器，即 ABR 路由器。

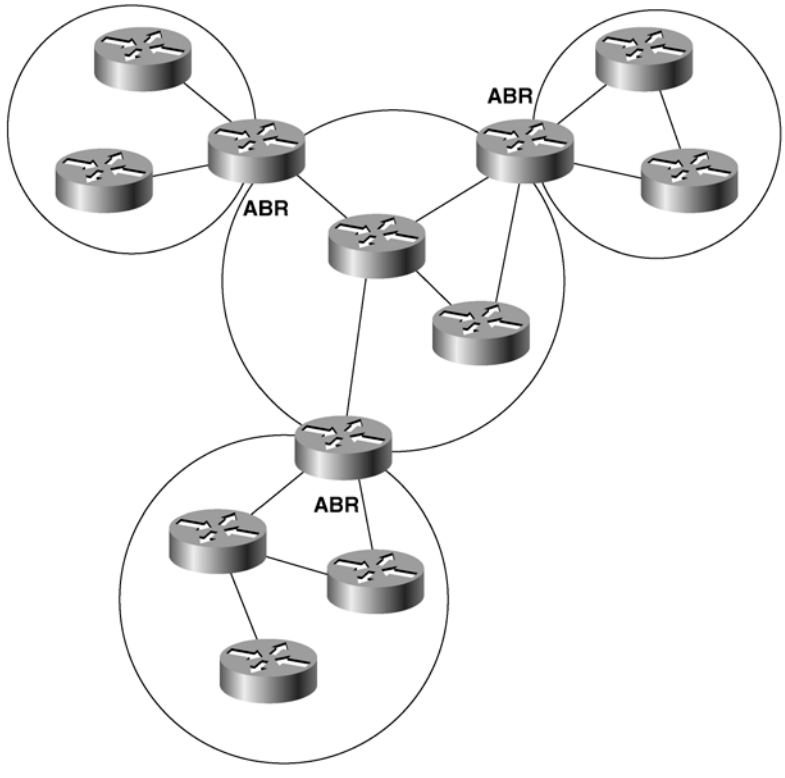


图 10-2 OSPF 区域边界是在路由器上，而与不同区域相连的路由器是 ABR 路由器

图 10-3 显示了与图 10-2 完全相同的网络拓扑，只是把它设计成了 IS-IS 区域。注意，所有的路由器都完全处在一个区域内部，并且区域的边界是在链路上，而不是在路由器上。IS-IS“骨干区域”是第 2 层区域，而非骨干区域是第 1 层区域。

一个中间系统可以是一台第 1 层的路由器（L1）、一台第 2 层的路由器（L2）或者两种类型皆是的路由器（L1/L2）。L1 路由器类似于 OSPF 协议中的非骨干内部路由器，而 L2 路由器类似于 OSPF 协议中的骨干路由器，同样地，L1/L2 路由器类似于 OSPF 协议中的 ABR 路由器。在图 10-3 中，L1/L2 路由器和 L1 路由器以及 L2 路由器相连。这些 L1/L2 路由器必须同时维护一个 L1 的链路状态数据库和一个 L2 的链路状态数据库，这种方式和 OSPF 协议中 ABR 路由器必须维护与之相连的每一个区域各自的数据库相类似。在 Cisco

路由器上，可以使用命令 **is-type** 来配置 L1-only、L2-only 或 L1/L2 类型，缺省情况下配置为 L1/L2。

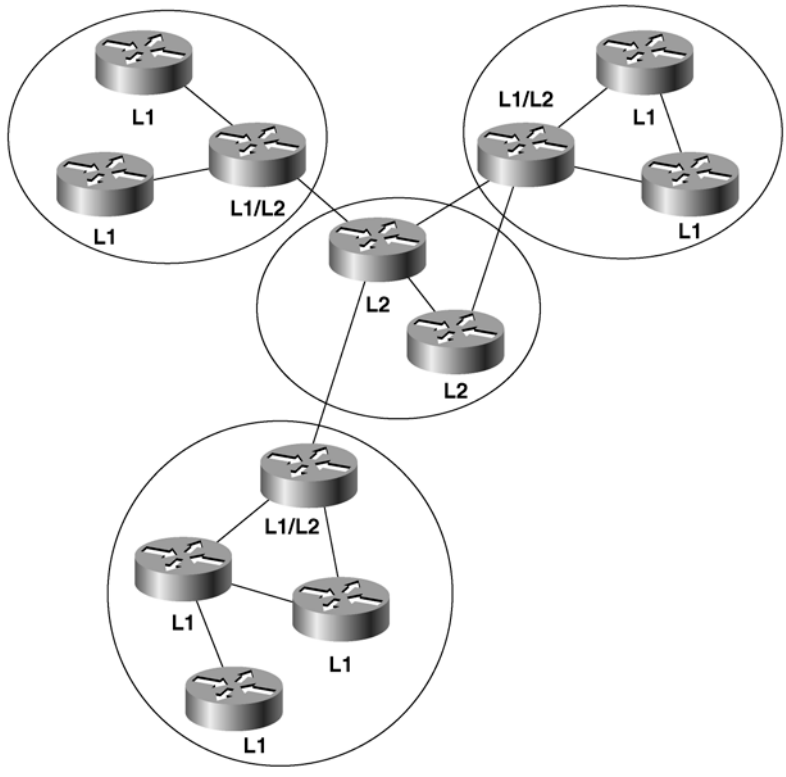


图 10-3 IS-IS 区域的边界在链路上，而与区域相连的路由器是第 2 层路由器

图 10-3 中的区域拓扑图对于比较 IS-IS 区域和 OSPF 区域是有帮助的，它也会带来一些误解；IS-IS 区域不像 OSPF 区域那样显得清楚明了。与其把 IS-IS 区域当作拓扑上的区域，倒不如把它们理解成一组邻接关系更好一些。一个邻接可以是 L1 邻接也可以是 L2 邻接。例如，一个特定的 L1 区域实际上是具有相同 AID 的路由器的一组相邻的 L1 邻接。L2 区域总是定义为一组 L2 邻接，理解这一点非常重要。一台 L1/L2 路由器是位于 L1 区域内的路由器，它可以同时具有一条或多条 L1 邻接和一条或多条 L2 邻接。一台 L2 路由器仅仅具有 L2 邻接关系。

在两个邻居之间同时存在一条 L1 邻接和一条 L2 邻接也是可能的。因为 IS-IS 区域被定义为一系列的邻接，在相同的两个邻居之间同时具有 L1 邻接和 L2 邻接意味着 IS-IS 区域是可以重叠的。与 OSPF 协议相比，IS-IS 的区域边界不像 OSPF 那么清楚。

与 OSPF 协议相同，区域间的通信量都必须经过 L2 区域，以便防止区域间路由选择环路。在一个区域内的每台 L1 路由器（包括区域内的 L1/L2 路由器）都会维护一个同样的链路状态数据库。但与 OSPF 协议中的 ABR 路由器不同，缺省情况下，L1/L2 路由器不需要通告 L2 类型的路由给 L1 类型的路由器。因此，一台 L1 路由器无法知晓它自己所在区域之外的目的路由。在这个意义上，一个 L1 区域就相当于 OSPF 协议中的完全末梢区域。为了路由转发数据包到其他的区域，L1 路由器必须转发数据包到一台 L1/L2 路由器上。当 L1/L2 路由器发送它的第 1 层 LSP 进入一个区域时，它将通过在 LSP 中设置一

个称为“区域关联位（Attached，ATT）”[1]的二进制位来通知其他 L1 路由器它可以到达其他的区域。

ISO 10589 描述了 IS-IS 协议路由器可以利用虚链路来修复被分段的区域，这和 OSPF 是一样的。但是这个特性在 Cisco 路由器和其他大多数厂商的路由器上都不支持，因而不在这里作进一步的描述。厂商不支持 IS-IS 虚链路的主要原因很简单，就是他们的客户不要求他们支持，这是 IS-IS 与 OSPF 在应用中的一个基本不同之处。OSPF 协议可以设置丰富的区域工具和特性，是企业网络选用的协议。另一方面，多区域的 IS-IS 运行更为复杂，因此很少在企业网中出现。运营商和 ISP 运行的是基于 BGP 协议的大型网络，他们的 IGP 协议主要用来寻找 BGP 会话的节点。因此，他们希望他们的 IGP 协议尽可能的简单——通常会把他们的整个路由选择域作为单个 IGP 区域。IS-IS 协议在很多方面无疑比 OSPF 协议更为简单，这对于“单个大型区域”这种类型的应用来说更具有可扩展性。因此，当读者在一个服务提供商的网络中碰到 IS-IS 协议，你通常会遇到单一的 L2 区域。[2]

由于一台 IS-IS 路由器可以完全地处于一个单一的区域内，因此区域 ID（或区域地址）将和整个路由器相关联，而不是和某一个接口相关联。IS-IS 协议的一个独特特性是，在缺省情况下一台路由器最多可以拥有 3 个区域地址，这在区域过渡期间是很有用的。利用 IOS 软件的命令 **max-area-addresses**，读者能够将区域地址的数目增加到最多 254 个。在 10.2.3 小节中演示了多个区域地址的使用。每台 IS-IS 路由器必须拥有一种在它所在的路由选择域内惟一地标识它本身的方法。这个惟一标识就是系统 ID 的功能，系统 ID（System ID）类似于 OSPF 协议中的路由器 ID。在一台 IS-IS 路由器上可以通过一个单一的地址同时定义区域 ID 和系统 ID，这个地址就是网络实体标题（Network Entity Title，NET）。

10.1.2 网络实体标题

即使 IS-IS 协议只用来为 TCP/IP 协议进行路由选择，它也依然是一个 ISO CLNP 协议。因此，IS-IS 协议对等体之间的通信数据包是 CLNS PDU；也就是说即使是在一个纯 IP 环境中，一台 IS-IS 路由器也必须有一个 ISO 地址。这个 ISO 地址是一个网络地址，称为网络实体标题（NET），并在 ISO 8348 中进行描述。[3] 一个 NET 地址的长度范围可以是 8～20 个八位组字节，并可以描述为区域 ID 和一台设备的系统 ID 两部分，如图 10-4 所示。

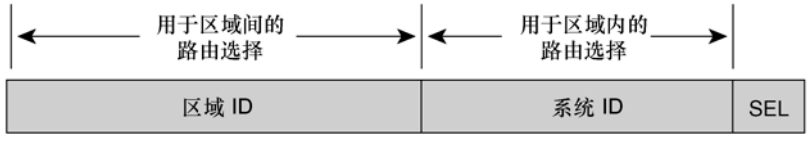


图 10-4 NET 地址指定了区域 ID 和一个 IS 或者 ES 的系统 ID

ISO 设计的 NET 地址可以在许多系统中做很多事情，这依赖于你个人的看法，要么认为这个地址的格式是非常灵活的和可扩展的，要么认为这个地址格式是一个麻烦的容易搞糊涂的变量字段。如图 10-5 所示，图中仅仅显示了一个 ISO NET 地址可能具有的多种格式中的 3

[1] 这里实际上是 4 个 ATT 位，并与不同的度量相关联。这些位的进一步解释在 10.1.4 小节中介绍。

[2] L2 区域用于为一个单独的区域提供必要的到区域外部的出口，新的区域可以作为 L1 类型，并简单地连接到已经存在的 L2 区域上。

[3] 国际标准化组织，“Network Service Definition Adden-dum 2：Network Layer Addressing”， ISO/IEC 8348/Add.2，1988 年。

种。虽然在每一个例子中系统 ID 前面的域是不同的，但是系统 ID 本身都是相同的。ISO 10589 指定了这个域的长度可以是从 1～8 个八位组字节，但是在一个路由选择域内的所有节点的系统 ID 必须使用相同的长度。实际上，这个系统 ID 的长度是 6 个八位组字节，[1] 并且经常是这台设备上的某个接口的 MAC 地址（Media Access Control，介质访问控制）。对于路由选择域内的每一个节点，这个系统 ID 必须是惟一的。

在图 10-5 的例子中还有一个需要注意的地方，就是 NSAP 选择符（SEL）。在所有的情况下，这个 1 个八位组字节的字段都被设置为 0x00。一个网络服务接入点（NSAP）所描述的都和某个节点在网络层上的一种特有服务相关联。因而，在一个 ISO 地址中，SEL 设置为大于 0x00 的某些值时，这个地址就是一台 NSAP 地址。这种情况和一个 IP 数据包内的 IP 目的地址与 IP 协议号的组合有些类似，它表明一台具体设备的 TCP/IP 协议栈的网络层上的一个具体服务。而在一个 ISO 地址的 SEL 设置为 0x00 时，这个地址就是一个 NET 地址，指明了某个节点网络层本身的地址。

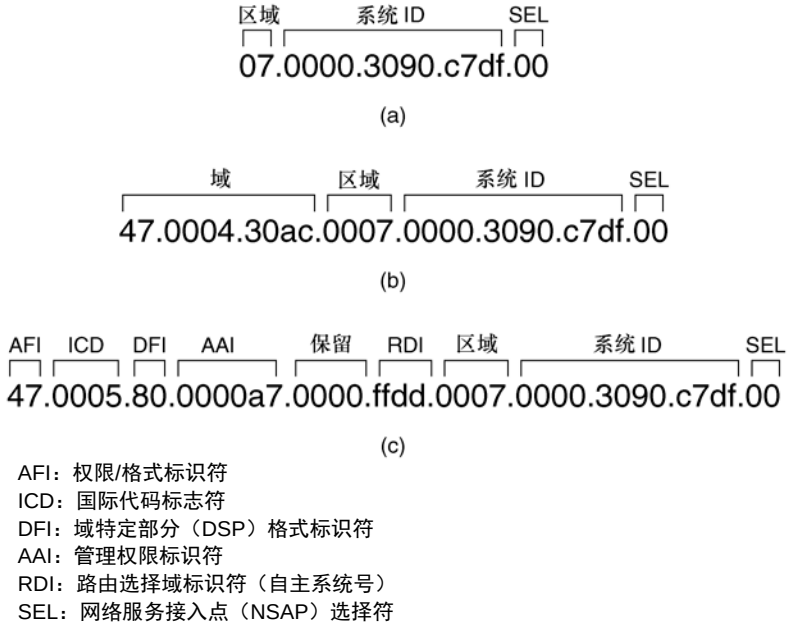


图 10-5　3 种 NET 格式：一个是简单的 8 个八位组字节区域 ID/系统 ID 格式（a）；一个是 OSI NSAP 格式（b）；还有一个是 GOSIP NSAP 格式[2]（c）

图 10-5 中只是显示了 NET 的多种格式，关于 NET 配置更详细的讨论已经超出了本书的讨论范围。如需要进一步的学习，RFC 1237 是一个不错的参考。[3] 在只有 IP 协议的环境中，NET 地址的设定可以基于某个标准，例如 GOSIP。如果你可以在一个纯 IP 环境中自由地选择任何 NET 地址的格式，那么将可以根据实际网络的需要选择最简单的格式。

无论是何种格式的地址，都需要满足下面的 3 个规则：

- NET 地址必须以一个单个八位组字节的域开始（例如，47.xxxx…）；

[1] Cisco 公司的 IS-IS 实现需要一个 6 个八位组字节的系统 ID。

[2] GOSIP Advanced Requirements Group（GOSIP 高级需求组），“Government Open Systems Interconnection Profile (GOSIP) Version 2.0 [Final Text]”，Federal Information Processing Standard（联邦信息处理标准）， U.S. Department of Commerce（美国商务部），National Institute of Stan-dards and Technology（美国国家标准和技术协会），1990 年 10 月。

[3] Richard Colella，Ella Gardner 和 Ross Callon，“Guidelines for OSI NSAP Allocation in the Internet”，RFC 1237，1991 年 7 月。

- NET 地址必须以一个单个八位组字节的域结束，并且应该设置为 0x00（…xxxx.00）。如果 SEL 是非零的，IS-IS 也会起作用，但是在一个 CLNP/IP 混和的路由器可能会出现一些问题；
- 在 Cisco 的路由器上，NET 地址的系统 ID 必须是 6 个八位组字节。

10.1.3 IS-IS 的功能结构

像 ISO 模型一样，之所以有一个分层的网络体系结构，其中有一个最主要的目的是为了使每一层的功能都独立于它下面的一层。例如，网络层必须适应大多数类型的数据链路或子网络。为了进一步满足这种适应性，网络层又由两个子层组成（如图 10-6 所示）。子网独立子层（subnetwork independent sublayer）为传输层提供了一致的和统一的网络服务；而子网依赖子层（subnetwork dependent sublayer）则为子网独立子层的需求而去存取数据链路层提供的服务。正如这两个命名所暗示的，子网依赖子层取决于数据链路的具体类型，而子网独立子层则能够独立于数据链路。

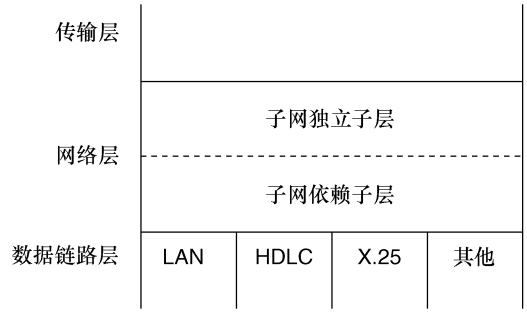


图 10-6 OSI 网络层由两个子层组成

网络层的结构是在 ISO 8648[1]中指定的，它实际上要比图 10-6 中显示的结构更为复杂。这里之所以提到这两个基本的子层，是因为在 ISO 10589 中对这些子层的功能架构中有关 IS-IS 的操作做了大量描述。

1. 依赖于子网的功能

依赖于子网的功能是指子网的功能依赖于它的下层；这是指为子网独立子层的功能层面隐藏掉不同种类数据链路（子网）的特征。下面的依赖于子网的功能对于路由选择是非常重要的：

- PDU 数据包的传送和接收是在一个具体相连的子网上；
- 通过 IS-IS 的 Hello PDU 数据包来发现邻居路由器并建立这个子网上的邻接关系；
- 邻接关系的维护；
- 链路的复用，或者说对于 OSI 进程转换为 OSI PDU 数据包，而对于 IP 进程转换为 IP 数据包。

（1）IS-IS 网络类型

相对于 OSPF 协议中定义的 4 种网络类型，IS-IS 协议只定义了两种类型：广播型子网和

[1] 国际标准化组织，“Internal Organisation of the Network Layer”，ISO 8648，1990 年。

点到点或一般拓扑子网。广播型子网的定义是和 OSPF 协议中的定义一样——就是支持多路广播的多路访问数据链路。而点到点子网（非广播子网）则可能是永久链路，例如 T1 链路，或者动态链接链路，例如 X.25 的 SVC 链路。

（2）邻居路由器和邻接关系

IS-IS 路由器是通过交换 IS-IS Hello PDU 数据包信息来发现邻居并形成邻接关系的。Hello 数据包每隔 10s 传送一次，在 Cisco 的路由器中，这个时间间隔可以基于每个接口通过命令 **isis hello-interval** 来改变。虽然 IS-IS Hello 数据包对于广播型子网和点到点子网略有差别，但是 Hello 数据包中包含的基本信息还是相同的，这将在 10.1.4 小节中讲述。一台 IS-IS 路由器使用它的 Hello PDU 数据包可以标识它本身和它的性能，以及描述发送该 Hello 数据包的接口的一些参数。如果两台邻居路由器关于它们各自的性能和接口的参数协商一致，那么它们就形成了邻接关系。然而，IS-IS 在形成邻接关系方面没有 OSPF 那样严格。在大多数实例中，一个邻居通告的特性如果其他邻居不支持的话不会阻止它们形成邻接关系；其中的特性可以被忽略。邻居之间甚至可以通告不同的 Hello 时间间隔。

对于第 1 层类型与第 2 层类型的邻居路由器，IS-IS 协议可以形成各自不同的邻接关系。L1-only 路由器可以和 L1 以及 L1/L2 邻居形成 L1 邻接关系，而 L2-only 路由器可以和 L2 以及 L1/L2 邻居形成 L2 邻接关系。L1/L2 路由器和它的邻居既可能形成 L1 邻接关系也可能形成 L2 邻接关系。但是，一台 L1-only 路由器和一台 L2-only 路由器不能形成一个邻接关系。正如前面所谈到的，缺省情况下，Cisco 路由器是 L1/L2 路由器。

当路由器的类型（L1-only、L2-only 或 L1/L2）影响所形成的邻接类型时，在两台邻居路由器上配置的区域 ID 也会对其产生影响。这可以应用以下的规则：

- 两台 L1-only 路由器只有在它们的 AID 匹配时才能形成一个 L1 邻接关系；
- 两台 L2-only 路由器即使它们的 AID 不同也能够形成一个 L2 邻接关系；
- 一台 L1-only 路由器和一台 L1/L2 路由器只有在它们的 AID 匹配时才能形成一个 L1 邻接关系；
- 一台 L2-only 路由器和一台 L1/L2 路由器即使在它们的 AID 不同时也能形成一个 L2 邻接关系；
- 如果两台 L1/L2 路由器的 AID 匹配，它们就可以同时形成 L1 和 L2 类型的邻接关系；
- 如果两台 L1/L2 路由器的 AID 不匹配，它们就只能形成 L2 类型的邻接关系。

一旦邻接关系建立成功，Hello 数据包将担当保活（keepalive）的功能。每一台路由器都在它的 Hello 数据包中发送一个抑制时间（hold time），用来通知它的邻居路由器在宣告这台路由器无效之前，应该等待多长的时间去侦听下一个 Hello 数据包。在 Cisco 路由器上，缺省的抑制时间是 Hello 时间间隔的 3 倍长，并且可以基于每接口通过命令 **isis hello-multiplier** 来改变。与 OSPF 的一个重要不同之处是，两个 IS-IS 邻居之间的 Hello 时间间隔与保持时间是不需要匹配的。每一台路由器都会认同它的邻居通告的保持时间。

OSPF 与 IS-IS 邻接的另一个有趣的不同之处是，两台路由器形成邻接关系的时候。OSPF 协议在这个地方有点混乱——两台路由器一旦建立了双向通信就认为形成了邻接关系，但直到数据库完成了同步才认为形成了完全邻接关系。而 IS-IS 协议在路由器之间能够交换 Hello 数据包时就认为它们形成了邻接关系。

如示例 10-1 所示，IS-IS 的邻居表可以通过命令 **show clns is-neighbors** 来查看。示例中显示的开始 4 列表明了每一台邻居路由器的系统 ID、与邻居相连的本地接口、邻接关系的状

态以及邻接关系的类型。这里的状态要么是 Init——表明邻居路由器是学习到了但是还没有形成邻接关系，要么是 Up——表明和邻居路由器成功建立了邻接关系。优先级是指在广播型网络上用来选举指定路由器的路由器优先级，这将在下面介绍。

示例 10-1 IS-IS 的邻居表可以通过命令 show clns is-neighbors 来显示

```
Brussels#show clns is-neighbors
System Id      Interface  State  Type  Priority  Circuit Id         Format
0000.0C04.DCC0 Se0        Up     L1    0         06                 Phase V
0000.0C04.DCC0 Et1        Up     L1    64        0000.0C76.5B7C.03  Phase V
0000.0C0A.2C51 Et0        Up     L2    64        0000.0C76.5B7C.02  Phase V
0000.0C0A.2AA9 Et0        Up     L1L2  64/64     0000.0C76.5B7C.02  Phase V
Brussels#
```

第 6 列显示的是电路 ID（Circuit ID），这是一个 1 个八位组字节的数字，路由器用它来惟一地标识 IS-IS 接口。如果该接口是和一个广播型多路访问网络相连的，那么这个电路 ID 是和该网络上的指定路由器的系统 ID 相连的，并把这个完全的数字称为 LAN ID。在这种用法中，更为正确的叫法应该把电路 ID 称为伪节点 ID（Pseudonode ID）。例如，在图 10-7 中，与接口 E0 相连的链路的 LAN ID 是 0000.0c76.5b7c.02。在这里，指定路由器的系统 ID 是 0000.0c76.5b7c，而伪节点 ID 是 02。

最后一列指出了邻接关系的格式。对于集成的 IS-IS 协议，这个格式将永远是 Phase V，用来说明是 OSI/DECnet Phase V。另外一个惟一的格式是 DECnet Phase IV。

（3）指定路由器

IS-IS 协议在一个广播型多路访问网络上选取指定路由器（更为准确地说，是一个指定 IS）的原因与 OSPF 协议相同。把网络本身看作是一台路由器或一个伪节点，要比局域网内的每一台路由器都要与该网络上相连的其余每台路由器形成一个邻接关系的方法好得多。包括指定路由器在内的每一台路由器都只需要通告单条链路到伪节点。指定路由器作为伪节点的代表也会通告一条链路到与之相连的所有路由器。

然而，与 OSPF 协议不同的是，与广播型多路访问网络相连的 IS-IS 路由器要与网络上的所有邻居建立邻接关系，而不仅仅是指定路由器。这和前面章节所讲述的是一致的，一旦可以交换 Hello 数据包就建立了 IS-IS 邻接关系。每一台路由器将以组播方式发送它的 LSP 数据包给所有的邻居路由器，并且指定路由器使用一个称为序列号 PDU（Sequence Number PDU，SNP）的数据包来确保 LSP 的泛洪是可靠的。这个可靠的泛洪过程和 SNP 数据包将在后面的“更新过程”部分中介绍。

IS-IS 协议指定路由器的选取过程非常简单。每一台 IS-IS 路由器接口都被指定一个 L1 类型的优先级和 L2 类型的优先级，它们的范围是 0～127。Cisco 路由器接口的优先级对于 L1 和 L2 类型的缺省值都是 64，并且可以通过命令 **isis priority** 来改变这个值。

路由器通过其每一个接口发送 Hello 数据包，并在 Hello 数据包中通告它的优先级——在 L1 类型的 Hello 数据包中通告 L1 类型的优先级，在 L2 类型的 Hello 数据包中通告 L2 类型的优先级。与 OSPF 不同的是，在 OSPF 中如果路由器的优先级是 0，那么它将没有资格成为一台指定路由器，而一台优先级为 0 的 IS-IS 路由器仅仅表示最低的优先级，但依然能够成为 DR 路由器。对于非广播型网络上的接口，不需要选举指定路由器，因此也将它们的优先级设置为 0（注意示例 10-1 中显示的串行接口的优先级）。拥有最高优先级的路由器将成为指定路由器。如果路由器的优先级相同，那么拥有在数值上具有最高的 MAC 地址的接口

的路由器将成为一台指定路由器。

与 L1 和 L2 类型的优先级相对应的是，需要在一个网络上为 L1 和 L2 分别选取单独的指定路由器。这种做法是必需的，因为在一个单一的局域网中存在着各自不同的 L1 和 L2 类型的邻接关系，如图 10-7 所示。由于一个接口对于每一层都具有单独不同的优先级，因此在同一个局域网上的 L1 类型的 DR 路由器和 L2 类型的 DR 路由器可能是同一台路由器，也可能不是。

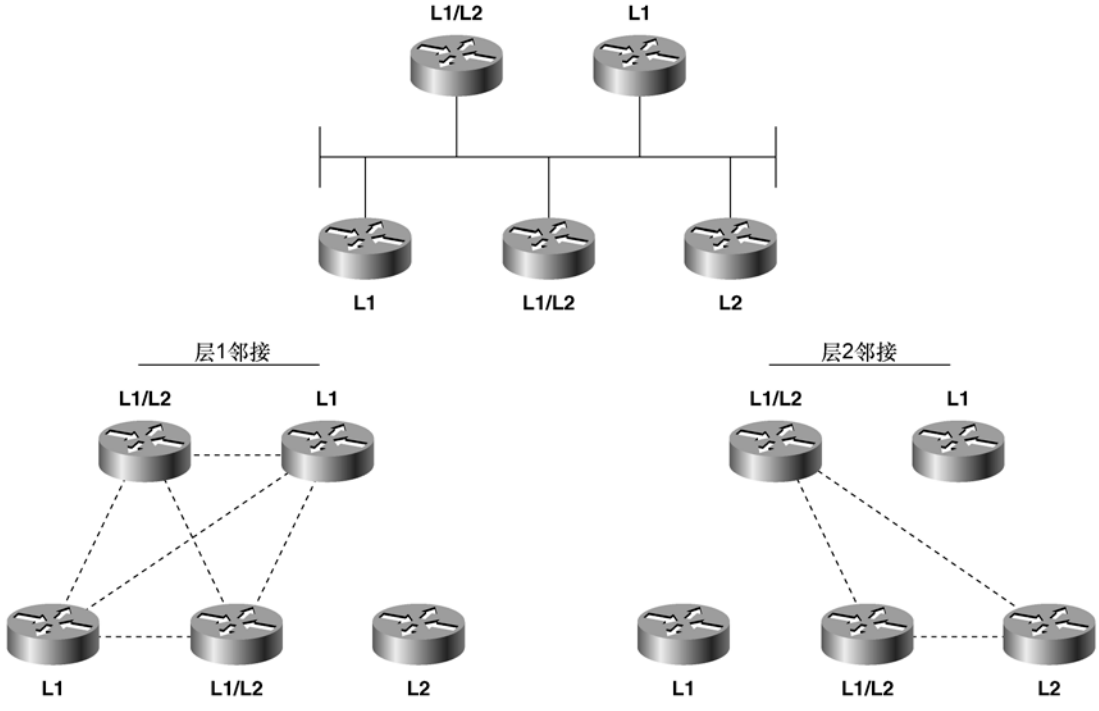


图 10-7 对于第 1 层和第 2 层建立的邻接关系是不同的，因而也就必须为第 1 层和第 2 层选取各自的指定路由器

指定路由器分配了所在网络的 LAN ID。正如前面章节所讨论的，LAN ID 是由该网络上的指定路由器的系统 ID 和它的伪节点 ID 连在一起得到的。该网络上其他所有的路由器都将使用指定路由器分配的这个 LAN ID。

如示例 10-2 所示，图中显示了一台路由器的 E0 接口的邻居表，这个 E0 接口是和示例 10-1 中显示的路由器的 E0 接口连接在同一个网络上的。通过比较这两个邻居表，可以发现总共有 3 台路由器连接在这个以太网上：0000.0c0a.2aa9、0000.0c0a.2c51 和 0000.0c76.5b7c。由于它们所有的优先级都是 64，因此在数值上具有最高的 MAC 地址的路由器将成为指定路由器，也就是路由器 0000.0c76.5b7c，并且它在这里利用设置为 02 的电路 ID 来标识这个网络。因此，示例 10-1 和示例 10-2 中显示的 LAN ID 都是 0000.0c76.5b7c.02。

示例 10-2 这台路由器的 E0 接口是和示例 10-1 中的路由器的 E0 接口连接在同一个网络上的

```
London#show clns is-neighbors
System Id       Interface  State  Type  Priority  Circuit Id          Format
0000.0C76.5B7C  Et0        Up     L2    64        0000.0C76.5B7C.02   Phase V
0000.0C0A.2AA9  Et0        Up     L2    64        0000.0C76.5B7C.02   Phase V
0000.3090.6756  Se0        Up     L1    0         02                  Phase V
London#
```

那个附加在系统 ID 后面的电路 ID 是必需的，因为同一台路由器可以是多个网络的指定路由器。注意在示例 10-1 中，与那台路由器的 E0 接口和 E1 接口相连的两个网络的指定路由器都是同一台路由器。与 E0 接口相连的网络的电路 ID 设置为 02，而与 E1 接口相连的网络的电路 ID 设置为 03，这样就可以使每一个网络的 LAN ID 保持惟一性。

IS-IS 协议的指定路由器处理过程相比 OSPF 协议的指定路由器处理过程来说，有两个方面是十分粗糙的（或者说是不够复杂的，这个观点因人而异）。首先，IS-IS 协议不选取备份指定路由器。如果 IS-IS 的指定路由器失效了，那么将选取一台新的指定路由器。其次，IS-IS 的指定路由器相对 OSPF 的指定路由器来说是不稳定的。如果一台 OSPF 路由器在一个已经存在指定路由器的网络上变成活动的了，即使它的优先级或路由器 ID 更高，新的路由器也不会成为一台指定路由器。结果是，OSPF 的指定路由器通常是网络上处于活动状态很长的路由器。与 OSPF 的规则相比，如果一台新的 IS-IS 路由器具有比现有指定路由器更高的优先级，或者优先级相同但是具有更高的 MAC 地址，那么这台新的 IS-IS 路由器将成为新的指定路由器。这样，每次指定路由器的更改都必须有一组新的 LSP 数据包进行泛洪扩散。

但是，假设增加一台新的路由器到一条广播链路上对于一个正在运行的网络来说是相对偶然情况，并且 DR 的选举过程在现代路由器中处理的非常快（一般在微秒级），那么缺少 BDR 与固定选取 DR 的情况就可以不予考虑。

2. 独立于子网的功能

子网独立子层的功能定义了 CLNS 怎样分发数据包通过整个 CLNP 的网络，以及怎样把这些服务提供给传输层。路由选择功能本身又被分成 4 个过程：更新过程、决策过程、转发过程和接收过程。正如后面两个过程的名称所暗示的，转发过程（forwarding process）的职责是传送 PDU 数据包，而接收过程（receive process）的职责是接收 PDU 数据包。这两个过程是在 ISO 10589 中描述的，并且相比 IP 数据包来说，它们和 CLNS NPDU 数据包的关系更密切，因此不再做进一步的讲述。

（1）更新过程

更新过程（update Process）的职责是构建 L1 和 L2 的链路状态数据库。为了做到这一点，L1 的 LSP 将在整个区域内进行泛洪，而 L2 的 LSP 将会在所有 L2 的邻接上进行泛洪。关于 LSP 数据包的具体字段的详细描述将在后面的 10.1.4 小节中讲述。

每一个 LSP 数据包都包含一个剩余生存时间、一个序列号和一个校验和。剩余生存时间（remaining lifetime）是一个老化时间或使用期限（age）。IS-IS LSP 数据包的剩余生存时间和 OSPF 协议中 LSA 数据包的老化时间参数的一个不同是：LSA 数据包的老化时间是从 0 到最大生存时间依次递增，而 LSP 数据包的剩余生存时间则是从最大生存时间开始，并递减到 0。在这里，IS-IS 的最大生存时间（MaxAge）是 1 200s（20min）。像 OSPF 协议一样，当 LSP 驻留在路由器的链路状态数据库中时，IS-IS 会随着时间的推移老化每一个 LSP，即递减它的剩余生存时间。并且，始发路由器必须周期性地刷新它的 LSP 以防止它的剩余生存时间减小到 0。IS-IS 的刷新时间间隔是 15min 减去一个最大不超过 25%的随机抖动变量。如果剩余生存时间减小到 0 了，那么这个过期的 LSP 将还会在路由器的链路状态数据库中保留 60s 的时间，这个时间称为“零老化生存时间”（ZeroAgeLifetime）。

与 OSPF LSA 的刷新过程相比，一个有趣而重要的不同是，IS-IS 的剩余生存时间由一个非 0 的数字开始，并递减到 0，这里非 0 的数字缺省是 1 200s 并能够更改。事实上，它可以

增大到 65 535s——这大约是 18.2 小时。这在一个非常大型的区域中对 IS-IS 的扩展性是一个重要而有帮助的因素。如果区域的链路相当稳定，那么增大刷新时间和剩余生存时间可以显著地减少 LSP 刷新引起的泛洪扩散负载。在本地始发的 LSP 中设置的初始剩余生存时间（MaxAge）值可以通过命令 **max-lsp-lifetime** 更改，本地始发的 LSP 中刷新之间的周期可以通过命令 **lsp-refresh-interval** 设置。如果改变缺省值，读者应该把设置的刷新时间间隔至少比剩余生存时间少几百秒，以便在原来的 LSP 实例过期之前，使新的刷新 LSP 可以到达区域内的所有路由器。

在原来的 IS-IS 过程中，如果一台路由器收到了一个带有错误校验和的 LSP，那么这台路由器可以通过将 LSP 的剩余生存时间设置为 0 并重新扩散出去来清除这条 LSP。清除的行为将会引起始发这条 LSP 的路由器发送一个新的关于这条 LSP 的实例（instance）。但是，ISO 10589 的第二版中将不再清除带有错误校验和的 LSP，因为在一个可能出现错误的子网中，允许接收路由器启动清除 LSP 的功能会显著地增加 LSP 的流量。

在 12.0 版之前的 IOS 软件实现中也遵循这个旧的清除过程，但是可以通过命令 **ignore-lsp-errors** 来忽略这个行为。当一台启动了该选项的路由器收到一条被破坏的 LSP 时，它就会丢弃它而不是清除它。但是，这条被破坏的 LSP 的始发路由器仍然会利用 SNP 了解到这条 LSP 没有被收到。SNP 将在本节后面的部分讲述。在 IOS 12.0 版本中实现了新的 IS-IS 过程，缺省情况下启动 **ignore-lsp-errors**。

然而，有关 LSP 清除的一个重要特点，也是 IS-IS 区别于 OSPF 协议的另一方面，是在 OSPF 协议里只有始发路由器才能清除这个 LSA。

序列号是一个 32 位的无符号线性数字。当一台路由器开始始发一条 LSP 时，它将使用设置为 1 的序列号，并且这条 LSP 的每一个后续实例的序列号都会递增 1。如果序列号递增达到了最大值（0xFFFFFFFF），那么这个 IS-IS 进程必须失效至少 21min（最大生存时间＋零老化生存时间），以便允许这条旧的 LSP 从所有的链路状态数据库中清除掉。

在一个点到点的子网上，路由器将直接发送 L1 和 L2 的 LSP 给它们的邻居路由器。在一个广播型的子网上，LSP 将以组播的方式发送到它所有的邻居路由器。携带 L1 LSP 的帧会有一个 0180.c200.0014 的目的 MAC 标识，称为 AllL1ISs。携带 L2 LSP 的帧会有一个 0180.c200.0015 的目的 MAC 标识，称为 AllL2ISs。

IS-IS 协议使用序列号数据包（SNP）来了解 LSP 的接收情况和维护链路状态数据库的同步情况。在这里有两种类型的 SNP：部分序列号报文（PSNP）和完全序列号报文（CSNP）。在一个点到点的子网上，路由器使用 PSNP 来明确地确认每一个 LSP 数据包是否收到。[1] PSNP 数据包是通过下面的信息来描述正在被确认的 LSP：

- LSP ID；
- LSP 的序列号；
- LSP 的校验和；
- LSP 的剩余生存时间。

当一台路由器在一个点到点的子网上发送一条 LSP 时，它会设置一个周期为 minimumLSPTransmissionInterval 的计时器。如果该计时器超时，路由器还没有收到一个关于确认收到这条 LSP 的 PSNP 数据包，那么将会发送一个新的 LSP 数据包。在 Cisco 路由器上，

[1] 有一个例外是，路由器收到了一条比它链路状态数据库中相同 LSP 的实例更旧的 LSP 实例。在这种情况下，该路由器将回复一个新的 LSP。

minimumLSPTransmissionInterval 的缺省值是 5s，并且可以基于每接口使用命令 **isis retransmit-interval** 来更改。缺省值通常几乎都是合适的值，偶尔的例外情况是某个邻居可能超负荷地处理大量的 LSP 数据包。这种情况下，路由器可能无法足够快地确认 LSP，从而触发一个重传数据包，并增加了它的故障。在这种情况下，增大 minimumLSPTransmissionInterval 是有帮助的；但最终真正解决这种问题的办法是升级低性能的邻居路由器。

在一个广播型子网上，LSP 不需要每一台接收它的路由器确认。作为替代，指定路由器将会周期性地以组播方式发送 CSNP，用来描述链路状态数据库中的每一个 LSP。发送 CSNP 的周期缺省的是 10s，并且可以通过命令 **isis csnp-interval** 来更改。L1 CSNP 以组播方式发送到 AllL1ISs（0180.c200.0014），而 L2 CSNP 以组播方式发送到 AllL2ISs（0180.c200.0015）。

当一台路由器收到一个 CSNP 时，它会把 PDU 中的 LSP 摘要与自己数据库中的 LSP 进行比较。如果发现在该路由器的数据库中存在 CSNP 中没有的 LSP，或者存在比 CSNP 中更新的 LSP 实例，那么该路由器将以组播方式在网络上发送这条 LSP。但是，如果其他的路由器开始发送更新的 LSP，那么该路由器将不会发送相同 LSP 的另一个拷贝。如果路由器的数据库中没有包含 CSNP 中列出的所有 LSP，或者数据库中包含的是某条 LSP 的旧实例，那么这台路由器将会以组播方式发送一个 PSNP 数据包，这个数据包中列出了该路由器所需要的所有 LSP。虽然 PSNP 数据包是以组播方式发送的，但是只有指定路由器才会使用包含相应 LSP 的数据包来响应。

IS-IS 具有一个有趣的特性，如果运行它的设备因内存不足而不能记录完整的链路状态数据库时，它具有通知其他路由器的能力。导致内存溢出或过载的原因可能是因为该路由器所在的区域变得过于庞大，路由器的内存不足，或者一些瞬间情况——像指定路由器失效等。在上面这些情况下，路由器如果不能完整地存储链路状态数据库，那么它将会在所发送的 LSP 数据包中设置一个称为过载（Overload，OL）的位。

OL 位用来指示路由器可能不能再进行正确的路由选择决策了，因为它的链路状态数据库已经不再完整。其他的路由器将仍然会转发数据包到这台过载路由器的直连网络上，但是，在这台过载的路由器发送一个清除 OL 位的 LSP 数据包之前，其他路由器不会再利用这台路由器转发经过它传送的数据流了。因为设置 OL 位可以避免过载的路由器被用作一条路由的下一跳，因此这个位又经常被称做 hippity 位（或称为 hippity-hop，随个人习惯叫法）。一般来说，路由器的内存应该平等地分配给 L1 和 L2 的数据库，但是，路由器能够在其中一个层的内存过载时，而保持其他层的内存处于正常状态。

过载特性是在路由器没有那么多内存时的产物。现代的路由器一般不太可能出现过载的情况。但是，OL 位在现代的网络中，特别是在 BGP 网络中还有另外一个非常有用的用途。为了了解这个问题，有必要先了解一些 BGP 协议的基本概念。虽然像 IS-IS 这样的链路状态协议收敛速度非常快，但 BGP 网络的收敛却比较慢——有时需要花上几分钟的时间。在一个 BGP 网络的内部，路由可能具有距离好几跳路由器的下一跳地址。当一个数据包到达时，将在 BGP 路由中查找它的目的地址。转发这个数据包之前，在 IGP 的路由表中必须能够查找到该 BGP 路由的下一跳。

现在假定有一台新路由器增加到某个网络上，并且 IGP 已经被收敛但 BGP 还没有完成收敛。如果另一台路由器根据收敛的 IGP 路由确认这台新增加的路由器是到达 BGP 下一跳的最优路径，那么它将把向下一跳地址转发的数据包转发到这台路由器上。但是，如果 BGP

还没有在新的路由器中完成收敛，该路由器就可能没有关于这个数据包的 BGP 路由，从而丢弃该数据包，结果变成路由业务黑洞。

在 BGP 完成收敛之前通过设置 IS-IS 的 OL 位，我们就可以避免这个问题。如果设置了 OL 位，那么其他路由器将会绕过这台新的路由器进行路由选择。一旦 BGP 完成收敛，OL 位将被清除，数据包将可以通过这台新路由器进行转发。

使用命令 **set-overload-bit on-startup** 可以指定一个秒数，用来说明在 IS-IS 启动后需要设置 OL 位的时间。当超过指定的秒数时，OL 位会被自动地清除。将 OL 位设置为大约 300～500s，可以确保 BGP 在 OL 位被清除前完成 BGP 的收敛。这里有另一个可选的命令 **set-overload-bit on-startup wait-for-bgp**，这样一旦 BGP 完成收敛就会清除 OL 位。虽然这比使用一个静态的秒数更可以动态地调整，但这种做法也有一个致命的缺点：如果某个 BGP 会话因为某些原因无法形成，那么 OL 位将永远都不会被清除。因此，通常的做法最好还是指定一个秒数代替 **wait-for-bgp** 选项。

我们也可以通过使用不指定其他选项的 **set-overload-bit** 手工地模糊设置 OL 位。这在我们希望把一台路由器连接到 IS-IS 网络上并接收全部的数据库，但又不希望这台路由器用来作为转发路径时比较有用。这种应用的一个例子就是，把一台试验的路由器连接到一个商用网络的情形。

使用命令 **show isis database** 可以显示一个 IS-IS 链路状态数据库的摘要，如示例 10-3 所示。其中显示了路由器 Brussels 是一台 L1/L2 路由器，因此它包含了 L1 和 L2 的数据库。在第一列显示的 LSP ID 是由始发路由器的系统 ID 连接了 2 个八位组字节构成的。在这里，跟在系统 ID 后的第一个八位组字节是伪节点 ID。如果这个八位组字节是非零的，LSP 则是由一台 DR 路由器始发的。这时，系统 ID 和非零的伪节点 ID 一起构成了一个广播型子网的 LAN ID。

示例 10-3 IS-IS 的数据库可以通过命令 show isis database 来查看

```
Brussels#show isis database
IS-IS Level-1 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
0000.0C04.DCC0.00-00  0x00000036   0x78AE        1152          0/0/0
0000.0C0A.2AA9.00-00  0x0000011B   0x057B        416           0/0/0
0000.0C76.5B7C.00-00* 0x00000150   0xD5D4        961           1/0/0
0000.0C76.5B7C.02-00* 0x00000119   0xD9C3        407           0/0/0
0000.0C76.5B7C.03-00* 0x000000FA   0x896E        847           0/0/0
IS-IS Level-2 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
0000.0C0A.2AA9.00-00  0x0000013E   0x319A        666           0/0/0
0000.0C0A.2C51.00-00  0x00000133   0x762D        654           0/0/0
0000.0C76.5B7C.00-00* 0x0000014C    0x4E91       886           0/0/0
0000.0C76.5B7C.02-00* 0x0000011F   0x3CC3        1174          0/0/0
0000.3090.C7DF.00-00  0x0000011A   0xDDF0        858           0/0/0
Brussels#
```

LSP ID 的最后一个八位组字节是 LSP 的编号。有时候一个 LSP 可能会很大，以至于超出了路由器缓冲区或数据链路所支持的 MTU 的大小。在这种情况下，LSP 将被分段传送——也就是说，LSP 的信息可以在多个 LSP 数据包中传送。这些 LSP 数据包的 LSP ID 由 3 部分组成：相同的系统 ID 和伪节点 ID，还有不同的 LSP 编号。

LSP ID 后面紧跟的星号，表示这条 LSP 数据包是始发于正在查看的数据库所在的路由

器。例如，在示例 10-3 中显示的数据库是来自路由器 Brussels 的。因此，LSP ID 为 0000.0c76.5b7c.00-00 的 L1 LSP 是始发于路由器 Brussels 的。

数据库的第 2 列和第 3 列显示了每一个 LSP 的序列号和校验和。第 4 列显示的是 LSP 抑制时间，也就是 LSP 的剩余生存时间，以秒数计。如果连续不断地重复输入 **show isis database** 命令，就会发现这个数字在不断减小。当重新刷新一条 LSP 时，LSP 的剩余生存时间将被重新设置为 1200s，并将序列号递加 1。

最后一列指明了每一个 LSP 的区域关联位（ATT 位）、区域分段位（Partition，P 位）和过载位（OL 位）。L2 和 L1/L2 路由器设置 ATT 位为 1 来指明它们含有到达其他区域的路由。P 位指出始发路由器具有支持区域分段修复的能力。Cisco 公司（和大多数其他厂商）并不支持这个功能，因此该位总是设置为 0。OL 位设置为 1，可能是始发路由器正处于内存过载状态，而这时链路状态数据库是不完整的，或者是被手工设置的。

参见示例 10-4 所示，使用带 level 和 LSP ID 的 **show isis database detail** 命令可以查看一个 LSP 的完整信息。LSP 中每一个单独字段的具体含义参见 10.1.4 小节。

示例 10-4 使用命令 show isis database detail 可以查看数据库中 LSP 的完整信息

```
London#show isis database detail level-2 0000.0C0A.2C51.00-00
IS-IS Level-2 LSP 0000.0C0A.2C51.00-00
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
0000.0C0A.2C51.00-00* 0x0000013B   0x6635        815           0/0/0
  Area Address: 47.0001
  NLPID:      0x81 0xCC
  IP Address: 10.1.3.2
  Metric: 10 IS 0000.0C76.5B7C.02
  Metric: 10 IP 10.1.3.0 255.255.255.0
  Metric: 20 IP 10.1.2.0 255.255.255.0
  Metric: 10 IP 10.1.255.4 255.255.255.252
  Metric: 20 IP 10.1.255.0 255.255.255.0
  Metric: 30 IP 10.1.255.8 255.255.255.252
London#
```

（2）决策过程

一旦更新过程建立了链路状态数据库，决策过程就将使用数据库中的信息去计算一个最短路径树。接着，该过程使用生成的最短路径树去构建一个转发数据库（路由表）。对于 L1 数据库和 L2 数据库，路由器将会执行不同的 SPF 计算。

ISO 10589 规定 IS-IS 协议使用下面的度量（一项是必须的，三项是可选的）来计算最短路径：

- **缺省度量（Default）**——这是每一台 IS-IS 路由器都必须支持和理解的度量。
- **时延度量（Delay）**——这是一个可选项，反映一个子网的传输时延。
- **代价度量（Expense）**——这是一个可选项，反映一个子网的成本代价。
- **差错度量（Error）**——这是一个可选项，反映一个子网的出错概率，类似于 IGRP/EIGRP 协议中的可靠度量。

每一种度量都使用一个范围在 0～63 之间的整数表示，并且每个路由都要为每种度量进行单独地计算。因此，如果一个系统同时支持这 4 种度量类型，那么路由器必须为 L1 数据库和 L2 数据库各运行 4 次 SPF 计算。由于对于每一个目的路由都可能需要进行多次反复地计算，结果会产生多个不同的路由表；因为可选的度量是用来支持根本没有发展起来的服务类型（ToS）的路由选择的，因而 Cisco 公司只支持缺省度量。

在 Cisco 的路由器上，不论接口的类型如何，都会指定每一个接口的缺省度量为 10。使用命令 **isis metric** 可以修改这个缺省度量的值，而且可以分别为第 1 层和第 2 层的接口修改它们的缺省值。如果对于每一个接口都保留使用它的缺省度量 10，那么每个子网的度量都可以被认为是等价的，并且每个子网的 IS-IS 度量可以看作是一个简单的跳数，其中每一跳的代价为 10。

这种情况下，一条路由的总代价就可以看作是沿此路由路径方向的每一个出站接口的单独度量简单相加。对于任何一条路由，IS-IS 最大的度量值是 1023。这个比较小的最大度量值经常被认为是 IS-IS 协议的一个限制，因为在一个大型的网络上它的度量尺度显得有点小了。

但是，IS-IS 中新的扩展允许非常大的度量空间，具有 32 位度量，称为扩展度量（wide metrics）。使用命令 **metric-style wide** 可以设置扩展度量。

IS-IS 协议的路由不仅分 L1 路由和 L2 路由，而且分内部路由和外部路由。内部路由是指到达 IS-IS 路由选择域内的目的地的路径，而外部路由是指到达 IS-IS 路由选择域外的目的地的路径。虽然 L2 路由可能是内部路由，也可能是外部路由，但 L1 路由却总是内部路由。使用路由选择策略我们能够把一条 L1 路由变成外部路由，但是只有在比较合理和比较小心的情况下才应该这样做。

如果到达某个具体的目的地存在多条可能的路由，那么 L1 的路由将优先于 L2 的路由。在同一层的多条路由中，支持可选度量的路径要优先于只支持缺省度量的路径（再次提示，Cisco 的路由器仅仅支持缺省度量，因此第二个优先顺序的排序和 Cisco 的路由器不相关）。在每一层所支持的度量中，具有最低度量的路由优先。如果经过这个决策处理后发现多条路径在同一层是等价的，那么它们都会被放入路由表中。在 Cisco 公司的 IS-IS 协议的实现中将执行等代价的负载均衡，并且最大支持 6 条等价负载均衡的路径。

在前面“更新过程”中谈到 LSP ID 的最后一个八位组字节是 LSP 编号（LSP Number），并用来跟踪分段的 LSP。决策过程关注这个 LSP 编号有几种原因。首先，如果一个 LSP 编号为 0 并且剩余生存时间不为 0 的 LSP 不在路由器的数据库中，那么决策过程将不会处理任何具有同样的系统 ID 但 LSP 编号不为 0 的 LSP。例如，假设在数据库中存在 LSP ID 为 0000.0c76.5b7c.00-01 和 0000.0c76.5b7c.00-02 的两条 LSP，但是在该数据库中没有包含 LSP ID 为 0000.0c76.5b7c.00-00 的 LSP，那么路由器将不会处理前面的两条 LSP。这种方法可以确保不会因不完整的 LSP 而导致不精确的路由选择决策。

另外，决策过程将仅从 LSP 编号为 0 的 LSP 中接受下面的信息：

- 数据库中过载位的设置信息；
- IS 类型字段的设置信息；
- 区域地址可选字段的设置信息。

但是在 LSP 编号不为 0 的 LSP 中，决策过程将忽略这些设置信息。换句话说，在一系列被分段的 LSP 中，将由第一个 LSP 来宣告所有分段 LSP 的这 3 个设置信息。

参见示例 10-5，图中显示了一台 Cisco 的 IS-IS 路由器的路由表。在这里我们注意到存在 L1 和 L2 的路由，并且有 3 个到达目的地的路由具有多条路径。每一条路由都带有一个相应的掩码，这表明它们是支持 VLSM 技术的。最后，在路由表中也指出了 IS-IS 路由的管理距离是 115。

示例 10-5 这个路由表显示了第 1 层和第 2 层 IS-IS 路由

```
Brussels#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
     10.0.0.0 is variably subnetted, 8 subnets, 3 masks
C       10.1.3.0 255.255.255.0 is directly connected, Ethernet0
i L2    10.1.2.0 255.255.255.0 [115/30] via 10.1.3.2, Ethernet0
                               [115/30] via 10.1.3.3, Ethernet0
i L1    10.1.5.0 255.255.255.0 [115/20] via 0.0.0.0, Serial0
                               [115/20] via 10.1.4.2, Ethernet1
C       10.1.4.0 255.255.255.0 is directly connected, Ethernet1
i L2    10.1.255.4 255.255.255.252 [115/20] via 10.1.3.2, Ethernet0
i L2    10.1.255.0 255.255.255.0 [115/30] via 10.1.3.2, Ethernet0
i L1    10.1.255.8 255.255.255.252 [115/20] via 10.1.3.3, Ethernet0
i L1    10.1.6.240 255.255.255.240 [115/20] via 0.0.0.0, Serial0
                                   [115/20] via 10.1.4.2, Ethernet1
Brussels#
```

对于 L1 的路由器来说，决策过程还有另外一项功能，就是为区域间路由选择计算到达最近的 L2 路由器的路径。正如前面所提到的，当一台 L2 或 L1/L2 路由器与其他区域相连时，路由器将通过在它的 LSP 中设置 ATT 位为 1 来通告这种情况。对于 L1 路由器，决策过程将选择度量最近的 L1/L2 路由器作为它缺省的区域间路由器。当使用 IS-IS 协议进行 IP 路由选择时，在路由器中会记录一条到达 L1/L2 路由器的 IP 缺省路由。例如，在示例 10-6 中显示了一台 L1 路由器的链路状态数据库和相应的路由表。LSP 0000.0c0a.2c51.00-00 的 ATT 位设置为 1。基于这个信息，决策过程将选择系统 ID 为 0000.0c0a.2c51 的路由器作为缺省的区域间路由器。在路由表中还显示了一条经过 10.1.255.6 可达的缺省路由（0.0.0.0），它的度量是 10。虽然在示例 10-6 的两个表中显示的信息不太容易对照，但实际上地址为 10.1.255.6 和系统 ID 为 0000.0c0a.2c51.00-00 的路由器指的是同一台路由器。

示例 10-6 如果 ATT 位设置为 1，集成 IS-IS 就会增加一条到达最近的 L1/L2 路由器的 IP 缺省路由

```
Paris#show isis database
IS-IS Level-1 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
0000.0C0A.2C51.00-00  0x0000016D   0xA093        730           1/0/0
0000.3090.6756.00-00* 0x00000167   0xC103        813           0/0/0
0000.3090.6756.04-00* 0x0000014E   0x227F        801           0/0/0
0000.3090.C7DF.00-00  0x00000158   0x78A6        442           0/0/0
Paris#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is 10.1.255.6 to network 0.0.0.0
     10.0.0.0 is variably subnetted, 5 subnets, 2 masks
i L1    10.1.3.0 255.255.255.0 [115/20] via 10.1.255.6, Serial0
C       10.1.2.0 255.255.255.0 is directly connected, TokenRing0
i L1    10.1.255.4 255.255.255.252 [115/20] via 10.1.255.6, Serial0
C       10.1.255.0 255.255.255.0 is directly connected, Serial0
i L1    10.1.255.8 255.255.255.252 [115/20] via 10.1.2.2, TokenRing0
i*L1 0.0.0.0 0.0.0.0 [115/10] via 10.1.255.6, Serial0
Paris#
```

示例 10-6 中显示的信息突出了采用 IS-IS 协议的一个有趣特性，特别是在进行故障排除时就更加突出。虽然 TCP/IP 协议是被路由转发的协议，但是决定路由选择策略的协议，包括所有路由的控制数据包和地址却都是 CLNS 协议。有时要把 CLNS 协议的信息和 IP 协议相关的信息关联起来是比较困难的。解决该问题的，有一条很有用的命令是 **which-route**。

这条命令主要用来确定某个具体的 CLNS 目的地址在路由表的定位。但是，利用 **which-route** 命令也可以了解到一些关于某个具体 CLNS 地址和相关 IP 地址的有用信息。如示例 10-7 所示，显示了使用系统 ID/电路标识 0000.0c0a.2c51.00 作为参数的 **which-route** 命令的输出信息，这里的系统 ID0000.0c0a.2c51.00 指的就是在前面示例 10-6 显示的数据库中 ATT=1 的那个 LSP。从输出结果可以看出，要查询的系统 ID 的下一跳 IP 地址是 10.1.255.6。

示例 10-7 使用 **which-route** 命令可以发现一些 CLNS 地址和 IP 地址的关联信息

```
Paris#which-route 0000.0C0A.2C51.00
Route look-up for destination 00.000c.0a2c.5100
  Using route to closest IS-IS level-2 router
Adjacency entry used:
System Id        SNPA      Interface   State   Holdtime   Type   Protocol
0000.0C0A.2C51   *HDLC*    Se0         Up      26         L1     IS-IS
  Area Address(es): 47.0001
  IP Address(es): 10.1.255.6
  Uptime: 22:08:52
Paris#
```

10.1.4 IS-IS 的 PDU 格式

IS-IS 协议使用 9 种 PDU 类型来进行它的控制信息处理，并使用一个 5 位的类型号来标识每一个 PDU 数据包。所有的 PDU 数据包可以归纳为 3 类，如表 10-1 所示。

在所有 IS-IS PDU 数据包起始的 8 个八位组字节都是该数据包的头部字段，并且对于所有的 PDU 数据包类型来说都是公用的，如图 10-8 所示。这里将先讲述这些起始字段，特有的 PDU 字段将在后续的章节讲述。

表 10-1 S-IS 协议的 PDU 数据包类型

IS-IS PDU	类型号
Hello PDU	
第 1 层 LAN 的 IS-IS Hello PDU	15
第 2 层 LAN 的 IS-IS Hello PDU	16
点到点的 IS-IS Hello PDU	17
链路状态 PDU	
第 1 层 LSP	18
第 2 层 LSP	20
序列号 PDU	
第 1 层完全序列号	24
第 2 层完全序列号	25
第 1 层部分序列号	26
第 2 层部分序列号	27

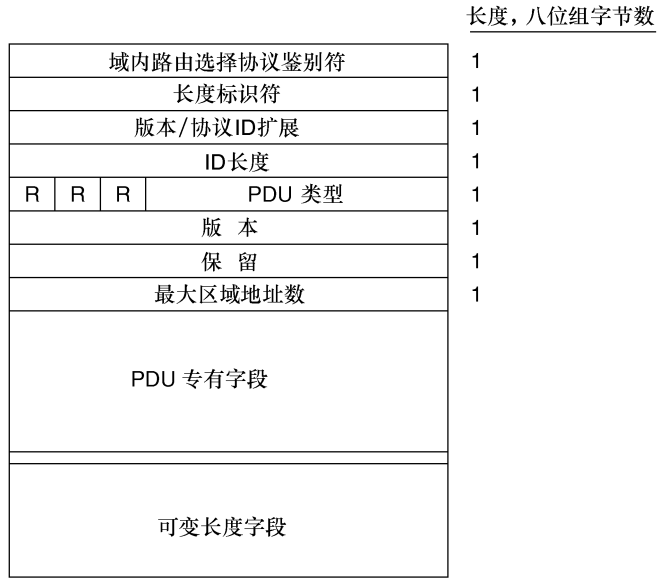


图 10-8 IS-IS PDU 数据包起始的 8 个八位组字节

- **域内路由选择协议鉴别符（Intradomain Routeing Protocol Discriminator）**——这是由 ISO 9577[1]分配的一个恒定不变的数值，用来标识网络层协议数据单元（NPDU）。在所有的 IS-IS PDU 中，该字段的值都是 0x83。
- **长度标识符（Length Indicator）**——标识该固定头部字段的长度，以八位组字节数表示。
- **版本/协议 ID 扩展**（Version/Protocol ID Extension）——当前始终设置为 1。
- **ID 长度（ID Length）**——用来标识该路由选择域内使用的 NSAP 地址和 NET 的系统 ID（System ID）的长度。该字段的取值可以是以下几个数值之一：
  - 1～8 的整数，表示系统 ID 字段具有相同长度的八位组字节数；
  - 0，表示系统 ID 字段的长度为 6 个八位组字节；
  - 255，表示系统 ID 字段为空（0 个八位组字节）。

在 Cisco 路由器中系统 ID 字段的长度固定为 6 个八位组字节，因此，在由 Cisco 路由器始发的 PDU 数据包中，这个 ID 长度字段的值将始终是 0。

- **PDU 类型**——是一个 5 位的字段，取值范围可以是表 10-1 中显示的 PDU 类型号中的任何一个。该字段的前 3 位（R）作为保留位，始终为 0。
- **版本号（Version）**——当前始终设置为 1，这和第 3 个八位组字节中的版本/协议 ID 扩展字段是一致的。
- **保留位**——当前设置为全 0。
- **最大区域地址数（Maximum Area Addresses）**——表示该 IS 区域所允许的最大区域地址数量。这个字段的值可以是下面数值之一：
  - 1～254 的整数，表示该区域实际所允许的最大区域地址数；
  - 0，表示该 IS 区域最多只支持 3 个区域地址数。

Cisco IOS 软件缺省情况下最多支持 3 个区域。因此，在由 Cisco 路由器始发的 IS-IS PDU

[1] 国际标准化组织，“Protocol Identification in the Network Layer”，ISO/IEC TR 9577，1990 年。

中，该最大区域地址数字段的值始终是 0，除非通过命令 **max-area-addresses** 改变缺省配置。

在图 10-9 中，显示了使用协议分析仪捕获到的某个 IS-IS PDU 起始的 8 个八位组字节。紧跟在公共头部字段之后的专有 PDU 字段也是头部的一部分。根据 PDU 类型的不同它们会有所变化，这将在讲述具体 PDU 类型的章节中介绍。

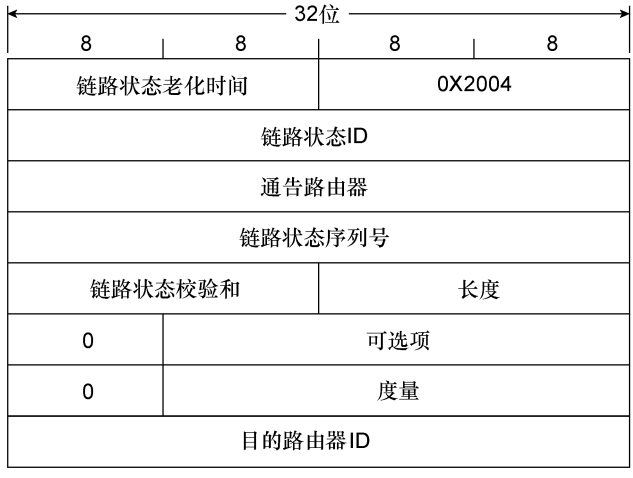


图 10-9 使用协议分析仪捕获到的某个 IS-IS PDU 起始的 8 个八位组字节

1. TLV 字段

紧跟在专有 PDU 字段之后的可变长度字段是类型/长度/值（Type/Length/Value，TLV）[1]这 3 个参数的不同组合，如图 10-10 所示。类型（或代码）[2]是一个指定值字段信息类型的数字，长度用来指定值字段的长度，而值字段就是它本身的信息内容。注意，长度字段只用一个八位组字节来表示，这意味着值字段的最大值为 255 个八位组字节。



图 10-10 IS-IS 的 TLV 参数的组合在 IS-IS 协议中的功能和 TLV 的组合在 EIGRP 协议中的功能相同

表 10-2 中列出了所有 IS-IS 协议里 TLV 的代码。在这个表中也指出了哪些 TLV 是在 ISO 10589 中指定的，哪些是在 RFC 1195 中指定的。ISO 指定的 TLV 是设计用于 CLNP 协议的，但它们中的大多数也用于 IP 协议。RFC 指定的 TLV 只设计用于 IP 协议。如果一台路由器不能识别一个特有的 TLV 代码，那么它将忽略这个 TLV。这种设计方法可以允许在同一个 PDU 中携带支持 CLNP 协议的 TLV、支持 IP 协议的 TLV，或者同时携带这两种 TLV。

---

[1] 读者已经在前面第 7 章中讲述 EIGRP TLV 的概念时熟悉了 TLV 的概念。事实上，在 RFC 1195 中，也把集成 IS-IS TLV 称为 TLV。

[2] 在 ISO 中使用术语“代码”，而在 IETF 中使用术语“类型”。由于“TLV”已经变得比较通用，因此本章通常使用术语“类型”。这与本书第一版不同，在第一版中作者选用了术语“代码”。

表 10-2 IS-IS 协议中使用的 TLV 代码

类型	TLV	ISO 10589	RFC 1195
1	区域地址	X	
2	中间系统邻居（LSP）	X	
3	终端系统邻居*	X	
4	区域分段指定的第 2 层中间系统**	X	
5	前缀邻居	X	
6	中间系统邻居（Hello）	X	
8	填充项	X	
9	LSP 条目	X	
10	认证信息	X	
14	LSP 缓存大小	X	
128	IP 内部可达性信息		X
129	支持的协议		X
130	IP 外部可达性信息		X
131	域间路由选择协议信息		X
132	IP 接口地址		X
133	认证信息***		X

*终端系统邻居和前缀邻居的 TLV 与 IP 路由选择没有关系，因此不在本书中讲述。

**这个 TLV 用作分段区域的修复，Cisco 的路由器并不支持。

***RFC 1195 为 IP 认证指定了这个代码，但大多数 IS-IS 的实现，包括 IOS 软件中都使用 ISO 指定的代码 10。

虽然大多数的 TLV 都在不止一种 IS-IS PDU 类型中使用，但是只有一个 TLV（认证信息 TLV）是在所有的 PDU 中都使用的。在下面讲述 IS-IS PDU 格式部分，将会列出每一种 PDU 用到的 TLV。每一种 TLV 的格式将只在它第一次出现时讲述一次。表 10-3 中总结了每种 PDU 所用到的 TLV。

表 10-3 每一种 IS-IS PDU 所用到的 TLV

TLV 的类型	PDU 类型								
	15	16	17	18	20	24	25	26	27
区域地址	X	X	X	X	X				
中间系统邻居（LSP）				X	X				
终端系统邻居				X					
区域分段指定的第 2 层 IS					X				
前缀邻居					X				
中间系统邻居（Hello）	X	X							
填充项	X	X	X						
LSP 条目						X	X	X	X
认证信息	X	X	X	X	X	X	X	X	X
IP 内部可达性信息				X	X				
支持的协议	X	X	X	X	X				
IP 外部可达性信息				X	X				
域间路由选择协议信息					X				
IP 接口地址	X	X	X	X	X				

使用 TLV 的最大好处是我们只需要增加新的 TLV 类型，就可以为 IS-IS 添加新特性。自从开始使用 TLV 以来，IS-IS 协议中已经增加了很多增强型特性。表 10-4 中列出了自从 ISO 10589 和 RFC 1195 规定 IS-IS 以来已经增加的许多 TLV 类型。表中同时也列出了 RFC 规定的新 TLV 和它们的用途。在一些实例中，扩展特性还非常新，在编写本书时它们还是 Internet 草案形式。但在读者阅读本章的时候，它们也许已经成为 RFC 了。这可以通过 IETF 的 Web 网站（www.ietf.org）查找。表 10-4 中列出的部分扩展特性在本章随后的一些章节会有更详细地描述。

表 10-4 TLV 用于扩展 IS-IS 特性

类型	TLV	RFC	描述
12	可选的校验和	3358	为 SNP 增加校验和的特性
22	扩展的 IS 可达性	3784	增加流量工程特性，替代类型 2 的 TLV
134	流量工程路由器 ID	3784	增加流量工程特性
135	扩展的 IP 可达性	3784	增加流量工程和扩展度量的特性，替代类型 128 和类型 130 的 TLV
137	动态主机名	2763	增加节点标识的能力，在像 **show isis database** 这样的命令中使用主机名而不是 SysID 来表示节点
211	重新启动	3847	增加优雅重启（graceful restart）的特性
222	MT 中间系统	草案	和类型 22 的 TLV 一起支持多拓扑
229	多拓扑	草案	增加对多拓扑的支持
232	IPv6 接口地址	草案	等价于类型 132 的 TLV，用来支持 IPv6
235	MT 可达的 IPv4 前缀	草案	和类型 135 的 TLV 一起支持多拓扑
236	IPv6 的可达性	草案	等价于类型 128 和类型 130，用来支持 IPv6
237	MT 可达的 IPv6 前缀	草案	使用类型 236 的 TLV 支持多拓扑
240	点到点的三方邻接	3373	增加三方握手的特性
250	实验用	草案	用于实验的扩展

2. IS-IS Hello PDU 格式

IS-IS Hello PDU 是同一条链路上的 IS-IS 路由器用来发现它们的 IS-IS 邻居路由器的。一旦路由器发现了它的邻居路由器并且成功建立邻接关系，Hello PDU 的工作就只担当 keepalive 的功能，从而维护已有的邻接关系并将邻接关系中任何参数的变化通知邻居。

一个 IS-IS PDU 大小的上限可以由始发路由器的缓冲区大小，或者传输该 PDU 的数据链路的 MTU 值来确定。ISO 10589 规定 IS-IS Hello 数据包必须使用类型 8 的填充 TLV，来填充一个小于该最大值的八位组字节，以便使路由器可以和它的邻居路由器以它的 MTU 进行通信。更为重要的是，发送达到或接近链路 MTU 大小的 Hello 数据包可以帮助检测这样一种链路的失效情形：较小的 PDU 可以通过，但是较大的 PDU 数据包会被丢弃。如果一个邻居的接口不支持所要求的最小 MTU，那么被填充的 Hello 将被丢弃，并无法建立邻接关系。

有两种类型的 IS-IS Hello 数据包：LAN Hello 数据包和点到点 Hello 数据包。LAN Hello 数据包可以进一步分为 L1 和 L2 的 LAN Hello 数据包。但是，这两种类型的 LAN Hello 数据包的格式是相同的，如图 10-11 所示。图 10-12 显示了协议分析仪捕获到的一个 L2 的 LAN Hello 数据包。

- **电路类型（Circuit Type）**——是一个 2 位的字段（前面 6 位是保留位，始终设置为 0），用来指定该路由器是 L1 路由器（01）、L2 路由器（02），还是 L1/L2 路

由器（11）。如果这两位都为 0（00），那么这个 PDU 数据包整个都会被忽略。

- **源 ID（Source ID）**——是指始发该 Hello 数据包的路由器的系统 ID。
- **抑制时间（Holding Time）**——是指邻居路由器在宣告始发路由器失效之前，它所等待接收下一个 Hello 数据包的时间间隔。
- **PDU 长度**——是指整个 PDU 数据包长度的八位组字节数。
- **优先级**——是一个用来选取 DR 的 7 位字段。这个字段可以设置成 0～127 之间的数值，而且数值越大就表示优先级越高。L1 的指定路由器是通过 L1 LAN Hello 数据包中的优先级特性选取出的，而 L2 的指定路由器是通过 L2 LAN Hello 数据包中的优先级特性选取出的。
- **LAN ID**——就是指定路由器（DR）的系统 ID 再加上一个八位组字节（伪节点 ID），用来区分这个 LAN ID 和同一台指定路由器上的其他 LAN ID。



图 10-11 IS-IS LAN Hello PDU 数据包的格式

在 IS-IS LAN Hello 数据包中可以使用下面的多种 TLV：[1]

- 区域地址（类型 1）；
- 中间系统邻居（类型 6）；
- 填充（类型 8）；
- 认证信息（类型 10）；
- 可选的校验和（类型 12）；
- 支持的协议（类型 129）；
- IP 接口地址（类型 132）；
- 重启（类型 211）；

[1] 作为提示，RFC 1195 规定了认证信息 TLV 的类型代码是 133。而 Cisco 路由器使用 ISO 规定的类型代码 10 来标识认证信息 TLV。

- 多拓扑（类型 229）；
- IPv6 接口地址（类型 232）；
- 实验用（类型 250）。

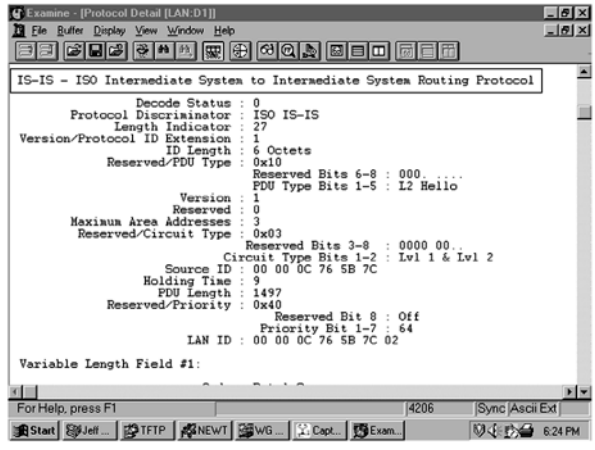


图 10-12 通过协议分析仪捕获到的 LAN Hello 数据包显示了只有 Hello PDU 数据包才有的字段

如图 10-13 所示，显示了 IS-IS 点到点 Hello PDU 数据包格式。点到点 Hello 数据包的格式和 LAN Hello 数据包相比，除了没有优先级字段外基本上是一样的，而且它用本地电路 ID 字段替代了 LAN ID 字段。与 LAN Hello 数据包不同，L1 和 L2 的信息是在同一个点到点 Hello PDU 数据包中传送的。除了中间系统邻居 TLV，点到点 Hello 数据包可以携带所有相同的 TLV。作为替代，如果 IS-IS 实现支持相关的扩展，点到点 Hello 数据包携带一个 P2P 三方邻接（类型 240）的 TLV。这个 TLV 的用途将在 10.1.5 小节有关“三方握手”部分讲述。

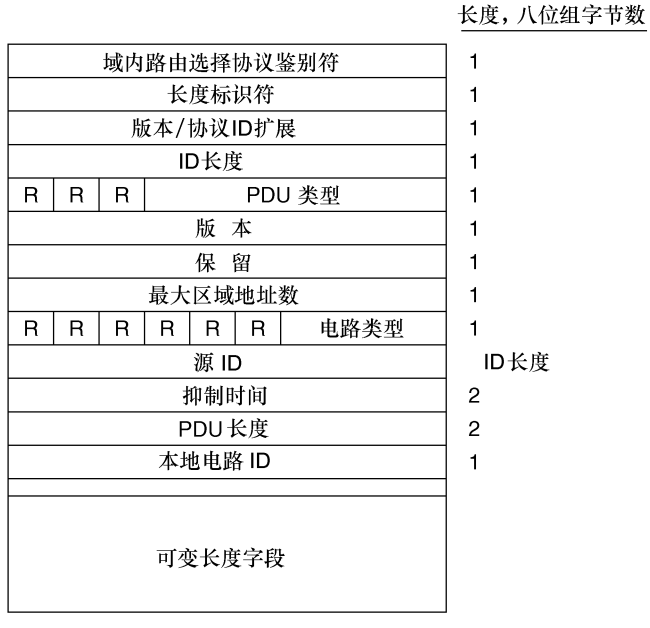


图 10-13 IS-IS 点到点 Hello PDU 数据包的格式

- **本地电路 ID（Local Circuit ID）**——是一个 1 个八位组字节的标识字段，由始发路由器发送 Hello 数据包时分配给这条电路，并且在路由器的接口上是惟一的。在点到

点链路的另一端，Hello 数据包中的本地电路 ID 可能包含，也可能不包含同样的值。

（1）区域地址 TLV

如图 10-14 所示，区域地址 TLV 是在始发路由器上配置的，并用来通告该区域的地址。正如多个地址长度/区域地址字段所表示的，一台路由器可以配置多个区域地址。

在图 10-15 中显示了一个 IS-IS Hello PDU 数据包的部分信息。“Variable Length Field #3”部分显示的就是一个区域地址 TLV 的信息，它总共有 6 个八位组字节长，包括两个区域地址：47.0002（3 个八位组字节）和 0（1 个八位组字节）。

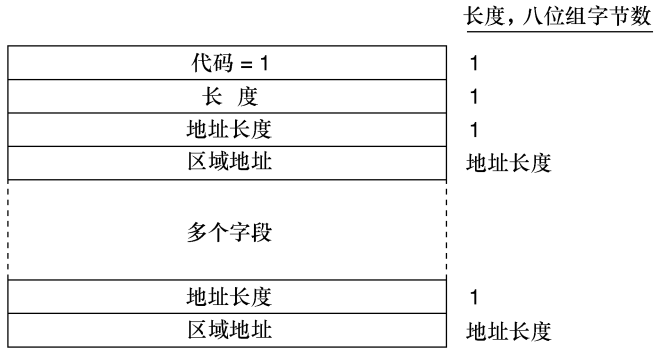


图 10-14　区域地址 TLV 的格式

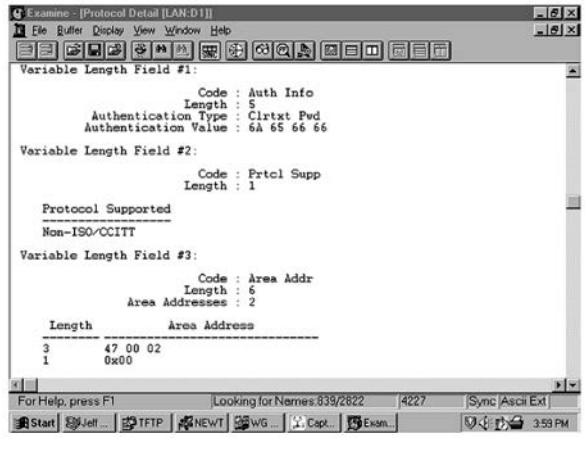


图 10-15　“Variable Length Field #3”部分显示的就是一个区域地址 TLV 的信息。为了便于观察，分析仪也在 TLV 中指出了所列出的所有地址数，但是这个信息并不是 TLV 字段的一部分

（2）中间系统邻居 TLV（Hello）

如图 10-16 所示，中间系统邻居 TLV 列出了本地路由器所有邻居的系统 ID，但是这些邻居路由器必须满足在最新的一个抑制时间间隔内，能够被本地路由器接收到它们发出的 Hello 数据包。这里可以注意到，这个 TLV 字段对于 IS-IS LAN Hello 数据包所起到的功能，其实在 OSPF 协议中也有相类似的功能：为了验证双向通信，本地路由器列出了最近发出的所有 Hello 数据包以及被其收到的邻居路由器。

这个 TLV 只能用在 LAN Hello 数据包中。由于点到点 Hello 数据包中没有指定路由器的选取过程，因而在点到点 Hello 数据包中没有这个 TLV。同时，这里的 TLV 也和 LSP 数据包中使用中间系统邻居 TLV 有所不同，这可以通过它们的类型号码区分开来。L1 LAN Hello

数据包只列出了 L1 的邻居，而 L2 LAN Hello 数据包也只列出了 L2 的邻居。在这个 TLV 中列出的邻居是由该 LAN 上它们接口的 MAC 地址标识的。图 10-17 中显示的“Variable Length Field #5”部分表示了一个中间系统邻居 TLV，它只列出了一个邻居——0000.0c0a.2aa9。

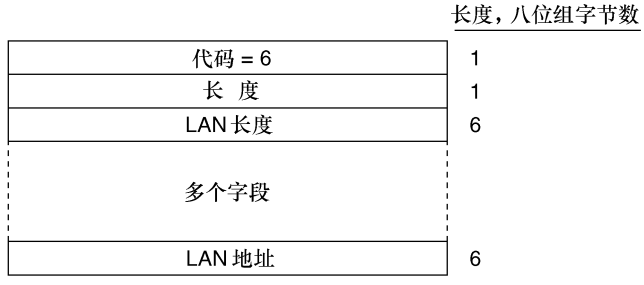


图 10-16　Hello PDU 数据包中的中间系统邻居 TLV 的格式

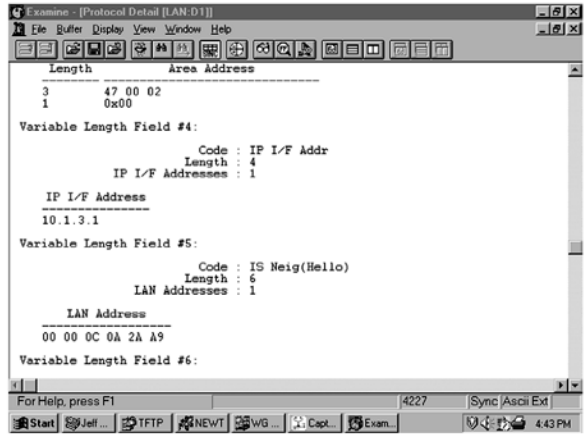


图 10-17　“Variable Length Field #5”部分显示了一个中间系统邻居 TLV

（3）填充 TLV

填充 TLV 是用来填充一个 Hello PDU 数据包，以使它达到允许的 IS-IS 大小的最小值（1492 字节），或者该链路的 MTU 大小。由于一个值字段的最大长度为 255 字节，因此经常会使用多个填充 TLV 字段。由于填充 TLV 的内容被路由器忽略，因此在它的值字段内可以设置任意的数值。在 Cisco 路由器中这些位的值都被设置为 0，如图 10-18 所示。

（4）认证信息 TLV

如图 10-19 所示，认证信息 TLV 只有在配置认证时才会使用。认证类型字段包含了一个 0～255 之间的数字，用来指定认证使用的类型，并且也指定认证值字段所包含的认证信息的类型。IOS 软件支持明文口令认证，或 HMAC-MD5 加密散列认证。明文口令认证是认证类型 1，而 HMAC-MD5 是认证类型 54。

如图 10-15 所示，“Variable Length Field #1”部分显示的就是一个认证信息 TLV。在这里，口令是“jeff”，并使用十六进制的 ASCII 码字符来表示。

（5）支持的协议 TLV

如图 10-20 所示，支持的协议 TLV 是由 RFC 1195 规定的，它用来指定 PDU 数据包的始发路由器所支持的协议——仅支持 CLNP、仅支持 IP 或同时支持这两种协议。从那个时候起，TLV 也用来对 IPv6 的支持。IPv4 NLPID 是 204（0xCC），而 IPv6 NLPID 是 142（0x8E）。

对于每一种所支持的协议，在 TLV 字段里都会有一个相应的 1 个八位组字节的网络层协议标识符（Network Layer Protocol Identifier，NLPID），这个标识符是由 ISO/TR 9577 规定的。IP 协议的网络层协议标识符是 0x81。在图 10-15 中的“Variable Length Field #2”部分显示的就是一个支持的协议 TLV。

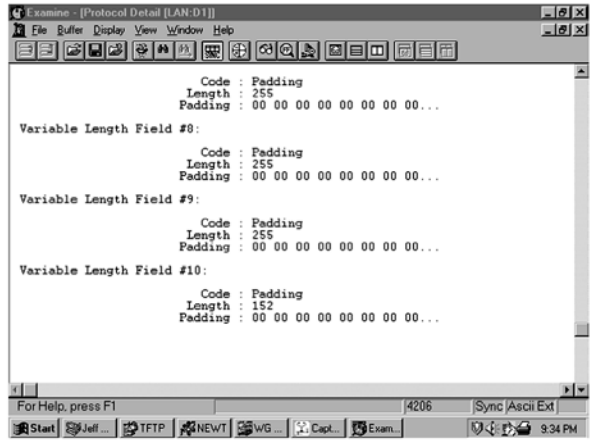


图 10-18　图 10-12 中显示的 Hello PDU 数据包末尾的可变长字段就是填充 TLV，它把图 10-12 中显示的 PDU 大小增大到 1497 个八位组字节。加上 3 个八位组字节的 LLC 头部和 18 个八位组字节的以太网头部，整个帧的大小就是最大为 1518 个八位组字节的以太网 MTU 的大小

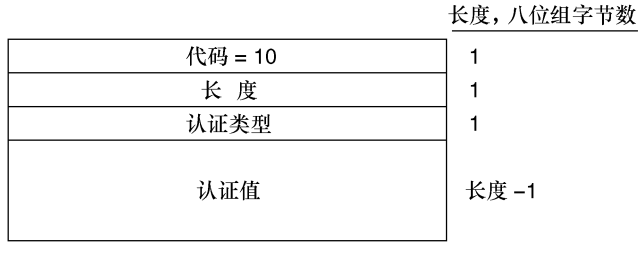


图 10-19　认证信息 TLV 的格式

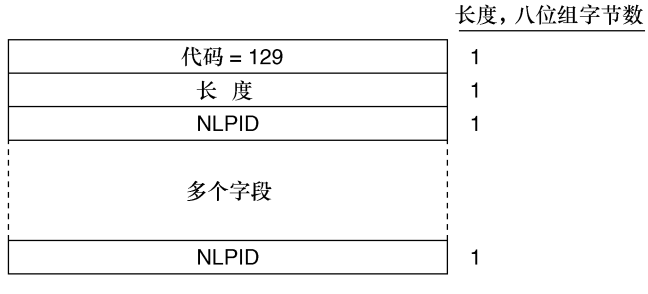


图 10-20　支持的协议 TLV 格式

（6）IP 接口地址 TLV

如图 10-21 所示，IP 接口地址 TLV 是指发出 PDU 数据包的接口地址或 IP 地址。因为长度字段是 1 个八位组字节，因而 IS-IS 路由器的接口理论上最多可以有 63 个 IP 地址。在图 10-17 中的“Variable Length Field #4”部分显示的就是一个 IP 接口地址 TLV，表明所捕获的 Hello PDU 数据包是由地址为 10.1.3.1 的接口发出的。

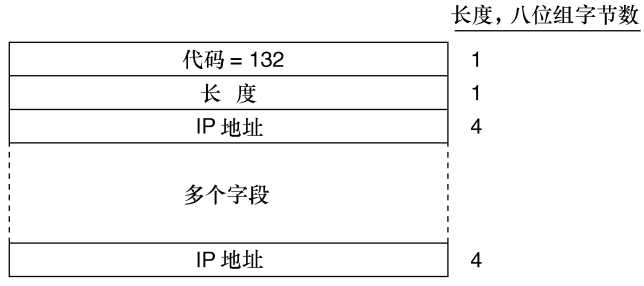


图 10-21 IP 接口地址 TLV 的格式

3. IS-IS 协议链路状态 PDU 格式

IS-IS 协议中 LSP 的功能在本质上和 OSPF 协议中 LSA 的功能是一样的。一台 L1 路由器把 L1 类型的 LSP 泛洪到整个区域，用来确定它的邻接关系和这些邻接关系的状态。一台 L2 路由器把 L2 类型的 LSP 泛洪到整个第 2 层的域，用来确定它与其他 L2 路由器的邻接关系，并标识出通告 L2 路由器能够到达的地址前缀。

如图 10-22 所示，显示了一个 IS-IS LSP 的格式。这个格式对于 L1 LSP 和 L2 LSP 都是相同的。



图 10-22 IS-IS LSP 的格式

- **PDU 长度**——是指整个 PDU 数据包的长度，用八位组字节数表示。
- **剩余生存时间（Remaining Lifetime）**——在确认一个 LSP 过期之前等待的秒数。
- **LSP ID**——可以是系统 ID、伪节点 ID 或 LSP 数据包的 LSP 编号。LSP ID 在“更新过程”小节中有更为详细的描述。
- **序列号**——是一个 32 位的无符号整数。
- **校验和**——对 LSP 内容的校验和。
- **P 位**——是指分段区域的修复位。虽然这一位存在于 L1 和 L2 的 LSP 数据包中，但

是它实际上只和 L2 的 LSP 数据包有关。当该位被设置为 1 时，表明始发路由器支持自动地修复区域的分段情况。Cisco IOS 软件并不支持这项功能，因此 Cisco 路由器始发的 LSP 数据包中，该位始终设置为 0。

- **区域关联位（ATT）**——是一个 4 位的字段，用来指明始发路由器是与一个或多个其他区域相连的。虽然区域关联位也存在于 L1 和 L2 的 LSP 数据包中，但是它们实际上只和 L1/L2 路由器始发的 L1 LSP 数据包有关。这 4 位用来表明相连的区域究竟使用哪一种度量方式。从左到右这 4 位依次表示：
  - 位 7：差错度量（Error）；
  - 位 6：代价度量（Expense）；
  - 位 5：时延度量（Delay）；
  - 位 4：缺省度量（Default）。

Cisco IOS 软件仅支持缺省度量，因此位 5～位 7 始终设置为 0。

- **过载位(OL)**——链路状态数据库的过载位。在一般情况下，该位设置为 0。收到过载位设置为 1 的 LSP 数据包的路由器，仍然会转发数据包到与这台始发过载信号的路由器直连的网络上，但是，不会再使用这台始发路由器作为转发数据包的透传路由器（transit router）了。
- **中间系统类型（IS Type）**——是一个两位的字段，用来指明始发路由器是 L1 路由器还是 L2 路由器：
  - 00=未用；
  - 01=L1；
  - 10=未用；
  - 11=L2。

一台 L1/L2 路由器可以根据收到的 LSP 是 L1 类型的 LSP，还是 L2 类型的 LSP 来确定设置这两位的值。

在 L1 的 LSP 数据包中可以使用下面的 TLV 字段：

- 区域地址（类型 1）；
- 中间系统邻居（类型 2）；
- 终端系统邻居（类型 3）；
- 认证信息（类型 10）；
- LSP 缓存大小（类型 14）；
- 扩展的 IS 可达性（类型 22）；
- IP 内部可达性信息（类型 128）；
- 支持的协议（类型 129）；
- IP 外部可达性信息（类型 130）；
- 流量工程路由器 ID（类型 134）；
- 扩展的 IP 可达性（类型 135）；
- 动态主机名（类型 137）；
- MT 中间系统（类型 222）；
- 多拓扑（类型 229）；
- IPv6 接口地址（类型 232）；

- MT 可达的 IPv4 前缀（类型 235）；
- IPv6 的 IP 可达性（类型 236）；
- MT 可达的 IPv6 前缀（类型 237）；
- 实验用（类型 250）。

在 L2 的 LSP 数据包中可以使用下面的 TLV 字段：

- 区域地址（类型 1）；
- 中间系统邻居（类型 2）；
- 区域分段指定的第 2 层中间系统（类型 4）；
- 前缀邻居（类型 5）；
- 认证信息（类型 10）；
- LSP 缓存大小（类型 14）；
- 扩展的 IS 可达性（类型 22）；
- IP 内部可达性信息（类型 128）；
- IP 外部可达性信息（类型 130）；
- 域间路由选择协议信息（类型 131）；
- 支持的协议（类型 129）；
- IP 接口地址（类型 132）；
- 流量工程路由器 ID（类型 134）；
- 扩展的 IP 可达性（类型 135）；
- 动态主机名（类型 137）；
- MT 中间系统（类型 222）；
- 多拓扑（类型 229）；
- IPv6 接口地址（类型 232）；
- MT 可达的 IPv4 前缀（类型 235）；
- IPv6 的 IP 可达性（类型 236）；
- MT 可达的 IPv6 前缀（类型 237）；
- 实验用（类型 250）。

如图 10-23 所示，显示了 L1/L2 路由器始发的一条 L1 类型的 LSP。

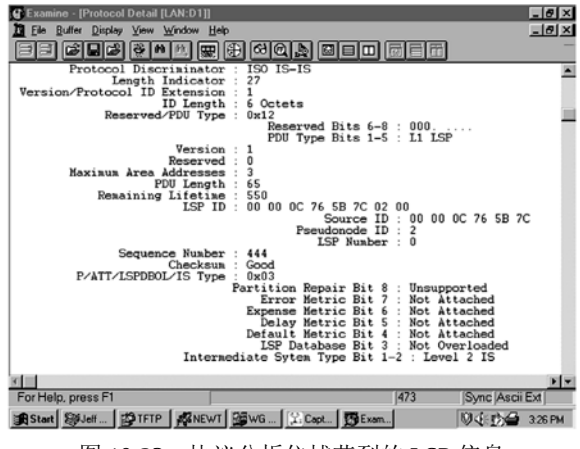


图 10-23　协议分析仪捕获到的 LSP 信息

（1）中间系统邻居 TLV（LSP）

中间系统邻居 TLV 用来在 LSP 中列出始发路由器的 IS-IS 邻居（包括伪节点），如图 10-24 所示。同时，它也列出了到达每一台邻居路由器的链路的度量。

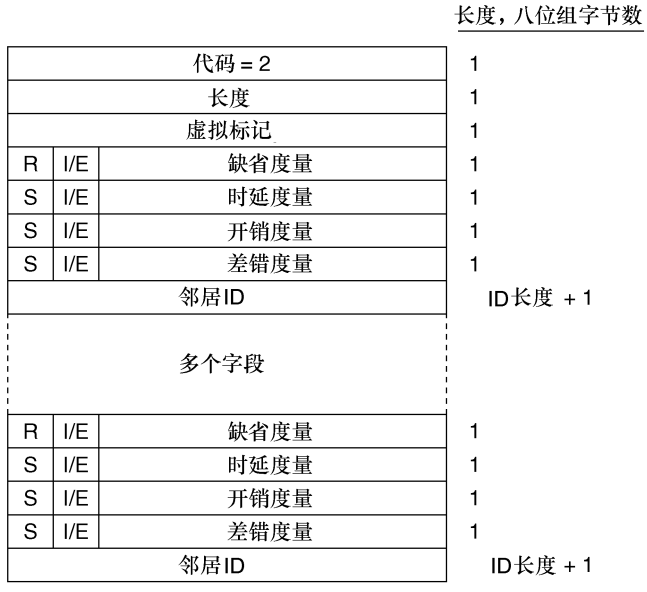


图 10-24　LSP 中的中间系统邻居 TLV 的格式

- **虚拟标记（Virtual Flag）**——这个字段虽然有 8 位长，但取值只有 0x01 或者 0x00。当这个字段设置为 0x01 时，表示该链路是一个用来修复分段区域的第 2 层类型的虚链路。这时该字段只和支持区域分段能力的 L2 路由器相关，由于 Cisco 路由器并不支持这一特性，因此在 Cisco 路由器始发的 LSP 中该字段始终设置为 0x00。
- **R 位**——是一个保留位，始终设置为 0。
- **I/E 位**——该位和每个度量有关，用来指明相关的度量是内部度量还是外部度量。该位在中间系统邻居 TLV 字段中没有意义，因为对 IS-IS 域来说所有的邻居路由器都被定义为内部的。因此，在中间系统邻居 TLV 字段中该位始终设置为 0。
- **缺省度量**——是一个 6 位的缺省度量，用来表示始发路由器到达所列出的邻居的链路度量，大小范围在 0～63 之间。
- **S 位**——该位和每一个可选度量有关，用来指明相关的度量是被支持（0）的，还是不被支持（1）的。由于 Cisco 路由器不支持其他所有的 3 种可选度量，因此这一位总是被设置为 1，并把相关的长度为 6 位的度量字段全部设置为 0。
- **邻居 ID**——是指邻居的系统 ID，再加上一个额外的八位组字节。如果该邻居是一台路由器，末尾的那个八位组字节就设置为 0x00。如果该邻居是一个伪节点，那么系统 ID 就是指定路由器（DR），末尾的那个八位组字节就是伪节点的 ID。

图 10-25 中显示了一个中间系统邻居 TLV 的部分信息。

（2）IP 内部可达性信息 TLV

如图 10-26 所示，IP 内部可达性信息 TLV 列出了与通告该 LSP 的、路由器直连的路由

选择域内的 IP 地址和相关的掩码信息。这个 TLV 使用在 L1 和 L2 类型的 LSP 中，但是从来不会出现在伪节点的 LSP 中。度量字段的含义是和中间系统邻居 TLV 中的度量字段一样的，但是它没有与可选度量相关联的 I/E 位。作为替代，这一位作为保留位并总是设置为 0。像中间系统邻居 TLV 一样，这个 TLV 字段中的 I/E 位也总是设置为 0，因为在这个 TLV 字段中通告的地址总是内部地址。如图 10-27 所示，显示了协议分析仪捕获到的一个 IP 内部可达性信息 TLV 的信息。

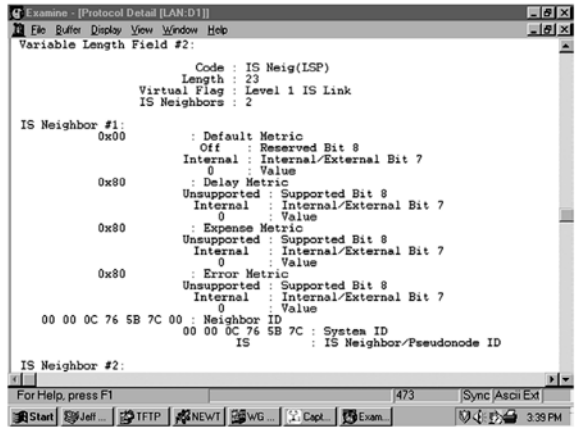


图 10-25　一个 LSP 中的中间系统邻居 TLV 的部分信息

			长度，八位组字节数
代码 = 128			1
长度			1
R	I/E	缺省度量	1
S	R	时延度量	1
S	R	开销度量	1
S	R	差错度量	1
IP 地址			4
子网掩码			4
多个字段			
R	I/E	缺省度量	1
S	R	时延度量	1
S	R	开销度量	1
S	R	差错度量	1
IP 地址			4
子网掩码			4

图 10-26　IP 内部可达性信息 TLV 的格式

（3）IP 外部可达性信息 TLV

IP 外部可达性信息 TLV 列出了到达 IS-IS 路由选择域外部的 IP 地址和相关的掩码，这些外部的目的地址可以通过始发路由器的某个接口到达。IP 外部可达性信息 TLV 的格式和图 10-26 中显示的 IP 内部可达性信息 TLV 的格式是相同的，但有一个例外，就是它的类型代码是 130。其中 I/E 位用来确定所有 4 种度量的类型——I/E=0 表示内部度量，而 I/E=1 表示外部度量。

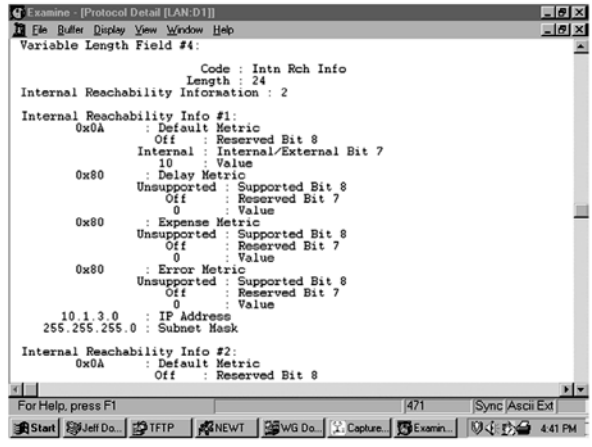


图 10-27 协议分析仪捕获到的一个 IP 内部可达性信息 TLV 的信息

（4）域间路由选择协议信息 TLV

如图 10-28 所示，域间路由选择协议信息 TLV 允许 L2 类型的 LSP 利用 IS-IS 域，来透传来自外部路由选择协议的信息。TLV 字段具有一个相同的用途，就是提供给 RIPv2、EIGRP 和 OSPF 协议数据包的路由标记（Route Tag）字段使用。路由标记将在第 14 章中介绍。

- **域间信息类型（Inter-Domain Information Type）**——指定了在可变长度的外部信息字段中包含的信息类型。如果该类型字段设置为 0x01，那么外部信息就会使用本地域间路由选择协议的方式。第 14 章包含了一个使用路由映射来设置这种本地信息的例子。如果该类型字段设置为 0x02，那么外部信息就是一个 16 位的自主系统号，用来标记所有后续的外部 IP 可达性条目，直到 LSP 的结尾或者下一个域间路由选择协议信息 TLV 的出现。

	长度，八位组字节数
代码 = 131	1
长度	1
域间信息类型	1
外部信息	可变的

图 10-28 域间路由选择协议信息 TLV 的格式

4. IS-IS 协议序列号 PDU 报文

SNP 通过描述数据库中的部分或者全部 LSP 的信息，对 IS-IS 链路状态数据库进行维护。在某些实例中，它们也用于请求 LSP，以及隐性或显性地确认接收到的 LSP。因此，SNP 具有和 OSPF 中链路状态请求、数据库描述和链路状态确认消息的功能。

如图 10-29 所示，一台指定路由器（DR）将会周期性地以组播方式发送完全序列号数据包（CSNP），来描述伪节点数据库中的所有 LSP 信息。由于存在 L1 类型和 L2 类型的数据库，因此完全序列号数据包也可能是 L1 类型或者 L2 类型的。有时，某些链路状态数据库的信息量太大，以至于 LSP 的信息无法使用单个完全序列号数据包来描述。基于这个原因，完全序列号包头最后两个字段作为起始 LSP ID（Start LSP ID）字段和结束 LSP ID（End LSP ID）字段，一起用来说明完全序列号数据包中描述的 LSP 的范围。如图 10-30 所示，图中显示了这两个字段是怎样使用的。在这个完全序列号数据包中将会描述完整的数据库信息，因此，LSP

ID 的范围将开始于 0000.0000.0000.00.00，并结束于 ffff.ffff.ffff.ff.ff。如果需要两个完全序列号数据包来描述这个数据库，那么第一个完全序列号数据包的范围可以是 0000.0000.0000.00.00～0000.0c0a.1234.00.00，而第二个完全序列号数据包的范围可以是 0000.0c0a.1234.00.01～ffff.ffff.ffff.ff.ff。

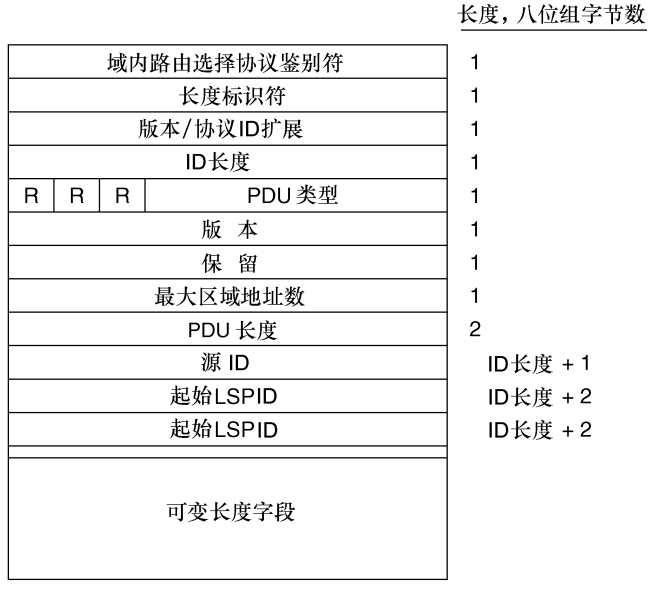


图 10-29　IS-IS 协议完全序列号数据包的格式

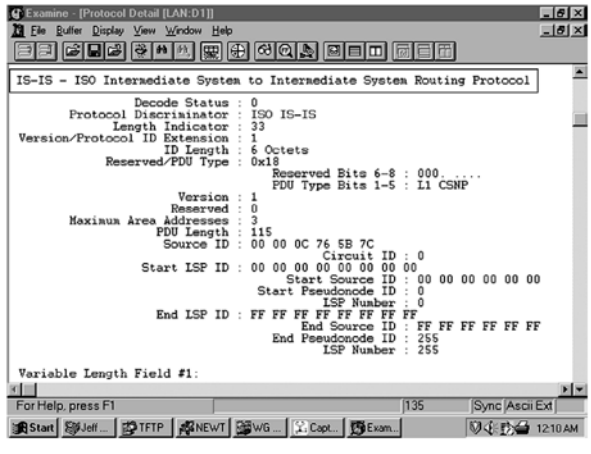


图 10-30　协议分析仪捕获的信息显示了一个 L1 类型的完全序列号数据包头部信息

如图 10-31 所示，一个部分序列号数据包（PSNP）除了像前面描述的只是携带部分 LSP 的信息，而不是整个数据库的信息外，其他与完全序列号数据包相似。因此，不必像完全序列号数据包那样需要起始和结束字段。一台路由器可以在一个点到点的子网上发送部分序列号数据包来确认收到的 LSP。在一个广播型的子网上，部分序列号数据包将会用来请求丢失的或者最新的 LSP。和完全序列号数据包一样，部分序列号数据包也存在 L1 类型和 L2 类型。



图 10-31 IS-IS PSNP 的格式

在 SNP 中，不论是 CSNP 还是 PSNP，也不管是 L1 类型还是 L2 类型，它们都只用到了 4 个 TLV 字段：

- LSP 条目（类型 9）；
- 认证信息（类型 10）；
- 可选的校验和（类型 12）；
- 实验用（类型 250）。

LSP 条目 TLV

如图 10-32 所示，LSP 条目 TLV 总结了一个 LSP 中列出的该 LSP 的剩余生存时间、LSP ID、序列号和校验和。这些字段信息不仅可以确定某个 LSP，而且可以完全地确定某个 LSP 的一个具体实例。如图 10-33 所示，显示了一个 LSP 条目 TLV 的部分信息。

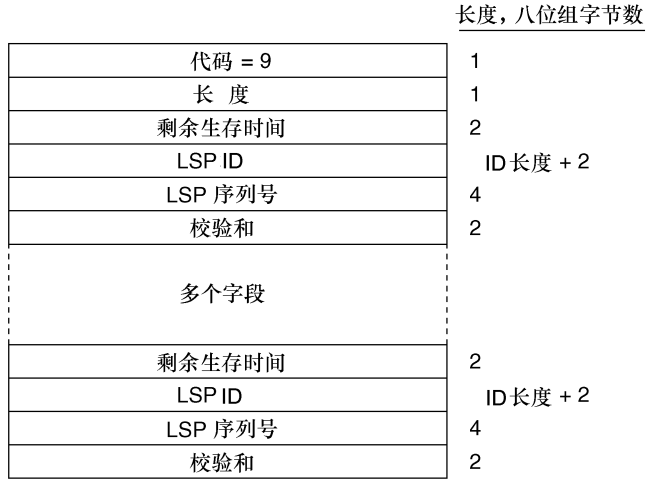


图 10-32 LSP 条目 TLV 的格式

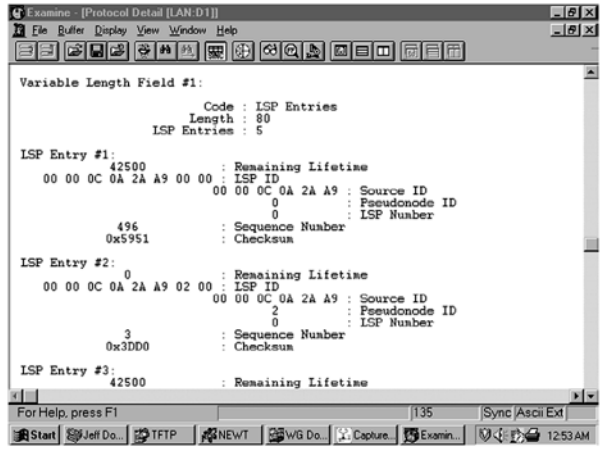


图 10-33 CSNP 的 LSP 条目 TLV 的部分信息，该 CSNP 如图 10-30 所示

10.1.5 IS-IS 的扩展属性

正如读者在表 10-4 中以及相关的讨论中所看到的，在 IS-IS 协议扩展用来支持 IPv4 时已经增加了许多扩展属性，这些扩展属性有的为了支持可选的性能，有的改善了 IS－IS 协议的基本机制。在本小节我们将探讨这些扩展特性中最重要的一些内容。

有关 IS-IS 协议基本机制方面的一个好的处理操作是，它和 OSPFv2 协议的 LSA 不同，IS-IS 协议碰到一个无法识别的 TLV 时，它只会忽略该 TLV。这一特性使 IS-IS 协议在处理扩展性和向后兼容性方面变得更加容易。假如读者需要迁移你的网络以便支持一些新的特性，或者只是希望有的路由器支持某个扩展属性而其他的路由器则不需要支持，那么担心对邻接关系的影响会比在 OSPFv2 协议中更少。在第 9 章中读者已看到，OSPFv3 至少采用了该属性的部分特性，这使 OSPFv3 协议比 OSPFv2 协议具有更好的扩展性和广泛性。

1. 三方握手

在邻居之间建立邻接关系之前，邻居必须确保它们之间形成了双向通信。确保该过程的形成称作*握手*。对于双向握手来说，仅仅简单地发送和接收 Hello 数据包是不够的。因为你的邻居可能并未接收你所发出的 Hello 数据包。因此，对于你的邻居是否收到你所发出的 Hello 数据包作一些明确的确认是必要的，这就是三方握手（three-way handshaking）。OSPF 协议总是使用三方握手，如果从邻居收到了 Hello 数据包，本地路由器就会在它所发送的 Hello 数据包中列出所有这样的邻居。因此，如果一台 OSPF 路由器看到它的 RID 在所收到的 Hello 数据包的邻居字段中列出，那么它就可以知道这个 Hello 数据包的始发路由器已经收到了这台路由器的 Hello 数据包。这时，双向通信状态就可以得到确认。

在广播型的网络中，IS-IS 协议也是使用三方握手机制的。如果从邻居收到了 Hello 数据包，本地路由器所发送的 LAN Hello 数据包就会在它的 IS 邻居 TLV 列出所有这样的邻居。而 TLV 也是在 OSPF Hello 数据包里列出邻居来达到同样的目的：IS-IS 路由器在它所收到的 LAN Hello 数据包的 IS 邻居 TLV 中看到它自己的 SysID，那么它就会认为双向通信状态已经确认建立了。

但是，正如已经所提及的那样，点到点 Hello 数据包是不携带 IS 邻居 TLV 的。ISO 10589

规定通过点到点链路只能建立双向握手，并且要求点到点的传输介质应该是可靠的。但是在现实当中，点到点链路可能经常是不可靠的；当然，我们也不会仅仅因为在网络中可能存在一条不能满足某些含糊的可靠性要求的链路，就不把 IS-IS 协议作为一个可能的 IGP 协议选择。

为了修正这个问题，RFC 3373 指定了一个点到点三方邻接 TLV（Point-to-Point Three-Way Adjacency TLV）。如图 10-34 所示，它是一个类型为 240 的 TLV，列出了发起它的路由器所知道的所有邻居的 SysID，并且它也指出了始发路由器在该链路上可能的邻接关系状态：正常、初始化，或失效（Up、Initializing，或 Down）。

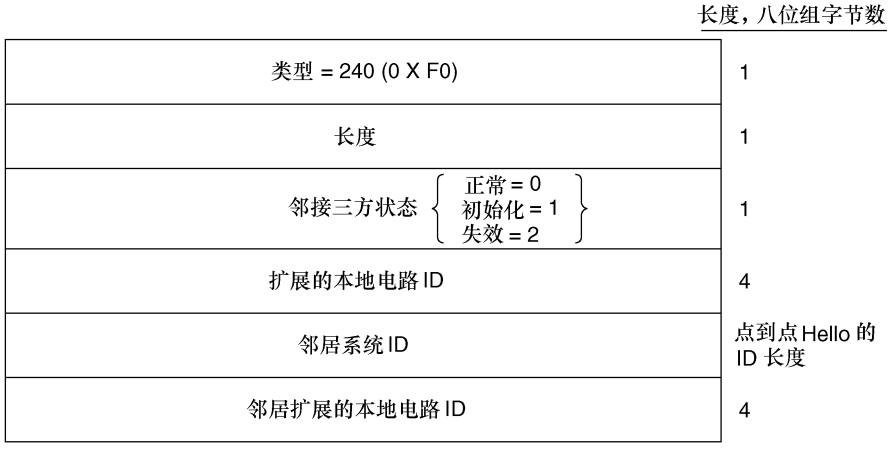


图 10-34 点到点三方邻接 TLV 的格式

2. 跨域范围（Domain-Wide）的前缀分发

正如读者在本章前面的部分了解到的，L1 区域的缺省特性和 OSPF 的完全末梢区域非常相似。换句话说，从区域 L2 到 L1 不通告任何前缀；相反的，L1/L2 路由器设置 ATT 位，并且 L1 路由器加载一条到达最近的 L1/L2 路由器的缺省路由。事实上，RFC 1195 规定，禁止从 L2 区域向 L1 区域通告前缀。

但是，这样的规定也不总是可接受的。如果存在多台 L1/L2 路由器，有时我们更希望选择到达目的地最近的那台 L1/L2 路由器，而不是选择所在区域离出口最近的那台 L1/L2 路由器。为了达到这个目的，L1 路由器必须拥有相应前缀的必要信息和它们在本地区域外的路径代价；这也就意味着必须把这个前缀从 L2 区域通告到 L1 区域。在 IS-IS 协议的专业术语中，这种情况称为“路由泄漏（route leaking）”。

在这里可能存在潜在的困难（这也是 RFC 1195 中禁止像这样“向下的”路由泄漏的原因），如果在 L1 区域内具有多台 L1/L2 路由器——这很可能是读者起初为什么从 L2 区域向 L1 区域泄漏路由的原因，这可能会引起潜在的区域间路由选择环路问题。如图 10-35 所示，图中演示了这个问题的存在。L1/L2 路由器 Hague 通过一些 L2 LSP 学习到前缀 192.168.1.0/24 的信息。如果需要将该前缀信息通告到它的 L1 区域内，那么它必须使用一个 IP 内部可达性 TLV 在它的 L1 LSP 中通告这个前缀信息。但这个 L1 TLV 在 L1 区域内扩散时，区域内的另一台 L1/L2 路由器 Rotterdam 将会收到它。然而，由于这条前缀信息是在 L1 LSP 中扩散的，路由器 Rotterdam 没有办法得知它是始发于 L1 区域外部的前缀。因此，这台路由器将会通过一条 L2 LSP 将这条前缀信息通告回 L2 区域。这样，在 L2 路由器上就存在路由选择环路了：L2 路由器认为到达前缀 192.168.1.0/24 的路径存在于 L1 区域内。

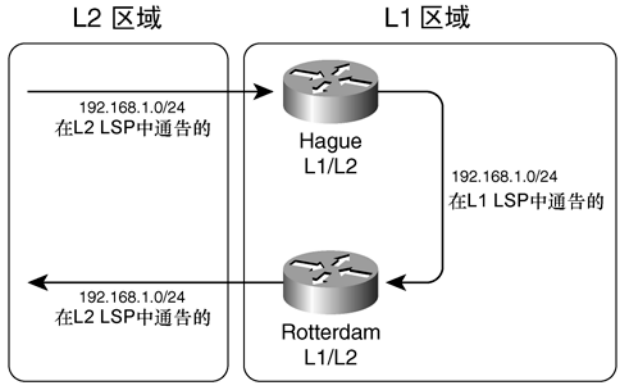


图 10-35 根据基本的 RFC 1195 规则，从 L2 区域向 L1 区域泄漏前缀信息会存在该前缀被通告回 L2 区域的风险，这样会产生一个路由选择环路

OSPF 协议并不存在这种问题，因为在 OSPF 协议中，对于非骨干区域外部的前缀都是通过类型 3 的 LSA 通告的。在一个非骨干区域内的其他 ABR 路由器将不会把从类型 3 的 LSA 学到的前缀信息通告到区域 0 中。

在 RFC 2666 中，通过在 IP 内部可达性 TLV 和 IP 外部可达性 TLV 中定义一个称为“Up/Down（U/D）”的位，来为该问题提供了一种解决方案。在图 10-26 中显示的格式中，查看 IP 内部可达性 TLV 的格式（并请记得 IP 外部可达性 TLV 的位格式也是相同的），请注意这个八位组包含了一个 I/E 位和一个 6 位的缺省度量字段，它们开始于一个保留位。在图 10-36 中，这个保留位就变成了 U/D 位。在指定了类型字段后，就可以为 IP 内部可达性 TLV 和 IP 外部可达性 TLV 定义该一位了。

当一台 L1/L2 路由器从 L2 区域向 L1 区域通告一条路由时，它将会设置这个 U/D 位。任何其他的 L1/L2 路由器收到在 L1 LSP 里设置了 U/D 位的前缀，都不会在 L2 LSP 中通告这个前缀。如果有一台 L1/L2 路由器不能识别这个 U/D 位，那么它将忽略这一位。因此，如果读者正在使用 L2 到 L1 的路由泄漏设计，那么确保你所有的 L1/L2 路由器都能识别这个扩展特性是很重要的。

在本章稍后的 10.2.8 小节中，将会示范怎样设置 L2 到 L1 的路由泄漏技术。

3. 扩展度量（Wide Metrics）

流量工程（Traffic Engineering，TE）是一个主要与多协议标签交换（Multiprotocol Label Switching，MPLS）网络相关联的功能，在 MPLS 网络中的数据包的一些子集能够依赖用户指定的约束条件以不同的方式进行转发。换句话说，不像 IGP 协议那样总是选择单一的最短路径通过一个网络，业务流（traffic flows）能够在不同的路径上传播。这种做法既可以帮助网络提高整个带宽的利用率，也能够提供区分服务——例如，确保时延敏感的流使用最短的路径，而其他的业务流使用较长的路径。

流量工程的一个关键是需要使用比度量更为详细的接口参数进行通信；使用用于共享路径设计的 IGP 协议和用于共享这些 TE 接口参数的接口信息就变得有意义了。OSPF 与 IS-IS 协议都被扩展以便能够携带 TE 接口参数；对于 IS-IS 协议来说，使用以下两种新的 TLV：

- 扩展的 IS 可达性（类型 22）；

- 扩展的 IP 可达性（类型 135）。

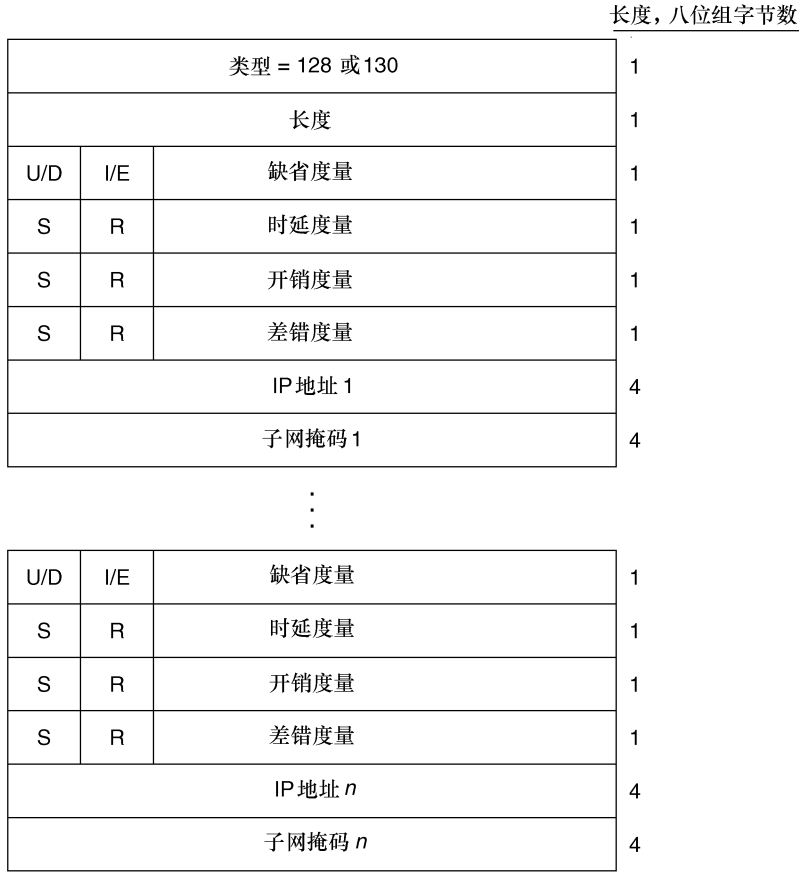


图 10-36 缺省度量字段的第八位，原来是保留的，现在被重新定义为 Up/Down 位

在 RFC 3784 中，详细描述了这些 TLV 以及它们所支持的 TE 参数。但是，MPLS 流量工程的讲解已经超出了本书的讲述范围。然而，我们关注这两个新的 TLV 是因为它们提供了一种新的重要性能，并且能够用在流量工程以外的地方。

正如本章前面所提及的，在 IS-IS 最初的格式里，有关 IS-IS 协议的一个问题是：仅仅 6 位的度量字段不能满足一个大型网络所需要的足够的度量尺度。这两个新的 TLV 则可以支持一个更大的度量尺度，因此称为“扩展度量（wide metrics）”。扩展的 IS 可达性 TLV 使用一个 24 位的度量字段，而扩展的 IP 可达性 TLV 使用一个 32 位的度量字段。

扩展的 IS 可达性 TLV 的格式如图 10-37 所示。当启用扩展度量时，这个 TLV 就用来在 LSP 中替代类型 2 的 IS 邻居 TLV。对于我们来说所关注的是类型 2 的 TLV，它列出了进行 SPF 计算的邻居以及它们的度量，注意是 24 位的度量字段。子 TLV（sub-TLV）字段是用于 TE 的参数信息，和这里的讨论无关。但是，这种 TLV 是允许嵌套的 TLV，就是说，是子 TLV（sub-TLV），或嵌入其他 TLV 内的 TLV，注意到这一点是有用的。这样的 TLV 将会为开发人员带来更多的灵活性。

扩展的 IP 可达性 TLV 的格式如图 10-38 所示。当启用扩展度量时，这个 TLV 就用来替代 IP 内部可达性信息和 IP 外部可达性信息 TLV。因此，这个 TLV 可以出现在 L1 和 L2 的

LSP 中。在图示中，读者可以看到通过这个 TLV 通告的前缀具有一个 32 位大小的度量值。对于从 L2 到 L1 区域的路由泄漏也具有一个 Up/Down 标记位。而在扩展的 IS 可达性 TLV 中，子 TLV 仅仅与流量工程相关联（S 位表示子 TLV 的当前状态）。



图 10-37 扩展的 IS 可达性 TLV

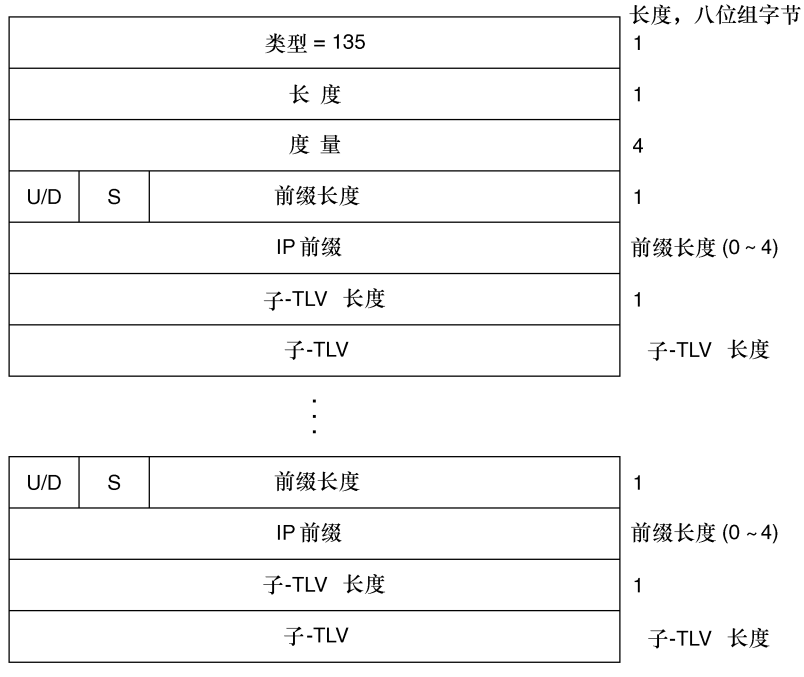


图 10-38 扩展的 IP 可达性 TLV

在 IOS 软件系统中启动扩展度量的命令是 **metric-style wide**。在本章后面的 10.2.6 小节和 10.2.7 小节将会讲述使用扩展度量的例子。

4. IPv6 的 IS-IS 路由选择

在第 9 章，读者已经看到 OSPF 协议需要一个全新的版本来支持 IPv6。而另一方面，IS-IS 协议通过两个新的 TLV 就可以很容易地进行扩展来支持 IPv6 了。在撰写本书时，IS-IS 协议的 IPv6 扩展属性依然是在 Internet 草案（Internet-Draft）阶段，但是很快它就可能形成一个 RFC——可能就在读者正在阅读本章的时候。

IS-IS 协议通过在它支持的协议 TLV 中包含 IPv6 NLPID 142（0x8E）来支持 IPv6。这两个支持 IPv6 的 TLV 是：

- IPv6 可达性（类型 236）；
- IPv6 接口地址（类型 232）。

如图 10-39 所示，IPv6 可达性 TLV 可以说是为了实现 IPv4 中 IP 内部可达性信息与 IP 外部可达性信息 TLV 相同的功能的。但是，同样相似的功能与类型 135 的扩展 IP 可达性 TLV 更加相近，这是因为以下两个原因：

- 这个 TLV 是用于同时通告内部和外部的前缀信息的；
- 这个 TLV 包括了一个 32 位的度量字段，因此支持扩展度量特性。



图 10-39　IPv6 可达性 TLV

从图中可以看到，对于每一个前缀都有一个 32 位的度量字段和用于从 L2 向 L1 路由泄漏的 Up/Down 位。X 位表示这个前缀是由内部始发的（X＝0），还是由外部始发的（X＝1）。S 位表示是否存在子 TLV。

如图 10-40 所示，图中显示了 IPv6 接口地址 TLV，它的功能等价于 IPv4 中类型为 132 的 IP 接口地址 TLV：它通告了始发这个 TLV 的接口的地址信息。并且和 IPv4 中相对应的，它能够同时被 Hello 数据包和 LSP 数据包携带。当 TLV 出现在 Hello 数据包中时，它所通告的地址就是始发接口的链路本地地址。如果 TLV 出现在 LSP 数据包中，那么它所通告的地址就不是始发接口的链路本地地址，而是地区或全球范围的地址。

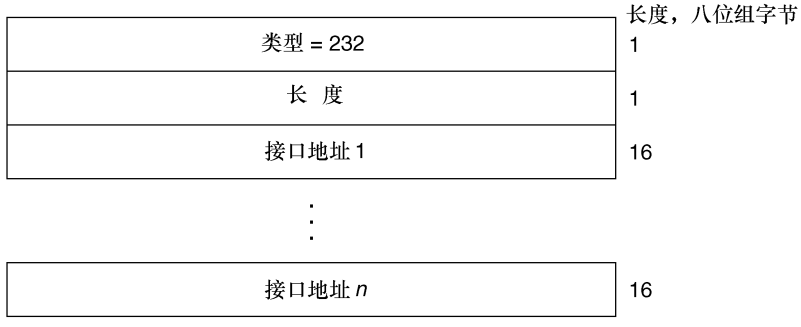


图 10-40 IPv6 接口地址 TLV

即使 IP 接口地址 TLV 能够携带最多 63 个 32 位的 IPv4 地址，但 IPv6 地址是这个长度的 4 倍，因此，这就限制了 IPv6 的接口地址 TLV 最多可以携带 15 个 IPv6 地址。

在本章随后 10.2.6 小节中讲述了一个有关 IPv6 的 IS-IS 的配置案例。

5. 动态主机名交换

使用 IS-IS 协议工作存在一个操作上的困难，就是它很难将多台不同路由器显示的 SysID 与正确的路由器相关联。标识 IPv4 地址是不容易的，但是我们往往为了方便而给它取一个长一点的比较熟悉的名字。对于 SysID，就像示例 10-1 中显示的那样确实存在实际的挑战和困难。

在 RFC 2763 中提供了一个动态主机名 TLV（类型 137）的说明，它提供了一个解决上述困难的简单方案。这个简单的 TLV 在它的数值字段提供了最多 255 位的空间用来携带一个路由器名字的 ASCII 码。通常情况下，支持这个扩展的实现方式是，将所配置的路由器主机名插入到这个字段中去；然后就可以在任何非伪 LSP 中携带这个 TLV。如示例 10-8 所示，当需要显示一台路由器时，动态主机名 TLV 中的 ASCII 码值就会用来替代它的 SysID，以便更加容易的进行识别。

示例 10-8 这个示例重新显示了示例 10-1 中 IS-IS 邻居表的内容，可以看出动态主机名交换机制的好处

```
Brussels#show clns is-neighbors

System Id      Interface   State  Type Priority  Circuit Id         Format
Dublin         Se0         Up     L1   0         06                 Phase V
Amsterdam      Et1         Up     L1   64        Brussels.03        Phase V
Rome           Et0         Up     L2   64        Brussels.02        Phase V
London         Et0         Up     L1L2 64/64     Brussels.02        Phase V

Brussels#
```

6. 多拓扑结构

现代 IP 网络通常会在一个通用的网络基础设施上提供多种网络服务，一般通过基本的网络服务模块如 IPv4、IPv6 和多播等实现。使用一个通用的网络基础设施提供所有的网络服务，而不愿为每一种服务组建各自独立的网络，这会在投资成本上带来巨大的好处：大大降低了投资开支，较少的设备维护，简化的操作流程，以及为数不多的操作人员。但是，各个基本的服务模块需要不同的路由选择拓扑结构。也许 IPv4 是无处不在的运行，但是 IPv6 应该只是局限在 IPv4 域的某个部分。多播服务可能也需要局限于 IPv4 域的某些子域，但是却与 IPv6 具有不同的拓扑结构。

多拓扑（Multi Topology，MT）路由选择允许读者在一个通用的网络基础设施上组建这些路由选择的网络子集。各个不同的拓扑可以使用多拓扑 ID（MT ID）来标识。当前定义的 MT ID 列在了表 10-5 中。这些 MT ID 是设计在路由器接口上的，用来表示这个接口属于什么拓扑；每一个接口都可以具有一个或多个 MT ID。邻接关系不是具体到特定的 MT 的，而仍然是在所有的 IS-IS 邻居之间建立的。但是，这些 LSP 都会被标记上相应的 MT ID，并且每一个拓扑结构都会运行各自的 SPF 计算。在每一台路由器上，都会为每一个拓扑维护各自的路由表，而且路由表中的每一个路由条目都是基于各自的 SPF 计算而产生的。

表 10-5 用于 MT IS-IS 的多拓扑 ID

MT ID	拓扑
0	标准（缺省）拓扑（IPv4 单播路由选择拓扑）
1	IPv4 带内管理（in-band management）
2	IPv6 单播路由选择拓扑
3	IPv4 多播路由选择拓扑
4	IPv6 多播路由选择拓扑
5～3995	IETF 统一保留
3996～4095	保留，用于开发、试验以及专有特性

IS-IS 支持的 MT 路由选择目前还处于 Internet 草案阶段。在这个草案中指定了几个支持 MT 的 TLV：

- 多拓扑中间系统（类型 222）；
- 多拓扑（类型 229）；
- 多拓扑可达的 IPv4 前缀（类型 235）；
- 多拓扑可达的 IPv6 前缀（类型 237）。

当一台 MT IS-IS 路由器发送 Hello 数据包时，它会包含一个或多个多拓扑 TLV，用于始发接口所属于的每一个拓扑，如图 10-41 所示。如果一台邻居路由器没有包含 Hello 数据包中的所有的多拓扑 TLV，那么这个邻居就会被认为仅仅属于缺省的 IPv4 拓扑（MT ID 为 0）。在一个点到点的链路上，如果两个邻居之间没有任何共同的 MT ID，那么在这两个邻居之间将不会形成邻接关系。但是，这和在一个广播型链路上的情况是不同的，在广播型链路上即使邻居之间没有任何共同的 MT ID，邻居之间也会形成一个邻接关系。这是因为指定路由器的选举是独立于 MT IS-IS 扩展特性的，一台不支持扩展特性的路由器依然能够被选举为指定

路由器（DR），因此在相同的层次的所有路由器一定是邻接的。

多拓扑 TLV 不仅可以由 Hello 数据包携带，也可以由 LSP 数据包携带。正如图 10-41 所显示的，这个 TLV 可以分别为每一个拓扑表示过载（使用 O 位）和 L2 关联（使用 A 位）。

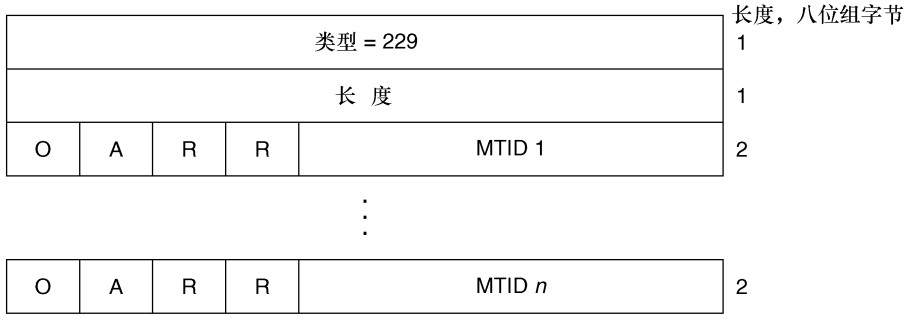


图 10-41 多拓扑 TLV 的格式

多拓扑中间系统 TLV 的格式如图 10-42 所示，它和前面讲到的扩展 IS 可达性 TLV 几乎完全相同，除了对于每一个始发接口所属的每种拓扑都要有各自单独的多拓扑中间系统 TLV 外，它和扩展的 IS 可达性 TLV 一样，也服务于同样的目的。



图 10-42 多拓扑中间系统 TLV 的格式

多拓扑可达的 IPv4 前缀 TLV 的格式如图 10-43 所示，多拓扑可达的 Ipv6 前缀 TLV 的格式如图 10-44 所示。多拓扑可达的 IPv4 前缀 TLV 和多拓扑可达的 IPv6 前缀 TLV，在功能上分别与扩展的 IP 可达性 TLV 和 IPv6 可达性 TLV 相同。它们同样通告前缀，但同每一个给定的拓扑相关联。

在本章后面的 10.2.7 小节中讨论了一个配置多拓扑模式 IS-IS 的示例。

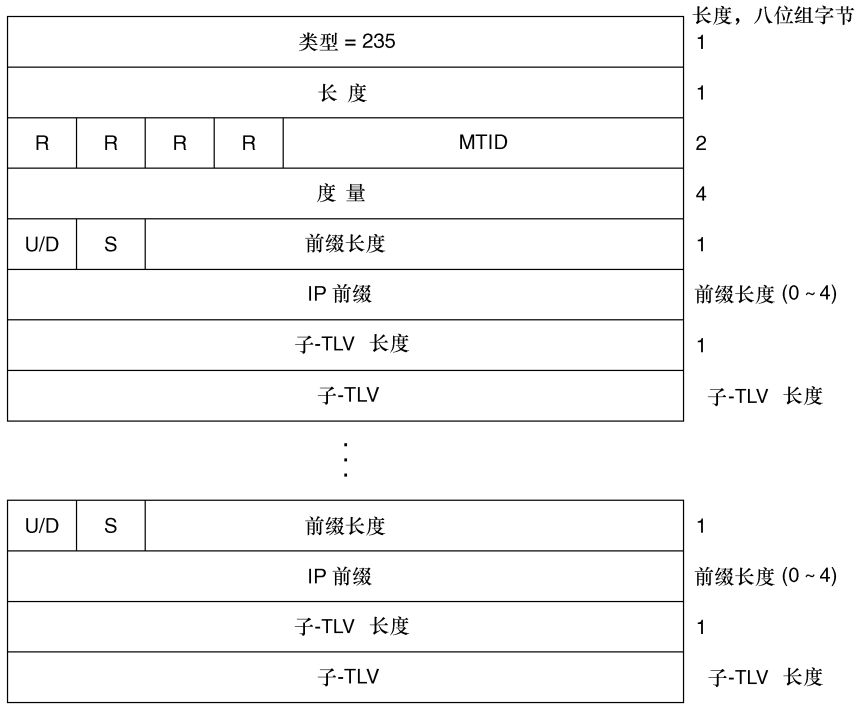


图 10-43 多拓扑可达的 IPv6 前缀 TLV 的格式

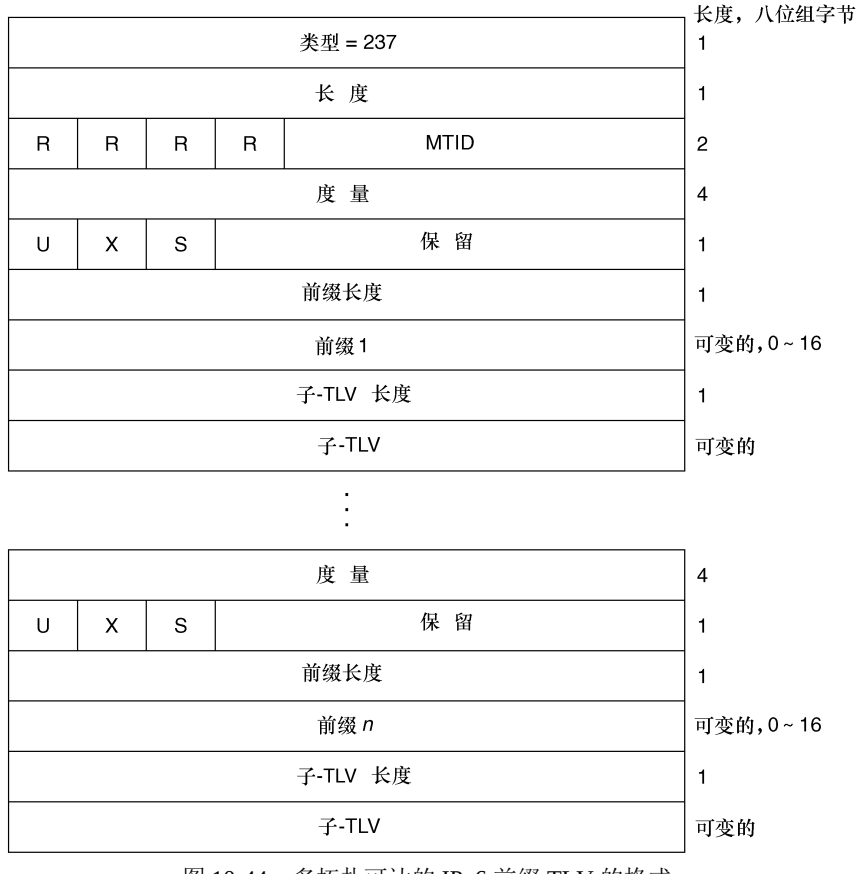


图 10-44 多拓扑可达的 IPv6 前缀 TLV 的格式

7. Mesh 组

虽然有利于 SONET 和 MPLS 的技术正在快速消退，但是帧中继和 ATM 网络却依然经常作为许多大型网络的核心传输介质。帧中继和 ATM 网络经常使用的拓扑结构如图 10-45 显示的那样，是用于提供连接性的全网状连接的虚电路。但是，过多的网状连接或全网状连接的网络容易受到过重的泛洪扩散负载影响。全网状连接是指所有的路由器都与其他所有的路由器相连接。

由于每一台路由器和其他任何一台路由器都进行连接，那么，在一台路由器扩散一条 LSP 时，其他路由器会立即收到该 LSP，如图 10-46 所示。这是进行泛洪扩散所需要的。

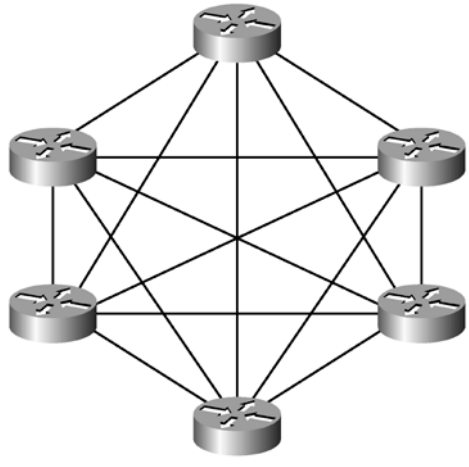


图 10-45 承载 IP 网络的 ATM 和帧中继网络基础设施经常配置成每一个节点都和其他所有节点进行连接，也就是说这些节点之间是全网状连接的

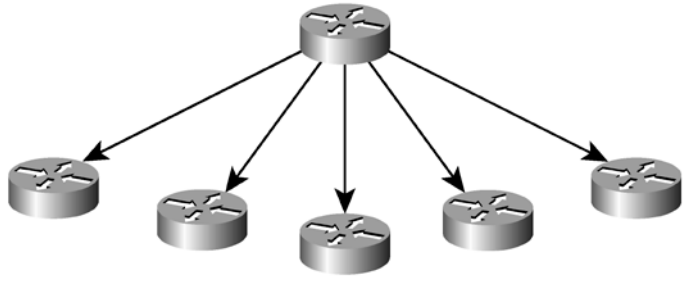


图 10-46 在一个全网状连接的网络里，来自某台路由器的传播会立即被其他所有路由器接收到

现在存在的问题是，接收路由器和其余的路由器之间都存在链路连接，而它们并没有办法知道所扩散的 LSP 已经被其他路由器收到。因此，根据泛洪扩散的基本水平分割规则，这些路由器将会把 LSP 从没有收到这个 LSP 的其他每个接口扩散出去，如图 10-47 所示。这个例子显示，如果某个网络上具有 *n* 台路由器，那么网络中的每一台路由器都要扩散 *n-2* 条不必要的 LSP：除了它本身和收到这条 LSP 的路由器，LSP 将会被扩散到其余的每一台路由器。根据二次方程式，可以计算出扩散了 $(n-1)(n-2)$ 或 n^2-3n+2 条不必要的 *LSP*。[1]

[1] 初级中学的数学教师喜欢用“语言问题”难为他们的学生，因此在这里等价的表达方式是：除了始发 LSP 的路由器，每一台路由器都会扩散 *n-2* 条不必要的 LSP。

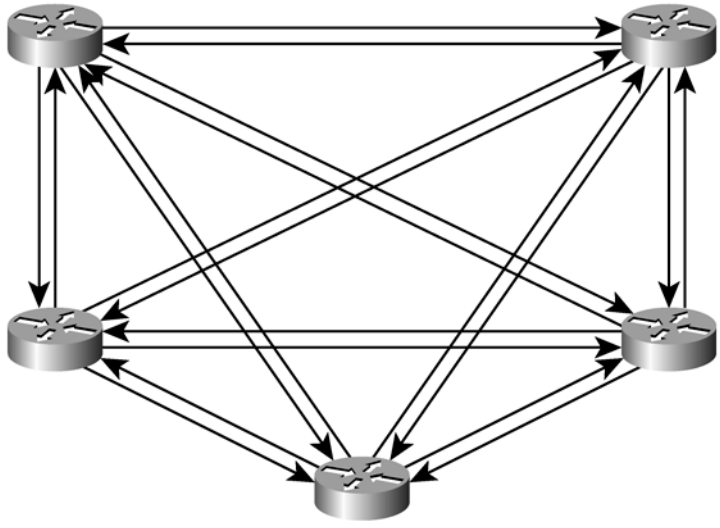


图 10-47 即使每一台路由器已经收到了扩散的 LSP，它们仍然会根据泛洪扩散的规则，来发送该 LSP 到它的每一个邻居，除了扩散该 LSP 的那个邻居

我们在这里检查一下上面网络的情况，它额外的扩散负载并不是太重：对于 6 台路由器，产生了 $n^2-3n+2=20$ 条不必要扩散的 LSP。但是，假定一个网络存在 100 台全网状连接的路由器，那么就会产生 $n^2-3n+2=9702$ 条不必要扩散的 LSP。而且这还仅仅是来自一台路由器刷新它的 LSP。考虑到每一台路由器都会刷新它们的 LSP，同时每台路由器可能有多条 LSP，另外除了刷新计时器触发的 LSP 扩散外还有其他一些活动，我们可以看出每一次泛洪扩散所产生的不必要的 LSP 是多么巨大。当然，在现代网络中，这个通信量与整个网络要处理的所有数据包的通信量相比也许不是不可接受的，但是对于许多网络架构设计师来说，任何无效率的设计都违背了审美的情趣。

对于这些为了简单地寻找更加简洁设计的工程师来说，在 RFC 2973 中提供了有关 IS-IS 协议的 *mesh 组（mesh groups）*。Mesh 组的机制允许读者在清楚地确认路由器的邻居已经收到某个 LSP 时可以避免再扩散这个 LSP。Mesh 组将一个接口定义为下面 3 种模式之一：

- 不活动模式（Inactive）；
- 阻塞模式（Blocked）；
- 设置模式（Set）。

不活动模式简单的来说就是虽然路由器支持 mesh 组，但是没有 mesh 组在接口上启用，LSP 是正常扩散的。

但一个接口处于阻塞模式时，它就不再扩散任何 LSP。如图 10-48 所示，图中显示了怎样通过阻塞接口来减少图 10-45 中网络的扩散负载。在这里，图 10-45 中全网状连接的拓扑已经简化成一个“环形”扩散拓扑；当然，正常的数据包转发依然能够使用全网状连接的网络结构。在这个扩散拓扑中，每一台路由器都有两个邻居而不再是 $n-1$ 个邻居了。这样虽说还有一点不必要的扩散，但是这已经大大减少了图 10-47 中所显示的情况。

对于图 10-48 中部分接口被阻塞的拓扑也是需要权衡的，或者说总是需要权衡的。首先，当每一台路由器只和其他两台路由器具有非阻塞的扩散连接时，如果这两条连接都失效了，那么，即使这台路由器还有其他完好的连接到达邻居，扩散也将会被破坏。其次，

就是对收敛时间的影响。图 10-46 中全网状连接的扩散拓扑，可以确保每一台路由器在要扩散的 LSP 一经发出时就可以被收到；但是在图 10-48 中，部分接口扩散被阻塞的拓扑意味着，当某些情况下要扩散的 LSP 需要通过一台或几台路由器传输后才会被所有的路由器接收到。

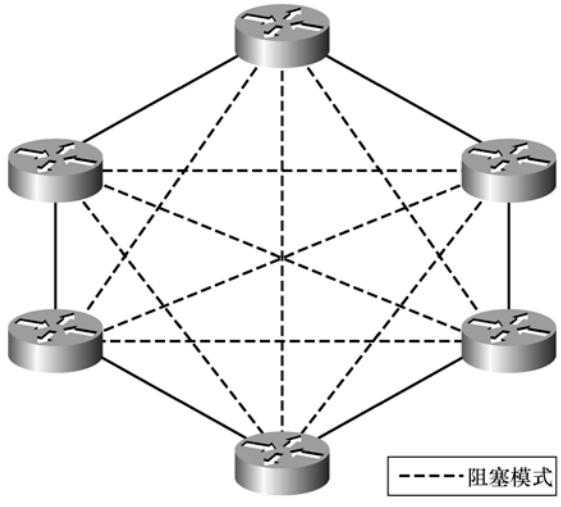


图 10-48 在一些接口上阻塞扩散可以减少整个扩散的负载

设置模式与阻塞模式相比，给我们提供了更多的灵活性，在仍然降低扩散负载的情况下（虽然没有阻塞模式降低的那么多），提供了在阻塞模式中存在的冗余性降低与收敛时间增加的一种折衷方案。在设置模式下，可以定义带有编号的 mesh 组，并分配相应的接口到对应的组。在收到一条 LSP 时，这条 LSP 就在属于某个编号 mesh 组的接口上被收到。然后，除了那些与接收到这条 LSP 的接口同属于一个组的接口外，这条 LSP 会在其他所有的接口上扩散出去。

在图 10-49 中，显示了将图 10-45 中的拓扑配置成了两个 mesh 组：组 1 和组 2。假设一台路由器正在发起一条 LSP，它把这条 LSP 扩散到每一个接口上而不管这个接口属于哪个组；这种最开始的扩散看起来比较像图 10-46 中显示的情形。一些邻居在属于组 1 的接口上收到这条 LSP，而一些邻居则在属于组 2 的接口上收到这条 LSP。

接着，接收 LSP 的路由器将这条 LSP 从与接收该 LSP 的接口所属的组不同的 mesh 组的接口上扩散出去。如果 LSP 是在属于组 1 的接口上收到的，那么它只能从属于组 2 的接口上扩散出去，反之亦然，如图 10-50 所示。所有这些扩散的 LSP 当然是不必要的，每一台路由器在始发路由器最开始扩散该 LSP 时就收到了它的一份拷贝。与图 10-48 中的扩散模型相比，可以看出虽然这些不必要的扩散比阻塞模式加重了一些负载，但是它仍然比图 10-47 中显示的负载小许多。而且，由于最开始的扩散到达所有的路由器，因此收敛时间没有受到影响。

不活动模式、阻塞模式和设置模式可以混和使用来组成一个复杂的拓扑，以便获取读者所希望的扩散模型。在实际使用中，阻塞模式比设置模式用的更为普遍。但是，如果读者真的选择使用 mesh 组，就应该意识到在所有的情况下，都是以牺牲全网状连接的扩散拓扑所具有的健壮性为代价来降低泛洪扩散的负载的。

在 IOS 软件系统中，读者可以在路由器的接口上使用命令 **isis mesh-group blocked** 来阻

塞扩散，或者使用命令 **isis mesh-group** *number* 来把某个接口分配到已编号的 mesh 组中。

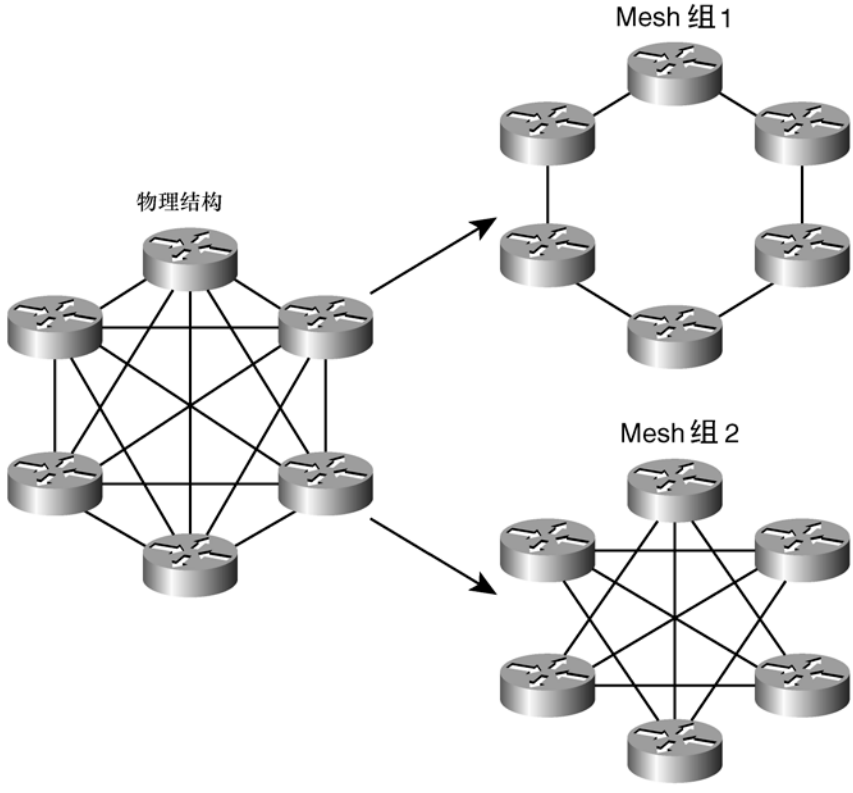


图 10-49 设置模式允许我们创建已编号的 mesh 组，并将接口分配到这些组内

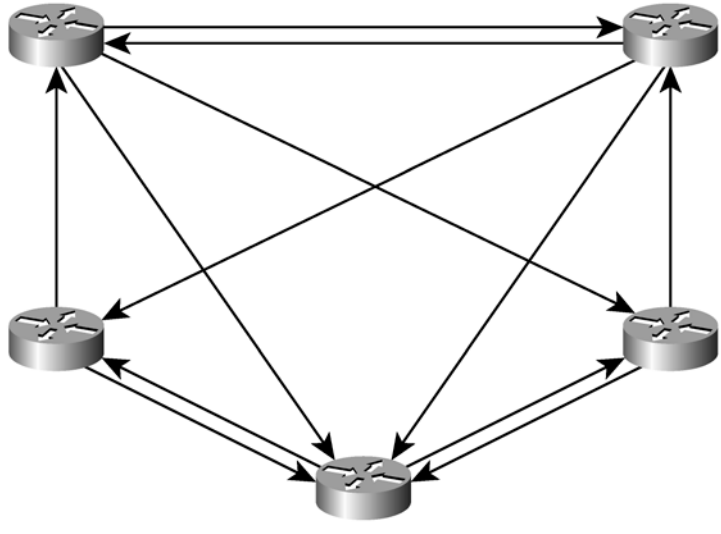


图 10-50 路由器将收到的 LSP 扩散到所有与接收该 LSP 的接口所属的组不同的 mesh 组的接口上

10.1.6 泛洪扩散的时延

虽然 mesh 组能够在比较密集的网状网络扩散过程中控制发送大量不必要的 LSP，但是，

如果一个网络不稳定的话泛洪扩散也会成为一个问题。例如，一个快速频繁波动的链路会引起大量的 LSP 扩散，每一次都会触发所有区域内的路由器进行 SPF 计算，并会耗尽网络资源。

IOS 软件系统提供了几种在这些情况下减少扩散影响的工具。第一个方法就是在路由器碰到不稳定的情况下减少本地始发的 LSP 的扩散，例如存在忽好忽坏波动的链路。这种情况下，在扩散一个 LSP 之前强制加上一个时延就可能会避免每一次波动都引起一次扩散的情况发生。例如，假设有一条链路每 20s 波动一次，而扩散每一分钟进行一次，这样就只会在每第 5 个波动才会引起一次扩散。

但是，我们并不希望总是延迟扩散：在网络处于稳定的情况下，等待一分钟才进行新的 LSP 的扩散，这会使收敛时间大大增加。因此，在 IOS 软件中利用一个指数补偿算法，在开始生成一个 LSP 之前增加一个非常小的时延，只有 50ms。如果在前一个 LSP 生成后的 5s 内由于某个事件产生了另一个 LSP，那么该路由器将会等待 5s 钟再生成新的 LSP。这个时延一直等到网络稳定 10s 后才取消，而重新恢复 50ms 的 LSP 生成间隔。

这个指数补偿算法的缺省状态可以通过命令 **lsp-gen-interval** 来改变，以下是该命令的可选项：

- **lsp-initial-wait**——是路由器产生一个 LSP 正常的等待时间。缺省值为 50ms，变化范围是 1～120 000ms。
- **lsp-second-wait**——是指路由器在产生第一个和第二个 LSP 之间所需要的时延。缺省值为 5 000ms（即 5s），它的变化范围是 1～120 000ms。在第二个 LSP 生成后，前面的时延就会加倍，并作为每一个后续 LSP 生成的时延。因此，如果第二个 LSP 的生成等待了 5s，那么在第二个和第三个 LSP 的生成之间就会等待 10s，在第三个和第四个 LSP 的生成之间等待 20s，在第四个和第五个 LSP 的生成之间等待 40s，依此类推，一直到 lsp-max-wait 所设定的时延。这个参数也给出了为什么叫指数补偿算法这个名字的原因。
- **lsp-max-wait**——是指 LSP 生成之间所允许的最长时延。如果在网络发生连续不稳定的情况时，为了使时延不会增加到影响生成 LSP 的长度，很有必要设置这个参数。这个参数的缺省值是 5s，它的变化范围在 1～120s 之间。

虽然在局部出现不稳定时可以采用 **lsp-gen-interval** 抑制 LSP 的生成，但在某台特定的路由器出现性能问题并因为大量的扩散产生过载时也可能会产生问题。惟一的解决办法是升级你的路由器。但是在过渡阶段可以利用一些可选项来控制泛洪扩散到邻居。第一个可选项是 **isis lsp-interval** 命令选项，它可以用来在一个接口上（因此也是对特定的邻居）改变 LSP 发送之间的缺省时延。缺省时延是 33ms。例如，可以将时延设置为 100ms，那么每 0.1s 只能传送 1 个 LSP，或每秒只能传送 10 个 LSP，而不能更快。

对于不能满足性能要求的那些邻居，LSP 重传也会产生问题。如果一台路由器正在吃力地处理接收到的 LSP，那么它可能会延迟确认这个 LSP，从而引起它的邻居重传这个 LSP。如果这台路由器从一开始就比较吃力，那么重传可能会使情况更加恶化。重传一条 LSP 的缺省等待时间是 5s。通过增加等待时间可以稍微保护一下性能不足的邻居；使用命令 **isis retransmit-interval** 可以增加等待时间，最大为 65 535s。

即使你使用命令 **isis retransmit-interval** 增加了重传一条 LSP 的等待时间，可能仍然会存在问题，等待时间可能会使多条 LSP 超时，而它们可能都应该被重传。读者可以使用命令 **isis retransmit-throttle-interval** 来调整传输这些需要重传的 LSP 之间的等待时间。这样我们就可以

利用命令 **isis retransmit-interval** 增加等待重传确认的时间，而使用命令 **isis retransmit-throttle-interval** 在等待时间超时的情况下，增加每一个重传的 LSP 之间的时间间隔。

虽然这些命令给了我们一些控制泛洪扩散的手段，但它们应该仅仅在极端的情况下使用。在绝大多数的情况下，缺省的参数值不应该改变；在看来有必要修补这些参数值的时候，读者应该找出改变的原因。通常情况下，进行路由器升级或提高链路的可靠性才是更好的解决方案。

10.1.7 提高 SPF 的效率

IOS 软件系统使用以下两个机制来提高 SPF 算法的效率：

- 递增 SPF（Incremental SPF，iSPF）；
- 部分路由计算（Partial Route Calculations，PRC）。

当将一台末梢路由器增加到网络中时——也就是说这台路由器仅通过一条链路增加到网络上——那么区域内的所有路由器都不需要重新计算 SPF。相反，只需要增加这台路由器到 SPF 树就足够了。如果还没有加载到 SPF 树上的链路在某些方面的变化不会影响 SPF 树，那么就根本没有必要为此而运行 SPF 计算。iSPF 考虑到这些情况，只根据拓扑改变的程度大小而运行 SPF 计算。iSPF 也能够限制 SPF 计算的范围。也就是说，如果一个变化只影响拓扑的有限部分，那么 iSPF 就可以把 SPF 的重新计算限制在拓扑变化的范围内。

另一个没有必要触发进行 SPF 计算的变化是 IP 前缀的增加、删除或度量变化。检测到这样一台变化的路由器会通过 IP 可达性 TLV（或功能上等价的 TLV）扩散一个 LSP，从而通告这个变化。但是这个 LSP 不需要触发进行 SPF 计算，这就是部分路由计算：首先检测所收到的 LSP，然后确认如果不是需要进行 SPF 计算的拓扑变化，例如一个新的 IP 可达性 TLV 或新的 IS 邻居 TLV，那么就不运行 SPF 计算。

在网络不稳定的时期，在 SPF 计算的运行之间加入更长的时延可能跨越多条 LSP 的接收时间。也就是说，如果很多 LSP 正在进行泛洪扩散，两个 SPF 计算的运行之间的等待可能意味着并不是每次收到一条 LSP 而运行 SPF 计算，而是意味着路由器可能收到了很多 LSP，并为所有这些 LSP 运行一次 SPF 计算。IOS 软件系统对于运行 SPF 计算也采用了类似抑制 LSP 生成的指数补偿算法，抑制 LSP 生成的指数补偿算法已经在前面章节中讲述过了。命令 **spf-interval** 应用于完整的 SPF 运行，而 **prc-interval** 则应用于部分路由计算。在这两种情况中，像抑制 LSP 的生成一样，读者可以指定一个初始化等待时间、为每一次后续运行间隔加倍的第二等待时间，以及不能超出的最大等待时间。SPF 指数补偿算法缺省的初始化等待时间是 5 500ms，第二等待时间的缺省值是 5 500ms，而最大等待时间的缺省值是 10s；对于 PRC 来说，它们的缺省值分别是 2 000ms、5 000ms 和 5s。但是，作为扩散的时延，改变 SPF 时延的缺省值通常不是一个好主意。解决频繁地进行 SPF 运行的实际可行的方法是，可靠的链路和网络部件，并且尽可能的使用区域来减少泛洪扩散的范围。

10.2 集成 IS-IS 协议的配置

集成 IS-IS 协议在本书介绍的 IP 路由选择协议中显得比较独特，这有两方面的原因。首

先，IS-IS 协议是惟一一个必须作为一个进程启动又要在单独的接口上启动的协议。其次，IS-IS 协议是惟一的一个开始并不是为 IP 协议设计的 IP 路由选择协议。由于集成 IS-IS 协议使用 CLNS PDU 数据包而不是 IP 数据包，因此 IS-IS 协议的配置就不如其他路由选择协议的配置那么清楚直观了。

10.2.1 案例研究：IPv4 集成 IS-IS 的基本配置

在一台 Cisco 路由器上配置一个集成 IS-IS，需要以下 4 个步骤：

步骤 1：确定路由器所在的区域和启动 IS-IS 协议的接口。

步骤 2：使用 **router isis** 命令来启动一个 IS-IS 进程。[1]

步骤 3：使用 **net** 命令来配置 NET 地址。

步骤 4：使用命令 **ip router isis** 在相应的接口上启动集成 IS-IS。这个命令不仅在转发接口（和 IS-IS 邻居相连的接口）上必须增加，而且在一个和末梢网络相连的接口也必须要配置，这里的末梢网络是指需要 IS-IS 协议来通告的 IP 地址。

如图 10-51 所示，显示了一个包括 6 台路由器的小型网络，它被分成了两个区域。使用 NET 地址表示方式，区域 1 和区域 2 将分别表示为 00.0001 和 00.0002，而它们各自的系统 ID 是每台路由器 E0 或 FastEthernet0/0 接口的 MAC 地址标识符。表 10-6 中显示了使用该信息进行编码得到的 NET 地址。

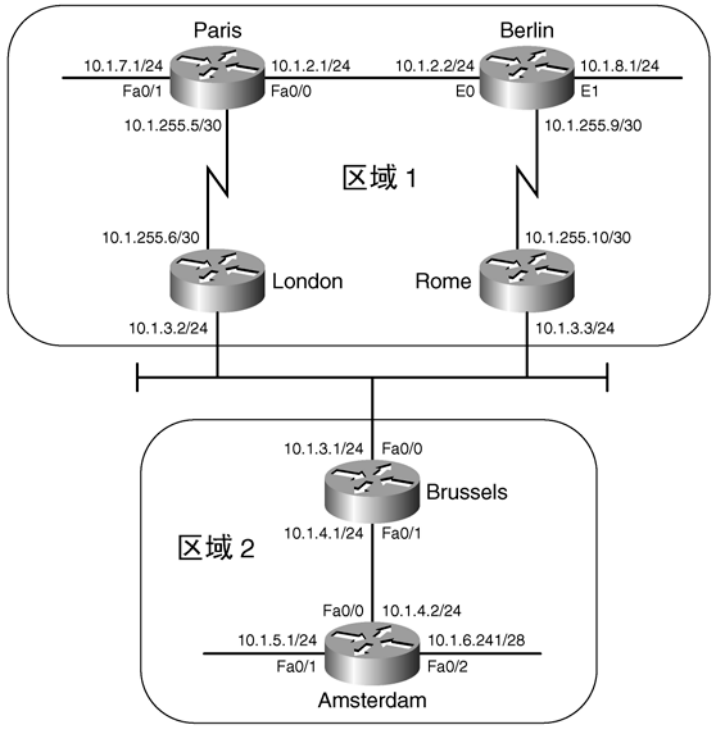


图 10-51 区域 1 表示成 NET 地址方式是 00.0001，区域 2 表示成 NET 地址方式是 00.0002。每一个 NET 的系统 ID 都是接口 E0 或 FastEthernet0/0 的 MAC 地址标识符

[1] **router isis** 命令也可以给一个名字，例如 **router isis Warsaw**。如果 IS-IS 和 ISO-IGRP 协议配置在同一台路由器上，那么必须为其中一个进程或者同时为两个进程命名。如果没有配置 ISO-IGRP 协议，则命名不是必须的。

表 10-6 图 10-51 中路由器 IS-IS 配置用到的 NET 地址

路由器	区域	MAC	NET
Paris	00.0001	0004.c150.e700	00.0001.0004.c150.e700.00
Berlin	00.0001	0010.7b3c.6bd3	00.0001.0010.7b3c.6bd3.00
London	00.0001	00b0.6430.1de0	00.0001.00b0.6430.1de0.00
Rome	00.0001	0004.c150.f1c0	00.0001.0004.c150.f1c0.00
Brussels	00.0002	0005.5e6b.50a0	00.0002.0005.5e6b.50a0.00
Amsterdam	00.0002	0000.0c8d.34f1	00.0002.0000.0c8d.34f1.00

在路由器 Paris、London、Brussels 和 Amsterdam 上的配置显示在示例 10-9～示例 10-12 中。

示例 10-9 路由器 Paris 的配置

```
interface Serial0/0.1 point-to-point
 ip address 10.1.255.5 255.255.255.252
 ip router isis
!
interface FastEthernet0/0
 ip address 10.1.2.1 255.255.255.0
 ip router isis
!
interface FastEthernet0/1
 ip address 10.1.7.1 255.255.255.0
 ip router isis
!
router isis
 net 00.0001.0004.c150.e700.00
```

示例 10-10 路由器 London 的配置

```
interface Ethernet0/0
 ip address 10.1.3.2 255.255.255.0
 ip router isis
!
interface Serial0/0.1 point-to-point
 ip address 10.1.255.6 255.255.255.252
 ip router isis
!
router isis
 net 00.0001.00b0.6430.1de0.00
```

示例 10-11 路由器 Brussels 的配置

```
interface FastEthernet0/0
 ip address 10.1.3.1 255.255.255.0
 ip router isis
!
interface FastEthernet0/1
 ip address 10.1.4.1 255.255.255.0
 ip router isis
!
router isis
 net 00.0002.0005.5e6b.50a0.00
```

示例 10-12 路由器 Amsterdam 的配置

```
clns routing
!
interface FastEthernet0/0
 ip address 10.1.4.2 255.255.255.0
 ip router isis
!
interface FastEthernet0/1
 ip address 10.1.5.1 255.255.255.0
 ip router isis
!
interface FastEthernet0/2
 ip address 10.1.6.241 255.255.255.240
 ip router isis
!
router isis
 net 00.0002.0000.0c8d.34f1.00
```

路由器 Berlin 和 Rome 的配置基本相似。在这里有一个需要注意的配置细节是，在 Amsterdam 路由器的配置中启动了 CLNS 协议的路由选择功能。CLNS 路由选择对于 IS-IS 协议处理 CLNS PDU 是必需的。当我们在创建一个 IS-IS 进程的时候，就自动地启动了 CLNS 路由选择。在 IOS 软件的某些版本中，我们可能在配置中看到命令 **clns routing**，虽然这在配置中并不是要求输入的。

参见示例 10-13，示例中显示了路由器 Paris 的路由表。请注意，这里路由表中同时包含了 L1 路由和 L2 路由。在缺省条件下，Cisco 路由器默认是 L1/L2 路由器。这个事实也可以通过查看路由器的 IS 邻居表清楚地看出来（参见示例 10-14）。

示例 10-13 路由器 Paris 的路由表中同时显示了 L1 路由和 L2 路由，表明这台路由器是一台 L1/L2 路由器

```
Paris#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
     10.0.0.0/8 is variably subnetted, 9 subnets, 3 masks
i L1    10.1.8.0/24 [115/20] via 10.1.2.2, FastEthernet0/0
i L1    10.1.3.0/24 [115/20] via 10.1.255.6, Serial0/0.1
C       10.1.2.0/24 is directly connected, FastEthernet0/0
C       10.1.7.0/24 is directly connected, FastEthernet0/1
i L2    10.1.5.0/24 [115/40] via 10.1.255.6, Serial0/0.1
i L2    10.1.4.0/24 [115/30] via 10.1.255.6, Serial0/0.1
C       10.1.255.4/30 is directly connected, Serial0/0.1
i L1    10.1.255.8/30 [115/20] via 10.1.2.2, FastEthernet0/0
i L2    10.1.6.240/28 [115/40] via 10.1.255.6, Serial0/0.1
Paris#
```

示例 10-14 路由器 Berlin 的 IS 邻居表显示出路由器 Paris 和 Rome 都是 L1/L2l 路由器

```
Berlin#show clns is-neighbors

System Id      Interface   State  Type Priority  Circuit Id         Format
Rome           Se0.1       Up     L1L2 0 /0      00                 Phase V
Paris          Et0         Up     L1L2 64/64     Berlin.01          Phase V
Berlin#
```

这里请注意，路由器 Berlin 的 IS 邻居表中列出了每一个邻居的名字。主机名是使用“动态主机名交换”小节中讲述的类型 137 的 TLV 动态交换的。为了显示与每个主机名相关联的系统标识符，可以使用命令 **show isis hostname**，参见示例 10-15 所示。

示例 10-15 使用命令 **show isis hostname** 可以显示与已知主机名相关联的系统 ID

```
Berlin#show isis hostname
Level  System ID      Dynamic Hostname  (notag)
 1     00B0.6430.1DE0 London
 2     0000.0C8D.34F1 Amsterdam
 1     0004.C150.E700 Paris
 1     0004.C150.F1C0 Rome
     * 0010.7B3C.6BD3 Berlin
Berlin#
```

由于在图 10-51 的网络中每一台路由器都是 L1/L2 类型的，因此，每一台路由器都会同时形成 L1 类型的邻接关系和 L2 类型的邻接关系。也正因为如此，每一台路由器也将会同时维护一个 L1 类型的链路状态数据库和一个 L2 类型的链路状态数据库。L1 区域和 L2 区域完全重叠了。例如，示例 10-16 中显示的路由器 Amsterdam 的链路状态数据库；其中，L1 类型的数据库中包含了一条始发于路由器 Amsterdam[1]的 LSP 数据包和一条始发于路由器 Brussels 的 LSP 数据包。同时，它还包含了一条始发于路由器 Brussels 的伪节点 LSP 数据包（Brussels.02-00），用来描述路由器 Brussels 和 Amsterdam 之间的以太网链路。请记住，作为伪节点 LSP 数据包的 LSP ID 是可以辨别出来的，因为伪节点 LSP 数据包的 LSP ID 的倒数第二个八位组字节（也就是伪节点 ID）是非零的。

示例 10-16 路由器 Amsterdam 同时具有一个 L1 类型的链路状态数据库和一个 L2 类型的链路状态数据库，这表明该路由器是一台 L1/L2 路由器

```
Amsterdam#show isis database

IS-IS Level-1 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Amsterdam.00-00     * 0x00000015   0x642E        1112              1/0/0
Brussels.00-00        0x00000016   0x9B81        1187              1/0/0
Brussels.02-00        0x00000013   0x46BD        1139              0/0/0
IS-IS Level-2 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Amsterdam.00-00     * 0x00000015   0xEAFF        1176              0/0/0
Paris.00-00           0x00000023   0x2B29        504               0/0/0
Rome.00-00            0x0000001D   0xD016        482               0/0/0
Brussels.00-00        0x00000019   0x45BB        1187              0/0/0
Brussels.02-00        0x00000013   0xD5B6        375               0/0/0
Berlin.00-00          0x0000001E   0x408B        506               0/0/0
Berlin.01-00          0x00000013   0xB10F        557               0/0/0
London.00-00          0x00000019   0xF1B1        463               0/0/0
London.01-00          0x00000013   0x9E92        844               0/0/0
Amsterdam#
```

这 3 条 L1 的 LSP 表明路由器 Amsterdam 除了它本身外，所知道的惟一的 L1 路由器是 Brussels。这个单一的节点是可以预料到的，因为在区域 2 中路由器 Brussels 是惟一另外的路由器。路由器 Amsterdam 的 L2 数据库显示 Amsterdam 和 IS-IS 域内的每一台路由器都形成了 L2 邻接，当然这也是可以预料到的，因为每一台路由器都是一台 L1/L2 类型的路由器。

[1] 正如前面讲到的，这个 LSP ID 后面的星号表示该 LSP 是由这台路由器本身始发的。

10.2.2 案例研究：更改路由器的类型

在如图 10-51 这样的小型网络中，在路由器上保留它们缺省的 IS-IS 类型是可以接受的。但是，当网络规模不断增大时，使用这些缺省的类型将越来越不能被接受。因为，这不仅要消耗大量的路由器 CPU 和内存去处理和维护两个链路状态数据库，而且要消耗大量的缓存和带宽去处理和泛洪每一台路由器始发的 L1 和 L2 类型的 IS-IS PDU。

在图 10-51 中的路由器 Paris、Berlin 和 Amsterdam 可以配置成 L1 路由器，因为它们都没有和其他区域直连的链路。在路由器上可以使用命令 **is-type** 来更改缺省的路由器类型。例如，要把路由器 Berlin 变成一台 L1 类型的路由器，它的配置如示例 10-17 所示。

示例 10-17 路由器 Berlin 配置为一台 L1 路由器

```
router isis
 net 00.0001.0000.3090.c7df.00
 is-type level-1
```

路由器 Paris 和 Amsterdam 的配置也和上面类似。比较一下路由器 Paris 在示例 10-18 中的路由表和在示例 10-13 中的路由表，可以发现 L2 类型的路由已经被删除了。同样地，比较一下路由器 Amsterdam 在示例 10-19 和在示例 10-16 中的路由表，可以发现现在路由器 Amsterdam 只剩下 L1 数据库了。

示例 10-18 当路由器 Paris 配置成一台 L1 路由器后，它的路由表中就只包含到达它本身所在区域内的目的地址的路由了

```
Paris#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 10.1.255.6 to network 0.0.0.0
     10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
i L1    10.1.8.0/24 [115/20] via 10.1.2.2, FastEthernet0/0
i L1    10.1.3.0/24 [115/20] via 10.1.255.6, Serial0/0.1
C       10.1.2.0/24 is directly connected, FastEthernet0/0
C       10.1.7.0/24 is directly connected, FastEthernet0/1
C       10.1.255.4/30 is directly connected, Serial0/0.1
i L1    10.1.255.8/30 [115/20] via 10.1.2.2, FastEthernet0/0
i*L1 0.0.0.0/0 [115/10] via 10.1.255.6, Serial0/0.1
Paris#
```

示例 10-19 当路由器 Amsterdam 配置成一台 L1 路由器后，它就只包含 L1 的链路状态数据库了

```
Amsterdam#show isis database

IS-IS Level-1 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Amsterdam.00-00     * 0x00000017   0x5644        1065              0/0/0
Brussels.00-00        0x00000018   0x9783        1063              1/0/0
Brussels.02-00        0x00000014   0x44BE        1063              0/0/0
Amsterdam#
```

回忆一下，在前面曾经讲述过 LSP 中的区域关联位（ATT 位），一台 L1/L2 路由器会通过设置 ATT 位来告知 L1 路由器它具有区域间的连接。示例 10-20 中显示了路由器 London 的 LSP 和路由器 Rome 的 LSP 都将 ATT 位设置为 1 了，即 ATT=1。因此，路由器 Paris 将会知道把区域间的通信量发送到路由器 London 或者 Rome。换句话说，路由器 Paris 会有一条到达路由器 London 或 Rome 的缺省路由，而且到达路由器 London 的路径将成为优先路径，因为路由器 Paris 到达它的度量更小。在示例 10-18 中，并没有在路由器 Paris 的路由表中显示出这条缺省路由（0.0.0.0）。

示例 10-20 路由器 London 和 Rome 始发的 L1 LSP 中设置了 ATT=1，这表明它们具有到达其他区域的连接

```
Paris#show isis database

IS-IS Level-1 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Paris.00-00         * 0x00000022   0xA4C8        857               0/0/0
Rome.00-00            0x00000018   0x139F        727               1/0/0
Berlin.00-00          0x0000001C   0x28B8        733               0/0/0
Berlin.01-00          0x00000018   0x161F        851               0/0/0
London.00-00          0x00000017   0xDD92        848               1/0/0
London.01-00          0x00000014   0x9F54        1080              0/0/0
Paris#
```

在 IOS 软件的旧版本中，IP 进程不能直接解释 ATT 位。ATT 位是一个 CLNS 的特性。当运行 IOS 软件的旧版本时，如果在 L1 路由器中没有自动地创建缺省路由，将有两种方法解决这个问题。第一种解决方法是在路由器的接口上除了启用 IP 协议的 IS-IS，另外再启用 CLNS 协议的 IS-IS。例如，修改路由器 London 和 Paris 的串行接口的配置，参见示例 10-21 和示例 10-22。

示例 10-21 在路由器 London 的接口上配置 CLNS 的 IS-IS，启动 ATT 位处理

```
interface Serial0/0.1
 ip address 10.1.255.6 255.255.255.252
 ip router isis
 clns router isis
```

示例 10-22 在路由器 Paris 的接口上配置 CLNS 的 IS-IS，启动 ATT 位处理

```
interface Serial0/0.1
 ip address 10.1.255.5 255.255.255.252
 ip router isis
 clns router isis
```

第一种解决方法需要 IS-IS 协议运行在一个 CLNP/IP 的混和环境里，但是如果 IS-IS 协议仅仅是作为一个单一的 IP 路由选择协议使用的话，那么启用 CLNS 路由选择仅仅为了生成一条 IP 缺省路由就显得十分没必要了。而第二种解决缺省路由问题的方法是在 L1/L2 路由器上配置一条静态路由，并且使用命令 **default-information originate** 配置 IS-IS 协议来通告这条缺省路由。在图 10-51 的区域 2 中使用这个方法，路由器 Brussels 的配置参见示例 10-23 所示。

示例 10-23 路由器 Brussels 配置成发起一条缺省路由

```
router isis
 net 00.0002.0000.0c76.5b7c.00
 default-information originate
!
ip route 0.0.0.0 0.0.0.0 Null0
```

关于缺省路由和 **default-information originate** 命令更为详细的介绍将在第 12 章中讲述。

10.2.3 案例研究：区域的迁移

在 OSPF 协议中如果更改区域的地址，就必须考虑和预计好网络中断的时间。但是，在 IS-IS 协议的设计中，能够在网络不中断的情况下允许更改区域地址。正如在 10.1 节中讲述的，Cisco 路由器可以最多配置 254 个 L1 区域地址。为了使两台路由器能够形成一个 L1 邻接，它们必须至少具有一个公用的区域地址。在允许具有多个区域地址的情况下，新的邻接关系能够在旧的邻接关系中断时替代它。这种方法在某些情况下会显得非常有用。例如，在合并区域或拆分区域的时候、在为一个区域重新编号的时候，或者在同一个 IS-IS 域内同时运行多个编址机构分配的区域地址的时候，等等。

举个例子，在图 10-52（a）中的路由器都具有一个区域地址 01（这些路由器当中的任何一台设备的 NET 地址看上去应该像 010000.0c12.3456.00 一样）。在图 10-52（b）中，这些路由器另外分配了一个区域地址 03。虽然实际上并没有形成多个邻接关系，但是这些路由器还是可以识别出它们具有多个公共的区域地址。在图 10-52（c）中，区域 01 已经从某一台路由器上移走了。这 3 台路由器仍然保留着邻接关系，因为它们至少还有一个公共的区域地址。最后，在图 10-52（d）中，区域地址 01 从这 3 台路由器上全部移走了，这时，这 3 台路由器全在区域 03 中。可以看出，在区域迁移期间并没有什么时候丢失邻接关系。

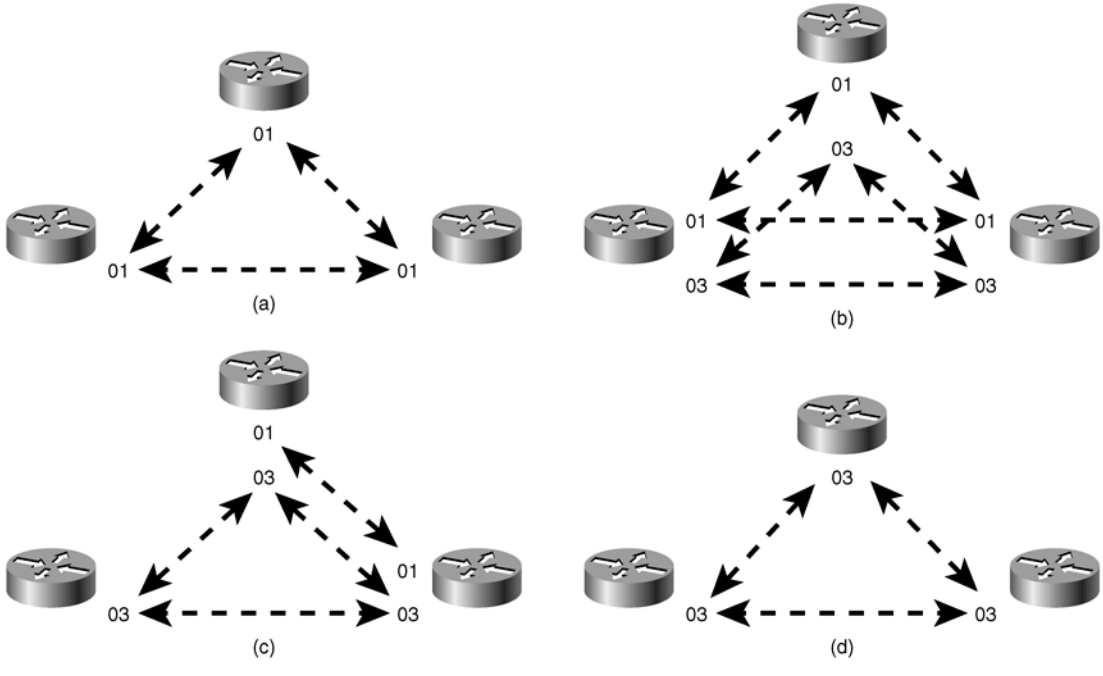


图 10-52 每台路由器都支持多个区域地址，可以使区域的更改变得比较容易

假定因为某些原因，认为图 10-51 中的网络正在使用的区域地址安排不合法，而被强令要求必须符合 GOSIP。这时通过注册 U.S.GSA 机构后，可以使用下面的资源来组成新的 NET

地址：

AFI： 47

IDI： 0005

DFI： 80

AAI： 00ab7c

Reserved： 0000

RDI： fffe9

Areas： 0001 （area 1），0002 （area 2）

根据上述信息，表 10-7 中显示了新的 NET 地址。

表 10-7 图 10-51 中的路由器分配到了新的 GOSIP 格式的 NET 地址

路由器	NET 地址
Paris	47.0005.80.00ab7c.0000.ffe9.0001.0004.c150.e700.00
Berlin	47.0005.80.00ab7c.0000.ffe9.0001.0010.7b3c.6bd3.00
London	47.0005.80.00ab7c.0000.ffe9.0001.00b0.6430.1de0.00
Rome	47.0005.80.00ab7c.0000.ffe9.0001.0004.c150.f1c0.00
Brussels	47.0005.80.00ab7c.0000.ffe9.0002.0005.5e6b.50a0.00
Amsterdam	47.0005.80.00ab7c.0000.ffe9.0002.0000.0c8d.34f1.00

更改区域地址的第一步是增加新的 NET 地址到路由器上，但是不改变原来的 NET 地址。路由器 Rome 上的 IS-IS 配置参见示例 10-24。

示例 10-24 在转换期间路由器 Rome 配置了两个 NET

```
router isis
 net 00.0001.0000.0c0a.2aa9.00
 net 47.0005.8000.ab7c.0000.ffe9.0001.0004.c150.f1c0.00
```

其他 5 台路由器的配置也和上面类似。可以使用带关键字 **detail** 的命令 **show isis database** 来查看结果（参见示例 10-25 所示），或者也可以使用命令 **show clns is-neighbors** 来查看结果（参见示例 10-26 所示）。在这两个数据库中，可以看出网络内的每一台路由器都是和多个区域相连的。

示例 10-25 在路由器 Rome 的链路状态数据库中，LSP 显示出图 10-51 中网络的所有路由器都正在通告两个区域地址

```
Rome#show isis database detail
IS-IS Level-1 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Paris.00-00           0x00000026   0xC3AA        886               0/0/0
  Area Address: 00.0001
  Area Address: 47.0005.8000.ab7c.0000.ffe9.0001
  NLPID:        0xCC
  Hostname: Paris
  IP Address:   10.1.7.1
  Metric: 10         IP 10.1.2.0 255.255.255.0
  Metric: 10         IP 10.1.255.4 255.255.255.252
  Metric: 10         IP 10.1.7.0 255.255.255.0
  Metric: 10         IS Berlin.01
  Metric: 10         IS London.00
```

（待续）

```
Rome.00-00          * 0x0000001E   0x4D64        688               1/0/0
  Area Address: 00.0001
  Area Address: 47.0005.8000.ab7c.0000.ffe9.0001
  NLPID:        0xCC
  Hostname: Rome
  IP Address:   10.1.255.10
  Metric: 10         IP 10.1.3.0 255.255.255.0
  Metric: 10         IP 10.1.255.8 255.255.255.252
  Metric: 10         IS Berlin.00
  Metric: 10         IS London.01
Berlin.00-00          0x00000024   0xC419        716               0/0/0
  Area Address: 00.0001
  Area Address: 47.0005.8000.ab7c.0000.ffe9.0001
  NLPID:        0xCC
  Hostname: Berlin
  IP Address:   10.1.8.1
  Metric: 10         IP 10.1.2.0 255.255.255.0
  Metric: 10         IP 10.1.8.0 255.255.255.0
  Metric: 10         IP 10.1.255.8 255.255.255.252
  Metric: 10         IS Berlin.01
  Metric: 10         IS Berlin.02
  Metric: 10         IS Rome.00
Berlin.01-00          0x0000001B   0x1022        572               0/0/0
  Metric: 0          IS Berlin.00
  Metric: 0          IS Paris.00
London.00-00          0x0000001A   0x1959        1044              1/0/0
  Area Address: 00.0001
--More—
```

示例 10-26　路由器 Rome 的 IS-IS 邻居表中也显示出每台邻居路由器都与多个地址相关联

```
Rome#show clns is-neighbors detail
System Id      Interface   State  Type Priority  Circuit Id         Format
Berlin         Se0/0.1     Up     L1   0         00                 Phase V
  Area Address(es): 00.0001 47.0005.8000.ab7c.0000.ffe9.0001
  IP Address(es):  10.1.255.9*
  Uptime: 00:30:49
London         Fa0/0       Up     L1L2 64/64     London.01          Phase V
  Area Address(es): 00.0001 47.0005.8000.ab7c.0000.ffe9.0001
  IP Address(es):  10.1.3.2*
  Uptime: 04:49:01
  NSF capable
Brussels       Fa0/0       Up     L2   64        London.01          Phase V
  Area Address(es): 00.0002 47.0005.8000.ab7c.0000.ffe9.0002
  IP Address(es):  10.1.3.1*
  Uptime: 04:51:46
  NSF capable
Rome#
```

区域迁移的最后一步是从所有的路由器上删除原来的 NET 地址语句。例如，在路由器 Rome 的 IS-IS 配置中输入命令 **no net 00.0001.0004.c150.f1c0.00**。在示例 10-27 中，显示了在路由器 Rome 上删除了以前的 NET 语句后，该路由器数据库中的一些 LSP 信息。

示例 10-27　路由器 Rome 的数据库中的 LSP 显示出只有一个区域地址

```
Rome#show isis data detail
IS-IS Level-1 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Paris.00-00           0x0000002D   0x412E        1171              0/0/0
```

（待续）

```
  Area Address: 47.0005.8000.ab7c.0000.ffe9.0001
  NLPID:        0xCC
  Hostname: Paris
  IP Address:   10.1.7.1
  Metric: 10         IP 10.1.2.0 255.255.255.0
  Metric: 10         IP 10.1.255.4 255.255.255.252
  Metric: 10         IP 10.1.7.0 255.255.255.0
  Metric: 10         IS Berlin.01
  Metric: 10         IS London.00
Rome.00-00          * 0x00000025   0x0BA7        1174              1/0/0
  Area Address: 47.0005.8000.ab7c.0000.ffe9.0001
  NLPID:        0xCC
  Hostname: Rome
  IP Address:   10.1.255.10
  Metric: 10         IP 10.1.3.0 255.255.255.0
  Metric: 10         IP 10.1.255.8 255.255.255.252
  Metric: 10         IS Berlin.00
  Metric: 10         IS London.01
Berlin.00-00         0x00000029   0x20C0        1097              0/0/0
  Area Address: 47.0005.8000.ab7c.0000.ffe9.0001
  NLPID:        0xCC
  Hostname: Berlin
  IP Address:   10.1.8.1
  Metric: 10         IP 10.1.2.0 255.255.255.0
  Metric: 10         IP 10.1.8.0 255.255.255.0
  Metric: 10         IP 10.1.255.8 255.255.255.252
  Metric: 10         IS Berlin.01
  Metric: 10         IS Berlin.02
  Metric: 10         IS Rome.00
Berlin.01-00         0x0000001D   0x0C24        1090              0/0/0
  Metric: 0          IS Berlin.00
  Metric: 0          IS Paris.00
London.00-00         0x0000001F   0xB7BD        1163              1/0/0
  Area Address: 47.0005.8000.ab7c.0000.ffe9.0001
  NLPID:        0xCC
  --More—
```

10.2.4 案例研究：路由汇总

在第 8 章中，已经介绍了在链路状态协议的区域之间如何进行路由汇总。关于路由汇总更完整的讲述将会在第 12 章讲述的缺省路由部分介绍。这里先简要地描述一下汇总路由的好处：

- 汇总路由可以有效地减小 LSP 的大小，这样也就减小了链路状态数据库的大小，从而也节省了路由器的 CPU 和内存消耗。
- 汇总路由可以隐藏掉区域内部网络的不稳定影响。如果仅仅是一个汇总地址范围内的地址发生了改变或一条链路的状态发生变化，那么并不会通告到做汇总的区域外部。

当然，汇总路由也有以下一些缺点：

- 汇总路由的效果依赖于能够进行汇总的连续的 IP 地址范围，因此地址分配必须进行仔细规划。
- 汇总路由由于隐藏区域内的细节因而减少了路由的精确性。如果具有多条进入汇总区域的路径，那么将无法确定最佳的路径。

路由汇总可以在 IS-IS 的配置下使用命令 **summary-address** 来启动。配置了这条语句后，任何在汇总地址范围内的更具体的目的地址都将被抑制，而汇总路由的度量会选择它所有更具体的地址中更小的度量。

在图 10-53 中，显示了包含 3 个区域的一个 IS-IS 网络。在这里，区域 1 内的地址可以汇总为 172.16.0.0/21，而区域 3 内的地址可以汇总为 172.16.16.0/21。路由器 Zurich、Madrid 和 Bonn 的配置参见示例 10-28～示例 10-30。[1]

示例 10-28 路由器 Zurich 的配置执行路由汇总

```
router isis
 net 01.0000.0c76.5b7c.00
 summary-address 172.16.0.0 255.255.248.0
```

示例 10-29 路由器 Madrid 的配置没有执行路由汇总

```
router isis
 net 02.0000.3090.6756.00
 is-type level-2-only
```

示例 10-30 路由器 Bonn 的配置执行路由汇总

```
router isis
 net 03.0000.0c0a.2aa9.00
 summary-address 172.16.16.0 255.255.248.0
```

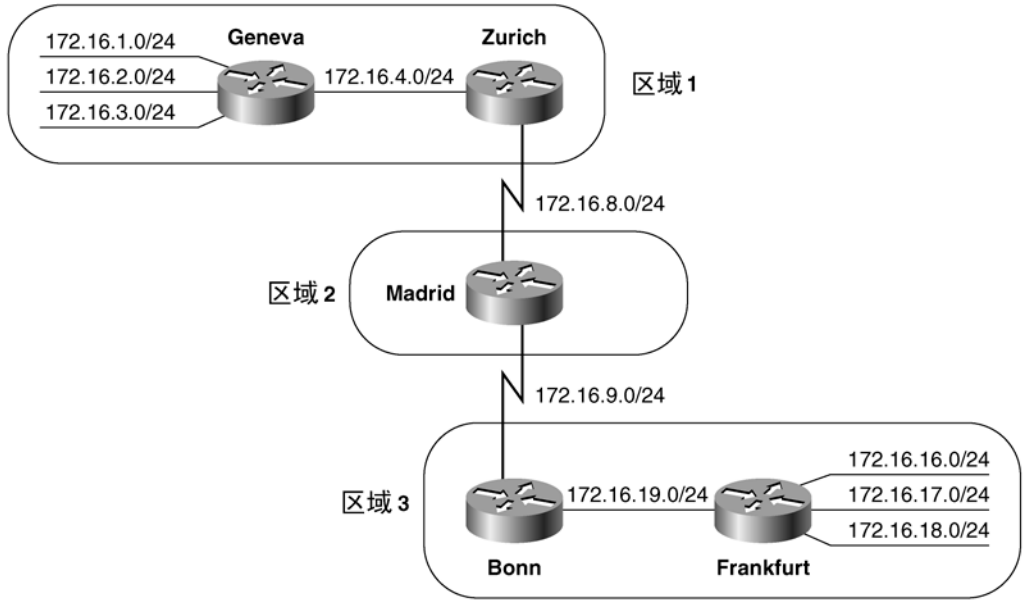


图 10-53 路由器 Zurich 和 Bonn 将区域 1 和区域 3 汇总到区域 2

这里注意，路由器 Madrid 由于没有 L1 类型的邻居，因而配置成一台 L2 路由器。路由器 Zurich 和 Bonn 正在汇总它们各自的区域到 L2 类型的骨干。参见示例 10-31 所示，在路由器 Madrid 的路由表中显示出了路由汇总的结果。

示例 10-31 路由器 Madrid 的路由表显示出路由器 Bonn 和 Zurich 通告的汇总地址

```
Madrid#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
```

（待续）

[1] 请注意，路由器 Madrid 由于没有 L1 类型的邻居，因而配置成一台 L2 路由器。

```
       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

     172.16.0.0/16 is variably subnetted, 4 subnets, 2 masks
i L2    172.16.16.0/21 [115/20] via 172.16.9.2, Serial0/0.2
C       172.16.8.0/24 is directly connected, Serial0/0.1
C       172.16.9.0/24 is directly connected, Serial0/0.2
i L2    172.16.0.0/21 [115/20] via 172.16.8.2, Serial0/0.1
Madrid#
```

10.2.5 案例研究：认证

IS-IS 协议的认证可以使用明文口令或 HMAC-MD5。有两种方法配置明文口令。这两种认证方式对于防止来自网络的攻击提供了一个很弱的安全机制，但是对于防止因为配置出错或未受权的路由器造成网络服务的中断还是比较有效的。其中一种明文配置模式使用钥匙链，它可以在配置文件中被加密以防止别人通过查看配置文件无意获取口令。没有钥匙链的明文认证被认为是“老式”口令。

Cisco IOS 软件支持 3 个级别的 IS-IS 认证：邻居之间、区域范围和 IS-IS 域范围。这 3 个级别的认证可以单独使用也可以一起使用。IS-IS 认证的规则如下：

- 当在邻居之间进行认证时，互相连接的路由器接口必须配置相同的口令；
- 当在邻居之间进行认证时，必须分别为 L1 和 L2 类型的邻接关系配置各自的认证；
- 当在邻居之间进行认证时，可以使用明文或 MD5 认证；
- 当认证在整个区域范围内有效时，区域内的每一台路由器都必须使用相同的认证模式和具有共同的钥匙串；
- 当认证在整个 IS-IS 域范围内有效时，IS-IS 域内的每一台 L2 和 L1/L2 类型的路由器都必须使用相同模式的认证，并使用共同的钥匙串。

在 IS-IS 中配置认证的步骤和在 RIPv2 与 EIGRP 中的配置步骤是一样的。这些步骤重复如下：

步骤 1：定义一个带有名字的钥匙链。

步骤 2：在钥匙链上定义一个或一组钥匙。

步骤 3：在某接口上，或者为 L1（区域范围）或 L2（域范围）的 IS-IS 实例启用认证，并指定使用的钥匙链。

步骤 4：为某接口，或者为 L1 或 L2 的 IS-IS 实例指定将使用明文或 MD5 认证。

步骤 5：可选地配置钥匙的管理。

在网络中配置认证可以不中断路由器之间的邻接关系。为了完成这项功能，命令 **isis authentication send-only** 必须首先配置在需要使用认证的所有路由器上。这条命令在接口上用来进行邻居之间的认证，在 ISIS 路由选择进程下用来进行区域范围或域范围的认证。在认证完全配置好之后可以删除这条命令。

如果在邻居之间进行认证，先配置钥匙链，命令 **isis authentication key-chain** 引用预先配置的钥匙链，接着使用命令 **isis authentication mode** 在直连的接口上配置认证类型。L1

与 L2 邻接可能引用不同的钥匙链。相邻的路由器必须共享共同的钥匙串或口令。在一个接口上可以指定任意一个或同时指定两个层的钥匙链，并且每一层的钥匙串可以相同也可以不同。当配置完后，钥匙串可以由 IS-IS 邻居之间的 L1 或 L2 Hello 数据包内的认证信息 TLV 携带。

举个例子，在图 10-53 中的路由器 Geneva、Zurich 和 Madrid 上配置了认证，这 3 台路由器的相关配置参见示例 10-32～示例 10-34。

示例 10-32 路由器 Geneva 的认证配置

```
interface FastEthernet0/0
 ip address 172.16.4.1 255.255.255.0
 ip router isis
 isis password magic
```

示例 10-33 路由器 Zurich 的认证配置

```
key chain troll
 key 1
  key-string magic

key chain fairy
 key 1
  key-string dust

interface Ethernet0/0
 ip address 172.16.4.2 255.255.255.0
 ip router isis
 isis authentication mode text
 isis authentication key-chain troll level-1
!
interface Serial0/0.1 point-to-point
 ip address 172.16.8.2 255.255.255.0
 ip router isis
 isis authentication mode text
 isis authentication key-chain fairy level-2
```

示例 10-34 路由器 Madrid 的认证配置

```
key chain fairy
 key 1
  key-string dust

interface Serial0/0.1 point-to-point
 ip address 172.16.8.1 255.255.255.0
 ip router isis
 isis authentication mode text
 isis authentication key-chain fairy level-2
```

因为路由器 Geneva 和 Zurich 之间的邻接关系是 L1 类型的，因此只指定了一个 L1 类型的钥匙链引用。路由器 Zurich 和 Madrid 之间只存在 L2 类型的邻接关系，因此只指定了一个 L2 类型的钥匙链引用。这里要注意，如果没有指定关键字 **level-1** 或者 **level-2**，那么命令 **isis password**、**isis authentication mode** 以及 **isis authentication key-chain** 将会默认为是 L1 和 L2。

注意路由器 Geneva 的配置。路由器 Geneva 连接到路由器 Zurich 的接口配置了老式明文口令。为了建立邻接关系，这个口令必须和路由器 Zurich 上定义的钥匙串相同。[1]

[1] 路由器之间的口令或钥匙串都必须是相同的。它们是区分大小写的。也就是说，空格和其他字符也是口令的一部分。如果配置文件是非在线创建而粘贴到路由器中的，就要小心尾部的空格。

如果要在一个区域内进行认证，可以使用命令 **isis authentication mode** *mode* **level-1** 和命令 **isis authentication key-chain** *name* **level-1** 来定义认证模式，并在 IS-IS 配置下引用一个钥匙链。当该模式是 **text** 时，这两条命令一起用来替代老式的命令 **area-password**。在接口配置模式下使用命令 **isis authentication** 指定的钥匙串是在 Hello 数据包中传送的，而在 IS-IS 进程下使用 level-1 认证命令指定的钥匙串是在所有的 L1 LSP、CSNP 和 PSNP 中传送的。因此，邻居级别的口令只可以用来验证邻接关系的建立，而区域级别的口令则可以用来验证 L1 类型的链路状态信息的交换。如果区域认证没有配置正确，路由器将仍然可以形成邻接关系，但是不会进行 L1 LSP 的交换。

在图 10-53 中的区域 3 上配置区域口令，路由器 Bonn 和 Frankfurt 的配置参见示例 10-35 和示例 10-36。

示例 10-35 路由器 Bonn 的区域口令配置

```
Key chain river
 Key 1
  Key-string Rhine

router isis
 net 03.0000.0c0a.2aa9.00
 authentication key-chain river level-1
 authentication mode text level-1
 summary-address 172.16.16.0 255.255.248.0
```

示例 10-36 路由器 Frankfurt 的区域口令配置

```
router isis
 net 03.0000.0c04.dcc0.00
 is-type level-1
 area-password Rhine
```

路由器 Frankfurt 的 IOS 软件不支持新式的认证，因此使用命令 **area-password**。注意该口令和路由器 Bonn 上定义的钥匙串是相同的。

如果要在 IS-IS 域范围内配置认证信息，可以在 IS-IS 进程下使用带关键字 **level-2** 的命令 **authentication key-chain** 和 **authentication mode**。这些配置命令可以和老式命令 **domain-password** 共同使用。但两种风格的命令不能同时配置在同一台路由器上。定义在钥匙链中被这些认证命令引用的钥匙串将在 L2 LSP、CSNP 和 PSNP 中传送。因而，IS-IS 域认证可以用来验证 L2 路由信息的交换。像区域认证一样，IS-IS 域认证不会去验证 L2 类型的邻接关系，但是会验证 L2 LSP 的交换。

在图 10-53 中的网络上配置 IS-IS 域认证，只需要在路由器 Zurich、Madrid 和 Bonn 上配置就可以了，因为路由器 Geneva 和 Frankfurt 是 L1 类型的。相关的配置参见示例 10-37～示例 10-39。

示例 10-37 路由器 Zurich 的域认证配置

```
key chain troll
 key 1
  key-string magic

key chain fairy
```

（待续）

```
 key 1
  key-string dust

key chain forest
 key 1
  key-string Blackforest

interface Ethernet0/0
 ip address 172.16.4.2 255.255.255.0
 ip router isis
 isis authentication mode text
 isis authentication key-chain troll level-1
!
interface Serial0/0.1 point-to-point
 ip address 172.16.8.2 255.255.255.0
 ip router isis
 isis authentication mode text
 isis authentication key-chain fairy level-2
!
router isis
 authentication key-chain forest level-2
 authentication mode md5 level-2
 net 01.0000.0c76.5b7c.00
 summary-address 172.16.0.0 255.255.248.0
```

示例 10-38　路由器 Madrid 的域认证配置

```
key chain fairy
 key 1
  key-string dust
key chain forest
 key 1
  key-string Blackforest

interface Serial0/0.1 point-to-point
 ip address 172.16.8.1 255.255.255.0
 ip router isis
 isis authentication mode text
 isis authentication key-chain fairy level-2
!
router isis
 net 02.0000.3090.6756.00
 is-type level-2-only
 authentication key-chain forest level-2
 authentication mode md5 level-2
```

示例 10-39　路由器 Bonn 的域认证配置

```
key chain river
 key 1
  key-string Rhine
key chain forest
 key 1
  key-string Blackforest
!
router isis
 net 03.0000.0c0a.2aa9.00
 authentication key-chain river level-1
 authentication key-chain forest level-2
 authentication mode text level-1
 authentication mode md5 level-2
 summary-address 172.16.16.0 255.255.248.0
```

10.2.6 案例研究：IPv6 集成 IS-IS 的基本配置

集成 IS-IS 对于 IPv4 和 IPv6 协议（还有 CLNS）只需计算单个 SPF 来创建单个拓扑。如果在网络中配置了 IPv4 和 IPv6，那么所有的接口和所有的路由器都必须配置这两种协议。

在路由器 Geneva 和 Madrid 之间增加一条链路到图 10-53 的网络中。路由器 Geneva 不再只是 L1 路由器了。IPv6 也添加到该网络中。在图 10-54 中显示了新的地址配置。图中的路由器已经配置了集成 IS-IS 协议。为了运行 IPv6 协议下的 IS-IS 协议，可以先利用命令 **ipv6 unicast-routing** 在全局模式下启动 IPv6 路由选择，然后在接口上启动 IPv6 的 IS-IS 协议。在每一个具有 IPv4 地址的接口上都增加了一个 IPv6 地址和 IPv6 IS-IS 协议。路由器 Geneva 的新配置参见示例 10-40。

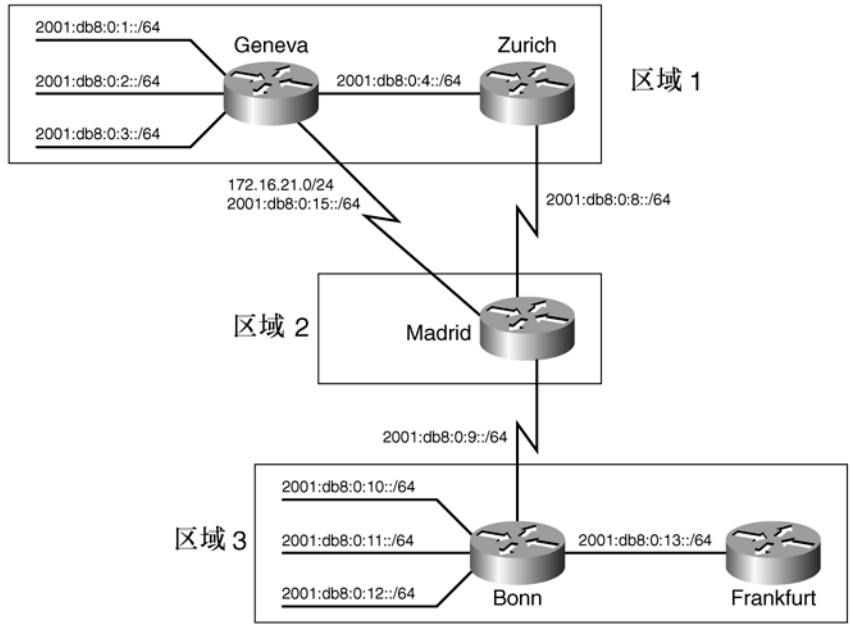


图 10-54 在网络上增加了 IPv6 协议的 IS-IS 协议

示例 10-40 路由器 Geneva 的 IPv6 IS-IS 配置

```
Hostname Geneva
ipv6 unicast-routing
!
interface FastEthernet0/0
 ip address 172.16.4.1 255.255.255.0
 ip router isis
 isis password magic
 ipv6 address 2001:db8:0:4::1/64
 ipv6 router isis
!
interface s 0/0.1 point-to-point
 ip address 172.16.21.1 255.255.255.0
 ip router isis
 ipv6 address 2001:db8:0:15::1/64
 ipv6 router isis
```

（待续）

```
!
interface FastEthernet0/1
 ip address 172.16.1.1 255.255.255.0
 ipv6 address 2001:db8:0:1::1/64
 ip router isis
 ipv6 router isis
!
interface FastEthernet0/2
 ip address 172.16.2.1 255.255.255.0
 ipv6 address 2001:db8:0:2::1/64
 ip router isis
 ipv6 router isis
!
interface FastEthernet0/3
 ip address 172.16.3.1 255.255.255.0
 ipv6 address 2001:db8:0:3::1/64
 ip router isis
 ipv6 router isis
!
router isis
 net 01.0004.c150.f1c0.00
 authentication mode md5
 authentication key-chain forest
```

IS-IS 路由选择进程没有改变。IPv6 地址已经添加到每一个接口上，并且在每一个接口上都运行了 IPv6 协议的 IS-IS。其他路由器的配置也与上面类似。

图 10-54 中的 IPv6 地址也可以类似 IPv4 地址那样进行汇总。路由器 Zurich、Geneva 和 Bonn 在将路由通告给 L2 路由器时进行了汇总。IPv6 地址的汇总是在全局 IS-IS 路由选择进程模式下，通过指定 IPv6 地址簇和配置希望汇总的地址范围来实现的（参见示例 10-41～示例 10-43）。

示例 10-41 路由器 Zurich 正在汇总 IPv6 前缀路由

```
router isis
 address-family ipv6
 summary-prefix 2001:db8::/62
```

示例 10-42 路由器 Geneva 正在汇总 IPv6 前缀路由

```
router isis
 address-family ipv6
 summary-prefix 2001:db8::/62
```

示例 10-43 路由器 Bonn 正在汇总 IPv6 前缀路由

```
router isis
 address-family ipv6
 summary-prefix 2001:db8:0:10::/62
```

在示例 10-44 中，显示了路由器 Madrid 的 IPv6 路由表，包含了路由器 Zurich、Geneva 和 Bonn 汇总的地址范围。

示例 10-44 路由器 Madrid 的 IPv6 路由表中显示了由路由器 Zurich、Geneva 和 Bonn 汇总的地址范围

```
Madrid#show ipv6 route
IPv6 Routing Table - 12 entries
Codes: C - Connected, L - Local, S - Static, R - RIP, B - BGP
       U - Per-user Static route
```

（待续）

```
       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
I2  2001:DB8::/62 [115/20]
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.3
I2  2001:DB8:0:4::/64 [115/20]
     via FE80::2B0:64FF:FE30:1DE0, Serial0/0.1
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.3
C   2001:DB8:0:8::/64 [0/0]
     via ::, Serial0/0.1
L   2001:DB8:0:8::1/128 [0/0]
     via ::, Serial0/0.1
C   2001:DB8:0:9::/64 [0/0]
     via ::, Serial0/0.2
L   2001:DB8:0:9::1/128 [0/0]
     via ::, Serial0/0.2
C   2001:DB8:0:15::/64 [0/0]
     via ::, Serial0/0.3
L   2001:DB8:0:15::2/128 [0/0]
     via ::, Serial0/0.3
I2  2001:DB8:0:10::/62 [115/20]
     via FE80::205:5EFF:FE6B:50A0, Serial0/0.2
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
Madrid#
```

IPv4 和 IPv6 共享相同的拓扑结构，因此也共享同样的度量值。IPv6 的 TLV 使用扩展的度量，但在缺省情况下，IPv4 使用较窄幅度的度量，最大值为 63。因此，IPv6 的度量也受限为 63。每一个接口的缺省值为 10。度量的类型可以通过命令 **metric-style** 来更改。关键字 **wide**、**transition** 或 **wide transition** 可以使路由器在网络重新配置期间发送和接收较窄和较宽幅度的度量。通过命令 **show clns protocol** 的输出可以显示度量的类型，参见示例 10-45 所示。

示例 10-45　路由器 Zurich 配置为缺省的较窄幅度的度量

```
Zurich#show clns protocol
IS-IS Router: <Null Tag>
  System Id: 0000.0C76.5B7C.00  IS-Type: level-1-2
  Manual area address(es):
        01
  Routing for area address(es):
        01
  Interfaces supported by IS-IS:
        Serial0/0.1 - IP - IPv6
        Ethernet0/0 - IP - IPv6
  Redistribute:
    static (on by default)
  Distance for L2 CLNS routes: 110
  RRR level: none
  Generate narrow metrics: level-1-2
  Accept narrow metrics:   level-1-2
  Generate wide metrics:   none
  Accept wide metrics:     none
Zurich#
```

正如示例 10-45 中所看到的，路由器 Zurich 仅仅发送和接收较窄幅度的度量。这里请注意度量的类型对于 IPv4 和 IPv6 并不是专有的。

参见示例 10-46 所示，显示了路由器 Zurich 改变后的度量类型配置。

示例 10-46 路由器 Zurich 的度量类型配置为 transition 模式，可以同时接收和发送宽幅与窄幅度的度量

```
router isis
 metric-style transition
 address-family ipv6
  summary-prefix 2001:db8::/62
```

关键字 **transition** 将会使该路由器发送和接收宽幅度的度量和窄幅度的度量。示例 10-47 显示了路由器 Zurich 的结果。

示例 10-47 宽幅度的度量和窄幅度的度量都可以生成和接收

```
Zurich#show clns protocol

IS-IS Router: <Null Tag>
  System Id: 0000.0C76.5B7C.00  IS-Type: level-1-2
  Manual area address(es):
        01
  Routing for area address(es):
        01
  Interfaces supported by IS-IS:
        Serial0/0.1 - IP - IPv6
        Ethernet0/0 - IP - IPv6
  Redistribute:
    static (on by default)
  Distance for L2 CLNS routes: 110
  RRR level: none
  Generate narrow metrics: level-1-2
  Accept narrow metrics:   level-1-2
  Generate wide metrics:   level-1-2
  Accept wide metrics:     level-1-2
Zurich#
```

10.2.7 案例研究：过渡到多拓扑模式

在图 10-54 中的网络出现了一个问题。路由器 Frankfurt 的 IOS 软件不支持 IPv6 协议。缺省情况下，当 IPv6 协议配置在路由器 Bonn 和网络中其余的路由器上后，IPv6 协议也就添加到由 IPv4 协议为 IS-IS 的每一层创建的 IS-IS 拓扑中了。因为单一的拓扑是为 IS-IS 的每一层存在的，因此这个拓扑是由 IPv4 和 IPv6 协议共同共享的，IPv4 和 IPv6 协议必须同时配置在每一条链路和每一台路由器上。如果使用单一的拓扑，并同时配置 IPv4 和 IPv6 协议，那么具有 IPv4 地址的链路也都必须具有 IPv6 地址。这时，在路由器 Bonn 上就可以看出问题了。当配置 IPv6 协议时，它和路由器 Frankfurt 之间的邻接关系就无法创建。参见示例 10-48，使用调试命令 **debug isis adj-packets** 显示了有关 IPv6 地址的问题所在。

示例 10-48 使用调试命令 **debug isis adj-packets** 显示了单一拓扑模式下有关 IPv6 地址的一个错误

```
Bonn#debug is adj-packets fastethernet0/0
IS-IS Adjacency related packets debugging is on
Bonn#
ISIS-Adj: Sending L1 LAN IIH on FastEthernet0/0, length 1497
ISIS-Adj: Sending L2 LAN IIH on FastEthernet0/0, length 1497
ISIS-Adj: Rec L1 IIH from 0000.0c8d.34f1 (FastEthernet0/0), cir type L1, cir id
0000.0C0A.2AA9.01, length 1497
ISIS-Adj: No usable IPv6 linklocal addresses in LAN IIH from FastEthernet0/0
```

IOS 软件系统除了支持缺省的单一拓扑模式，还支持多拓扑模式的 IS-IS。[1] 在单一拓扑模式下，IPv4、IPv6 和 CLNS 都共享相同的 IS-IS 拓扑。在配置了多拓扑模式后，IOS 软件系统可以支持两种拓扑：一种与 IPv6 相关，另外一种与 IPv4 和 CLNS 相关。对于每一种拓扑都会有各自独立的 SPF 计算。

正如在“多拓扑结构”部分中讲述的，单拓扑的 IS-IS 和多拓扑的 IS-IS 使用不同的 TLV 交换信息。一台不能支持多拓扑模式的路由器将无法理解多拓扑 TLV。一台多拓扑模式的路由器可以处理单拓扑模式的 TLV：如果在邻接连接上没有收到多拓扑 TLV，那么这台接收路由器就假定它的这个邻居是 IPv4/CLNS 拓扑的一部分。这在单拓扑模式和 IPv6 都已经配置在网络中，并且你希望把网络迁移到多拓扑模式的时候可能会出现问题。在迁移期间，IPv6 网络的区域会变得不可到达。

IOS 软件系统提供了一个过渡模式（transition mode），在这种模式下可以发送单拓扑和多拓扑的 TLV，但是路由器只能在多拓扑模式下操作。在所有的路由器都配置完成后，将会删除过渡模式。

在配置多拓扑模式之前，在路由器上必须配置用在多拓扑 TLV 中的扩展度量。

在图 10-54 中显示的网络上配置了多拓扑模式。所有路由器最初的配置都是过渡模式，具体配置参见示例 10-49。

示例 10-49　图 10-54 中的所有路由器都具有以下配置

```
router isis
 metric-style wide transition
 address-family ipv6
  multi-topology transition
```

路由器 Frankfurt 不支持 IPv6 和多拓扑模式，它惟一的配置变化就是将度量改变为宽幅度的度量类型。由于路由器 Frankfurt 不发送多拓扑 TLV，所以 Frankfurt 路由器和它的链路都保留为缺省的 IPv4 拓扑模式。

从单拓扑模式到多拓扑模式的转换需要两条命令：第一条命令将度量类型从窄幅度改变为宽幅度，第二条命令启动多拓扑模式。新的拓扑共享 NET 和 IPv4 拓扑中的 L1/L2 边界。

在路由器 Bonn 上可以看到两个不同的拓扑。路由器 Bonn 上的 IS-IS IPv6 拓扑显示了从 L1 路由器到 L2 路由器的 IPv6 路径。这里有两条 L1 路由器的条目。一条是对应路由器 Bonn 的，另一条是对应路由器 Frankfurt 的。星号“*”表示这台路由器没有 IPv6 路径，但存在链路。在示例 10-50 中显示了命令 **show isis ipv6 topology** 和 **show isis topology** 的输出，读者可以比较一下 IPv6 的拓扑和 IPv4 的拓扑。命令 **show isis topology** 显示了缺省的 IPv4 的拓扑。

示例 10-50　在路由器 Bonn 上显示了 IPv4 的拓扑和 IPv6 的拓扑

```
Bonn#show isis ipv6 topology

IS-IS IPv6 paths to level-1 routers
System Id            Metric     Next-Hop             Interface   SNPA
Bonn                 --
Frankfurt            **
IS-IS IPv6 paths to level-2 routers
System Id            Metric     Next-Hop             Interface   SNPA
```

（待续）

[1] 在 IOS 软件的 12.2(15)T、12.2(18)S、12.0(26)S、12.3、12.3(2)T，以及以后的版本支持多拓扑模式。

```
Bonn                --
Zurich              20         Madrid              Se0/0.1     DLCI 171
Madrid              10         Madrid              Se0/0.1     DLCI 171
Geneva              20         Madrid              Se0/0.1     DLCI 171
Bonn#show isis topology
IS-IS paths to level-1 routers
System Id           Metric     Next-Hop            Interface   SNPA
Bonn                --
Frankfurt           10         Frankfurt           Fa0/0       0000.0c8d.34f1
IS-IS paths to level-2 routers
System Id           Metric     Next-Hop            Interface   SNPA
Bonn                --
Zurich              20         Madrid              Se0/0.1     DLCI 171
Madrid              10         Madrid              Se0/0.1     DLCI 171
Geneva              20         Madrid              Se0/0.1     DLCI 171
Bonn#
```

如果在点到点链路上至少存在一个拓扑，就可以形成单个邻接关系。在示例 10-51 中使用命令 **show clns is-neighbors detail** 显示了路由器 Bonn 的接口 Serial0/0.1 上有一条与路由器 Madrid 之间的邻接，但是 IPv4 和 IPv6 拓扑都共享这个邻接。

示例 10-51 在路由器 Bonn 和 Madrid 之间形成单个邻接，却由 IPv4 和 IPv6 拓扑共享这个邻接

```
Bonn#show clns is-neighbors detail
System Id      Interface   State  Type Priority  Circuit Id         Format
Madrid         Se0/0.1     Up     L2   0         00                 Phase V
  Area Address(es): 03
  IP Address(es):  172.16.9.1*
  IPv6 Address(es): FE80::204:C1FF:FE50:E700
  Uptime: 00:23:14
  NSF capable
  Topology: IPv4, IPv6
Frankfurt      Fa0/0       Up     L1   64        Bonn.01            Phase V
  Area Address(es): 03
  IP Address(es):  172.16.19.2*
  Uptime: 00:23:17
Bonn#
```

10.2.8 案例研究：层之间的路由泄漏

回忆当一台 L1/L2 路由器发送 L1 LSP 到一个区域时，它会通过在 LSP 中设置 ATT 位来通知其他的 L1 路由器它能够到达另一个区域。当需要转发数据包到区域外部时，L1 路由器会把数据包转发到最近的 L2 路由器（到设置 ATT 位的路由器具有最小代价的路径）。

考虑图 10-55 中的网络。在区域 1，路由器 Prague 和 Bucharest 都是 L1/L2 路由器。它们都是在 L1 更新数据包中设置了 ATT 位，并发送到路由器 Belgrade 和 Zagreb，并且都是 L1 路由器。示例 10-52 中显示了路由器 Belgrade 的 IS 数据库。

示例 10-52 区域 1 中的一台 L1 路由器的 IS-IS 数据库显示了两台都设置了 ATT 位的 L1/L2 路由器

```
Belgrade#show is database

IS-IS Level-1 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
```

（待续）

```
Bucharest.00-00          0x00000052   0xE802          1104              1/0/0
Prague.00-00             0x00000052   0xEDC1          1099              1/0/0
Zagreb.00-00             0x0000003B   0x13A5          1076              0/0/0
Belgrade.00-00         * 0x00000050   0x5EC1          485               0/0/0
Belgrade#
```

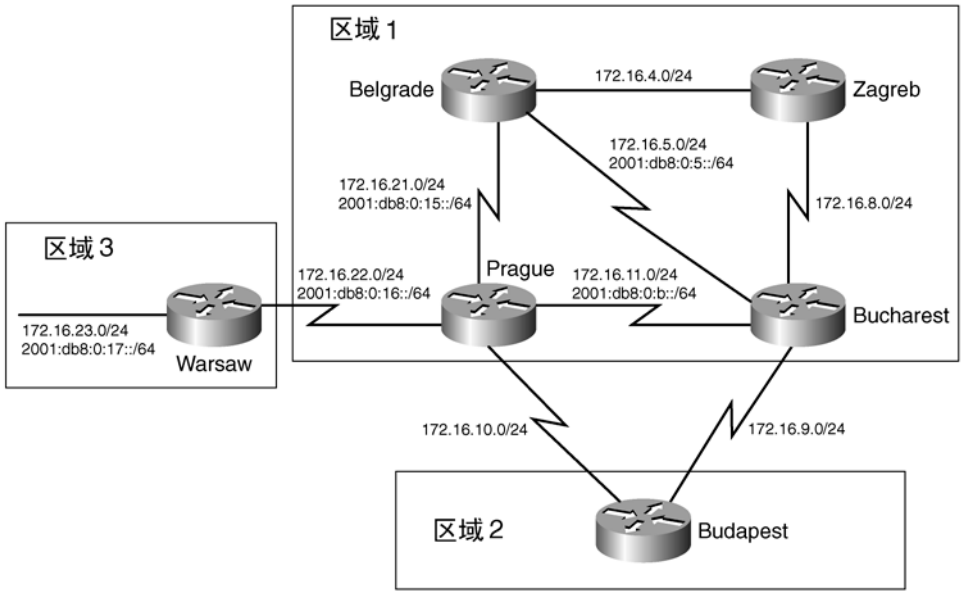


图 10-55 在区域 1 中的网络具有多台 L1/L2 路由器，这将会导致从区域 1 到其他区域的路由选择没有效率

路由器 Belgrade 的 IS-IS 数据库显示了区域 1 中的所有路由器。路由器 Prague 和 Bucharest 都是 L1/L2 路由器，从而都设置了 ATT 位。回忆 L1 路由器使用最近的 L1/L2 路由器来转发业务量到区域外部。这两台 L1/L2 路由器到达路由器 Belgrade 具有相等的距离。路由器 Belgrade 的 IPv4 路由表（参见示例 10-53）和 IPv6 路由表（参见示例 10-54）显示了路由器 Bucharest 和 Prague 都被用来转发业务量到区域外部。

示例 10-53 L1 路由器 Belgrade 上的 IPv4 路由表显示了到达区域外部的两条路径，表示具有两条缺省路由

```
Belgrade#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 172.16.5.1 to network 0.0.0.0
     172.16.0.0/24 is subnetted, 8 subnets
C       172.16.21.0 is directly connected, Serial0/0.1
i L1    172.16.22.0 [115/20] via 172.16.21.2, Serial0/0.1
i L1    172.16.8.0 [115/20] via 172.16.4.2, Serial0/0.2
                    [115/20] via 172.16.5.1, Serial0/0.3
i L1    172.16.9.0 [115/20] via 172.16.5.1, Serial0/0.3
i L1    172.16.10.0 [115/20] via 172.16.21.2, Serial0/0.1
i L1    172.16.11.0 [115/20] via 172.16.21.2, Serial0/0.1
                    [115/20] via 172.16.5.1, Serial0/0.3
```

（待续）

```
C       172.16.4.0 is directly connected, Serial0/0.2
C       172.16.5.0 is directly connected, Serial0/0.3
i*L1 0.0.0.0/0 [115/10] via 172.16.5.1, Serial0/0.3
               [115/10] via 172.16.21.2, Serial0/0.1
Belgrade#
```

示例 10-54 L1 路由器 Belgrade 上的 IPv6 路由表显示了到达区域外部的两条路径，表示具有两条缺省路由

```
Belgrade#show ipv6 route
IPv6 Routing Table - 9 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
I1  ::/0 [115/10]
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.3
C   2001:DB8:0:5::/64 [0/0]
     via ::, Serial0/0.3
L   2001:DB8:0:5::2/128 [0/0]
     via ::, Serial0/0.3
I1  2001:DB8:0:B::/64 [115/20]
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.3
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
C   2001:DB8:0:15::/64 [0/0]
     via ::, Serial0/0.1
L   2001:DB8:0:15::1/128 [0/0]
     via ::, Serial0/0.1
I1  2001:DB8:0:16::/64 [115/20]
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
Belgrade#
```

正如读者在示例 10-53 和示例 10-54 中所看到的，路由器 Belgrade 的路由表中只有 L1 路由。所有区域外部的地址都跟在由缺省路由指定的路径后。缺省路由（IPv4：0.0.0.0/0，IPv6：::/0）显示了两个下一跳地址，一个来自路由器 Bucharest（S0/0.3），一个来自路由器 Prague（S0/0.1）。

地址 172.16.23.0/24 和 2001:db8:0:17::/64 是区域 3 中和路由器 Warsaw 相连的地址。对于路由器 Belgrade 来说，到达这些地址的最优路径是通过路由器 Prague。但是，由于对于路由器 Belgrade 来说，路由器 Bucharest 和 Prague 是相等距离的 L1/L2 路由器，因此在路由器 Belgrade 发送业务量到地址 172.16.23.0/24 和 2001:db8:0:17::/64 时将会在两台 L1/L2 路由器之间进行负载均分共享。一半的业务流量通过较优的路径，另一半则通过一条较长的路径。示例 10-55 中显示了路由器 Belgrade 上 ping172.16.23.1 和记录的路由，可以看出，一条路由通过 Bucharest（172.16.11.2），而另一条路由通过 Prague（172.16.22.2）。

示例 10-55 使用带路由记录选项的 ping 显示了一个数据包从路由器 Belgrade 到 Warsaw 的往返路由

```
Belgrade#ping
Protocol [ip]:
Target IP address: 172.16.23.1
Extended commands [n]: y
```

（待续）

```
Loose, Strict, Record, Timestamp, Verbose[none]: r
Loose, Strict, Record, Timestamp, Verbose[RV]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.23.1, timeout is 2 seconds:
<...>Reply to request 0 (416 ms).  Received packet has options
 Total option bytes= 40, padded length=40
 Record route:
   (172.16.5.2)
   (172.16.11.2)
   (172.16.22.2)
   (172.16.23.1)
   (172.16.22.1)
   (172.16.21.2)
   (172.16.5.2) <*>
   (0.0.0.0)
   (0.0.0.0)
 End of list
Reply to request 1 (216 ms).  Received packet has options
 Total option bytes= 40, padded length=40
 Record route:
   (172.16.21.1)
   (172.16.22.2)
   (172.16.23.1)
   (172.16.22.1)
   (172.16.21.2)
   (172.16.21.1) <*>
   (0.0.0.0)
   (0.0.0.0)
   (0.0.0.0)
 End of list
```

通过将 172.16.23.0 的地址通告到区域 1 的 L1 路由器，可以控制从路由器 Belgrade 到这个地址的业务流路径。这个把 L2 的路由通告给 L1 路由器的做法称为路由泄漏（route leaking）。下行方向的路由泄漏要求 L1 路由器需要明确地知道附加的地址和这些地址的路径。

路由泄漏的配置方式是，首先通过创建一个包含所要通告到该区域的地址列表，然后配置这个列表分发到 L1 路由器上。路由器 Prague 的配置参见示例 10-56 所示。

示例 10-56　路由器 Prague 配置了路由泄漏，将 L2 的路由泄漏到 L1 区域

```
access-list 100 permit ip 172.16.23.0 0.0.0.255 any
ipv6 prefix-list warsaw seq 5 permit 2001:DB8:0:17::/64
router isis
 redistribute isis ip level-2 into level-1 distribute-list 100
 address-family ipv6
  redistribute isis level-2 into level-1 distribute-list warsaw
 exit-address-family
```

通过访问列表编号 100 定义了所要通告的 IPv4 地址，另外通过名字为 warsaw 的前缀列表定义了需要通告的 IPv6 地址。**Redistribute isis ip** 命令检查匹配列表 100 的 L2 IPv4 地址，并将它们通告到该区域内的 L1 路由器上。在命令 **address-family ipv6** 模式下使用 **Redistribute isis** 命令检查匹配名字为 warsaw 的前缀列表的那些 L2 IPv6 地址，并将它们通告到 L1 区域内的 L1 路由器上。访问列表的讲述参见附录 B，有关路由重新分配的更详细讲述参见第 11 章。

泄漏的路由通过和其他地址一样的方式进行通告：使用类型为 128、130 或 135 的 IPv4 可达性 TLV，以及类型为 236 和 237 的 IPv6 可达性 TLV。类型为 128 和 130 的 TLV 使用在窄幅度的度量，而类型为 135 的 TLV 使用在宽幅度的度量中。为了把泄漏的路由和其他相连

接的 IP 路由或外部的 IP 路由区分开来，这里定义了一个 Up/Down 位，这个位的定义已经在前面的章节“跨域范围的前缀分发”中讲述过了。如果 Up/Down 位设置为 0，那么这条路由就是始发于本地 L1 区域的。如果 Up/Down 位设置为 1，那么说明这条路由是由 L2 泄漏到 L1 的。设置该位有助于避免产生路由环路。一台 L1/L2 路由器将不会把设置了 Up/Down 位的 L1 路由地址通告给其他的 L2 路由器。

泄漏到 L1 的路由在路由表和 IS-IS 数据库中已有标识。在示例 10-57 和示例 10-58 中分别显示了路由器 Belgrade 的 IPv4 路由表和 IPv6 路由表。而在示例 10-59 中显示了路由器 Belgrade 的 IS-IS 数据库信息。

示例 10-57 在路由器 Belgrade 的路由表中泄漏的路由被标识为“ia”路由，即区域间的路由

```
Belgrade#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 172.16.5.1 to network 0.0.0.0
     172.16.0.0/24 is subnetted, 9 subnets
C       172.16.21.0 is directly connected, Serial0/0.1
i L1    172.16.22.0 [115/20] via 172.16.21.2, Serial0/0.1
i ia    172.16.23.0 [115/30] via 172.16.21.2, Serial0/0.1
i L1    172.16.8.0 [115/20] via 172.16.4.2, Serial0/0.2
                   [115/20] via 172.16.5.1, Serial0/0.3
i L1    172.16.9.0 [115/20] via 172.16.5.1, Serial0/0.3
i L1    172.16.10.0 [115/20] via 172.16.21.2, Serial0/0.1
i L1    172.16.11.0 [115/20] via 172.16.21.2, Serial0/0.1
                    [115/20] via 172.16.5.1, Serial0/0.3
C       172.16.4.0 is directly connected, Serial0/0.2
C       172.16.5.0 is directly connected, Serial0/0.3
i*L1 0.0.0.0/0 [115/10] via 172.16.5.1, Serial0/0.3
               [115/10] via 172.16.21.2, Serial0/0.1
Belgrade#
```

示例 10-58 在路由器 Belgrade 的路由表中泄漏的 IPv6 地址路由被标识为“IA”路由，即区域间的路由

```
Belgrade#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
I1  ::/0 [115/10]
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.3
C   2001:DB8:0:5::/64 [0/0]
     via ::, Serial0/0.3
L   2001:DB8:0:5::2/128 [0/0]
     via ::, Serial0/0.3
I1  2001:DB8:0:B::/64 [115/20]
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.3
```

（待续）

```
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
C    2001:DB8:0:15::/64 [0/0]
     via ::, Serial0/0.1
L    2001:DB8:0:15::1/128 [0/0]
     via ::, Serial0/0.1
I1   2001:DB8:0:16::/64 [115/20]
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
IA   2001:DB8:0:17::/64 [115/30]
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
L    FE80::/10 [0/0]
     via ::, Null0
L    FF00::/8 [0/0]
     via ::, Null0
Belgrade#
```

示例 10-59 在路由器 Belgrade 的 IS-IS 数据库中泄漏的地址被标识为“IP-inter-area”地址

```
Belgrade#show is database Prague.00-00 detail l1
IS-IS Level-1 LSP Prague.00-00
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Prague.00-00          0x0000006D   0x7A37        655               1/0/0
  Area Address: 01
  Topology:     IPv4 (0x0) IPv6 (0x4002 ATT)
  NLPID:        0xCC 0x8E
  Hostname: Prague
  IP Address:   172.16.11.1
  Metric: 10         IP 172.16.22.0/24
  Metric: 10         IP 172.16.21.0/24
  Metric: 10         IP 172.16.10.0/24
  Metric: 10         IP 172.16.11.0/24
  IPv6 Address: 2001:DB8:0:15::2
  Metric: 10         IPv6 (MT-IPv6) 2001:DB8:0:16::/64
  Metric: 10         IPv6 (MT-IPv6) 2001:DB8:0:15::/64
  Metric: 10         IPv6 (MT-IPv6) 2001:DB8:0:B::/64
  Metric: 10         IS-Extended Belgrade.00
  Metric: 10         IS-Extended Bucharest.00
  Metric: 10         IS (MT-IPv6) Belgrade.00
  Metric: 10         IS (MT-IPv6) Bucharest.00
  Metric: 20         IP-Interarea 172.16.23.0/24
  Metric: 20         IPv6-Interarea (MT-IPv6) 2001:DB8:0:17::/64
Belgrade#
```

在路由表中，将 IPv4 的泄漏路由指定为“ia”类型，将 IPv6 的泄漏路由指定为“IA”类型，从而可以显示出这些 IPv4 和 IPv6 泄漏路由为 IS-IS 区域间路由。路由器把由路由泄漏发起的数据库条目显示为 IP-Interarea 和 IPv6-Interarea 地址。

10.2.9 案例研究：多个 L1 区域运行在同一台路由器上

IOS 软件系统具有在单台路由器上支持多个 L1 区域同时运行的能力：在 IOS 路由器上最多可配置 29 个 L1 区域，但是只能配置一个 L2 区域。在缺省情况下，第一个配置的 IS-IS 路由选择进程定义为 L2 区域。它也可以定义单个 L1 区域。为了在路由器上增加运行另外的 L1 区域，需要定义另外的 IS-IS 路由选择进程。在所配置的每一个 L1 区域内的路由器都是通过这台 L1/L2 路由器来通信的。因为每一个区域都不需要具有单一的 L1/L2 路由器到达网络的其他部分，因此 L1/L2 路由器的数量可以减到最少。

在图 10-55 的网络中，路由器 Warsaw 位于区域 3。对于该区域，路由器 Warsaw 是一台 L1/L2 路由器。从路由器 Warsaw 到网络的其他部分的惟一连接是通过路由器 Prague 的。改变路由器 Prague 的配置，使 Prague 同时成为区域 1 和区域 3 的 L2 路由器。路由器 Prague 新的配置参见示例 10-60。

示例 10-60 路由器 Prague 配置了多个 L1 区域

```
router isis
 net 01.0004.c150.e700.00
 metric-style wide
 address-family ipv6
  multi-topology

router isis three
 net 03.0004.c150.e700.00
 metric-style wide
 address-family ipv6
  multi-topology

interface Serial0/0.1 point-to-point
 description Warsaw
 ip address 172.16.22.2 255.255.255.0
 ipv6 address 2001:db8:0:16::2/64
 ip router isis three
 ipv6 router isis three
```

配置的第二个 IS-IS 路由选择进程命名为“three”。同时，把连接路由器 Prague 和 Warsaw 的接口 Serial0/0.1 指定运行 *IS-IS three* 进程。*IS-IS “three”* 自动地配置为 L1 区域，这可以在路由器 Prague 上通过命令 **show clns neighbor** 看到，参见示例 10-61 所示。

示例 10-61 在路由器 Prague 上配置了多个 L1 区域

```
Prague#show clns neighbor
Area null:
System Id      Interface   SNPA            State  Holdtime  Type Protocol
Budapest       Se0/0.3     DLCI 115        Up     24        L2   IS-IS
Belgrade       Se0/0.2     DLCI 116        Up     26        L1   M-ISIS
Bucharest      Se0/0.4     DLCI 113        Up     22        L1L2 M-ISIS
Area three:
System Id      Interface   SNPA            State  Holdtime  Type Protocol
Warsaw         Se0/0.1     DLCI 117        Up     28        L1   M-ISIS
Prague#
```

因为路由器 Warsaw 所有的邻居都出现在区域 3 内，因此它现在被配置作为 L1 路由器，参见示例 10-62 所示。在这个简单的网络中，这一步骤是不需要的，因为路由器 Warsaw 只有一个和路由器 Prague 相连的连接通向区域 3 外部，路由器 Prague 和路由器 Warsaw 只需要使用 L1 进行通信。在一个大型的网络中可能希望具有这一步骤，以便确保路由器 Warsaw 不会试图通过 L2 连接到其他路由器。

示例 10-62 路由器 Warsaw 配置作为一台 L1 路由器

```
router isis
 is-type level-1
```

示例 10-63 显示了路由器 Warsaw 作为一台 L1/L2 路由器时的 IS-IS 数据库和路由表。

示例 10-64 显示了路由器 Warsaw 作为一台 L1 路由器时的 IS-IS 数据库和路由表。

示例 10-63 路由器 Warsaw 的数据库包含了区域 3 路由器和所有 L2 路由器的条目

```
Warsaw#show is database
IS-IS Level-1 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Prague.00-00          0x0000000A   0x08B1        1048              1/0/0
Warsaw.00-00        * 0x00000075   0x3D13        1085              1/0/0
IS-IS Level-2 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Bucharest.00-00       0x0000007A   0x59D0        1060              0/0/0
Prague.00-00          0x00000090   0x433A        1078              0/0/0
Warsaw.00-00        * 0x00000001   0xA828        1080              0/0/0
Budapest.00-00        0x00000073   0xFE95        344               0/0/0
Budapest.01-00        0x00000066   0xFF8B        444               0/0/0
Warsaw#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
     172.16.0.0/24 is subnetted, 10 subnets
i L2    172.16.21.0 [115/20] via 172.16.22.2, Serial0/0.1
C       172.16.22.0 is directly connected, Serial0/0.1
C       172.16.23.0 is directly connected, FastEthernet0/0
i L2    172.16.12.0 [115/30] via 172.16.22.2, Serial0/0.1
i L2    172.16.8.0 [115/30] via 172.16.22.2, Serial0/0.1
i L2    172.16.9.0 [115/30] via 172.16.22.2, Serial0/0.1
i L2    172.16.10.0 [115/20] via 172.16.22.2, Serial0/0.1
i L2    172.16.11.0 [115/20] via 172.16.22.2, Serial0/0.1
i L2    172.16.4.0 [115/30] via 172.16.22.2, Serial0/0.1
i L2    172.16.5.0 [115/30] via 172.16.22.2, Serial0/0.1
Warsaw#
```

示例 10-64 路由器 Warsaw 的数据库仅仅包含了区域 3 的路由器和链路，而它的路由表仅仅包含了不属于区域 3 的地址的缺省路由

```
Warsaw#show is database

IS-IS Level-1 Link State Database:
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Prague.00-00          0x00000009   0x2BDF        923               1/0/0
Warsaw.00-00        * 0x00000073   0x6CF3        924               0/0/0
Warsaw#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 172.16.22.2 to network 0.0.0.0
     172.16.0.0/24 is subnetted, 2 subnets
C       172.16.22.0 is directly connected, Serial0/0.1
C       172.16.23.0 is directly connected, FastEthernet0/0
i*L1 0.0.0.0/0 [115/10] via 172.16.22.2, Serial0/0.1
Warsaw#
```

当路由器 Warsaw 作为区域 3 的 L1/L2 路由器时，它的 IS 数据库包含了区域 3 内每一台路由器的条目和每一台 L2 路由器的条目，因此那些 L2 路由器的所有地址都可以到达。它的

路由表包含了网络中每一个可达的地址的路由条目。路由器 Warsaw 作为 L1 路由器时，明显地减小了它的 IS-IS 数据库和路由器的大小。

10.3 集成 IS-IS 协议的故障诊断

IS-IS 协议故障诊断的基本方法和第 8 章讲述的 OSPF 协议的故障诊断方法十分相似。集成 IS-IS 协议和其他 IP 路由选择协议的故障诊断相比，一个主要的不同之处是 IS-IS 协议使用的是 CLNS PDU 数据包，而不是 IP 数据包。如果你是在对协议本身做故障诊断的话，记住你是在做 CLNS 协议的故障诊断，而不是 IP 协议。

正如所有的路由选择协议一样，故障诊断的第一步是检查路由表来获取精确的信息。如果一个预期的路由条目在路由表中丢失了或变得不正确，那么故障诊断剩下来的任务就是确定引起故障的源头了。

在检查过路由表之后，查看链路状态数据库就是获取故障诊断信息的一个最重要的来源。正如在第 8 章中所建议的，一个比较好的实际经验是，为每一个区域保存一份 L1 类型的链路状态数据库的拷贝，以及保存一份 L2 类型的链路状态数据库的拷贝。这些保存的数据库拷贝应该有规律地进行更新，并作为日常工作的一部分。这样在网络出现问题或错误时，这些保存的数据库拷贝将可以提供一个稳定状态的参考。在检查一台单独的路由器配置时，可以考虑以下问题：

- 在 IS-IS 协议配置下面的 **net** 语句是否指定了正确的 NET 地址？在该路由器上配置的区域 ID 和系统 ID 是否正确无误？配置的 NET 地址是否符合所在网络上正在使用的 CLNS 编址约定？
- 是否在正确的接口上使用命令 **ip router isis** 或 **ipv6 router isis** 启动 IS-IS 协议？
- IP 地址和子网掩码或前缀的配置是否正确？在一个集成 IS-IS 环境里检查这些配置显得加倍重要，因为配置错误的 IP 地址不会妨碍部分地建立一个 IS-IS 邻接关系。

10.3.1 IS-IS 邻接关系的故障诊断

使用命令 **show clns is-neighbors** 可以显示 IS-IS 的邻居表。缺省条件下显示的是整张邻居表，当然也可以指定显示一个具体接口的邻居表。从这个表中，我们可以看出所有预期的邻居是否都出现了？并且它们的类型是否正确？为了获取更详细的信息，可以使用命令 **show clns is-neighbors detail** 来显示与每一个邻居相关联的区域地址和 IP 地址，以及每一个邻居的上线时间，等等。

在检查邻接关系时，可以考虑以下问题：

- 路由器的层（level）是否配置正确？L1 路由器只能和 L1 与 L1/L2 类型的路由器建立邻接关系，而 L2 路由器只能和 L2 与 L1/L2 类型的路由器建立邻接关系。
- 是否这两台邻居路由器都在发送 Hello 数据包？Hello 数据包的层（level）是否正确？Hello 数据包包含的参数是否正确？调试命令 **debug isis adj-pachets** 是用来查看 Hello 数据包的一个比较有用的命令。
- 如果使用了认证，那么在邻居之间的口令和认证是否相同？记住区域（第 1 层）和域（第 2 层）认证不是验证普通的邻接关系的，它们只验证 LSP 的交换。

- 是否存在任何阻塞 IS-IS 或者 CLNS 协议的访问列表？

在图 10-54 中的路由器 Bonn 发生了一个变化。Bonn 和 Frankfurt 不再能够从其他地点通过 IPv4 访问了。参考示例 10-65，从路由器 Bonn 的 IPv4 路由表中显示，其中没有经过路由器 Madrid（接口 Serial0/0.1）学到的路由。

示例 10-65 路由器 Bonn 的 IPv4 路由表显示了从路由器 Frankfurt 的接口 FA0/0 学到的 IS-IS 路由，但从路由器 Madrid 的接口 Serial0/0.1 没有学到路由

```
Bonn#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
     172.16.0.0/16 is variably subnetted, 8 subnets, 2 masks
C       172.16.24.0/24 is directly connected, Loopback1
C       172.16.25.0/24 is directly connected, Loopback2
i L1    172.16.16.0/24 [115/20] via 172.16.19.2, FastEthernet0/0
i su    172.16.16.0/21 [115/10] via 0.0.0.0, Null0
i L1    172.16.17.0/24 [115/20] via 172.16.19.2, FastEthernet0/0
i L1    172.16.18.0/24 [115/20] via 172.16.19.2, FastEthernet0/0
C       172.16.19.0/24 is directly connected, FastEthernet0/0
C       172.16.9.0/24 is directly connected, Serial0/0.1
Bonn#
```

命令 **show clns is-neighbors** 的输出显示了一条从路由器 Bonn 到 Madrid 的现有邻接，如示例 10-66 所示。但是，请注意路由器 Madrid 的类型是 IS 而不是 L1 也不是 L2。这表明该邻接出现了问题。使用调试命令 **debug isis adj-packets** 可以给我们一点提示查找问题出在哪里，参见示例 10-67 所示。

示例 10-66 路由器 Bonn 的 CLNS IS-IS 邻居表显示它到 Madrid 和 Frankfurt 的邻接，但是到 Madrid 的邻接出了一些问题

```
Bonn#show clns is-neighbors
System Id      Interface   State  Type Priority  Circuit Id         Format
Madrid         Se0/0.1     Up     IS   0         00                 Phase V
Frankfurt      Fa0/0       Up     L1   64        Bonn.03            Phase V
Bonn#
```

示例 10-67 利用调试命令 **debug isis adj-packets** 观察 IS-IS Hello（IIHs）的详细信息。这个输出显示来自于图 10-54 的路由器 Bonn，串行接口 Serial0/0.1 的 IP 地址出现了问题

```
Bonn#debug isis adj-packets
IS-IS Adjacency related packets debugging is on
Bonn#
ISIS-Adj: Sending serial IIH on Serial0/0.1, length 1499
ISIS-Adj: Rec serial IIH from DLCI 171 (Serial0/0.1), cir type L2, cir id 01, length 1499
ISIS-Adj: No usable IP interface addresses in serial IIH from Serial0/0.1
ISIS-Adj: Sending L1 LAN IIH on FastEthernet0/0, length 1497
ISIS-Adj: Sending L2 LAN IIH on FastEthernet0/0, length 1497
ISIS-Adj: Sending L1 LAN IIH on FastEthernet0/0, length 1497
ISIS-Adj: Rec L1 IIH from 0000.0c8d.34f1 (FastEthernet0/0), cir type L1, cir id
0000.0C0A.2AA9.03, length 1497
```

回顾一下路由器 Bonn 上的 IPv4 地址，说明串行接口 Serial0/0.1 的 IP 地址被无意改变了。IOS 软件在建立邻接关系之前会检查 TLV 中的地址。如果这个接口地址 TLV（类型 132）中的 IPv4 地址和接收它的接口不属于同一个子网，那么邻接关系会建立，但邻接关系的类型是 IS 而不是 L1 或 L2，这表明存在影响邻接关系的配置问题。改正了这个问题后，路由器 Bonn 的 CLNS IS 邻居表和 IPv4 路由表就正常了，参见示例 10-68 所示。

示例 10-68 解决问题后的 CLNS 邻居表显示出邻接类型是 L1、L2，或 L1/L2。路由器 Bonn 的路由表显示了来自这两个邻接邻居的 IP 路由

```
Bonn#show clns is-neighbors
System Id      Interface   State  Type Priority  Circuit Id         Format
Madrid         Se0/0.1     Up     L2   0         00                 Phase V
Madrid         Fa0/0       Up     L1   64        Bonn.03            Phase V
Bonn#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
     172.16.0.0/16 is variably subnetted, 11 subnets, 2 masks
C       172.16.24.0/24 is directly connected, Loopback1
C       172.16.25.0/24 is directly connected, Loopback2
i L2    172.16.21.0/24 [115/20] via 172.16.9.1, Serial0/0.1
i L1    172.16.16.0/24 [115/20] via 172.16.19.2, FastEthernet0/0
i su    172.16.16.0/21 [115/10] via 0.0.0.0, Null0
i L1    172.16.17.0/24 [115/20] via 172.16.19.2, FastEthernet0/0
i L1    172.16.18.0/24 [115/20] via 172.16.19.2, FastEthernet0/0
C       172.16.19.0/24 is directly connected, FastEthernet0/0
i L2    172.16.8.0/24 [115/20] via 172.16.9.1, Serial0/0.1
C       172.16.9.0/24 is directly connected, Serial0/0.1
i L2    172.16.0.0/21 [115/30] via 172.16.9.1, Serial0/0.1
Bonn#
```

在问题解决后的路由器 Bonn 上执行命令 **debug isis adj-packets**，输出信息参见示例 10-69。该信息是在串行接口 Serial0/0.1 上接收的。

示例 10-69 在配置正确的路由器 Bonn 上观察 IS-IS Hello（IIHs）的详细信息

```
Bonn#debug isis adj-packets
IS-IS Adjacency related packets debugging is on
Bonn#
ISIS-Adj: Sending L1 LAN IIH on FastEthernet0/0, length 1497
ISIS-Adj: Sending L2 LAN IIH on FastEthernet0/0, length 1497
ISIS-Adj: Rec L1 IIH from 0000.0c8d.34f1 (FastEthernet0/0), cir type L1, cir id
0000.0C0A.2AA9.03, length 1497
ISIS-Adj: Rec serial IIH from DLCI 171 (Serial0/0.1), cir type L2, cir id 01, length 1499
ISIS-Adj: Local mode (IP, IPv6), remote mode (IP, IPv6)
ISIS-Adj: rcvd state UP, old state UP, new state UP
ISIS-Adj: Action = ACCEPT
ISIS-Adj: Sending L1 LAN IIH on FastEthernet0/0, length 1497l
ISIS-Adj: Sending serial IIH on Serial0/0.1, length 1499
Bonn#
```

如果没有在配置了 IPv4 地址的每一条链路上配置 IPv6 地址，类似的邻接问题也存在于单一拓扑模式下的 IPv6 中。当图 10-54 的网络配置成单一拓扑模式的 IPv6，路由器 Bonn 与 Frankfurt 之间就出问题了。路由器 Frankfurt 不支持 IPv6。示例 10-48 显示了路由器 Bonn 上

调试命令 **debug isis adj-packets** 的输出信息。调试信息说明存在一个链路本地地址的问题："在来自接口 FastEthernet0/0 的 LAN IIH 内没有可用的 IPv6 链路本地地址"。路由器 Bonn 上的 IOS 软件在建立邻接关系之前会检查地址和地址簇的有效性。在单一拓扑模式下，如果在一台路由器上同时配置了 IPv4 与 IPv6，那么每一条具有 IPv4 地址的链路都必须具有一个 IPv6 的地址，每一条具有 IPv6 地址的链路也必须具有一个 IPv4 的地址，同时邻居路由器也必须具有有效的 IPv4 和 IPv6 地址。在 Hello 数据包中，类型 232 的 TLV 包含了 IPv6 的链路本地地址。在单一拓扑模式下，如果一台邻居路由器在它的 Hello 数据包中没有包含有效的 IPv6 链路本地地址，那么将会形成 IS 类型的邻接关系（再次表明配置有问题）。

log-adjacency-changes 是一个 IS-IS 进程配置模式的命令，该命令将会在日志中记录所有邻接关系的变化，日志可能在缓存区也可能发送到一台日志服务器上。仔细阅读这样的日志记录对处理过去某段时间发生的邻接问题是很有用的。

10.3.2 IS-IS 链路状态数据库的故障诊断

IS-IS 链路状态数据库的信息可以通过命令 **show isis database** 来查看。如果一台路由器是 L1/L2 路由器，那么缺省情况下将会同时显示 L1 和 L2 类型的数据库。如果只需要查看其中一个数据库，可以使用 **level-1** 或 **level-2** 关键字。如果需要查看 LSP 更详尽的信息，可以使用 **detail** 关键字。如果指定一个 LSP ID，也可以查看单个 LSP 的信息，参见示例 10-70 所示。

示例 10-70 这个 LSP 来自于图 10-54 中路由器 Zurich 的 L2 类型的数据库

```
Zurich#show isis database detail Madrid.00-00
IS-IS Level-2 LSP Madrid.00-00
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Madrid.00-00          0x0000007C   0x5874        1023              0/0/0
  Auth:         Length: 17
  Area Address: 03
  Topology:     IPv4 (0x0) IPv6 (0x2)
  NLPID:        0xCC 0x8E
  Hostname: Madrid
  IP Address:   172.16.21.2
  IPv6 Address: 2001:DB8:0:15::2
  Metric: 10         IS-Extended Zurich.00
  Metric: 10         IS-Extended Bonn.00
  Metric: 10         IS-Extended Geneva.00
  Metric: 10         IS (MT-IPv6) Zurich.00
  Metric: 10         IS (MT-IPv6) Bonn.00
  Metric: 10         IS (MT-IPv6) Geneva.00
  Metric: 10         IP 172.16.21.0/24
  Metric: 10         IP 172.16.8.0/24
  Metric: 10         IP 172.16.9.0/24
  Metric: 10         IPv6 (MT-IPv6) 2001:DB8:0:8::/64
  Metric: 10         IPv6 (MT-IPv6) 2001:DB8:0:9::/64
  Metric: 10         IPv6 (MT-IPv6) 2001:DB8:0:15::/64
Zurich#
```

如果一个序列号显著地高于其他 LSP 的序列号，就可能表明这个区域不稳定或者是 L2 类型的骨干不稳定。另一个网络不稳定的提示是，某个 LSP 的抑制时间从来不会变得很小。如果怀疑网络不稳定，可以使用命令 **show isis spf-log** 列出这台路由器上最近执行的所有 SPF 计算。

在示例 10-71 中，显示了图 10-54 中路由器 Geneva 的 SPF 日志。在显示这个日志大约

3min 后，除了每隔 15min 由数据库重新刷新触发的周期性的 SPF 计算，并没有显示其他内容。在那个时间段，频繁的 SPF 计算开始产生，这表明网络发生了频繁的变化。[1]

示例 10-71 这个 SPF 的日志记录揭示了图 10-54 中区域 1 的不稳定性

```
Geneva#sh isis spf-log
     Level 1 SPF log
   When      Duration   Nodes   Count    Last trigger LSP     Triggers
02:43:09          12       3       1                         PERIODIC
02:28:08          12       3       1                         PERIODIC
02:13:06          12       3       1                         PERIODIC
01:58:05          12       3       1                         PERIODIC
01:43:03          12       3       1                         PERIODIC
01:28:02          12       3       1                         PERIODIC
01:13:00          12       3       1                         PERIODIC
00:57:59          12       3       1                         PERIODIC
00:42:58          12       3       1                         PERIODIC
00:27:56          12       3       1                         PERIODIC
00:12:55          12       3       1                         PERIODIC
00:03:08           8       3       1   0000.0C76.5B7C.00-00  LSPHEADER
00:02:35           8       3       1   0000.0C76.5B7C.00-00  LSPHEADER
00:02:23           8       3       1   0000.0C76.5B7C.00-00  LSPHEADER
00:01:50           8       3       1   0000.0C76.5B7C.00-00  LSPHEADER
00:01:14           4       1       1   0000.0C0A.2C51.00-00  TLVCONTENT
00:00:46           4       2       2   0000.0C0A.2C51.04-00  NEWLSP TLVCONTENT
00:00:20           4       1       3   0000.0C0A.2C51.00-00  NEWADJ TLVCONTENT
00:00:08           8       3       1   0000.0C76.5B7C.02-00  TLVCONTENT
Geneva#
```

为了进一步跟踪 SPF 日志暴露的网络震荡问题，还有 3 个有用的调试命令可以使用。示例 10-72、示例 10-73，以及示例 10-74 中显示了这 3 个调试命令的输出结果。在每一个输出中，从路由器 Frankfurt 的角度来看，调试信息都显示了图 10-54 中路由器 Bonn 的串行接口断开和重新连接的结果。首先，**debug isis spf-triggers** 命令显示了与触发一个 SPF 计算有关的事件消息，参见示例 10-72 所示。第二条命令是 **debug isis spf-events**，这个命令显示了触发事件引起 SPF 计算的一个详细报告，参见示例 10-73 所示。第三条命令是 **debug isis spf-statistics**，它显示了有关 SPF 计算本身的信息，参见示例 10-74 所示。这里值得特别注意的是，这次进行的是完全的计算，这可能给路由器带来性能上的问题。

示例 10-72 **debug isis spf-triggers** 命令显示了触发一个 SPF 计算的事件消息

```
Frankfurt#debug isis spf-triggers
IS-IS SPF triggering events debugging is on
Frankfurt#
ISIS-Spf: L1, LSP fields changed 0000.0C0A.2AA9.00-00
ISIS-Spf: L1, LSP fields changed 0000.0C0A.2AA9.00-00
Frankfurt#
```

示例 10-73 **debug isis spf-events** 显示了一个 SPF 计算的详细信息

```
Frankfurt#debug isis spf-events
IS-IS SPF events debugging is on
Frankfurt#
ISIS-Spf: L1 LSP 4 (0000.0C0A.2AA9.00-00) flagged for recalculation from 37D73FA
ISIS-Spf: Calculating routes for L1 LSP 4 (0000.0C0A.2AA9.00-00)
```

（待续）

[1] 开始的 4 个触发事件是由路由器 Zurich 的串行接口几次状态变化引起的，接下来的 3 个事件是由于删除和接着错误配置了链路口令引起的，最后一个事件是在配置了正确的口令时引起的。

```
ISIS-Spf: Add 172.16.19.0/255.255.255.0 to IP RIB, metric 20
ISIS-Spf: Next hop 0000.0C0A.2AA9/172.16.19.1 (Ethernet0) (rejected)
ISIS-Spf: Redundant IP route 172.16.24.0/255.255.255.0, metric 20, not added
ISIS-Spf: Redundant IP route 172.16.25.0/255.255.255.0, metric 20, not added
ISIS-Spf: Aging L1 LSP 4 (0000.0C0A.2AA9.00-00), version 105
ISIS-Spf: Aging IP 172.16.9.0/255.255.255.0, next hop 172.16.19.1
ISIS-Spf: Deleted NDB
ISIS-Spf:  Compute L1 SPT
ISIS-Spf: 3 nodes for level-1
ISIS-Spf: Move 0000.0C8D.34F1.00-00 to PATHS, metric 0
ISIS-Spf: Add 0000.0C0A.2AA9.03-00 to TENT, metric 10
ISIS-Spf: Move 0000.0C0A.2AA9.03-00 to PATHS, metric 10
ISIS-Spf: thru 0/2147483647/0, delay 0/0/0, mtu 0/2147483647/0, hops 0/0/0, ticks
0/0/0
ISIS-Spf: considering adj to 0000.0C0A.2AA9 (Ethernet0) metric 10, level 1, circuit
3, adj 1
ISIS-Spf:   (accepted)
ISIS-Spf: Add 0000.0C0A.2AA9.00-00 to TENT, metric 10
ISIS-Spf:   Next hop 0000.0C0A.2AA9 (Ethernet0)
ISIS-Spf: Move 0000.0C0A.2AA9.00-00 to PATHS, metric 10
ISIS-Spf: Add 172.16.19.0/255.255.255.0 to IP RIB, metric 20
ISIS-Spf: Next hop 0000.0C0A.2AA9/172.16.19.1 (Ethernet0) (rejected)
ISIS-Spf: Redundant IP route 172.16.24.0/255.255.255.0, metric 20, not added
ISIS-Spf: Redundant IP route 172.16.25.0/255.255.255.0, metric 20, not added
ISIS-Spf: Aging L1 LSP 2 (0000.0C8D.34F1.00-00), version 101
ISIS-Spf: Aging L1 LSP 3 (0000.0C0A.2AA9.03-00), version 99
ISIS-Spf: Aging L1 LSP 4 (0000.0C0A.2AA9.00-00), version 106
ISIS-Spf: Remove stale 0/0 from IP RIB
ISIS-Spf: L1 LSP 4 (0000.0C0A.2AA9.00-00) flagged for recalculation from 37D73FA
ISIS-Spf: Calculating routes for L1 LSP 4 (0000.0C0A.2AA9.00-00)
ISIS-Spf: Add 172.16.19.0/255.255.255.0 to IP RIB, metric 20
ISIS-Spf: Next hop 0000.0C0A.2AA9/172.16.19.1 (Ethernet0) (rejected)
ISIS-Spf: Add 172.16.9.0/255.255.255.0 to IP RIB, metric 20
ISIS-Spf: Next hop 0000.0C0A.2AA9/172.16.19.1 (Ethernet0) (accepted)
ISIS-Spf: Redundant IP route 172.16.24.0/255.255.255.0, metric 20, not added
ISIS-Spf: Redundant IP route 172.16.25.0/255.255.255.0, metric 20, not added
ISIS-Spf: Aging L1 LSP 4 (0000.0C0A.2AA9.00-00), version 107
```

示例 10-74 **debug isis spf-statistics** 显示了有关 SPF 计算本身的统计信息

```
Frankfurt#debug isis spf-statistics
IS-IS SPF Timing and Statistics Data debugging is on
Frankfurt#
ISIS-Spf:  Compute L1 SPT
ISIS-Spf: Complete L1 SPT,
ISIS-Spf:  Compute time 0.032/0.032, 2/1 nodes, 0/2 links, 0 suspends
ISIS-Spf:  Compute L1 SPT
ISIS-Spf: Complete L1 SPT,
ISIS-Spf:  Compute time 0.036/0.036, 2/1 nodes, 0/2 links, 0 suspends
```

在一个区域内的每一台路由器都必须维护一个同样的链路状态数据库。另外，IS-IS 域内的每一台 L1/L2 和 L2 路由器都必须维护一个同样的 L2 类型的数据库。如果你怀疑某台路由器的链路状态数据库不能正确同步，可以检查它的 LSP ID 以其校验和。相同的 LSP ID 应该存在于每一个数据库中，并且在每一个数据库中的每一个 LSP 的校验和也都应该相同。

有两条调试命令可以帮助我们查看数据库的同步过程。第一条命令是 **debug isis update-packets**，它显示了路由器接收和发送 SNP 与 LSP 的有关信息，参见示例 10-75 所示。第二条命令是 **debug isis snp-packets**，它显示了路由器接收和发送某个指定的 CSNP 与 PSNP 的有关信息，参见示例 10-76 所示。

示例 10-75 **debug isis update-packets** 显示了路由器接收和发送 SNP 与 LSP 的有关信息

```
Frankfurt#debug isis update-packets
IS-IS Update related packet debugging is on
Frankfurt#
ISIS-Upd: Rec L1 LSP 0000.0C0A.2AA9.00-00, seq 10, ht 1199,
ISIS-Upd: from SNPA 0005.5e6b.50a0 (Ethernet0)
ISIS-Upd: LSP newer than database copy
ISIS-Upd: Leaf routes changed
ISIS-Upd: Rec L1 LSP 0000.0C0A.2AA9.00-00, seq 11, ht 1199,
ISIS-Upd: from SNPA 0005.5e6b.50a0 (Ethernet0)
ISIS-Upd: LSP newer than database copy
ISIS-Upd: Important fields changed
ISIS-Upd: Full SPF required
ISIS-Upd: Rec L1 LSP 0000.0C0A.2AA9.00-00, seq 12, ht 1199,
ISIS-Upd: from SNPA 0005.5e6b.50a0 (Ethernet0)
Frankfurt#
```

示例 10-76 **debug isis snp-packets** 显示了路由器接收和发送 CSNP 与 PSNP 的有关详细信息

```
Frankfurt#debug isis snp-packets
IS-IS CSNP/PSNP packets debugging is on
Frankfurt#
ISIS-Snp: Rec L1 CSNP from 0000.0C0A.2AA9 (Ethernet0)
ISIS-Snp: CSNP range 0000.0000.0000.00-00 to FFFF.FFFF.FFFF.FF-FF
ISIS-Snp: Same entry 0000.0C04.DCC0.00-00, seq 9
ISIS-Snp: Same entry 0000.0C0A.2AA9.00-00, seq 1A
ISIS-Snp: Same entry 0000.0C0A.2AA9.01-00, seq 6
ISIS-Snp: Rec L1 CSNP from 0000.0C0A.2AA9 (Ethernet0)
ISIS-Snp: CSNP range 0000.0000.0000.00-00 to FFFF.FFFF.FFFF.FF-FF
ISIS-Snp: Same entry 0000.0C04.DCC0.00-00, seq 9
ISIS-Snp: Entry 0000.0C0A.2AA9.00-00, seq 1B is newer than ours (seq 1A), sending PSNP
ISIS-Snp: Same entry 0000.0C0A.2AA9.01-00, seq 6
ISIS-Snp: Build L1 PSNP entry for 0000.0C0A.2AA9.00-00, seq 1A
ISIS-Snp: Sending L1 PSNP on Ethernet0
ISIS-Snp: Rec L1 CSNP from 0000.0C0A.2AA9 (Ethernet0)
ISIS-Snp: CSNP range 0000.0000.0000.00-00 to FFFF.FFFF.FFFF.FF-FF
ISIS-Snp: Same entry 0000.0C04.DCC0.00-00, seq 9
ISIS-Snp: Same entry 0000.0C0A.2AA9.00-00, seq 1B
ISIS-Snp: Same entry 0000.0C0A.2AA9.01-00, seq 6
ISIS-Snp: Rec L1 CSNP from 0000.0C0A.2AA9 (Ethernet0)
ISIS-Snp: CSNP range 0000.0000.0000.00-00 to FFFF.FFFF. FFFF.FF-FF
ISIS-Snp: Same entry 0000.0C04.DCC0.00-00, seq 9
ISIS-Snp: Same entry 0000.0C0A.2AA9.00-00, seq 1B
ISIS-Snp: Same entry 0000.0C0A.2AA9.01-00, seq 6
Frankfurt#
```

10.3.3 案例研究：运行于 NBMA 网络上的集成 IS-IS

在图 10-56 中，显示了 4 台运行 IS-IS 协议的路由器，它们之间通过一个部分网状连接的帧中继网络相连。IP 地址、DLCI 和 NET 地址都标注在图上。所有路由器的 IS-IS 配置都已经确认是正确的，而且没有配置任何认证信息。

这个网络的故障是无法发现路由，参见示例 10-77 所示。邻居路由器的帧中继接口的 IP 地址可以 ping 通，但是邻居路由器上的其他接口地址都不能 ping 通，参见示例 10-78 所示。这些 ping 的结果表明帧中继 PVC 是正常运作的，IP 也是工作的，但是路由器却没有进行路由选择。

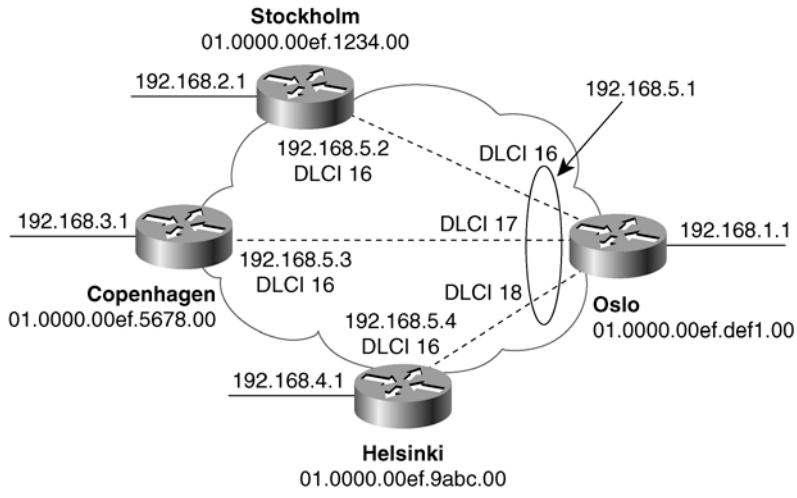


图 10-56 IS-IS 协议在帧中继网络上没有建立邻接关系

示例 10-77 图 10-56 中路由器 Oslo 的路由表没有包含任何 IS-IS 路由

```
Oslo#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
C    192.168.1.0 is directly connected, TokenRing0
C    192.168.5.0 is directly connected, Serial0
Oslo#
```

示例 10-78 与帧中继网络相连的其他接口可以 ping 通，但是路由器的可达地址却不能 ping 通

```
Oslo#ping 192.168.5.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.5.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/65/68 ms
Oslo#ping 192.168.2.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.2.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
Oslo#ping 192.168.5.3
Type escape sequence to 5, 100-byte ICMP Echos to 192.168.5.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/66/68 ms
Oslo#ping 192.168.3.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.3.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
Oslo#ping 192.168.5.4
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.5.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/65/68 ms
Oslo#ping 192.168.4.1
```

（待续）

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.4.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
Oslo#
```

接下来的一步就是检查 IS-IS 的邻居表了。路由器 Oslo 的邻居表显示已经收到 Hello 数据包（参见示例 10-79），而且这台路由器能够学习到邻居路由器的系统 ID。同时，邻居表也显示了邻居路由器的 IP 地址和区域地址是正确的。但是，所有邻居路由器的状态都是 Init，这表明还没有建立一个完全的邻接关系。查看一下链路状态数据库并确认不存在的邻接关系，路由器 Oslo 的数据库中的惟一 LSP 就是路由器本身的 LSP，参见示例 10-80 所示。

示例 10-79　路由器 Oslo 的 IS-IS 邻居表显示了可以收到 Hello 数据包，但是不能完全建立邻接关系

```
Oslo#show clns is-neighbors detail
System Id          Interface   State   Type Priority   Circuit Id         Format
0000.00EF.5678     Se0         Init    L1L2 0 /0       0000.0000.0000.00 Phase V
  Area Address(es): 01
  IP Address(es): 192.168.5.3
  Uptime: 1:11:20
0000.00EF.1234     Se0         Init    L1L2 0 /0       0500.0000.0000.00 Phase V
  Area Address(es): 01
  IP Address(es): 192.168.5.2
  Uptime: 1:11:15
0000.00EF.9ABC     Se0         Init    L1L2 0 /0       0700.0000.0000.00 Phase V
  Area Address(es): 01
  IP Address(es): 192.168.5.4
  Uptime: 1:11:20
Oslo#
```

示例 10-80　路由器 Oslo 的链路状态数据库没有包含任何来自于邻居的 LSP

```
Oslo#show isis database
IS-IS Level-1 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
0000.00EF.DEF1.00-00* 0x0000001F   0x8460        947           0/0/0
0000.00EF.DEF1.02-00* 0x00000010   0x695E        896           0/0/0
0000.00EF.DEF1.04-00* 0x00000002   0x2F2E        887           0/0/0
0000.00EF.DEF1.05-00* 0x00000008   0x1C3A        847           0/0/0
IS-IS Level-2 Link State Database
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime  ATT/P/OL
0000.00EF.DEF1.00-00* 0x00000013   0x81BE        829           0/0/0
Oslo#
```

事实上，路由器正在接收 Hello 数据包，但邻接关系尚未建立成功，因此问题就指向了 Hello 数据包本身。如果 Hello 数据包中的参数不正确，这个 PDU 就会被丢弃。因此可以使用 **debug isis adj-packets** 命令来观察 Hello 数据包。在这里需要特别注意的是，调试输出显示了“封装失败（encapsulation failed）”的消息，参见示例 10-81 所示。这些消息显示路由器显然不能解释收到的 Hello 数据包，因而丢弃了这些 Hello 数据包。

示例 10-81　打开调试命令 **debug isis adj-packets**，输出结果显示了由于封装失败而正在丢弃 Hello 数据包

```
Oslo#debug isis adj-packets
IS-IS Adjacency related packets debugging is on
Oslo#
```

（待续）

```
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Encapsulation failed for level 2 IIH on Serial0
ISIS-Adj: Rec serial IIH from DLCI 17 on Serial0, cir type 3, cir id 00
ISIS-Adj: rcvd state 2, old state 1, new state 1
ISIS-Adj: Action = 1, new_type = 3
ISIS-Adj: Sending L2 IIH on TokenRing0
ISIS-Adj: Encapsulation failed for level 1 IIH on Serial0
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Rec serial IIH from DLCI 18 on Serial0, cir type 3, cir id 07
ISIS-Adj: rcvd state 2, old state 1, new state 1
ISIS-Adj: Action = 1, new_type = 3
ISIS-Adj: Encapsulation failed for level 2 IIH on Serial0
ISIS-Adj: Sending L2 IIH on TokenRing0
ISIS-Adj: Encapsulation failed for level 1 IIH on Serial0
ISIS-Adj: Rec serial IIH from DLCI 16 on Serial0, cir type 3, cir id 05
ISIS-Adj: rcvd state 2, old state 1, new state 1
ISIS-Adj: Action = 1, new_type = 3
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Encapsulation failed for level 2 IIH on Serial0no debu
ISIS-Adj: Sending L2 IIH on TokenRing0
ISIS-Adj: Encapsulation failed for level 1 IIH on Serial0g a
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Rec serial IIH from DLCI 17 on Serial0, cir type 3, cir id 00
ISIS-Adj: rcvd state 2, old state 1, new state 1
ISIS-Adj: Action = 1, new_type = 3
ISIS-Adj: Encapsulation failed for level 2 IIH on Serial0ll
```

任何时候看到封装失败的消息都应该怀疑数据链路的问题和所连接的接口问题。使用命令 **show interface serial** 检查接口，没有发现存在严重的误码率，因此不像是帧中继 PVC 链路破坏了 Hello 数据包。下一步需要检查接口的配置。图 10-56 中的 4 台路由器的接口配置参见示例 10-82～示例 10-85。

示例 10-82　路由器 Oslo 的接口配置

```
interface Serial0
 ip address 192.168.5.1 255.255.255.0
 ip router isis
 encapsulation frame-relay
 frame-relay interface-dlci 16
 frame-relay interface-dlci 17
 frame-relay interface-dlci 18
```

示例 10-83　路由器 Stockholm 的接口配置

```
interface Serial0
 no ip address
 encapsulation frame-relay
!
interface Serial0.16 point-to-point
 ip address 192.168.5.2 255.255.255.0
 ip router isis
 frame-relay interface-dlci 16
```

示例 10-84　路由器 Copenhagen 的接口配置

```
interface Serial0
 no ip address
 encapsulation frame-relay
!
interface Serial0.16 point-to-point
 ip address 192.168.5.3 255.255.255.0
 ip router isis
 frame-relay interface-dlci 16
```

示例 10-85 路由器 Helsinki 的接口配置

```
interface Serial0
 no ip address
 encapsulation frame-relay
!
interface Serial0.16 point-to-point
 ip address 192.168.5.4 255.255.255.0
 ip router isis
 frame-relay interface-dlci 16
```

通过比较这些配置发现了一个问题，虽然这个问题不一定是很明显。路由器 Stockholm、Copenhagen 和 Helsinki 都配置成了点到点的子接口，但路由器 Oslo 没有使用子接口。在缺省情况下，Cisco 路由器的串行接口在封装成帧中继时是一个多点接口。因此，路由器 Stockholm、Copenhagen 和 Helsinki 发送点到点 IS-IS Hello 数据包，而路由器 Oslo 发送 L1 和 L2 类型的 IS-IS LAN Hello 数据包。

IS-IS 协议的配置中没有一个类似于 OSPF 协议中的 **ip ospf network** 命令的配置选项，因此，路由器 Oslo 必须重新配置成点到点子接口，并且必须更改 IP 地址，以便使每一个 PVC 链路都在不同的子网中，参见示例 10-86 所示。

示例 10-86 在路由器 Oslo 配置成点到点子接口后，并且进行重新编址以使每一个 PVC 成为一个单独的子网后，IS-IS 路由选择就起作用了

```
Oslo#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
C    192.168.1.0 is directly connected, TokenRing0
i L1 192.168.2.0 [115/20] via 192.168.5.5, Serial0.16
i L1 192.168.3.0 [115/20] via 192.168.5.9, Serial0.17
i L1 192.168.4.0 [115/20] via 192.168.5.13, Serial0.18
     192.168.5.0 is variably subnetted, 4 subnets, 2 masks
C       192.168.5.12 255.255.255.252 is directly connected, Serial0.18
C       192.168.5.8 255.255.255.252 is directly connected, Serial0.17
C       192.168.5.4 255.255.255.252 is directly connected, Serial0.16
C       192.168.5.0 255.255.255.0 is directly connected, Serial0
Oslo#
```

10.4 展　　望

到现在为止，所有的 IP IGP 协议都已经论述完了，下一步将开始研究一些有效的工具来帮助控制我们的网络。第三部分“路由控制和互操作性”涵盖了路由再分配、缺省路由、按需路由选择、路由过滤和路由映射。

10.5 总结表：第 10 章命令总结

命令	描述
address-family ipv6	指定跟在其后的命令应用于 IPv6 地址簇

续表

命令	描述
area-password *password*	配置 IS-IS 区域（第 1 层）认证
authentication key-chain *name-of-chain* [**level-1 | level-2**]	指定预先配置的钥匙链，以便用于对 L1 或 L2 的路由器进行认证。L1 表示区域范围的认证，而 L2 表示域范围的认证
authentication mode {md5|text} [**level-1 | level-2**]	指定用于一个区域内或域内路由器之间的认证模式
authentication send-only [**level-1 | level-2**]	在配置认证而不需要使现有的数据库无效时使用，这条命令发送认证信息，但不要求接收匹配认证信息
debug isis adj-packets	显示 IS-IS Hello PDU 的行为
debug isis spf-events	显示触发一个 IS-IS SPF 计算的事件的详细信息
debug isis snp-packets	显示路由器接收和发送 SNP 的有关信息
debug isis spf-statistics	显示一个有关 IS-IS SPF 计算的统计信息
debug isis spf-triggers	显示触发 IS-IS SPF 计算的事件
debug isis update-packets	显示路由器接收和发送 LSP、CSNP 和 PSNP 的有关信息
default-information originate [**route-map** *map-name*]	生成一条进入 IS-IS 域的缺省 IP 路由
domain-password *password*	配置 IS-IS 域（L2）认证
ignore-lsp-errors	配置一台 IS-IS 路由器忽略错误的 LSP 而不是触发一个 LSP 的清除
ip router isis [*tag*]	在一个接口上启动 IPv4 的 IS-IS 路由选择
ipv6 router isis [*tag*]	在一个接口上启动 IPv6 的 IS-IS 路由选择
isis authentication key-chain *name-of-chain* [**level-1 | level-2**]	在某个接口上指定预先配置的钥匙链，以便用于对 L1 或 L2 的路由器进行认证。L1 表示区域范围的认证，而 L2 表示域范围的认证
isis authentication mode {**md5|text**} [**level-1 | level-2**]	在某个接口上指定它的认证模式
isis authentication send-only [**level-1 | level-2**]	在配置认证而不需要中断现有的邻接时使用，这条命令发送认证信息，但不要求接收匹配的认证信息
isis csnp-interval *seconds* {**level-1|level-2**}	指定一台 IS-IS 确定路由器发送 CSNP 的时间间隔，以秒数计
isis hello-interval *seconds* {**level-1|level-2**}	指定 IS-IS Hello PDU 重传的时间间隔，以秒数计
isis hello-multiplier *multiplier* {**level-1|level-2**}	指定一台邻居路由器在宣告它与始发路由器邻接关系失效之前，必须错过的 IS-IS Hello PDU 的数目
isis lsp-interval *milliseconds*	在一个接口上改变 LSP 的传输之间的缺省时延
isis mesh-group [*number*|**blocked**]	给一个接口分配一个数值的 mesh 组，或在接口上完全阻塞 LSP 扩散
isis metric *default-metric* {**level-1|level-2**}	指定一个接口的 IS-IS 缺省度量
isis password *password* {**level-1|level-2**}	在两台 IS-IS 邻居路由器之间配置认证
isis priority *value* {**level-1|level-2}**	指定一个接口用来选取指定路由器的优先级
isis retransmit-interval *seconds*	指定一台路由器在一个点到点链路上发送一条 LSP 后需要等待确认的时间，如果超过这个时间还没收到确认，路由器将会重传这条 LSP
is-type {**level-1|level-1-2|level-2-only**}	配置一台路由器作为一台 L1、L1/L2 或者 L2 类型的 IS-IS 路由器
log-adjacency-changes	发送所有的邻接变化事件到日志记录中
lsp-refresh-interval *seconds*	改变本地始发 LSP 的刷新之间的缺省周期
lsp-gen-interval [**level-1|level-2**] *lsp-max-wait* [**lsp-initial-wait lsp-second-wait**]	改变调整 LSP 生成的指数补偿算法的缺省值
max-lsp-lifetime *seconds*	改变本地始发 LSP 的剩余生存时间的缺省值
metric-style [narrow|wide|transition]{**level-1| level-2| level-1-2**}	指定所使用的度量风格的类型
multi-topology {**transition**}	启动 IPv6 的多拓扑
net *network-entity-title*	配置一台 IS-IS 路由器的 NET 地址

续表

命令	描述
prc-interval *prc-max-wait* [**prc-initial-wait prc-second-wait**]	改变用于部分路由计算的指数补偿算法的缺省值
redistribute isis {**ip**} **level-2 into level-2 distribute-list** [**access-list number**|**prefix-list name**]	启动从 IS-IS level-2 进入 level-1 区域的路由泄漏
router isis [*tag*]	启动一个 IS-IS 路由选择进程
set-overload-bit[**on-startup**{*seconds*}|**wait-for-bgp**] [**suppress** {[**interlevel**] | [**external**]}]	在路由器的 LSP 中手工设置过载位为 1
show clns is-neighbor [*type number*][**detail**]	显示一个 IS-IS 邻居表
show clns protocol	显示路由器上如何配置 clns 和集成 IS-IS 的信息
show isis database [**level-1**][**level-2**][**l1**][**l2**][**detail**] [*lspid*]	显示一个 IS-IS 链路状态数据库
show isis hostname	显示网络中路由器的系统 ID 和主机名。这个信息是通过类型 137 的 TLV 获取的
show isis ipv6 topology	显示 IPv6 的拓扑
show isis spf-log	显示路由器怎样以及为什么要运行一个完全的 SPF 计算
show isis topology	显示 IPv4/CLNS 的拓扑
spf-interval [**level-1**|**level-2**] *spf-max-wait* [**spf-initial-wait spf-second-wait**]	更改用于 SPF 计算的指数补偿算法的缺省参数
summary-address *address-mask* {**level-1**|**level-1-2** |**level-2**}	配置 IP 的地址汇总
summary-prefix *ipv6_prefix/prefix_length* {**level-1**| **level-1-2** | **level-2**}	配置 IPv6 的地址汇总
which-route {*nsap-address*|*clns-name*}	查找并显示一个指定的 CLNS 目的地址在路由表中对应的 IP 地址和区域地址的详细信息

10.6 复 习 题

1．什么是中间系统？

2．什么是网络协议数据单元？

3．L1、L2 和 L1/L2 类型的路由器有什么不同？

4．什么是 Cisco 路由器的缺省层？

5．说明 IS-IS 区域和 OSPF 区域的不同之处。

6．在具有相同区域 ID 的两台 L1/L2 路由器之间可以形成什么类型的邻接？如果 L1/L2 路由器的区域 ID 不同呢？

7．在具有相同区域 ID 的两台 L2-only 路由器之间可以形成什么类型的邻接？如果 L2-only 路由器的区域 ID 不同呢？

8．什么是网络实体标题（NET）？

9．在 NET 中必须将 NSAP 选择符设置成什么值？

10．系统 ID 的用途是什么？

11．一台路由器是怎样确定它所在的区域的？

12．IS-IS 协议在一个广播型子网上选取备份指定路由器吗？

13．伪节点 ID 的用途是什么？

14．一个 IS-IS LSP 缺省的最大老化时间（MaxAge）是多少？配置的 MaxAge 的最大值

是多少？

15．OSPF 协议老化它的 LSA 和 IS-IS 协议老化它的 LSP 的方式有什么基本的不同？

16．一台 IS-IS 路由器多长时间重刷新一次它的 LSP？

17．什么是完全序列号数据包（CSNP）？它有什么用途？

18．什么是部分序列号数据包（PSNP）？它有什么用途？

19．过载位（OL）的用途是什么？

20．区域关联位（ATT）的用途是什么？

21．Up/Down 位的用途是什么？

22．ISO 中规定的 IS-IS 度量有哪些？在 Cisco IOS 中支持哪些？

23．IS-IS 的两种缺省度量类型是什么？它们的最大值是多少？

24．一条 IS-IS 路由的最大度量值是多少？

25．L1 类型的 IS-IS 度量和 L2 类型的 IS-IS 度量有什么不同？

26．内部 IS-IS 度量和外部 IS-IS 度量有什么不同？

27．在单一拓扑模式下，同时运行 IPv4 和 IPv6 的两台路由器之间的单条链路上可以创建几个邻接？如果是多拓扑模式呢？

28．在单独一台路由器上可以有几个活动的 L1 区域？可以有几个活动 L2 区域？

29．除了 Inactive 外，另外两个 mesh 组模式是什么？它们各有何利弊？

10.7 配置练习

1．表 10-8 中显示了 11 台路由器的接口、接口地址和子网掩码。这个表还指定了属于同一个区域的路由器。使用下面的指导策略，写出每台路由器的集成 IS-IS 的配置。

- 为每台路由器配置自己的系统 ID；
- 使用尽可能短的 NET 地址；
- 适当地把这些路由器配置成 L1、L2 或者 L1/L2 类型的路由器。

提示：首先画一张路由器和子网的图形。

表 10-8 配置练习 1～5 的路由器信息

	区域	接口	地址/掩码
A	0	E0	192.168.1.17/28
		E1	192.168.1.50/28
B	0	E0	192.168.1.33/28
		E1	192.168.1.51/28
C	0	E0	192.168.1.49/28
		S0	192.168.1.133/30
D	2	S0	192.168.1.134/30
		S1	192.168.1.137/30
E	2	S0	192.168.1.142/30
		S1	192.168.1.145/30
		S2	192.168.1.138/30

续表

	区域	接口	地址/掩码
F	2	S0	192.168.1.141/30
		S1	192.168.1.158/30
G	1	E0	192.168.1.111/27
		S0	192.168.1.157/30
H	1	E0	192.168.1.73/27
		E1	192.168.1.97/27
I	3	E0	192.168.1.225/29
		E1	192.168.1.221/29
		S0	192.168.1.249/30
		S1	192.168.1.146/30
J	3	E0	192.168.1.201/29
		E1	192.168.1.217/29
K	3	E0	192.168.1.209/29
		S0	192.168.1.250/30

2．在每一台路由器上配置多拓扑模式的 IPv6。使用地址 2001:db8:0:x::/64，这里的 x 是每一条链路上对应于 IPv4 地址的子网（只有第 4 个八位组）的十六进制值。

3．在表 10-8 的区域 2 内的所有路由器上配置认证。路由器 D 和路由器 E 之间使用口令“Eiffel”，路由器 E 和路由器 F 之间使用口令“Tower”。使用的认证是 md5 类型。

4．在表 10-8 中区域 1 上配置 L1 类型的认证，使用口令“Scotland”。使用没有钥匙链的明文口令。

5．在表 10-8 的路由器上配置 L2 类型的认证，使用口令“Vienna”。使用带有钥匙链的明文口令。

6．配置表 10-8 中区域 0、区域 1 和区域 3 的 L1/L2 路由器，同时汇总它们 IPv4 和 IPv6 的 L1 地址。

10.8 故障诊断练习

1．在示例 10-87 和示例 10-88 中，显示了路由器 A 和路由器 B 的 IS-IS 邻居表，这两台路由器是通过一个串行接口相连的。出现了什么错误？

2．在示例 10-89 中，显示了一台路由器的调试信息，这台路由器没有和它 TO0 接口上的邻居路由器成功建立邻接关系。出现了什么错误？

示例 10-87 故障诊断练习 1，路由器 A 的 IS-IS 邻居表

```
Router_A#show clns is-neighbors detail
System Id      Interface  State  Type Priority  Circuit Id        Format
0000.00EF.DCBA To0        Up     L1L2 64/64     0000.00EF.DCBA.04 Phase V
  Area Address(es): 01
  IP Address(es): 192.168.11.2
  Uptime: 0:09:25
0000.00EF.5678 Se0.17     Up     IS   0 /0      00                Phase V
```

（待续）

```
  Area Address(es): 01
  IP Address(es): 192.168.5.9
  Uptime: 1:28:22
0000.00EF.9ABC Se0.18     Up     L1L2 0 /0      07                Phase V
  Area Address(es): 01
  IP Address(es): 192.168.5.13
  Uptime: 1:29:45
0000.00EF.1234 Se0.16     Up     L1L2 0 /0       06                Phase V
  Area Address(es): 01
  IP Address(es): 192.168.5.5
  Uptime: 1:29:45
Router_A#
```

示例 10-88　故障诊断练习 1，路由器 B 的 IS-IS 邻居表

```
Router_B#show clns is-neighbors detail
System Id      Interface  State  Type Priority  Circuit Id         Format
0000.00EF.DEF1 Se0.17     Up     IS 0 /0        00                 Phase V
  Area Address(es): 01
  IP Address(es): 10.1.1.1
  Uptime: 0:11:06
Router_B#
```

示例 10-89　故障诊断练习 2 的调试输出信息

```
Router_B#debug isis adj-packets
IS-IS Adjacency related packets debugging is on
Router_B#
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Sending L1 IIH on TokenRing1
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Sending L1 IIH on TokenRing1
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Sending L1 IIH on TokenRing1
ISIS-Adj: Rec L2 IIH from 0000.3090.c7df (TokenRing0), cir type 2, cir id 0000.0
0EF.DCBA.04
ISIS-Adj: is-type mismatch
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Sending L1 IIH on TokenRing1
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Sending L1 IIH on TokenRing1
ISIS-Adj: Sending L1 IIH on TokenRing1
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Rec L2 IIH from 0000.3090.c7df (TokenRing0), cir type 2, cir id 0000.0
0EF.DCBA.04
ISIS-Adj: is-type mismatch
ISIS-Adj: Sending L1 IIH on TokenRing1
ISIS-Adj: Sending L1 IIH on TokenRing0
ISIS-Adj: Sending L1 IIH on TokenRing1
ISIS-Adj: Sending L1 IIH on TokenRing0
 IIH on TokenRing0L1 IIH on TokenRing1
```



第三部分

路由控制和互操作性

第 11 章　路由重新分配

第 12 章　缺省路由和按需路由选择

第 13 章　路由过滤

第 14 章　路由映射

本章包括以下主题：

- 重新分配的原则；
- 配置重新分配。



第 11 章

路由重新分配

当路由器使用路由选择协议通告从其他方式学习到的路由时，路由器将执行重新分配。这里所谓的其他方式可能是另外一个路由选择协议、静态路由或直连目标网络。例如，路由器可能同时运行 OSPF 进程和 RIP 进程。如果设置 OSPF 进程通告来自 RIP 进程的路由，这就叫做重新分配 RIP。

在整个IP网络中，如果从配置管理和故障管理的角度看，我们通常更愿意运行一种路由选择协议，而不是多种路由选择协议。然而，现代的网络又常常迫使我们接受多协议 IP 路由选择域这一现实。当部门、分公司乃至整个公司合并时，必须统一它们原来的自主网络。

在大部分案例中，将要被合并的网络在实现和发展上都不相同，它们满足不同的需求，是不同设计理念的产物。这种差异性使得向单一路由选择协议的迁移成为一项复杂的任务。在某些案例中，公司的策略可能会强制使用多种路由选择协议。而在少数场合还会出现因网络管理员不能很好地协同工作而采用多种路由选择协议。

多厂商环境是需要重新分配路由的另一个因素。例如，一个运行 Cisco EIGRP 的网络可能会与使用另一个厂商路由器的网络合并，而这种路由器仅支持 RIP 和 OSPF。如果不进行重新分配，那么 Cisco 路由器需要使用一种公开的协议重新配置或者用 Cisco 路由器替代非 Cisco 路由器。

当多种路由选择协议“被拼凑”在一起时，使用重新分配是很有必要的，而且重新分配也是一个严谨网络设计的一部分。图 11-1 给出了一个例子，这里把两个 OSPF 进程域连接在一起，但是 OSPF 进程之间并不直接通信，取而代之的是在每台路由器上配置静态路由，指向其他OSPF域内的被选网络。

例如，路由器 Spalding 包含指向网络 192.168.11.0 和 192.168.12.0 的路由， Spalding 把这些静态路由重新分配到 OSPF 中，OSPF 又向 OSPF 10 内的其他路由器通告这些路由。

这样做的结果是向 OSPF 10 隐瞒了 OSPF 20 中的其他网络。重新分配使得 OSPF 的动态特性和静态路由的精确控制性融合在一起。

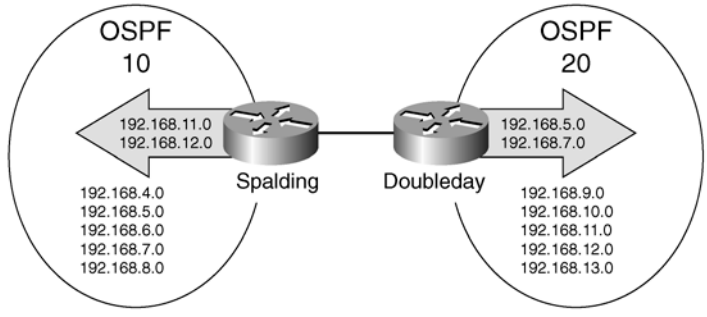


图 11-1 在这个网络中，在每台路由器上都配置了静态路由，并且向 OSPF 重新分配这些路由。结果是，可以做到精确地控制在两个 OSPF 域之间的前缀通告

如果不是非要使用动态路由选择协议的话，在拨号环境中向动态路由选择协议重新分配静态路由也是非常有用的。动态协议周期性的管理流量会导致拨号线路始终保持接通状态。通过阻止路由更新和 Hello 信息通过线路，并在两边配置静态路由，管理员可以确保线路只在有用户流量通过时才接通。向动态路由选择协议重新分配静态路由，可以使拨号线路两边的所有路由器都知道链路对方的所有网络。

注意到，除了少数特例外，[1]在相同路由器上存在不止一种路由选择协议并不意味着重新分配自动发生。重新分配必须被明确地配置。在没有使用重新分配的单一路由器上配置多种路由选择协议的方法叫做午夜航船（Ships In the Night，SIN）路由选择。路由器将会在每个进程域内向它的对等路由器传递路由，但是进程域之间却一无所知——这就好比黑暗中航行的船只一样。虽然 SIN 路由选择法通常指在相同路由器上多种路由选择协议为多种可路由协议进行路由选择（例如 OSPF 为 IP 和 NLSP 为 IPX 进行路由选择），但是它也可以指在单一路由器上两个 IP 协议为单独的 IP 域进行路由选择。

11.1 重新分配的原则

IP 路由选择协议的能力相差非常大。对重新分配影响最大的协议特性是度量和管理距离的差异性，以及每种协议的有类别和无类别能力。在重新分配时如果忽略了对这些差异性的考虑将导致以下后果，最好情况会出现某些或全部路由交换失败，最坏情况将造成路由环路和黑洞。

11.1.1 度量

图 11-1 中的路由器正在向 OSPF 重新分配静态路由，然后它们会向其他 OSPF 路由器通告这些路由。虽然静态路由没有相关联的度量，但每条 OSPF 路由必须有一个代价值。有关度量冲突的另一个例子是向 EIGRP 重新分配 RIP 路由，RIP 的度量是跳数，而 EIGRP 使用带宽和时延。在这两种情况中，接收被重新分配路由的协议必须能够将自己的度量与这些路

[1] 在 IP 中，自治系统号相同的 IGRP 和 EIGRP 进程可以自动重新分配。

由关联起来。

所以执行重新分配的路由器必须为被重新分配的路由指派度量。图 11-2 给出了一个例子，这里 EIGRP 被重新分配到 OSPF，同时 OSPF 也被重新分配到 EIGRP。OSPF 不能理解 EIGRP 的复合度量， EIGRP 也不能理解 OSPF 的代价。因此在重新分配进程中，路由器必须在向 OSPF 传递 EIGRP 路由之前为每一条 EIGRP 路由分配代价度量。同样的，路由器在向 EIGRP 传递 OSPF 路由之前也必须为每一条 OSPF 路由分配带宽、时延、可靠性、负载和 MTU 度量值。如果分配了不正确的度量，重新分配将会失败。

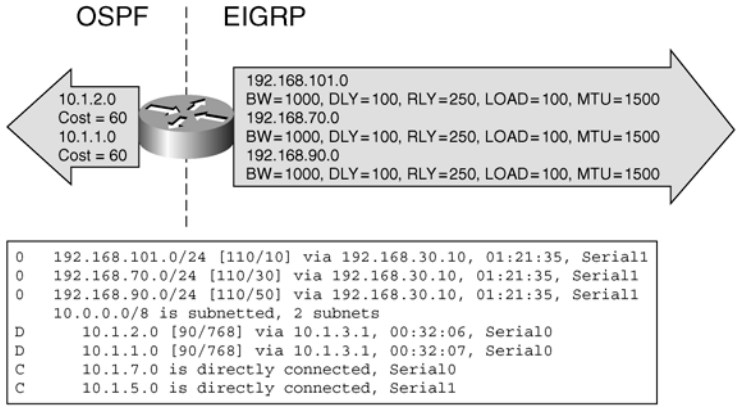


图 11-2 当重新分配路由时，必须为路由分配一个接收协议可以理解的度量值

本章后面的案例分析将会讨论为了达到路由重新分配的目标，应该如何配置路由器用来分配度量。

11.1.2 管理距离

度量的差异性产生了另一个问题：如果路由器正在运行多个路由选择协议，并从每个协议都学习到一条到达相同目标网络的路由，那么应该选择哪一条路由呢？每一个路由选择协议均使用自己的度量方案定义最优路径。比较不同度量的路由，例如代价和跳数，就像比较苹果和橙子一样。

这个问题的答案是管理距离。正像为路由分配度量以便可以确定首选路由一样，我们可以向路由源分配管理距离值以便确定首选路由源。管理距离被看作是一个可信度测度，管理距离越小，协议的可信度越高。

例如，假设运行 RIP 和 EIGRP 的路由器从邻居 RIP 路由器那里学习到一条指向网络 192.168.5.0 的路由，从邻居 EIGRP 路由器那里学习到一条指向相同网络的路由。由于 EIGRP 的复合度量，使得该协议更有可能确定最佳路由。因此，EIGRP 比 RIP 更可信。

表 11-1 列出了缺省的 Cisco 管理距离。EIGRP 的管理距离为 90，而 RIP 的管理距离为 120。因此，可以认为 EIGRP 比 RIP 更值得信赖。

表 11-1 Cisco 缺省管理距离

路由源	管理距离
直连接口 *	0
静态路由	1

续表

路由源	管理距离
EIGRP 汇总路由	5
外部 BGP	20
EIGRP	90
IGRP	100
OSPF	110
IS-IS	115
RIP	120
EGP	140
外部 EIGRP	170
内部 BGP	200
未知	255

*回忆一下第 3 章，当静态路由使用接口替代下一跳地址时，目的网络即被认为是直连网络。

虽然管理距离帮助解决了不同度量带来的混乱，但是它又为重新分配带来了问题。例如，在图 11-3 中，Gehrig 和 Ruth 都在向 IGRP 重新分配 RIP 路由。Gehrig 通过 RIP 学习到网络 192.168.1.0，并且将其通告给 IGRP 域。结果是，Ruth 不仅通过 RIP 从 Combs 处学习到网络 192.168.1.0，而且还通过 IGRP 从 Meusel 那里也学习到该网络。

示例 11-1 给出了 Ruth 的路由表。注意，指向网络 192.168.1.0 的路由是一条 IGRP 路由。Ruth 之所以选择 IGRP 路由是因为 IGRP 比 RIP 具有更小的管理距离。Ruth 将经过 Meusel 沿着这条“风景优美的路线”发送所有数据包，代替直接向 Combs 发送数据包。

水平分隔阻止了在图 11-3 的网络中路由环路的发生。Gehrig 和 Ruth 最初都向 IGRP 域通告网络 192.168.1.0，并且最终 4 台 IGRP 路由器都收敛到一条到达该网络的路径。然而，这种收敛是不可预知的。重新启动 Lazzeri 和 Meusel 可以看到这一情形。在重新启动后，Ruth 的路由表显示到达网络 192.168.1.0 的下一跳路由器是 Combs（参见示例 11-2）。

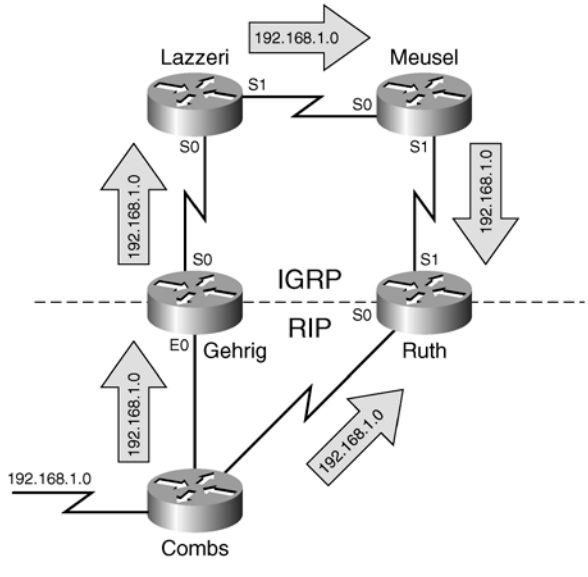


图 11-3 通过 RIP 和 IGRP 向 Ruth 通告网络 192.168.1.0

示例 11-1 虽然从 Ruth 到达网络 192.168.1.0 的最佳路径是从 S0 出发经过 Combs 去往目标网络，但是 Ruth 却选择了从 S1 出发经过 Meusel 的路径

```
Ruth#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
I    192.168.1.0/24 [100/16100] via 192.168.5.1, 00:00:00, Serial1
I    192.168.2.0/24 [100/12576] via 192.168.5.1, 00:00:00, Serial1
I    192.168.3.0/24 [100/12476] via 192.168.5.1, 00:00:01, Serial1
I    192.168.4.0/24 [100/10476] via 192.168.5.1, 00:00:01, Serial1
C    192.168.5.0/24 is directly connected, Serial1
C    192.168.6.0/24 is directly connected, Serial0
Ruth#
```

示例 11-2 图 11-3 中的网络收敛不可预知。在路由器重新启动后，Ruth 现在选择经过 Combs 到达网络 192.168.1.0

```
Ruth#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
R    192.168.1.0/24 [120/1] via 192.168.6.2, 00:00:23, Serial0
I    192.168.2.0/24 [100/12576] via 192.168.5.1, 00:00:22, Serial1
I    192.168.3.0/24 [100/12476] via 192.168.5.1, 00:00:22, Serial1
I    192.168.4.0/24 [100/10476] via 192.168.5.1, 00:00:22, Serial1
C    192.168.5.0/24 is directly connected, Serial1
C    192.168.6.0/24 is directly connected, Serial0
Ruth#
```

在重新启动后收敛不仅难以预知，而且很慢。示例 11-3 显示了重新启动完毕大约又经过 3min 后 Gehrig 的路由表。它使用 Lazzeri 作为到达网络 192.168.1.0 的下一跳路由器，但是 ping 该网络中的一个在线地址却发生失败。从 Lazzeri 的路由表（参见示例 11-4）可以看出问题所在：Lazzeri 使用 Gehrig 作为下一跳路由器，因此存在路由环路。

示例 11-3 重新启动后不久，Gehrig 为数据包选择经 Lazzeri 到达网络 192.168.1.0 的路径

```
Gehrig#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
I    192.168.1.0/24 [100/16100] via 192.168.3.2, 00:02:38, Serial0
C    192.168.2.0/24 is directly connected, Ethernet0
C    192.168.3.0/24 is directly connected, Serial0
I    192.168.4.0/24 [100/10476] via 192.168.3.2, 00:00:29, Serial0
I    192.168.5.0/24 [100/12476] via 192.168.3.2, 00:00:29, Serial0
I    192.168.6.0/24 [100/14476] via 192.168.3.2, 00:00:39, Serial0
Gehrig#ping 192.168.1.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
Gehrig#
```

示例 11-4 Lazzeri 路由为数据包选择经 Gerig 到达 192.168.1.0 的路径，因而产生路由环路。注意路由的年龄

```
Lazzeri#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
I    192.168.1.0/24 [100/12100] via 192.168.3.1, 00:04:21, Serial0
I    192.168.2.0/24 [100/8576] via 192.168.3.1, 00:00:33, Serial0
C    192.168.3.0/24 is directly connected, Serial0
C    192.168.4.0/24 is directly connected, Serial1
I    192.168.5.0/24 [100/10476] via 192.168.4.2, 00:00:53, Serial1
I    192.168.6.0/24 [100/12100] via 192.168.3.1, 00:02:32, Serial0
Lazzeri#
```

下面是导致环路的事件顺序：

1．当 Lazzeri 和 Meusel 重新启动时，Gehrig 和 Ruth 的路由条目显示经过 Combs 可以到达网络 192.168.1.0。

2．随着 Lazzeri 和 Meusel 启动完毕，Gehrig 和 Ruth 发送包括网络 192.168.1.0 的 IGRP 更新信息，仅仅由于运气，Ruth 比 Gehrig 更早一点发送了更新信息。

3．Meuse 接收到 Ruth 的更新信息，把 Ruth 作为下一跳路由器，并且向 Lazzeri 发送更新信息。

4．Lazzeri 接收到 Meusel 的更新信息，把 Meusel 作为下一跳路由器。

5．Lazzeri 和 Gehrig 在差不多相同的时刻互相发送了更新信息。Lazzeri 把 Gehrig 作为到达网络 192.168.1.0 的下一跳路由器，因为这条路由比 Meusel 的路由更接近目标。Gehrig 把 Lazzeri 作为到达网络 192.168.1.0 的下一跳路由器，因为 Lazzeri 的 IGRP 通告的管理距离比 Combs 的 RIP 通告的小。至此环路产生。

水平分隔和失效计时器最终将会解决这一问题。虽然 Lazzeri 正在向 Meusel 通告 192.168.1.0，但是 Meusel 仍会继续使用经过 Ruth 的更近的路径。由于 Ruth 是下一跳路由器，所以在 Meusel 的接口 S1 上，水平分隔对 192.168.1.0 有效。Meusel 还向 Lazzeri 通告 192.168.1.0，但是 Lazzeri 认为 Gehrig 更接近目的网络。

由于 Lazzeri 和 Gehrig 都将对方看作是去往 192.168.1.0 的下一跳路由器，所以它们相互不通告此路由。保存在它们路由表中的这条路由一直老化到失效计时器超时为止（参见示例 11-5）。

示例 11-5 在指向 192.168.1.0 路由的失效计时器超时后，这条路由将会被声明为不可达，并且抑制计时器被启动

```
Lazzeri#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
```

（待续）

```
I    192.168.1.0/24 is possibly down, routing via 192.168.3.1, Serial0
I    192.168.2.0/24 [100/8576] via 192.168.3.1, 00:00:57, Serial0
C    192.168.3.0/24 is directly connected, Serial0
C    192.168.4.0/24 is directly connected, Serial1
I    192.168.5.0/24 [100/10476] via 192.168.4.2, 00:01:25, Serial1
I    192.168.6.0/24 is possibly down, routing via 192.168.3.1, Serial0
Lazzeri#
```

在 Lazzeri 的失效计时器超时后，指向 192.168.1.0 的路由将被抑制。虽然 Meusel 正在通告指向该网络的路由，但是 Lazzeri 直到抑制计时器超时才能接收该通告。示例 11-6 显示出 Lazzeri 最终接收了这条来自 Meusel 的路由，示例 11-7 显示出 Gehrig 通过 Lazzeri 可以成功地到达 192.168.1.0。但是这两台路由器花费了 9min 时间才得以收敛，而且还使用了一条非最佳路由。

示例 11-6 在网络 192.168.1.0 的抑制计时器超时后，Lazzeri 接受 Meusel 通告的路由

```
Lazzeri#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
I    192.168.1.0/24 [100/14100] via 192.168.4.2, 00:00:27, Serial1
I    192.168.2.0/24 [100/8576] via 192.168.3.1, 00:00:02, Serial0
C    192.168.3.0/24 is directly connected, Serial0
C    192.168.4.0/24 is directly connected, Serial1
I    192.168.5.0/24 [100/10476] via 192.168.4.2, 00:00:28, Serial1
I   192.168.6.0/24 [100/12476] via 192.168.4.2, 00:00:28, Serial1
Lazzeri#
```

示例 11-7 Gehrig 现在可以通过 Lazzeri 到达网络 192.168.1.0

```
Gehrig#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
I    192.168.1.0/24 [100/16100] via 192.168.3.2, 00:00:32, Serial0
C    192.168.2.0/24 is directly connected, Ethernet0
C    192.168.3.0/24 is directly connected, Serial0
I    192.168.4.0/24 [100/10476] via 192.168.3.2, 00:00:33, Serial0
I    192.168.5.0/24 [100/12476] via 192.168.3.2, 00:00:33, Serial0
I    192.168.6.0/24 [100/14476] via 192.168.3.2, 00:00:33, Serial0
Gehrig#ping 192.168.1.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 52/72/108 ms
Gehrig#
```

管理距离导致的问题会比前面例子所述的非最佳路径、不可预知的行为以及慢收敛问题更严重。例如，图 11-4 给出的网络与图 11-3 中的网络本质上是相同，除了 IGRP 路由器之间的链路为帧中继永久虚电路。在缺省情况下帧中继接口上的 IP 水平分隔功能是关闭的，结果是在 Lazzeri 和 Gehrig 之间以及 Meuse 和 Ruth 之间会形成永久的路由选择环路，并且从 IGRP 域无法到达网络 192.168.1.0。

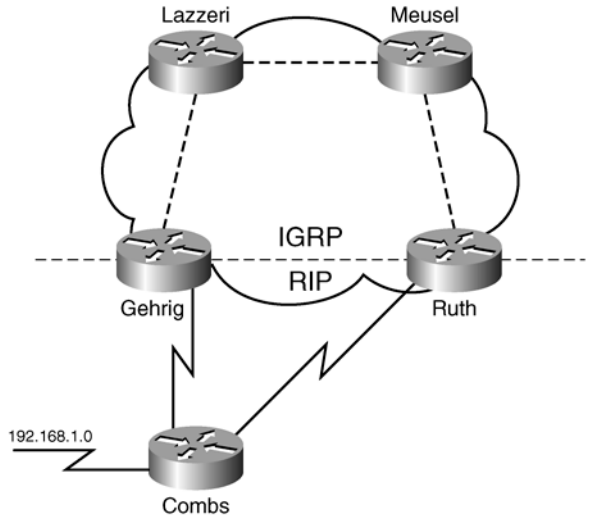


图 11-4 因为在帧中继接口上水平分隔功能在缺省状态下是关闭的，所以在这个网络上将会形成永久路由选择环路

在重新分配时有几种工具和策略可以避免路由选择环路，比如使用操作管理距离、路由过滤和路由映射。第 13 章讨论路由过滤，第 14 章讨论路由映射。这两章还示范了修改管理距离的技术。

11.1.3 从无类别协议向有类别协议重新分配

从无类别路由选择进程域向有类别域重新分配路由会产生哪些影响，这值得我们仔细地考虑。为了理解为什么这样做，首先有必要理解有类别路由选择协议怎样应对变长子网划分。回想第 5 章的内容，有类别路由选择协议不能通告携带子网掩码的路由。对于有类别路由器所接收到的每一条路由，无外乎是下面两种情况之一：

- 路由器将有一个或多个接口连接到主网上；
- 路由器将没有接口连接到主网上。

在第一种情况下，为了正确地确定数据包目标地址的子网，路由器必须使用它自己的主网掩码。在第二种情况下，公告信息中仅包含主网地址，因为路由器不知道使用哪一个子网掩码。

图 11-5 给出了路由器有 4 个接口分别连接到 192.168.100.0 的各子网上。该网络采用了变长子网划分——两个接口的子网掩码为 27 位，另两个为 30 位。如果路由器运行有类别协议，例如 IGRP，那么它将不能从 27 位掩码推出 30 位掩码的子网，并且也不能从 30 位掩码推出 27 位掩码的子网。那么，协议如何处理冲突的掩码呢？

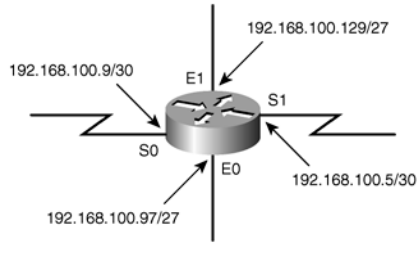


图 11-5 如果路由器运行有类别路由选择协议，它应该选择什么掩码呢

在示例 11-8 中，使用调试手段观察图 11-5 中路由器发出的 IGRP 通告。注意，子网 192.168.100.128/27 的通告从接口 E0 发出，网络使用了 27 位掩码，但是没有从该接口发送 192.168.100.4/30 和 192.168.100.8/30 通告。类似的，192.168.100.8/30 的通告从接口 S1 发出，掩码为 30 位，但是没有从该接口发送 192.168.100.96/27 和 192.168.100.128/27 通告。相同的情形同样适用于所有 4 个接口。在 192.168.100.0 的子网中，仅那些掩码与接口掩码相同的子网才会从此接口通告。最后结果是接口 E0 和 E1 的 IGRP 邻居路由器不知道掩码为 30 位的子网，接口 S0 和 S1 的 IGRP 邻居路由器也不知道掩码为 27 位的子网。

示例 11-8 有类别路由选择协议将不在掩码一致的接口之间通告路由

```
O'Neil#debug ip igrp transactions
IGRP protocol debugging is on
O'Neil#
IGRP: sending update to 255.255.255.255 via Ethernet0 (192.168.100.97)
      subnet 192.168.100.128, metric=1100
IGRP: sending update to 255.255.255.255 via Ethernet1 (192.168.100.129)
      subnet 192.168.100.96, metric=1100
IGRP: sending update to 255.255.255.255 via Serial0 (192.168.100.9)
      subnet 192.168.100.4, metric=8476
IGRP: sending update to 255.255.255.255 via Serial1 (192.168.100.5)
      subnet 192.168.100.8, metric=8476
O'Neil#
```

当从无类别路由选择协议向有类别路由选择协议重新分配路由时，仅在掩码相同的接口之间通告路由这一特性将得到应用。在图 11-6 中，OSPF 域的子网是经过变长子网划分得到的，Paige 将来自 OSPF 的路由信息向 IGRP 重新分配。

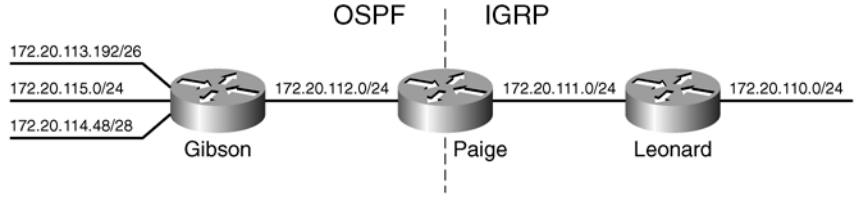


图 11-6 Paige 将来自 OSPF 的路由重新分配进 IGRP

如示例 11-9 所示，Paige 知道 OSPF 和 IGRP 域内的所有子网。因为 OSPF 是无类别协议，所以路由器知道连接到 Gibson 的每个子网的相应掩码。由于 Paige 的 IGRP 进程使用 24 位掩码，因此 172.20.113.192/26 和 172.20.114.48/28 不一致，所以不能被通告（参见示例 11-10）。注意，IGRP 对 172.20.112.0/24 和 172.20.115.0/24 进行通告，结果在 OSPF 域内 Leonard 仅知道掩码为 24 位的子网（参见示例 11-11）。

示例 11-9 Paige 知道图 11-6 中的所有 6 个子网，它们不是来自 OSPF、IGRP，就是直接连接

```
Paige#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
```

（待续）

```
     172.20.0.0/16 is variably subnetted, 6 subnets, 3 masks
O       172.20.113.192/26 [110/74] via 172.20.112.1, 00:01:35, Ethernet1
C       172.20.112.0/24 is directly connected, Ethernet1
O       172.20.115.0/24 [110/80] via 172.20.112.1, 00:01:35, Ethernet1
I       172.20.110.0/24 [100/1600] via 172.20.111.1, 00:00:33, Ethernet0
C       172.20.111.0/24 is directly connected, Ethernet0
O       172.20.114.48/28 [110/74] via 172.20.112.1, 00:01:35, Ethernet1
Paige#
```

示例 11-10　在来自 OSPF 的路由信息中，仅掩码为 24 位的路由被成功地重新分配进入 IGRP 域，该 IGRP 域也使用 24 位掩码

```
Paige#debug ip igrp transactions
IGRP protocol debugging is on
Paige#
IGRP: received update from 172.20.111.1 on Ethernet0
      subnet 172.20.110.0, metric 1600 (neighbor 501)
IGRP: sending update to 255.255.255.255 via Ethernet0 (172.20.111.2)
      subnet 172.20.112.0, metric=1100
      subnet 172.20.115.0, metric=1100
Paige#
```

示例 11-11　Leonard 仅知道掩码为 24 位且在 OSPF 域内的子网，从 Leonard 这里不能到达网络 172.20.113.192/26 和 172.20.114.48/28

```
Leonard#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
     172.20.0.0/24 is subnetted, 4 subnets
I       172.20.112.0 [100/1200] via 172.20.111.2, 00:00:48, Ethernet1
I       172.20.115.0 [100/1200] via 172.20.111.2, 00:00:48, Ethernet1
C       172.20.110.0 is directly connected, Ethernet0
C       172.20.111.0 is directly connected, Ethernet1
Leonard#
```

配置部分包括的案例研究将示范从无类别路由选择协议向有类别路由选择协议可靠地进行重新分配的方法。

11.2　配置重新分配

配置重新分配分为两步：

步骤 1：在路由选择协议中配置接收重新分配的路由，其中使用命令 **redistribute** 指定路由源点。

步骤 2：为重新分配的路由指定度量值。

图 11-7 中 Lajoie 的 EIGRP 配置见示例 11-12。

示例 11-12　Lajoie 的 EIGRP 配置显示出 OSPF 的路由被重新分配到 EIGRP 进程

```
router eigrp 1
 redistribute ospf 1 metric 10000 100 255 1 15003
 passive-interface Ethernet1
 network 172.20.0.0
```

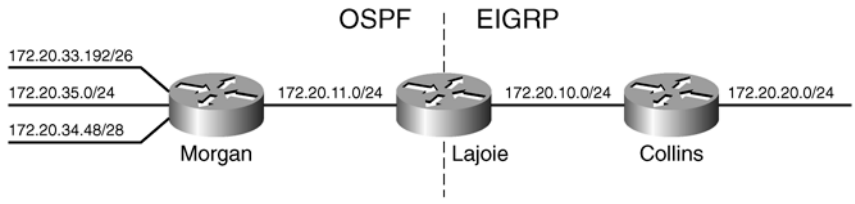


图 11-7 为简单网络配置路由重新分配

示例 11-12 中的配置把 OSPF 进程 1 发现的路由向 EIGRP 进程 1 重新分配。命令的 **Metric** 部分为路由分配了 EIGRP 度量值。按照顺序，命令中各数字分别表示：

- 带宽，单位是 kbit/s；
- 时延，单位是 10μs；
- 可靠性，为 255 的若干分之一；
- 负载，为 255 的若干分之一；
- MTU，单位为八位组字节。

Lajoie 的 OSPF 配置见示例 11-13。

示例 11-13 Lajoie 的 OSPF 配置显示出 EIGRP 的路由被重新分配到 OSPF

```
router ospf 1
 redistribute eigrp 1 metric 30 metric-type 1 subnets
 network 172.20.11.2 0.0.0.0 area 0
```

上面的配置将 EIGRP 进程 1 发现的路由重新分配到 OSPF 进程 1。命令的 Metric 部分为每一条被重新分配的路由分配了 OSPF 代价值 30。重新分配使得 Lajoie 成为 OSPF 域的 ASBR，并且被重新分配的路由是作为外部路由进行通告的。命令的 **metric-type** 部分指明了外部路由的类型为 E1。关键字 **subnets** 仅当向 OSPF 重新分配路由时使用，它指明子网的细节将被重新分配；没有它，仅重新分配主网地址。在案例研究中会更多地讨论关键字 **subnets**。

另一种分配度量的方法是使用 **default-metric** 命令。例如，前面的 OSPF 配置也可以改写为示例 11-14 的形式：

示例 11-14 在 Lajoie 的 OSPF 配置中使用 default-metric 命令为所有被重新分配到 OSPF 的路由指定 OSPF 度量

```
router ospf 1
 redistribute eigrp 1 metric-type 1 subnets
 default-metric 30
 network 172.20.11.2 0.0.0.0 area 0
```

该配置同前面配置所产生的结果完全相同。当重新分配来自多个源点的路由时，命令 **default-metric** 显得十分有用。例如在图 11-7 中，假设路由器 Lajoie 不仅运行 EIGRP 和 OSPF，而且还运行 RIP。相应的 OSPF 配置见示例 11-15。

示例 11-15 对于所有被重新分配到 OSPF 的 EIGRP 和 RIP 路由，在 Lajoie 的 OSPF 配置中使用 default-metric 命令为它们指定 OSPF 度量

```
router ospf 1
 redistribute eigrp 1 metric-type 1 subnets
 redistribute rip metric-type 1 subnets
default-metric 30
 network 172.20.11.2 0.0.0.0 area 0
```

在上面的配置中，对于所有来自 EIGRP 和 RIP 的路由，所分配的度量均为 OSPF 代价 30。

在这里，两种分配度量的方法也可以相互使用。例如，假设配置 Lajoie 向 EIGRP 重新分配 OSPF 和 RIP 路由，但要求对 RIP 路由进行通告时，所使用的度量要不同于 OSPF，相应的配置见示例 11-16。

示例 11-16 Lajoie 的 EIGRP 配置使用带度量参数的 redistribution 命令为从 RIP 重新分配来的地址指定度量值，而使用 default metric 命令对所有其他重新分配的地址分配 EIGRP 度量

```
router eigrp 1
 redistribute ospf 1
 redistribute rip metric 50000 500 255 1 1500
 default-metric 10000 100 255 1 1500
 passive-interface Ethernet1
 network 172.20.0.0
```

在上面的配置中，使用命令 **redistribute** 中关键字 **metric** 分配度量值优于 **defualt-metric** 命令分配度量值。来自 RIP 的路由被通告到 EIGRP，这些路由所使用的度量在配置行 **redistribute rip** 中指明。而来自 OSPF 的路由则使用 **defualt-metric** 命令指定的度量被通告。

如果关键字 **metric** 和命令 **defualt-metric** 都没有指定度量，那么被重新分配到 OSPF 的路由的度量缺省值为 20，而其他协议路由度量的缺省值为 0。IS-IS 可以理解 0 度量，但是 RIP 不能，因为它的跳数在 1 到 16 之间。0 度量与 EIGRP 的多度量格式也不兼容。因此这两种协议都必须为重新分配的路由分配合适的度量，否则重新分配将不能进行。下面的案例研究将会分析向各种 IP IGP 重新分配路由的配置方法。此外，案例研究中还更多地安排了关于从有类别向有类别、从无类别向无类别以及从无类别向有类别重新分配路由等常见问题的分析。

11.2.1 案例研究：重新分配 IGRP 和 RIP

在图 11-8 的网络中，Ford 运行 IGRP，Berra 运行 RIP。Mantle 的路由配置见示例 11-17。

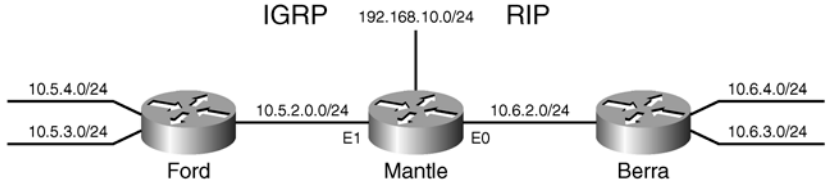


图 11-8 Ford 运行 IGRP，Berra 运行 RIP。Mantle 正在进行路由重新分配

示例 11-17 Mantel 同时运行 IGRP 和 RIP，并在两个协议之间重新分配路由，其中使用缺省度量和 redistribution 命令指定的度量

```
router rip
 redistribute igrp 1 metric 5
 passive-interface Ethernet1
 network 10.0.0.0
!
router igrp 1
 redistribute rip
 default-metric 1000 100 255 1 1500
 passive-interface Ethernet0
 network 10.0.0.0
```

这里同时使用两种分配度量的方法是出于示范目的，在大部分情况下，如果重新分配方案和本例一样简单的话，可以使用一种方法。

注意，Mantle 还被连接到一个末梢网络（192.168.10.0/24）。在本案例中，要求向 IGRP 域通告末梢网络，但是不能向 RIP 域通告。一种实现方法是仅在 IGRP 中添加适当的网络语句。然而，这样做会在末梢网络中造成不必要的 IGRP 广播。另一个实现方法是使用重新发配，具体配置见示例 11-18。

示例 11-18 路由器 Mantel 向 IGRP 重新分配直连末梢网络

```
router rip
 redistribute igrp 1 metric 5
 passive-interface Ethernet1
 network 10.0.0.0
!
router igrp 1
 redistribute connected
 redistribute rip
 default-metric 1000 100 255 1 1500
 passive-interface Ethernet0
 network 10.0.0.0
```

命令 **redistribute connected** 将会重新分配所有直连网络。如果要向 IGRP 域和 RIP 域通告网络 192.168.10.0/24，那么具体配置见示例 11-19。

示例 11-19 从路由器 Mantle 的配置可以看出，直连末梢网络被通告到 RIP 和 IGRP，其中向 RIP 通告的路由度量值是在重新分配过程中指定的，而向 IGRP 通告的路由度量值由 default metric 命令指定

```
router rip
 redistribute connected metric 5
 redistribute igrp 1 metric 5
 passive-interface Ethernet1
 network 10.0.0.0
!
router igrp 1
 redistribute connected
 redistribute rip
 default-metric 1000 100 255 1 1500
 passive-interface Ethernet0
 network 10.0.0.0
```

11.2.2 案例研究：重新分配 EIGRP 和 OSPF

图 11-9 的网络中，有一个 OSPF 域和两个 EIGRP 域。路由器 Hodges 运行 OSPF 进程 1，Podres 运行 EIGRP 进程 1，Snider 和 Campanella 运行 EIGRP 进程 2。

Robinson 的配置见示例 11-20。

示例 11-20 Robinson 向 EIGRP1、EIGRP2 和 OSPF 进程重新分配路由

```
router eigrp 1
 redistribute ospf 1 metric 1000 100 1 255 1500
 redistribute eigrp 2 passive-interface FastEthernet0/0
 network 192.168.3.0
!
```

（待续）

```
router eigrp 2
 redistribute ospf 1 metric 1000 100 1 255 1500
 redistribute eigrp 1  network 192.168.4.0
 network 172.16.0.0
!
router ospf 1
 redistribute eigrp 1
 redistribute eigrp 2 metric 100
 default-metric 50
 network 192.168.3.33 0.0.0.0 area 0
```

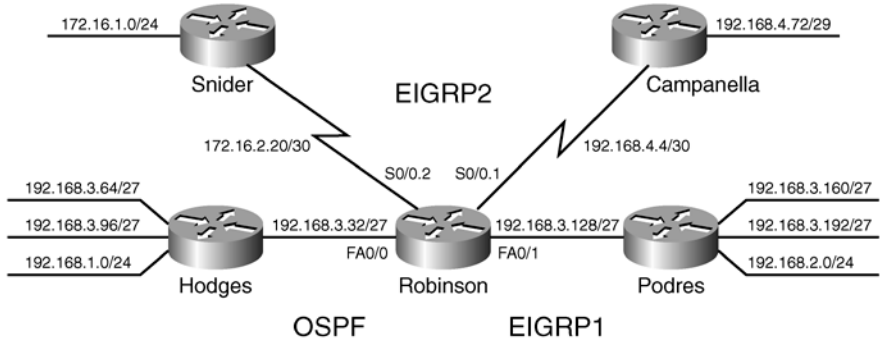


图 11-9 Hodges 运行 OSPF，Podre 运行 EIGRP1，Snider 和 Campanella 运行 EIGRP2

注意，尽管在 EIGRP 进程之间必须配置重新分配，但是不需要配置度量。因为这些进程使用相同的度量，所以能够穿过重新分配边界准确地跟踪度量。示例 11-21 显示了 Podres 的路由表，重新分配的路由被标记为 EIGRP 外部路由。

示例 11-21 图 11-9 中 Podres 的路由表

```
Podres#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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

172.16.0.0/16 is variably subnetted, 3 subnets, 3 masks
D EX    172.16.2.20/30 [170/2195456] via 192.168.3.129, 00:00:26, Ethernet0
D EX    172.16.0.0/16 [170/2195456] via 192.168.3.129, 00:00:26, Ethernet0
D EX    172.16.1.0/24 [170/2221056] via 192.168.3.129, 00:00:26, Ethernet0
     192.168.4.0/24 is variably subnetted, 3 subnets, 3 masks
D EX    192.168.4.72/29 [170/2221056] via 192.168.3.129, 00:00:26, Ethernet0
D EX    192.168.4.4/30 [170/2195456] via 192.168.3.129, 00:00:26, Ethernet0
D EX    192.168.4.0/24 [170/2195456] via 192.168.3.129, 00:00:27, Ethernet0
D EX 192.168.1.0/24 [170/2611200] via 192.168.3.129, 00:00:28, Ethernet0
C    192.168.2.0/24 is directly connected, Ethernet3
     192.168.3.0/24 is variably subnetted, 7 subnets, 2 masks
D EX    192.168.3.96/27 [170/2611200] via 192.168.3.129, 00:00:28, Ethernet0
D EX    192.168.3.64/27 [170/2611200] via 192.168.3.129, 00:00:28, Ethernet0
```

（待续）

```
D       192.168.3.32/27 [90/284160] via 192.168.3.129, 00:00:35, Ethernet0
D       192.168.3.0/24 is a summary, 00:07:18, Null0
C       192.168.3.192/27 is directly connected, Ethernet2
C       192.168.3.160/27 is directly connected, Ethernet1
C       192.168.3.128/27 is directly connected, Ethernet0
Podres#
```

示例 11-22 显示了 Hodges 的路由表，这里存在一些问题。回忆一下第 8 章的内容，被重新分配到 OSPF 域的路由类型可以是类型 1（E1）或类型 2（E2）外部路由。在这里，被重新分配且标记为 E2 的路由仅是主网地址 192.168.2.0/24、172.16.0.0 和 192.168.4.0/24。造成这种现象的原因是，在 Robinson 的配置语句中缺少关键字 **subnets**。如果没有该关键字，那么被重新分配的地址仅包括那些在非 OSPF 进程内的主网地址。

示例 11-22 Hodges 的路由表仅包括重新分配的主网路由，标记为 E2

```
Hodges#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
O E2 172.16.0.0/16 [110/100] via 192.168.3.33, 00:24:41, FastEthernet0/0
O E2 192.168.4.0/24 [110/100] via 192.168.3.33, 00:24:41, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0
O E2 192.168.2.0/24 [110/50] via 192.168.3.33, 00:02:57, FastEthernet0/0
     192.168.3.0/27 is subnetted, 3 subnets
C       192.168.3.96 is directly connected, Serial0/0
C       192.168.3.64 is directly connected, Serial0/0
C       192.168.3.32 is directly connected, FastEthernet0/0
Hodges#
```

修改后的 Robinson 配置见示例 11-23，增加了关键字 **subnets**。

示例 11-23 当向 OSPF 重新分配路由时，为了使子网地址和主网地址一起被重新分配，Robinson 使用关键字 subnets 来解决这个问题

```
router eigrp 1
 redistribute ospf 1 metric 1000 100 1 255 1500
 redistribute eigrp 2
 passive-interface FastEthernet0/0
 network 192.168.3.0
!
router eigrp 2
 redistribute ospf 1 metric 1000 100 1 255 1500
 redistribute eigrp 1
 network 192.168.4.0
 network 172.16.0.0
!
router ospf 1
 redistribute eigrp 1 subnets
 redistribute eigrp 2 metric 100 subnets
 default-metric 50 network 192.168.3.33 0.0.0.0 area 0
```

修改的结果使图 11-9 中的子网出现在 Hodeges 的路由表中（参见示例 11-24）。

示例 11-24 在关键字 **subnets** 被添加到 Robinson 的重新分配配置中后，Hodegs 便可以知道所有子网信息

```
Hodges#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
     172.16.0.0/16 is variably subnetted, 3 subnets, 3 masks
O E2    172.16.2.20/30 [110/100] via 192.168.3.33, 00:00:59, FastEthernet0/0
O E2    172.16.0.0/16 [110/100] via 192.168.3.33, 00:30:20, FastEthernet0/0
O E2    172.16.1.0/24 [110/100] via 192.168.3.33, 00:00:59, FastEthernet0/0
     192.168.4.0/24 is variably subnetted, 3 subnets, 3 masks
O E2    192.168.4.72/29 [110/100] via 192.168.3.33, 00:00:59, FastEthernet0/0
O E2    192.168.4.4/30 [110/100] via 192.168.3.33, 00:01:00, FastEthernet0/0
O E2    192.168.4.0/24 [110/100] via 192.168.3.33, 00:30:22, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0
O E2 192.168.2.0/24 [110/50] via 192.168.3.33, 00:08:38, FastEthernet0/0
     192.168.3.0/27 is subnetted, 6 subnets

C       192.168.3.96 is directly connected, Serial0/0
C       192.168.3.64 is directly connected, Serial0/0
C       192.168.3.32 is directly connected, FastEthernet0/0
O E2    192.168.3.192 [110/50] via 192.168.3.33, 00:01:12, FastEthernet0/0
O E2    192.168.3.160 [110/50] via 192.168.3.33, 00:01:12, FastEthernet0/0
O E2    192.168.3.128 [110/50] via 192.168.3.33, 00:01:12, FastEthernet0/0
Hodges#
```

缺省情况下，外部路由作为类型 2 路由被重新分配到 OSPF。正如第 8 章所讨论的，E2 路由仅包括路由的外部代价。如图 11-10 所示，当存在不止一条外部路由可以到达单一目标网络时，这一事实将显得十分重要。在图 11-10 的网络中，一台路由器正在重新分配代价为 50，指向 10.2.3.0/24 的路由；另一台路由器也正在重新分配代价为 100 并且指向相同目标网络的另一条路由。如果这条路由被作为 E2 通告，那么在 OSPF 域内的链路代价将不会被计入。结果在 OSPF 域内的路由器将会选择路由 1 到达 10.2.3.0/24。

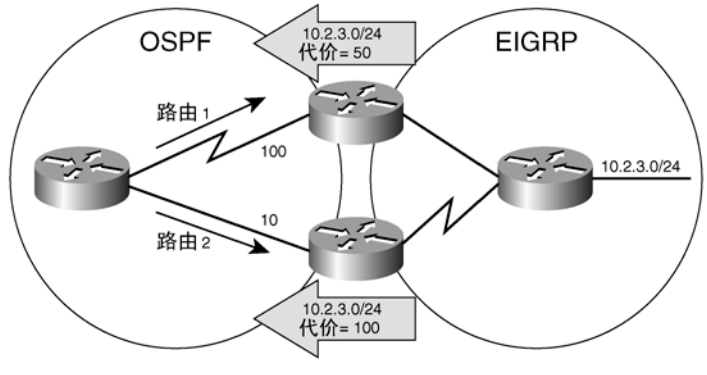


图 11-10 如果去往 10.2.3.0/24 的路由被作为 E2 通告，那么路由 1 的代价将会是 50，路由 2 的代价为 100。如果该路由被作为 E1 通告，那么路由 1 的代价为 150，路由 2 的代价为 110

如果在图 11-10 中，指向 10.2.3.0/24 的路由被作为 E1 重新分配，那么在 OSPF 域内的链路代价将被计入重新分配代价。结果是，OSPF 域内的路由器将选择路由 2，代价为 110（100

+10）；而不是路由 1，代价为 150（100+50）。

图 11-9 中的路由器 Robinson 正在重新分配代价为 50 的 EIGRP 1 和代价为 100 的 EIGRP 2。示例 11-24 显示出，在 Hodegs，指向 EIGRP 1 子网的路由代价仍然为 50，而指向 EIGRP 2 子网的路由代价为 100。因为在 Hodegs 和 Robinson 之间的快速以太网链路代价没有被计入。

为了将路由作为 E1 重新分配到 OSPF，可以在重新分配命令中添加关键字 **metric-type 1**。在示例 11-25 的配置中，Robinson 继续把 EIGRP 1 作为 E2 重新分配，但是把 EIGRP 2 作为 E1 重新分配。

示例 11-25 经配置，Robinson 把 EIGRP2 路由作为类型 1 外部路由向 OSPF 重新分配

```
router eigrp 1
 redistribute ospf 1 metric 1000 100 1 255 1500
 redistribute eigrp 2
 passive-interface FastEthernet0/0
 network 192.168.3.0
!
router eigrp 2
 redistribute ospf 1 metric 1000 100 1 255 1500
 redistribute eigrp 1
 network 192.168.4.0
 network 172.16.0.0
!
router ospf 1
 redistribute eigrp 1 subnets
 redistribute eigrp 2 metric 100 metric-type 1 subnets
 default-metric 50 network 192.168.3.33 0.0.0.0 area 0
```

在重新配置 Robinson 之后，示例 11-26 给出了 Hodegs 的路由表。在 EIGRP 1 域内所有指向目标网络的路由代价仍旧为 50，但是在 EIGRP 2 域内指向目标网络的路由代价现在为 101（重新分配代价加上 Robinson 和 Hodegs 之间快速以太网链路的缺省代价 1）。

示例 11-26 修改 Robinson 的配置以便把子网 192.168.4.0 和 172.16.0.0 作为类型 1 外部路由通告

```
Hodges#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
     172.16.0.0/16 is variably subnetted, 3 subnets, 3 masks
O E1    172.16.2.20/30 [110/101] via 192.168.3.33, 09:01:00, FastEthernet0/0
O E1    172.16.0.0/16 [110/101] via 192.168.3.33, 00:01:02, FastEthernet0/0
O E1    172.16.1.0/24 [110/101] via 192.168.3.33, 09:01:00, FastEthernet0/0
     192.168.4.0/24 is variably subnetted, 3 subnets, 3 masks
O E1    192.168.4.72/29 [110/101] via 192.168.3.33, 00:00:57, FastEthernet0/0
O E1    192.168.4.4/30 [110/101] via 192.168.3.33, 00:01:03, FastEthernet0/0
O E1    192.168.4.0/24 [110/101] via 192.168.3.33, 00:01:03, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0
O E2 192.168.2.0/24 [110/50] via 192.168.3.33, 09:01:01, FastEthernet0/0
     192.168.3.0/27 is subnetted, 6 subnets
C       192.168.3.96 is directly connected, Serial0/0
C       192.168.3.64 is directly connected, Serial0/0
C       192.168.3.32 is directly connected, FastEthernet0/0
```

（待续）

```
O E2    192.168.3.192 [110/50] via 192.168.3.33, 09:01:06, FastEthernet0/0
O E2    192.168.3.160 [110/50] via 192.168.3.33, 09:01:06, FastEthernet0/0
O E2    192.168.3.128 [110/50] via 192.168.3.33, 09:01:06, FastEthernet0/0
Hodges#
```

11.2.3 案例研究：重新分配和路由汇总

Cisco 对 EIGRP、OSPFv2、OSPFv3 和 IS-IS 的实现都可以汇总被重新分配的路由。这个案例研究将分析在 IPv4 下 EIGRP 和 OSPF 的路由汇总；下面两个案例研究将分析 IPv6 下 OSPFv3 和 IS-IS 的路由汇总。

第一个要注意的事情是汇总要想起作用，先决条件是 IP 子网地址已为汇总进行过规划。例如，在图 11-9 中 OSPF 域内，192.168.3.0 的子网全部都被汇总地址 192.168.3.0/25 所包含。在 EIGRP 1 域内相同主网地址的所有子网也都被汇总地址 192.168.3.128/25 覆盖。如果子网 192.168.3.0/27 被连接到 Podres，那么必须从汇总地址中将这个单一目标分离出来之后，才能进行通告。虽然通告单一目标几乎不会有什么不利影响，但是通告大量汇总地址范围之外的子网将会减少汇总带来的好处。

命令 **summary-address** 为 OSPF 进程指定了一个汇总地址和掩码。任何在指定汇总地址范围内的更精确的地址都会被禁止。注意，此命令仅用在 ASBR 汇总外部路由；在 ABR 内部 OSPF 路由的汇总可以通过命令 **area range** 实现，详见第 8 章。

在图 11-9 中路由器 Robinson 上，EIGRP 1 的子网在进入 OSPF 域时被汇总为 192.168.3.128/25，EIGRP 2 的子网被汇总为 172.16.0.0/16，具体配置见示例 11-27。

示例 11-27 Robinson 对通告给 OSPF 的外部地址进行汇总

```
router eigrp 1
 redistribute ospf 1 metric 1000 100 1 255 1500
 redistribute eigrp 2
 passive-interface FastEthernet0/0
 network 192.168.3.0
!
router eigrp 2
 redistribute eigrp 1
 network 192.168.4.0
 network 172.16.0.0
!
router ospf 1
 summary-address 192.168.3.128 255.255.255.128
 summary-address 172.16.0.0 255.255.0.0
 redistribute eigrp 1 subnets
 redistribute eigrp 2 metric 100 metric-type 1 subnets
 default-metric 50
 network 192.168.3.33 0.0.0.0 area 0
```

比较示例 11-28 和示例 11-26，在示例 11-28 中，Hodges 的路由表包含指定的汇总地址，汇总地址范围以内的子网地址在重新分配点被禁止。注意，由于没有对 192.168.4.0/24 进行汇总配置，所以该主网地址的所有子网仍出现在路由表中。

对于 EIGRP 的汇总是指定接口的。也就是不在路由进程下指明汇总地址和掩码，而是在独立的接口下指明。这个系统提供了更大的灵活性，可以在同一进程的不同接口通告不同的汇总地址。命令 **ip summary-address eigrp** *process-id* 指定了汇总地址、掩码和汇总所要通告的 EIGRP 进程。

在示例 11-29 的配置中，Robinson 将向 EIGRP 1 通告汇总地址 192.168.3.0/25、172.16.0.0/16 和 192.168.4.0/24。

示例 11-28 Robinson 正在汇总 192.168.3.128/25 和 172.16.0.0/26，因此在 Hodges 的路由表中不会出现在此范围内的更精确的地址

```
Hodges#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
O E1 172.16.0.0/16 [110/101] via 192.168.3.33, 00:11:55, FastEthernet0/0
     192.168.4.0/24 is variably subnetted, 3 subnets, 3 masks
O E1    192.168.4.72/29 [110/101] via 192.168.3.33, 00:03:56, FastEthernet0/0
O E1    192.168.4.4/30 [110/101] via 192.168.3.33, 00:11:55, FastEthernet0/0
O E1    192.168.4.0/24 [110/101] via 192.168.3.33, 00:11:55, FastEthernet0/0
C    192.168.1.0/24 is directly connected, Serial0/0
O E2 192.168.2.0/24 [110/50] via 192.168.3.33, 00:03:57, FastEthernet0/0
     192.168.3.0/24 is variably subnetted, 4 subnets, 2 masks
C       192.168.3.96/27 is directly connected, Serial0/0
C       192.168.3.64/27 is directly connected, Serial0/0
C       192.168.3.32/27 is directly connected, FastEthernet0/0
O E2    192.168.3.128/25 [110/50] via 192.168.3.33, 00:00:37, FastEthernet0/0
```

示例 11-29 Robinson 对通告到 EIGRP1 的路由进行汇总

```
interface FastEthernet0/0
 ip address 192.168.3.33 255.255.255.224
!
interface FastEthernet0/1
 ip address 192.168.3.129 255.255.255.224
 ip summary-address eigrp 1 192.168.3.0 255.255.255.128
 ip summary-address eigrp 1 172.16.0.0 255.255.0.0
 ip summary-address eigrp 1 192.168.4.0 255.255.255.0
!
interface Serial0/0.1
 ip address 192.168.4.5 255.255.255.252
 ip summary-address eigrp 2 192.168.3.0 255.255.255.0
!
interface Serial0/0.2
 ip address 172.16.2.21 255.255.255.252
 ip summary-address eigrp 2 192.168.0.0 255.255.0.0
!
router eigrp 1
 redistribute ospf 1 metric 1000 100 1 255 1500
 redistribute eigrp 2
 passive-interface FastEthernet0/0
 network 192.168.3.0
!
router eigrp 2
 redistribute eigrp 1
 network 192.168.4.0
 network 172.16.0.0
!
router ospf 1
 summary-address 192.168.3.128 255.255.255.128
 summary-address 172.16.0.0 255.255.0.0
```

（待续）

```
redistribute eigrp 1  subnets
redistribute eigrp 2 metric 100 metric-type 1 subnets
default-metric 50
network 192.168.3.33 0.0.0.0 area 0
```

示例 11-30 给出了 Podres 的路由表。正如 OSPF 汇总一样，EIGRP 的汇总禁止通告汇总范围以内的子网。但与 OSPF 不一样的是，Podres 的路由表显示向 EIGRP 通告的汇总路由没有被标记为外部路由。

示例 11-30 Podres 的路由表显示了汇总路由 192.168.3.0/25、192.168.4.0/24 和 192.172.16.0.0/16

```
Podres#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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
D    172.16.0.0/16 [90/2195456] via 192.168.3.129, 00:00:12, Ethernet0
D    192.168.4.0/24 [90/2195456] via 192.168.3.129, 00:00:12, Ethernet0
D EX 192.168.1.0/24 [170/2611200] via 192.168.3.129, 00:00:12, Ethernet0
C    192.168.2.0/24 is directly connected, Ethernet3
     192.168.3.0/24 is variably subnetted, 6 subnets, 3 masks
D       192.168.3.0/25 [90/284160] via 192.168.3.129, 00:00:12, Ethernet0
D       192.168.3.0/24 is a summary, 00:00:12, Null0
C       192.168.3.192/27 is directly connected, Ethernet2
C       192.168.3.160/27 is directly connected, Ethernet1
D EX    192.168.3.128/25 [170/2611200] via 192.168.3.129, 00:00:13, Ethernet0
C       192.168.3.128/27 is directly connected, Ethernet0
D EX 192.168.0.0/16 [170/284160] via 192.168.3.129, 00:00:13, Ethernet0
Podres#
```

Robinson 正在向 Campanella 通告 EIGRP 汇总路由 192.168.3.0/24，同时向 Snider 通告 192.168.0.0/16。示例 11-31 给出了 Campanella 的路由表，示例 11-32 给出了 Snider 的路由表。

示例 11-31 在 Robinson 配置汇总之后的 Campanellade 的路由表

```
Campanella#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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
D EX 172.16.0.0/16 [170/2681856] via 192.168.4.5, 00:35:12, Serial0.1
     192.168.4.0/24 is variably subnetted, 3 subnets, 3 masks
C       192.168.4.72/29 is directly connected, Ethernet0
C       192.168.4.4/30 is directly connected, Serial0.1
D EX    192.168.4.0/24 [170/2681856] via 192.168.4.5, 00:35:12, Serial0.1
D EX 192.168.1.0/24 [170/3097600] via 192.168.4.5, 00:35:12, Serial0.1
D EX 192.168.2.0/24 [170/2300416] via 192.168.4.5, 00:35:12, Serial0.1
D    192.168.3.0/24 [90/2172416] via 192.168.4.5, 00:35:13, Serial0.1
Campanella#
```

示例 11-32 在 Robinson 配置汇总之后的 Snider 的路由表

```
Snider#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
     172.16.0.0/16 is variably subnetted, 3 subnets, 3 masks
C       172.16.2.20/30 is directly connected, Serial0/0.1
D EX    172.16.0.0/16 [170/2681856] via 172.16.2.21, 00:07:59, Serial0/0.1
C       172.16.1.0/24 is directly connected, Ethernet0/0
D    192.168.0.0/16 [90/2195456] via 172.16.2.21, 00:07:59, Serial0/0.1
Snider#
```

回过头再看示例 11-30，注意指向 192.168.3.128/25 的汇总路由，它可能会使你感到惊讶，因为汇总地址被通告到 OSPF，而不是 EIGRP。另外还要注意，这条路由被标记为外部路由，这表明它已经被重新分配到 EIGRP 域了。这里所发生的情况是，汇总路由被通告到 OSPF，接着又从 OSPF 域被重新分配到 EIGRP。所以在 Podres 出现了不期望的路由条目。这就是 172.16.0.0/16 和 192.168.4.0/24 作为外部路由出现在 Campanella 的原因。这些汇总地址先是通告给 Podres 的 EIGRP1，接着又被重新分配给 EIGRP2。Snider 路由表中有指向 172.16.0.0/16 的路由，但没有指向 192.168.4.0/24 的路由，因为 Snider 有一条汇总地址为 192.168.0.0/16 的路由。

现在假设子网 192.168.3.192/27 变为不可访问。Podres 将按照较精确的路由 192.168.3.128/25 转发去往该子网的数据包。数据包将被发送到 OSPF 域，你可能会认为在 OSPF 域中汇总路由 192.168.3.128/25 将导致数据包被送回 Podres。

事实上，这种情形不会发生。Robinson 的路由表（参见示例 11-33）中有许多汇总路由条目都把 Null0 接口作为连接接口。空接口是一个不知道去往哪里的软件接口——路由到它的数据包将会被丢弃。除了某些特例外，[1]每当路由器产生一条汇总路由，路由器同时还会生成一条指向空接口的路由。如果 Robinson 接收到一个去往 192.168.3.192/27 的数据包并且该子网不再可达，那么路由器将转发数据包至空接口。路由环路将在这一跳被打断。

示例 11-33 Robinson 的路由表。由于路由器产生了许多汇总路由，因此相应地有许多连接到空接口的汇总路由条目。这样可以防止路由环路

```
Robinson#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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
     172.16.0.0/16 is variably subnetted, 3 subnets, 3 masks
C       172.16.2.20/30 is directly connected, Serial0/0.2
D       172.16.0.0/16 is a summary, 00:19:44, Null0
D       172.16.1.0/24 [90/2195456] via 172.16.2.22, 00:17:08, Serial0/0.2
     192.168.4.0/24 is variably subnetted, 3 subnets, 3 masks
D       192.168.4.72/29 [90/2172416] via 192.168.4.6, 00:15:17, Serial0/0.1
```

（待续）

[1] 例如，OSPF 内部区域汇总并不自动生成到空接口的汇总路由。它必须被静态配置。

```
C       192.168.4.4/30 is directly connected, Serial0/0.1
D       192.168.4.0/24 is a summary, 00:20:53, Null0
O    192.168.1.0/24 [110/74] via 192.168.3.34, 00:21:14, FastEthernet0/0
D    192.168.2.0/24 [90/665600] via 192.168.3.130, 00:19:41, FastEthernet0/1
     192.168.3.0/24 is variably subnetted, 8 subnets, 3 masks
O       192.168.3.96/27 [110/74] via 192.168.3.34, 00:21:15, FastEthernet0/0
O       192.168.3.64/27 [110/74] via 192.168.3.34, 00:21:15, FastEthernet0/0
C       192.168.3.32/27 is directly connected, FastEthernet0/0
D       192.168.3.0/24 is a summary, 00:19:01, Null0
D       192.168.3.0/25 is a summary, 00:19:47, Null0
D       192.168.3.192/27 [90/665600] via 192.168.3.130, 00:19:42, FastEthernet0/1
D       192.168.3.160/27 [90/665600] via 192.168.3.130, 00:19:42, FastEthernet0/1
C       192.168.3.128/27 is directly connected, FastEthernet0/1
D    192.168.0.0/16 is a summary, 00:19:00, Null0
Robinson#
```

指向空接口的汇总路由对于防止环路非常有用，关于空接口的使用详见第 12 章的内容。然而，重新分配不正确的路由选择信息是根本不允许发生的。假设 Podres 到 Robinson 不是 1 跳而是 10 跳，那么方向错误的消息要经过很长的路线之后才会被丢弃。这个例子证明了在互相进行重新分配时——也就是当两个路由选择协议互相向对方重新分配它们各自的路由时——需要仔细地控制路由通告。在这种情况下，使用路由过滤（详见第 13 章）或路由映射（详见第 14 章）是绝对必要的。

前面的情景还展示了为使用汇总而付出的代价。虽然路由表的大小被减少，节约了内存和处理器的循环周期，但路由的精度却降低了；随着网络变得更加复杂，细节的损失将会增加路由错误的可能性。

11.2.4 案例研究：重新分配 OSPFv3 和 RIPng

图 11-11 是一个 IPv6 的网络。路由器 Griffey 是 3 个分离网络的连接点。两个网络运行 OSPFv3，另一个运行 RIPng。Griffey 在路由选择进程之间重新分配路由。

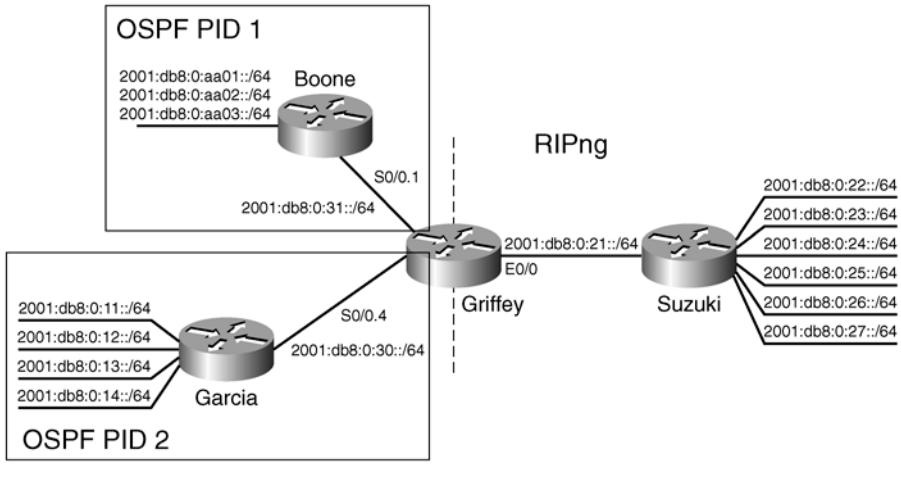


图 11-11 IPv6 通过 OSPFv3 和 RIPng 进行路由，其中一台路由器在路由选择进程之间进行路由的重新分配

路由器 Griffey 的配置见示例 11-34。

示例 11-34 Griffey 上配置了 IPv6、OSPFv3 进程 1、OSPFv3 进程 2 和 RIPng，Griffey 在各进程之间进行路由的重新分配

```
interface Ethernet 0/0
 ipv6 address 2001:db8:0:21::1/64
 ipv6 rip Mariners enable
!
interface Serial 0/0.1 point-to-point
 ipv6 address 2001:db8:0:31::1/64
 ipv6 ospf 1 area 1
!
interface Serial 0/0.4 point-to-point
 ipv6 address 2001:db8:0:30::1/64
 ipv6 ospf 2 area 20
!
ipv6 router ospf 1
 redistribute rip Mariners metric-type 1 tag 4
!
ipv6 router ospf 2
 redistribute rip Mariners
!
ipv6 router rip Mariners
 redistribute ospf 1 metric 5
 redistribute ospf 2 metric 10
```

Griffey 把来自 RIPng 进程 Mariners 的路由重新分配到进程 ID 为 1 的 OSPFv3 进程，路由的度量类型为外部类型 1。而且，这些路由被添加了标记，值为 4。在后面这些标记将被 RIPng 进程用来标识被通告的路由。Boone 的 IPv6 路由表（参见示例 11-35）给出了这些路由。

示例 11-35 IPv6 的路由表显示出来自 RIP 被重新分配到 OSPF 的路由，标记值为 4，度量类型为外部类型 1

```
Boone#show ipv6 route ospf
IPv6 Routing Table - 26 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
OE1  2001:DB8:0:22::/64 [110/84], tag 4
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE1  2001:DB8:0:23::/64 [110/84], tag 4
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE1  2001:DB8:0:24::/64 [110/84], tag 4
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE1  2001:DB8:0:25::/64 [110/84], tag 4
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE1  2001:DB8:0:26::/64 [110/84], tag 4
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE1  2001:DB8:0:27::/64 [110/84], tag 4
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
Boone#
```

Griffey 向 OSPFv3 进程（进程 ID 为 2）通告 RIPng 路由，路由度量类型为缺省值（外部类型 2），且没有标记值。在 Garcia 的路由表中可以看到这些被重新分配的路由（参见示例 11-36）。

示例 11-36 IPv6 路由表给出了度量类型为缺省类型——外部类型 2 的被重新分配的路由

```
Garcia#show ipv6 route ospf
IPv6 Routing Table - 26 entries
```

（待续）

```
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
OE2  2001:DB8:0:22::/64 [110/20]
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE2  2001:DB8:0:23::/64 [110/20]
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE2  2001:DB8:0:24::/64 [110/20]
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE2  2001:DB8:0:25::/64 [110/20]
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE2  2001:DB8:0:26::/64 [110/20]
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
OE2  2001:DB8:0:27::/64 [110/20]
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
Garcia#
```

对 IPv6 来说，关键字 **subnets** 不能和 OSPFv3 一起使用。所有 IPv6 的路由选择协议都包括 IPv6 前缀，这些前缀通常包括地址和前缀长度。除非执行过滤机制限制前缀被通告，否则协议之间的路由重新分配会包括所有前缀。第 13 章和 14 章将会讨论路由过滤机制。

Griffey 也向 RIPng 重新分配从两个 OSPFv3 进程获知的路由。在被通告的路由中，来自 OSPF1 的路由度量值为 5，来自 OSPF2 的路由度量值为 10。示例 11-37 给出了 Suzuki 的 IPv6 路由表。

示例 11-37　从 RIPng 的路由表可以看出，被重新分配到 RIPng 的路由带有不同的度量值

```
Suzuki#show ipv6 route rip
IPv6 Routing Table - 19 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
R   2001:DB8:0:11::/64 [120/11]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
R   2001:DB8:0:12::/64 [120/11]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
R   2001:DB8:0:13::/64 [120/11]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
R   2001:DB8:0:14::/64 [120/11]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
R   2001:DB8:0:AA01::/64 [120/6]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
R   2001:DB8:0:AA02::/64 [120/6]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
R   2001:DB8:0:AA03::/64 [120/6]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
Suzuki#
```

来自 Boone 的路由的度量值为 6，来自 Garcia 的路由的度量值为 11。

在 Griffey 上将要配置路由汇总。当 RIPng 重新分配来自其他协议的路由时，RIPng 没有办法对这些通告来的路由进行汇总。但是在 Griffey 连接到 Suzuki 的以太网接口上，我们可以通过配置汇总从该接口通告出去的 RIPng 前缀。当 RIPng 前缀被通告进入路由进程时，OSPFv3 可以对它们进行汇总。Griffey 具体的配置见示例 11-38。

示例 11-38 当 RIPng 路由从 Ethernet0/0 被通告或被重新分配到 OSPFv3 时，Griffey 可以对它们进行汇总

```
interface Ethernet 0/0
 ipv6 address 2001:db8:0:21::1/64
 ipv6 rip Mariners enable
 ipv6 rip Mariners summary-address 2001:DB8:0:AA00::/62
 ipv6 rip Mariners summary-address 2001:DB8:0:10::/61
!
interface Serial 0/0.1 point-to-point
 ipv6 address 2001:db8:0:31::1/64
 ipv6 ospf 1 area 1
!
interface Serial 0/0.4 point-to-point
 ipv6 address 2001:db8:0:30::1/64
 ipv6 ospf 2 area 20
!
ipv6 router ospf 1
 redistribute rip Mariners metric-type 1 tag 4
 summary-prefix 2001:DB8:0:20::/61 tag 4
!
ipv6 router ospf 2
 redistribute rip Mariners
 summary-prefix 2001:DB8:0:20::/61
!
ipv6 router rip Mariners
 redistribute ospf 1 metric 5
 redistribute ospf 2 metric 10
```

在示例 11-39 中，汇总路由出现在新的路由表中，而更精确的路由被删除了。

示例 11-39 路由表中包括汇总路由，但是不再包含属于汇总范围的更精确的路由

```
Boone#show ipv6 route ospf
IPv6 Routing Table - 21 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
OE1  2001:DB8:0:20::/61 [110/84], tag 4
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
Boone#
```

```
Garcia#show ipv6 route ospf
IPv6 Routing Table - 21 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
OE2  2001:DB8:0:20::/61 [110/20]
     via FE80::207:85FF:FE6B:EA20, Serial0/0.1
Garcia#
```

```
Suzuki#show ipv6 route rip
IPv6 Routing Table - 19 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
```

（待续）

```
R   2001:DB8:0:10::/61 [120/11]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
R   2001:DB8:0:20::/61 [120/6]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
R   2001:DB8:0:AA00::/62 [120/6]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
Suzuki#
```

注意，Griffey 把汇总路由 2001:db8:0:20::/61 通告给 Suzuki。这是为通告到 OSPFv3 的前缀建立的汇总路由。OSPF 又把这个汇总路由通告给 RIPng。RIPng 域内的路由器仍然能够正确的路由更加精确的前缀。从 Suzuki 的 IPv6 路由表（参见示例 11-40）可以看出，汇总 2001:db8:0:20::/61 所包含的前缀是直接连接到路由器的。所以路由器将会把流量转发给更精确的前缀，而不是给汇总前缀。让汇总重新被通告回 RIPng 域的不利之处表现为，如果数据包的目标地址在汇总范围内，但是这个目标实际在网络中并不存在，那么数据包在被丢弃之前会被转发给 Griffey。我们可以在路由重新分配期间对路由进行过滤，以便被通告到 RIPng 的前缀仅是那些已知存在于 OSPFv3 域内的前缀。重新分配期间的路由过滤将在第 13 章和 14 章讨论。

示例 11-40 Suzuki 的路由表中即包括汇总路由，也包括属于汇总范围的更精确的路由。路由器将尽可能的向更精确的路由转发流量

```
Suzuki#show ipv6 route
IPv6 Routing Table - 19 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
R   2001:DB8:0:10::/61 [120/11]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
R   2001:DB8:0:20::/61 [120/6]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
C   2001:DB8:0:21::/64 [0/0]
     via ::, Ethernet0/0
L   2001:DB8:0:21::2/128 [0/0]
     via ::, Ethernet0/0
C   2001:DB8:0:22::/64 [0/0]
     via ::, Serial0/0
L   2001:DB8:0:22::1/128 [0/0]
     via ::, Serial0/0
C   2001:DB8:0:23::/64 [0/0]
     via ::, Serial0/0
L   2001:DB8:0:23::1/128 [0/0]
     via ::, Serial0/0
C   2001:DB8:0:24::/64 [0/0]
     via ::, Serial0/0
L   2001:DB8:0:24::1/128 [0/0]
     via ::, Serial0/0
C   2001:DB8:0:25::/64 [0/0]
     via ::, Serial0/0
L   2001:DB8:0:25::1/128 [0/0]
     via ::, Serial0/0
C   2001:DB8:0:26::/64 [0/0]
     via ::, Serial0/0
L   2001:DB8:0:26::1/128 [0/0]
     via ::, Serial0/0
C   2001:DB8:0:27::/64 [0/0]
```

（待续）

```
     via ::, Serial0/0
L    2001:DB8:0:27::1/128 [0/0]
     via ::, Serial0/0
R    2001:DB8:0:AA00::/62 [120/6]
     via FE80::207:85FF:FE6B:EA20, Ethernet0/0
L    FE80::/10 [0/0]
     via ::, Null0
L    FF00::/8 [0/0]
     via ::, Null0
Suzuki#
```

11.2.5 案例研究：重新分配 IS-IS 和 RIP/RIPng

在图 11-12 的网络中，Aaron 运行 IS-IS，Williams 为 IPv4 运行 RIPv1，为 IPv6 运行 RIPng，Mays 正在重新分配路由。

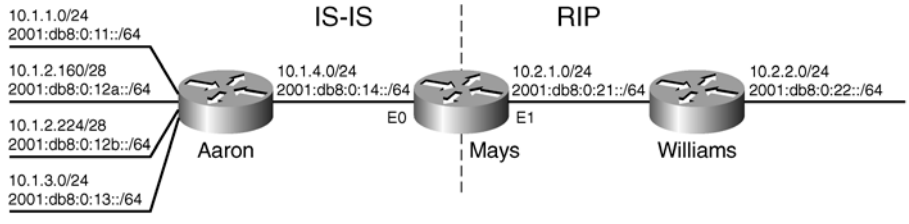


图 11-12 路由器 Mays 正在向 IS-IS 重新分配 RIP 路由，同时向 RIP 重新分配 IS-IS 路由

Mays 的 IS-IS 和 RIP 配置见示例 11-41。

示例 11-41 路由器 Mays 的 IS-IS、RIP 和 RIPng 配置

```
interface Ethernet1
 description to Williams
 ip address 10.2.1.1 255.255.255.0
 ipv6 address 2001:db8:0:21::1/64
 ipv6 rip baseball enable
!
interface Ethernet0
 description to Aaron
 ip address 10.1.4.2 255.255.255.0
 ip router isis
 ipv6 address 2001:db8:0:14::2/64
 ipv6 router isis

router rip
 passive-interface Ethernet0
 network 10.0.0.0
 redistribute isis level-1-2 metric 1
!
router isis
 net 01.0001.0000.0c76.5432.00
 redistribute rip metric 0 metric-type internal level-2
 address-family ipv6
  redistribute rip baseball metric 0 metric-type internal level-2
!
ipv6 router rip baseball
 redistribute isis level-1-2 metric 1
```

路由可能被作为内部或外部路由（缺省是内部）、第 1 层或第 2 层（缺省是第 1 层）路由向 IS-IS 重新分配。度量类型决定被重新分配路由的度量值基数。内部度量类型的度量值小

于 64，外部类型的度量值在 64 到 128 之间。如果度量类型是内部类型，那么被重新分配路由的初始度量将是度量值指定的数字（在本示例中度量值为 0 ），如果没有指定度量值，那么则为 0 。在这个例子中，被重新分配的路由被添加进 IS-IS 数据库，表项的度量值为 0。如果度量值为 5，那么在 Mays 的 IS-IS 数据库中的表项度量值将是 5。如果度量的外部类型基数从 64 开始且指定的度量值为 5，那么 Mays 的 IS-IS 数据库中的表项度量值将是 64＋5 或 69。在所示例子中，RIP 路由作为内部 L2 路由被重新分配，且使用缺省度量 0。示例 11-42 给出了 Aaron 的 IPv4 路由表中被重新分配的路由，示例 11-43 给出了 Aaron 的 IPv6 路由表中被重新分配的路由。

示例 11-42 Araon 的路由表显示了被重新分配的 RIP 路由

```
Aaron#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
     10.0.0.0/8 is variably subnetted, 7 subnets, 2 masks
C     10.1.3.0/24 is directly connected, Ethernet4
i L2  10.2.1.0/24 [115/10] via 10.1.4.2, Ethernet0
i L2  10.2.2.0/24 [115/10] via 10.1.4.2, Ethernet0
C     10.1.1.0/24 is directly connected, Ethernet1
C     10.1.4.0/24 is directly connected, Ethernet0
C     10.1.2.160/28 is directly connected, Ethernet2
C     10.1.2.224/28 is directly connected, Ethernet3
Aaron#
```

示例 11-43 Aaron 的 IPv6 路由表给出了被重新分配的 RIP 路由

```
Aaron#show ipv6 route isis
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
I2  2001:DB8:0:22::/64 [115/10]
     via FE80::204:C1FF:FE50:E700, Ethernet0
Aaron#
```

Aaron 的 IPv6 路由表显示，连接到 Mays 上的地址 2001:da8:0:21::/64 没有被通告到 IS-IS。为了重新分配这些直连地址，在示例 11-44 中附加了命令 **redistribute connected**，示例 11-45 给出了因此而产生的 IPv6 路由表。

示例 11-44 因为在 Mays 上添加了 redistribute connected，所以直连 IPv6 前缀被通告到 IS-IS

```
router isis
 net 01.0001.0000.0c76.5432.00
 redistribute rip metric 0 metric-type internal level-2
 address-family ipv6
  redistribute rip baseball metric 0 metric-type internal level-2
  redistribute connected metric 0 metric-type internal level-2
!
ipv6 router rip baseball
 redistribute isis level-1-2 metric 1
 redistribute connected metric 1
```

示例 11-45 Aaron 的 IPv6 路由表给出了被重新分配的 RIP 路由和直连地址

```
Aaron#show ipv6 route isis
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
I2  2001:DB8:0:21::/64 [115/10]
     via FE80::204:C1FF:FE50:E700, Ethernet0
I2  2001:DB8:0:22::/64 [115/10]
     via FE80::204:C1FF:FE50:E700, Ethernet0
Aaron#
```

因为 RIP 对于 IS-IS 路由域来说是外部路由，把它们作为外部路由重新分配到路由域可以最好地反映这一点。详见示例 11-46。

示例 11-46 Mays 把从 RIP 重新分配到 IS-IS 的路由配置为外部路由

```
router isis
 redistribute rip metric 0 metric-type external level-2
 net 01.0001.0000.0c76.5432.00
 address-family ipv6
  redistribute rip baseball metric 0 metric-type external level-2
  redistribute connected metric 0 metric-type external level-2
!
router rip
 redistribute isis level-1-2 metric 1
 passive-interface Ethernet0
 network 10.0.0.0
!
ipv6 router rip baseball
 redistribute isis level-1-2 metric 1
 redistribute connected metric 1
```

示例 11-47 给出了 Aaron 改变后的路由表。与示例 11-42 相比，惟一改变的是被重新分配的路由度量值变大了，并且大于 64，这表明（在这个小型网络中）它们是外部路由。示例 11-48 给出了 Mays 的 IS-IS 数据库表项，其中被重新分配的路由表项被表示为 IP-External 和 IPv6-Ext。

示例 11-47 在路由被通告为外部路由后，去往 10.2.1.0/24 和 10.2.2.0/24 的路由度量值变为 138

```
Aaron#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
     10.0.0.0/8 is variably subnetted, 7 subnets, 2 masks
C       10.1.3.0/24 is directly connected, Ethernet4
i L2    10.2.1.0/24 [115/74] via 10.1.4.2, Ethernet0
i L2    10.2.2.0/24 [115/74] via 10.1.4.2, Ethernet0
C       10.1.1.0/24 is directly connected, Ethernet1
C       10.1.4.0/24 is directly connected, Ethernet0
```

（待续）

```
C       10.1.2.160/28 is directly connected, Ethernet2
C       10.1.2.224/28 is directly connected, Ethernet3
Aaron#
```

示例 11-48 在 Mays 的 IS-IS 数据库中，被重新分配的路由被表示为 IP-External 和 IPv6-Ext

```
Mays#show isis database detail level-2 Mays.00-00
IS-IS Level-2 LSP Mays.00-00
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Mays.00-00          * 0x0000005F   0xA0E5        1186              0/0/0
  Area Address: 01.0001
  NLPID:        0xCC 0x8E
  Hostname: Mays
  IP Address:   10.1.4.2
  IPv6 Address: 2001:DB8:0:14::2
  Metric: 10         IS Aaron.00
  Metric: 20         IP 10.1.3.0 255.255.255.0
  Metric: 64         IP-External 10.2.1.0 255.255.255.0
  Metric: 64         IP-External 10.2.2.0 255.255.255.0
  Metric: 20         IP 10.1.1.0 255.255.255.0
  Metric: 10         IP 10.1.4.0 255.255.255.0
  Metric: 20         IP 10.1.2.160 255.255.255.240
  Metric: 20         IP 10.1.2.224 255.255.255.240
  Metric: 20         IPv6 2001:DB8:0:11::/64
  Metric: 20         IPv6 2001:DB8:0:13::/64
  Metric: 10         IPv6 2001:DB8:0:14::/64
  Metric: 0          IPv6-Ext 2001:DB8:0:21::/64
  Metric: 0          IPv6-Ext 2001:DB8:0:22::/64
  Metric: 20         IPv6 2001:DB8:0:12A::/64
  Metric: 20         IPv6 2001:DB8:0:12B::/64
Mays#
```

从图 11-12 还可以看出，RIP 域内的 IPv4 和 IPv6 子网被汇总为 10.2.0.0/16 和 2001:db8:0:20::/62。命令 **summary-address** 用于把 IPv4 路由汇总到 IS-IS；OSPF 也使用这个命令，而且还可以指定把汇总路由发送到哪一层。如果没有指定，那么地址将汇总到第 2 层。同 OSPFv3 一样，对于 IPv6 地址簇，将使用 **summary-prefix** 汇总 IPv6 路由到 IS-IS。在示例 11-49 的配置中，RIP 的 IPv4 路由被作为 L1 重新分配和汇总，RIPng 的 IPv6 路由被作为 L2 重新分配和汇总。

示例 11-49 在 RIP 路由被重新分配进入 IS-IS 后，Mays 对它们进行汇总

```
router isis
 summary-address 10.2.0.0 255.255.0.0 level-1
 redistribute rip metric 0 metric-type external level-1
 net 01.0001.0000.0c76.5432.00
 address-family ipv6
  redistribute rip baseball metric 0 metric-type external level-2
  redistribute connected metric 0 metric-type external level-2
  summary-prefix 2001:db8:0:20::/62 level-2
router rip
 redistribute isis level-1-2 metric 1
 passive-interface Ethernet0
 network 10.0.0.0
!
ipv6 router rip baseball
 redistribute isis level-1-2 metric 1
 redistribute connected metric 1
```

示例 11-50 给出了 Aaron 的 IPv4 路由表的汇总路由。示例 11-51 给出了 Aaron 的 IPv6

路由表。像 OSPF 和 EIGRP 一样，这会导致在汇总范围内更精确的路由被抑制。

示例 11-50 Aaron 的 IPv4 路由表中有一条指向 RIP 域子网的 L1 汇总路由

```
Aaron#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
     10.0.0.0/8 is variably subnetted, 6 subnets, 3 masks
i L1    10.2.0.0/16 [115/138] via 10.1.4.2, Ethernet0
C       10.1.3.0/24 is directly connected, Ethernet4
C       10.1.1.0/24 is directly connected, Ethernet1
C       10.1.4.0/24 is directly connected, Ethernet0
C       10.1.2.160/28 is directly connected, Ethernet2
C       10.1.2.224/28 is directly connected, Ethernet3
Aaron#
```

示例 11-51 Aaron 的 IPv6 路由表中有一条指向 RIP 域子网的 L2 汇总路由

```
Aaron#show ipv6 route isis
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
I2  2001:DB8:0:20::/62 [115/10]
     via FE80::204:C1FF:FE50:E700, Ethernet0
Aaron#
```

当 IS-IS 被重新分配到其他协议时，必须为重新分配的路由指定层数。到目前为止，在我们所举的例子中，重新分配到 RIP 的路由都被指定为 L1 和 L2。

11.2.6 案例研究：重新分配静态路由

示例 11-52 给出了在图 11-12 中 Williams 的路由表。注意缺少了子网 10.1.2.160/28 和 10.1.2.224/28。这些子网的掩码与配置在 Mays 接口 E1 上的 24 位掩码不一致，所以从该接口发送的 RIP 更新消息中没有包含这些路由。这个例子再一次说明了从无类别协议向有类别协议重新分配变长子网路由所存在的问题，这在本章开头曾讨论过。

示例 11-52 没有向 RIP 域重新分配掩码不是 24 位的子网路由

```
Williams#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
     10.0.0.0/8 is subnetted, 5 subnets
R       10.1.3.0 [120/1] via 10.2.1.2, 00:00:01, Ethernet0
C       10.2.1.0 is directly connected, Ethernet0
R       10.1.1.0 [120/1] via 10.2.1.2, 00:00:02, Ethernet0
```

（待续）

```
C       10.2.2.0 is directly connected, Ethernet1
R       10.1.4.0 [120/1] via 10.2.1.2, 00:00:02, Ethernet0
Williams#
```

解决这个问题的一种方案是使用 24 位地址 10.1.2.0/24 汇总两个 28 位子网。因为 RIP 没有这个汇总命令，所以实现该汇总的办法是配置一条指向汇总地址的静态路由，接着向 RIP 重新分配该路由。详细的配置见示例 11-53。

示例 11-53 在 Mays 上使用静态路由为 IPv4 的子网建立了一条汇总路由，并且把这条静态路由重新分配到 RIP 中

```
router isis
 summary-address 10.2.0.0 255.255.0.0 level-1
 redistribute rip metric 0 metric-type external level-1
 net 01.0001.0000.0c76.5432.00
!
router rip
 redistribute static metric 1
 redistribute isis level-1-2 metric 1
 passive-interface Ethernet0
 network 10.0.0.0
!
ip route 10.1.2.0 255.255.255.0 10.1.4.1
```

示例 11-54 给出了 Williams 的路由表，其中包含了该汇总路由。

示例 11-54 子网 10.1.2.160/28 和 10.1.2.224/28 被汇总为 10.1.2.0/24

```
Williams#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
     10.0.0.0/8 is subnetted, 6 subnets
R       10.1.3.0 [120/1] via 10.2.1.2, 00:00:03, Ethernet0
R       10.1.2.0 [120/1] via 10.2.1.2, 00:00:03, Ethernet0
C       10.2.1.0 is directly connected, Ethernet0
R       10.1.1.0 [120/1] via 10.2.1.2, 00:00:03, Ethernet0
C       10.2.2.0 is directly connected, Ethernet1
R       10.1.4.0 [120/1] via 10.2.1.2, 00:00:03, Ethernet0
Williams#
```

正如第 3 章所讨论的，静态路由的一种变化形式是使用出站端口代替路由条目中的下一跳地址。这种静态路由也可以被重新分配，但是配置上稍微有一点不同。Mays 的配置可参见示例 11-55。

示例 11-55 在 Mays 上使用出站接口代替静态路由中的下一跳地址

```
router isis
 summary-address 10.2.0.0 255.255.0.0 level-1
 redistribute rip metric 0 metric-type external level-1
 net 01.0001.0000.0c76.5432.00
!
router rip
 redistribute isis level-1-2 metric 1
 passive-interface Ethernet0
 network 10.0.0.0
!
ip route 10.1.2.0 255.255.255.0 Ethernet0
```

这里的静态路由指向 Mays 的接口 E0，而不是指向下一跳地址 10.1.4.1。由于在 RIP 的配置模式下不再使用命令 **rdistribute static**，所以 Williams 的路由表看上去和示例 11-54 一样。

这个静态路由仍然会被重新分配的原因是当静态路由指向出站接口时，目标网络被认为是直接连接到路由器的（参见示例 11-56）；又因为网络 10.0.0.0 的语句出现在 RIP 的配置中，所以 RIP 将通告 10.0.0.0 的直连子网。

示例 11-56 Mays 认为汇总地址 10.1.2.0/24 被直接连接到 Ethernet 0

```
Mays#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
     10.0.0.0/8 is variably subnetted, 8 subnets, 2 masks
i L1    10.1.3.0/24 [115/20] via 10.1.4.1, Ethernet0
S       10.1.2.0/24 is directly connected, Ethernet0
C       10.2.1.0/24 is directly connected, Ethernet1
i L1    10.1.1.0/24 [115/20] via 10.1.4.1, Ethernet0
R       10.2.2.0/24 [120/1] via 10.2.1.1, 00:00:21, Ethernet1
C       10.1.4.0/24 is directly connected, Ethernet0
i       L1 10.1.2.160/28 [115/20] via 10.1.4.1, Ethernet0
i       L1 10.1.2.224/28 [115/20] via 10.1.4.1, Ethernet0
Mays#
```

假设 Williams 接收到数据包的目的地址是 10.1.2.5，并且数据包匹配到汇总地址 10.1.2.0/24，那么数据包将被转发给 Mays。在 Mays，目的地址不能匹配到更精确的子网，因而最终匹配到这条静态路由。Mays 将在接口 E0 发送 ARP 请求，试图发现主机 10.1.2.5（或者发现将发送一个代理 ARP 回应的路由器）。如果没有发现，那么路由器将不知道该如何处理此数据包。ICMP 目标不可达信息将不会被发送给源点。

回忆一下，当使用汇总命令时，它们会在路由表中创建一个指向空接口的路由条目。

注意：空接口应该和静态汇总路由联合使用。

对于静态汇总路由，也应该做同样的工作（参见示例 11-57）。

示例 11-57 Mays 的静态汇总路由指向空接口

```
router isis
 summary-address 10.2.0.0 255.255.0.0 level-1
 redistribute rip metric 0 metric-type external level-1
 net 01.0001.0000.0c76.5432.00
!
router rip
 redistribute isis level-1-2 metric 1
 passive-interface Ethernet0
 network 10.0.0.0
!
ip route 10.1.2.0 255.255.255.0 Null0
```

现在，在路由器 Mays，任何不能发现最精确匹配的目的地址都会被路由到空接口后丢弃，同时目标不可达的 ICMP 消息将会被发送给源点。

11.3 展 望

本章讨论了在重新分配路由时会出现的几个问题。为了避免或纠正故障，除了最简单的重新分配方案之外，通常几乎在所有的方案中都包括对路由过滤和路由映射的使用，它们分别在第 13 章和第 14 章讨论。这些章节包括了更加复杂的重新分配方案以及如何进行故障诊断。首先，第 12 章将研究缺省路由——缺省路由被认为是汇总路由的最普遍的形式。

11.4 总结表：第 11 章命令总结

命令	描述
default-metric *bandwidth delay reliability load mtu*	为重新分配到 IGRP 和 EIGRP 的路由指定缺省度量
default-metric *number*	为重新分配到 OSPF 和 RIP 的路由指定缺省度量
ip summary-address eigrp *autonomous-system-number address mask*	在接口上配置一条 EIGRP 汇总路由
redistribute connected	重新分配所有直连网络
redistribute *protocol* [*process-id*] {**level-1**|**level-1-2**|**level-2**} [**metric** *metric-value*][**metric-type** *type-value*][**match**{**internal**| **external 1**|**external 2**}][**tag** *tag-value*] [**route-map** *map-tag*][**weight** *weight*][**subnets**]	配置向一个路由选择协议重新分配路由，并且指定被重新分配路由的源
summary-address *address mask* {**level-1**|**level-1-2**|**level-2**} *prefix mask* [**not-advertise**] [**tag** *tag*]	为 IS-IS 和 OSPF 配置路由汇总
summary-prefix *prefix/length* **{level-1 | level-1-2 | level-2}**	为 IPv6 的 IS-IS 配置汇总路由
summary-prefix *prefix/length* **[not-advertise | tag** *tag-value***]**	为 IPv6 的 OSPFv3 配置汇总路由

11.5 复 习 题

1．来自什么样信息源的路由可以被重新分配？

2．管理距离的目的是什么？

3．在重新分配时管理距离是如何导致故障的？

4．从无类别路由选择协议向有类别路由选择协议重新分配是怎样导致故障的？

5．哪一种 IP 的 IGP 可以使用缺省重新分配度量，为了重新分配工作正常，哪一种 IGP 必须配置度量？

6．使用带关键字 **metric** 的 **redistribute** 命令和 **default-metric** 命令有什么区别？

7．在重新分配 OSPF 时，关键字 **subnets** 的作用是什么？

8．在汇总路由时空接口是如何起作用的？

11.6 配置练习

1．在图 11-13 中，路由器 A 运行 OSPFv2 和 OSPFv3，路由器 C 运行 RIPv1 和 RIPng。为了使所有子网相互连通，请给出路由器 B 的配置。

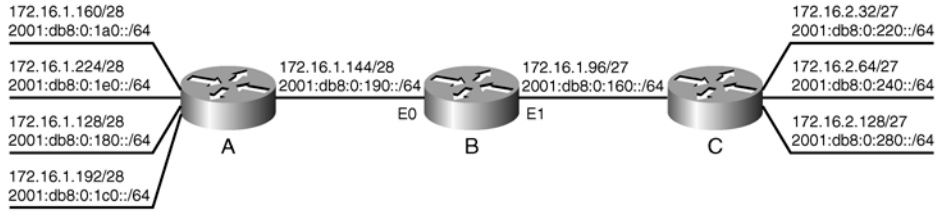


图 11-13 配置练习 1～3 所使用的网络

2．如图 11-13 所示，请尽可能在路由器 B 配置 IPv6 汇总路由。

3．在图 11-13 中路由器 A 为 IPv4 和 IPv6 运行 IS-IS。路由器 C 为 IPv4 运行 EIGRP，为 IPv6 运行 RIPng。在路由器 B 上，所有 IS-IS 的路由均为第 1 层。请在路由器 B 配置相互重新分配时尽可能使用汇总路由。EIGRP 路由要求被作为外部路由通告到 IS-IS 域。

11.7 故障诊断练习

1．在 11.2.1 小节中，下面给出图 11-8 中路由器 Mantle 的配置：

```
router rip
 redistribute igrp 1 metric 5
 passive-interface Ethernet1
 network 10.0.0.0
!
router igrp 1
 redistribute connected
 redistribute rip
 default-metric 1000 100 255 1 1500
 passive-interface Ethernet0
 network 10.0.0.0
```

由于路由可以被重新分配到 IGRP，接着再次被重新分配到 RIP，那么 RIP 域可以知道末梢网络 192.168.10.0/24 吗？

2．在故障诊断练习 1 中，如果在 IGRP 的配置中取消命令 **redistribute rip**，那么为了向 RIP 域进行通告，命令 **redistribute connected** 是否可以满足要求？

3．在示例 11-21 中，为什么子网 192.168.3.32/27 没有被标记为 EIGRP 外部路由？

4．在示例 11-21 中，有一条指向 192.168.3.0 的汇总路由，是什么导致产生了这一条目？

5．根据以下配置，问区域 1 区的 L1 路由器是否可以知道汇总路由？

Router A:

```
router isis
 net 01.0001.0000.0c76.5432.00
 address-family ipv6
  redistribute rip baseball metric-type external
  summary-prefix 2001:db8:0:20::/62 level-1
```

本章包括以下主题：

- 缺省路由基本原理；
- 按需路由基本原理；
- 配置缺省路由和 ODR。

第 12 章

缺省路由和按需路由选择

到目前为止，我们已经在几个章节中对路由汇总进行了讨论。汇总可以减小路由表的大小和路由通告内容，从而节省了网络资源。路由表越小、越简单，那么管理和故障诊断也越容易。

一个汇总地址可以表示几个或更多个更加精确的地址。例如，下面的 4 个子网可以用单一地址 192.168.200.128/25 来汇总。

```
192.168.200.128/27
192.168.200.160/27
192.168.200.192/27
192.168.200.224/27
```

当使用二进制方式查看地址时，我们可以看出汇总地址不太准确，因为汇总地址所包含的网络和子网位要比原地址少。因此，如果用粗略的方式表达，可以说向主机空间添加越多的 0 位，被使用的网络位就越少，那么可以汇总的地址就越多。按照这种想法，如果许多 0 位被添加到主机空间以至于没有剩余的网络位将会怎么样？换言之，如果汇总地址包括 32 个 0 位和前缀长度为 0（0.0.0.0/0）又会怎样呢？这个地址将会汇总所有可能的 IPv4 地址。

0.0.0.0/0 是 IPv4 的缺省地址，指向 0.0.0.0/0 的路由是缺省路由。[1]类似的，缺省 IPv6 地址::/0 可以汇总所有 IPv6 地址。其他每个 IP 地址都比缺省地址更准确，所以当路由表中存在缺省路由时，如果不能寻找到一个更加匹配的路由，那么都会匹配到缺省路由上。

[1] 在所有开放式 IP 路由选择协议中均使用这个地址。Cisco 的 IGRP 和 EIGRP 使用一个真实的网络地址，作为外部路由进行通告。

12.1 缺省路由基本原理

当将路由器与 Internet 相连时，缺省路由是非常有用的。使用了缺省路由，路由器仅需要知道它自己内部管理系统中的目标网络。缺省路由将把去往其他地址的数据包转发给 Internet 服务提供商。这样可以没有必要同服务提供商使用边界网关协议（BGP）去学习 Internet 路由表中的所有前缀——Internet 路由表包含 100 000 以上的前缀，并且可能很快会达到 200 000。在处理大型路由表时，拓扑变化所产生的影响远远大于对内存的需求。在大型网络中，拓扑频繁地发生变化，从而导致通告以及处理这些变化的系统活动明显增加。使用缺省路由可以有效地“隐藏”更精确路由的变化，使得具有缺省路由的网络更加稳定。

在单一自主系统中，缺省路由在更小的程度上还是有用的。在小型网络中，缺省路由可以减少对内存和 CPU 的使用，尽管随着路由数目的减少这种好处也会相应减弱。

缺省路由在星型（hub-and-spoke）拓扑结构中也非常有用，如图 12-1 所示。在这里，中心路由器包含指向每一个远程子网的静态路由。每当一个新的子网在线，中心路由器上就会被输入一条新的静态路由。虽然这一管理任务是微不足道的，但是要在每台分支路由器上添加路由可能会很耗时。如果在分支路由器上使用缺省路由，那么仅中心路由器需要有关每个子网的路由。当分支路由器收到去往未知目标网络的数据包时，它将把数据包转发至中心路由器，中心路由器接着将数据包发送到正确的目的地。

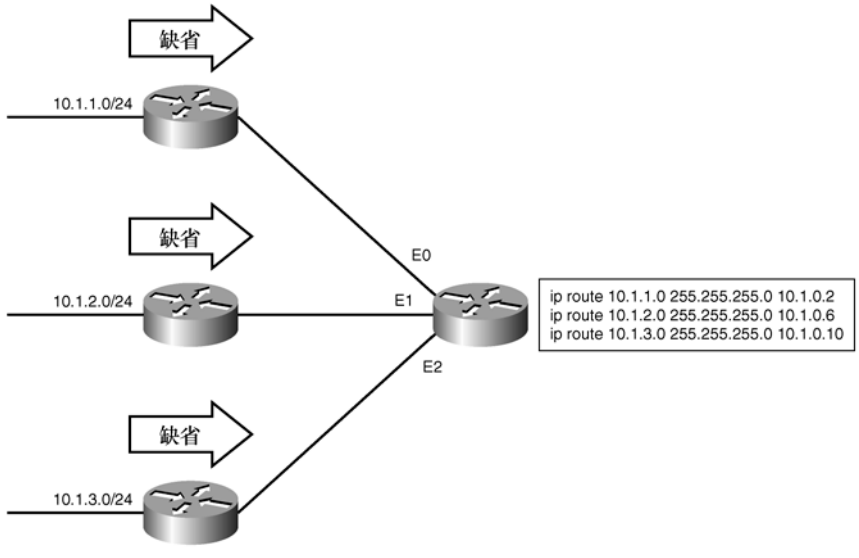


图 12-1 缺省路由大大地简化了星型拓扑网络的静态路由管理

在图 12-1 中，分支路由器更正确地应该被称为末梢路由器。末梢路由器到其他路由器仅存在一条连接。在这种设备上路由决策就变得非常简单：目标网络要么是路由器的直连网络（末梢网络）之一，要么经邻居路由器可达；而且如果这台邻居路由器是下一跳路由器的惟一选择，那么末梢路由器就不需要详细的路由表，一条缺省路由常常就足够了。

正如使用汇总路由一样，缺省路由也会造成路由细节的损失。例如在图 12-1 中的末梢路由器将不知道某个目标网络是否可达。所有去往未知目标网络的数据包都要被转发到中心路

由器，然后确定网络是否可达。在网络中，很少会发生向不存在的地址发送数据包的情况。但如果万一发生，那么更好的设计选择是让末梢路由器运行路由协议，并从中心路由器获取路由以便尽快地确定未知网络。对于像图 12-1 这样的网络，你的设计选择或者是把未知目标网络的数据包转发给中心路由器，由它负责丢弃，或者在末梢路由器和中心路由器之间运行动态路由选择协议，并且末梢路由器就地丢弃去往未知目标网络的数据包，这两种方式哪种更经济。虽然运行动态路由选择协议所需要的资源和运作代价通常是很小的，但是缺省路由依然更可能是最佳选择。

图 12-2 给出了路由细节损失所引起的另一个问题。这些路由器形成了一个全国性的骨干网络，而且把大量本地网络连接到这些骨干路由器上。在洛杉矶（Los Angeles）骨干路由器上存在指向旧金山（San Francisco）和圣地亚哥（San Diego）的缺省路由。如果洛杉矶必须要向西雅图（Seattle）转发数据包，而它仅有两条缺省路由，那么它无法得知经过旧金山才是最佳的路由。洛杉矶有可能会向圣地亚哥转发数据包，在这种情况下，数据包将使用部分非常昂贵的链路带宽，而且在数据包延迟到达目标网络之前将遭遇不必要的传播时延。虽然在骨干网中使用缺省路由是一个不好的设计决策，[1]但是却说明了使用缺省路由隐藏路由细节可能会导致不理想的路由选择。

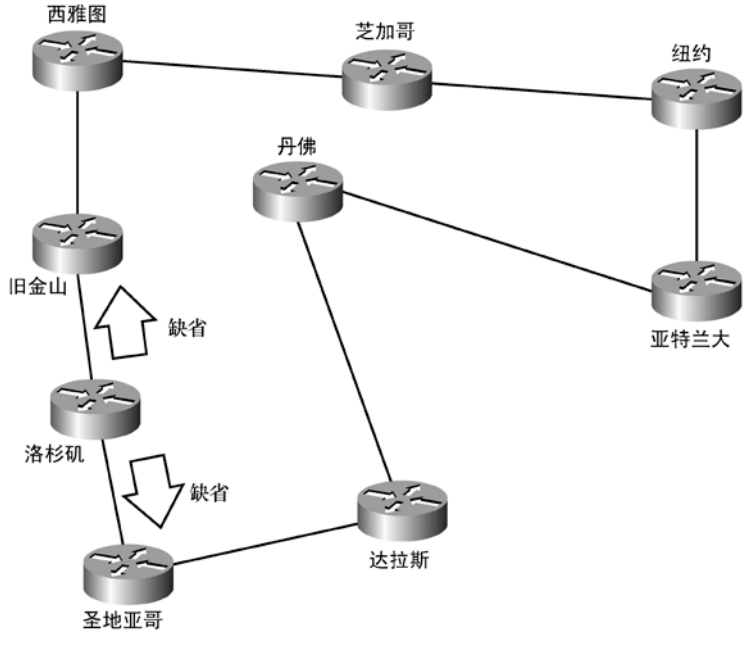


图 12-2 如果洛杉矶仅知道指向旧金山和圣地亚哥通告的缺省路由，而不知道这两台路由器后面网络拓扑的更多细节，那么它将不能够有效地进行路由选择

12.2 按需路由基本原理

如图 12-1 所示，虽然在中心路由器上配置静态路由非常简单，但是许多网络管理员仍然

[1] 另一方面，让每台骨干路由器仅向它的本地网络通告一条缺省路由将会是一个非常好的设计选择，这可以限制本地路由表的大小。

不喜欢使用静态路由。困难不在于每当新的末梢网络在线时需要添加路由，而是在末梢网络或末梢路由器离线时忘记删除路由。从 IOS 11.2 起，Cisco 开始向中心路由器提供另一种专有技术，叫做按需路由（On-Deman Routing，ODR）。

当末梢路由器仍然使用指向中心路由器的缺省路由时，中心路由器使用 ODR 可以自动地发现末梢网络。ODR 仅传送地址前缀，即地址的网络号，而不是整个地址，因此路由器必须支持 VLSM。由于在末梢路由器和中心路由器之间的链路上传输的路由信息非常少，所以节省了带宽。

ODR 不是真正意义上的路由选择协议。它可以发现有关末梢网络的信息，但是 ODR 不能向末梢路由器提供任何路由选择信息。链路信息通过数据链路协议进行传输，因而从末梢路由器到中心路由器后不会做进一步的传输。然而，在后面的案例研究中将会讨论，ODR 发现的路由可以被重新分配到动态路由选择协议中。

示例 12-1 给出了一个包含 ODR 表项的路由表，该表显示的管理距离为 160，路由的度量值为 1。因为 ODR 路由总是从中心路由器到末梢路由器，所以度量值（跳数）将永远不会超过 1。路由还显示出支持 VLSM。

示例 12-1 这个路由表显示了几个 ODR 表项

```
Router#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
    192.168.1.0/24 is variably subnetted, 3 subnets, 2 masks
o      192.168.1.40/30 [160/1] via 192.168.1.37, 00:00:27, Serial0
C      192.168.1.36/30 is directly connected, Serial0
C      192.168.1.192/27 is directly connected, Ethernet1
o   192.168.3.0/24 [160/1] via 192.168.1.37, 00:00:27, Serial0
    192.168.4.0/24 is variably subnetted, 2 subnets, 2 masks
o      192.168.4.48/29 [160/1] via 192.168.1.37, 00:00:27, Serial0
o      192.168.4.128/27 [160/1] via 192.168.1.37, 00:00:27, Serial0
Router#
```

ODR 路由的传输机制是 Cisco 发现协议（Cisco Discovery Protocol，CDP），CDP 是一种专用的数据链路协议，它可以收集有关邻居网络设备[1]的信息。示例 12-2 给出了 CDP 所收集信息的类型。

示例 12-2 CDP 收集有关邻居 Cisco 网络设备的信息

```
Bumble#show cdp neighbors detail
-------------------------
Device ID: P8R1
Entry address(es):
  IP address: 10.131.223.226
Platform: Cisco 2620,  Capabilities: Router
Interface: Serial0/0.708,  Port ID (outgoing port): Serial0/0.807
Holdtime : 144 sec

Version :
```

（待续）

[1] CDP 不仅可以运行在路由器上，而且还可以运行在 Cisco 的交换机和接入服务器上。

```
Cisco Internetwork Operating System Software
IOS (tm) C2600 Software (C2600-J1S3-M), Version 12.3(6), RELEASE SOFTWARE (fc3)
Copyright (c) 1986-2004 by Cisco Systems, Inc.
Compiled Wed 11-Feb-04 19:24 by kellythw

advertisement version: 2

-------------------------
Device ID: Blathers
Entry address(es):
  IP address: 192.168.3.2
Platform: cisco 2610,  Capabilities: Router
Interface: Serial0/0.1,  Port ID (outgoing port): Serial0/0.2
Holdtime : 122 sec

Version :
Cisco Internetwork Operating System Software
IOS (tm) C2600 Software (C2600-J1S3-M), Version 12.3(10a), RELEASE SOFTWARE (fc2
)
Copyright (c) 1986-2004 by Cisco Systems, Inc.
Compiled Fri 22-Oct-04 20:43 by kellythw

advertisement version: 2

-------------------------

Bumble#
-------------------------
```

CDP 运行在任何支持子网访问协议（SNAP）的介质上，这意味着 ODR 还依赖于 SNAP 的支持。虽然在所有运行 IOS 10.3 或更高版本 IOS 的 Cisco 设备的所有接口上，CDP 缺省是被启用的，但是从 IOS11.2 才开始支持 ODR。配置案例研究部分将会显示 ODR 仅能配置在中心路由器上，为了中心路由器能发现末梢路由器所连接的网络，末梢路由器必须运行 IOS 11.2 或更高版本的操作系统。

12.3 配置缺省路由和 ODR

缺省路由可以配置在每台需要缺省路由的路由器上，或者配置在向其对等路由器依次通告路由的路由器上。本节的案例研究将分析这两种方法。

回忆第 5 章讨论的有类别路由查询，路由器将首先匹配主网地址，然后匹配子网。如果子网不匹配，那么数据包将被丢弃。有类别路由查询是 Cisco 路由器 IOS 11.3 及后继版本的缺省模式。对于早期版本，可以通过命令 **ip classless** 把有类别查询变换到无类别（甚至对有类别路由选择协议）查询方式。

任何使用缺省路由的路由器必须执行无类别路由查询，图 12-3 解释了这样做的原因。在这个网络中，Memphis 同 Tanis、Giza 使用动态路由选择协议，但是 Memphis 并不从 Thebes 接收路由。为了向 BigNet 路由数据包，Memphis 有一条指向 Thebes 的缺省路由。如果 Memphis 接收到目的地址是 192.168.1.50 的数据包，并且它正在执行有类别路由查询，那么它将会首先匹配主网地址 192.168.1.0，在路由表中存在该主网地址的几个子网。接着 Memphis 试图寻找有关子网 192.168.1.48/28 的路由，但是因为 Memphis 并不从 Thebes 接收路由，所以该子网不在路由表中，因此数据包会被丢弃。

如果 Memphis 配置了 **ip classless**，那么它首先不会匹配主网络，而是为 192.168.1.48/28 寻找最精确的匹配。如果在路由表中没有发现，那么它将匹配到缺省路由，最终数据包被转发到 Thebes。

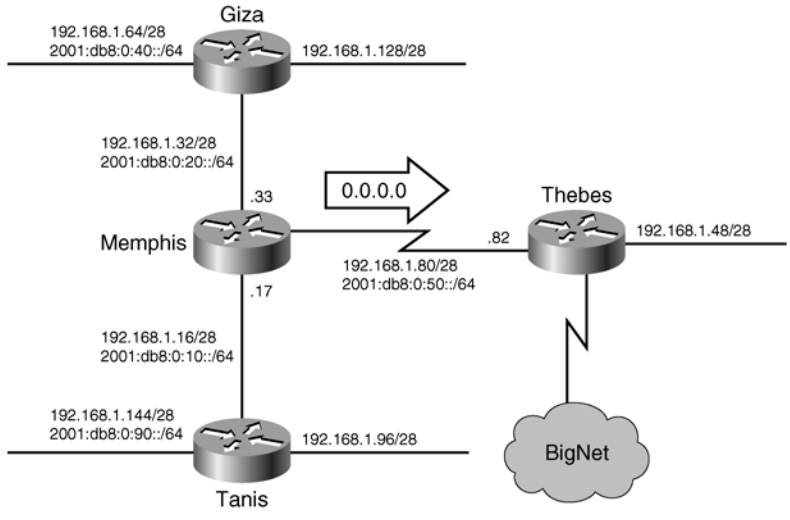


图 12-3 Memphis 使用缺省路由向 Thebes 转发数据包。如果 Memphis 使用有类别路由查询，子网 192.168.1.48/28 将不可达

12.3.1 案例研究：静态缺省路由

图 12-3 中 Memphis 的配置见示例 12-3。

示例 12-3 在 Memphis 上使用静态 IPv4 和 IPv6 路由建立缺省路由

```
interface serial 0/0.1
 ip address 192.168.1.33 255.255.255.240
 ipv6 address 2001:db8:0:20::1/64
 ipv6 rip egypt enable
!
interface serial 0/0.2
 ip address 192.168.1.81 255.255.255.240
 ipv6 address 2001:db8:0:50::1/64
 ipv6 rip egypt enable
!
interface serial 0/0.3
 ip address 192.168.1.17 255.255.255.240
 ipv6 address 2001:db8:0:10::1/64
 ipv6 rip egypt enable
!
router rip
network 192.168.1.0
!
ip classless
ip route 0.0.0.0 0.0.0.0 192.168.1.82
ipv6 route ::/0 2001:DB8:0:50::2
```

静态路由配置了缺省路由地址 0.0.0.0 和::/0，并且使用的掩码也是 0.0.0.0（对 IPv6 来说前缀长度为 0）。第一次配置缺省路由的人常犯的一个错误是使用全 1 掩码，而不是使用全 0 掩码，例如：

```
ip route 0.0.0.0 255.255.255.255 192.168.1.82
```

全 1 掩码将会设置一条指向 0.0.0.0 的主机路由，惟有那些目的地址为 0.0.0.0 的数据包才能匹配到该地址。另一方面，全 0 掩码全部是由“不关心”位组成，它可以在任意位置匹配到任意位。本章开头曾讨论过缺省地址是汇总地址的一种极端形式，即每一个位都会被 0 汇总。这里缺省路由的掩码也是汇总掩码的一种极端形式。

Memphis 缺省路由的下一跳地址在 Thebes 上。这个下一跳地址指的是最后可选网关或缺省路由器。示例 12-4 给出了 Memphis 的 IPv4 路由表。指向 0.0.0.0 的路由被标记为候选缺省，在表的开头指明了最后可选网关。示例 12-5 给出了 IPv6 的路由表。

示例 12-4 Memphis 的 IPv4 路由表，给出了缺省路由和最后可选网关

```
Memphis#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is 192.168.1.82 to network 0.0.0.0
    192.168.1.0/28 is subnetted, 7 subnets
R      192.168.1.96 [120/1] via 192.168.1.18, 00:00:15, Ethernet0
R      192.168.1.64 [120/1] via 192.168.1.34, 00:00:27, Ethernet1
C      192.168.1.80 is directly connected, Serial0
C      192.168.1.32 is directly connected, Ethernet1
C      192.168.1.16 is directly connected, Ethernet0
R      192.168.1.128 [120/1] via 192.168.1.34, 00:00:27, Ethernet1
R      192.168.1.144 [120/1] via 192.168.1.18, 00:00:15, Ethernet0
S*  0.0.0.0/0 [1/0] via 192.168.1.82
Memphis#
```

示例 12-5 Memphis 的 IPv6 给出了指向缺省地址::/0 的静态路由表项

```
Memphis#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
S   ::/0 [1/0]
     via 2001:DB8:0:50::2

C   2001:DB8:0:10::/64 [0/0]
     via ::, Serial0/0.3
L   2001:DB8:0:10::1/128 [0/0]
     via ::, Serial0/0.3
C   2001:DB8:0:20::/64 [0/0]
     via ::, Serial0/0.1
L   2001:DB8:0:20::1/128 [0/0]
     via ::, Serial0/0.1
R   2001:DB8:0:40::/64 [120/2]
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.1
C   2001:DB8:0:50::/64 [0/0]
     via ::, Serial0/0.2
L   2001:DB8:0:50::1/128 [0/0]
     via ::, Serial0/0.2
```

（待续）

```
R   2001:DB8:0:90::/64 [120/2]
     via FE80::205:5EFF:FE6B:50A0, Serial0/0.3
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
Memphis#
```

通过向 RIP 重新分配静态路由，可以把缺省路由通告给剩余的 RIP 路由器。但 Memphis 不会向 Tanis 和 Giza 通告缺省路由，除非静态路由被重新分配到 RIP 协议[1]。在示例 12-6 中向 Memphis 配置中添加了 IPv4 和 IPv6 的重新分配命令。

示例 12-6 向 Memphis 添加重新分配命令使得 RIP 通告静态缺省路由

```
router rip
redistribute static
!
ipv6 router rip egypt
redistribute static
```

虽然 OSPF 和 IS-IS 不使用命令 **redistribution** 通告缺省路由，但是它们也产生缺省路由，这可以在后继案例研究中看到。示例 12-7 和示例 12-8 分别给出了向 RIP 重新分配静态缺省路由后 Tanis 的 IPv4 和 IPv6 的路由表。

示例 12-7 Tanis 的 IPv4 路由表显示了缺省路由是通过 RIP 协议从 Memphis 那里学习到的

```
Tanis#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is 192.168.1.17 to network 0.0.0.0
     192.168.1.0/28 is subnetted, 9 subnets
C       192.168.1.96 is directly connected, Ethernet1
R       192.168.1.64 [120/2] via 192.168.1.17, 00:00:01, Ethernet0
R       192.168.1.80 [120/1] via 192.168.1.17, 00:00:01, Ethernet0
R       192.168.1.32 [120/1] via 192.168.1.17, 00:00:01, Ethernet0
R       192.168.1.48 [120/2] via 192.168.1.17, 00:00:01, Ethernet0
C       192.168.1.16 is directly connected, Ethernet0
R       192.168.1.224 [120/1] via 192.168.1.17, 00:00:01, Ethernet0
R       192.168.1.128 [120/2] via 192.168.1.17, 00:00:01, Ethernet0
C       192.168.1.144 is directly connected, Ethernet2
R*    0.0.0.0/0 [120/1] via 192.168.1.17, 00:00:02, Ethernet0
Tanis#
```

示例 12-8 Tanis 的 IPv6 路由表显示了缺省路由是通过 IPv6 RIP 协议从 Memphis 那里学习到的

```
       Tanis#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
```

（待续）

[1] 在 IOS12.0T 之前的版本中，如果路由表中存在缺省路由，则不需要向 RIP、IGRP 和 EIGRP 重新分配静态路由，它们就会自动向邻居通告。

```
        O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
        ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
R   ::/0 [120/2]
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
C   2001:DB8:0:10::/64 [0/0]
     via ::, Serial0/0.1
L   2001:DB8:0:10::2/128 [0/0]
     via ::, Serial0/0.1
R   2001:DB8:0:20::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
R   2001:DB8:0:40::/64 [120/3]
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
R   2001:DB8:0:50::/64 [120/2]
     via FE80::204:C1FF:FE50:E700, Serial0/0.1
C   2001:DB8:0:90::/64 [0/0]
     via ::, FastEthernet0/0
L   2001:DB8:0:90::1/128 [0/0]
     via ::, FastEthernet0/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
Tanis#
```

缺省路由对于连接无类别路由选择域也是非常有用的，在图 12-4 中，Chimu 将一个 RIP 域和一个 EIGRP 域连接到一起。虽然在 RIP 域主网 192.168.25.0 的掩码是一致的，但是在 EIGRP 域对该主网进行了变长子网划分。此外，VLSM 方式对进入 RIP 的汇总没有任何帮助。

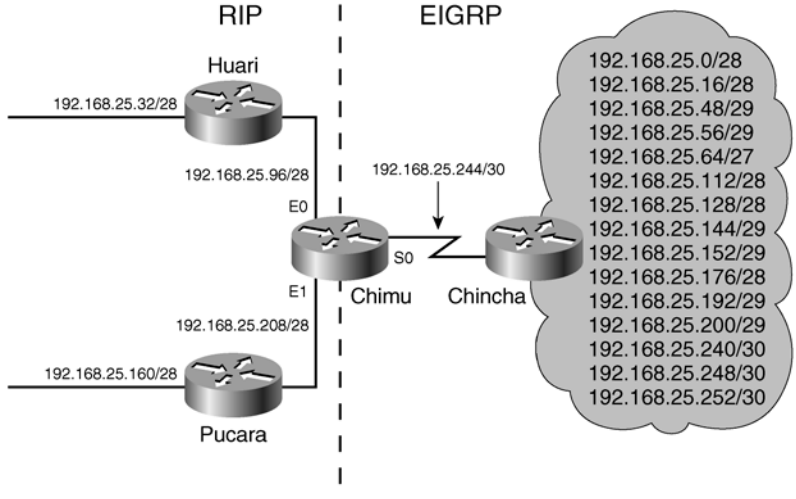


图 12-4 缺省路由使 RIP 可以路由到变长子网化的 EIGRP 域

Chimu 的配置见示例 12-9。

示例 12-9 Chimu 把 RIP 路由重新分配到 EIGRP，但除了缺省路由外所有 EIGRP 路由都没有被重新分配到 RIP 域

```
router eigrp 1
redistribute rip metric 1000 100 255 1 1500
passive-interface Ethernet0
passive-interface Ethernet1
network 192.168.25.0
```

（待续）

```
!
router rip
passive-interface Serial0
network 192.168.25.0
redistribute static
!
ip classless
ip route 0.0.0.0 0.0.0.0 Null0
```

Chimu 有一套来自 EIGRP 域的完整路由，但是 Chimu 没有将它们重新分配到 RIP。相反，Chimu 仅通告了一条缺省路由。RIP 路由器将向 Chimu 转发所有去往未知网络的数据包，然后 Chimu 查找路由表，寻找一条到 EIGRP 域的最精确的路由。

Chimu 的静态路由不是指向下一跳地址，而是指向空接口。如果转发给 Chimu 的数据包的目标属于一个不存在的子网，例如 192.168.25.224/28，那么数据包不会被转发到 EIGRP 域，而是被丢弃。

12.3.2 案例研究：缺省网络命令

配置缺省路由的另一种方法是使用命令 **ip default-network**。该命令指明了用作缺省网络的网络地址。这个网络可以是由静态路由指定的直连网络，也可以是通过动态路由选择协议发现的网络。开始介绍这个命令是和 IGRP 一起使用的，IGRP 不用 0.0.0.0 作为缺省路由，而是把实际的网络标记为缺省网络。该命令仅用于 IGRP、EIGRP 和 RIP。

命令 **ip default-network** 是一条全局命令，所以它会导致在支持这个命令的路由器上配置的所有路由协议都通告一条缺省路由。如果协议是 IGRP 和 EIGRP，那么缺省路由就是命令参数指定的网络，如果是 RIP，则是 0.0.0.0。

在图 12-5 中，Athens 的配置中使用了 RIP 和命令 **ip default-network**，具体配置见示例 12-10。

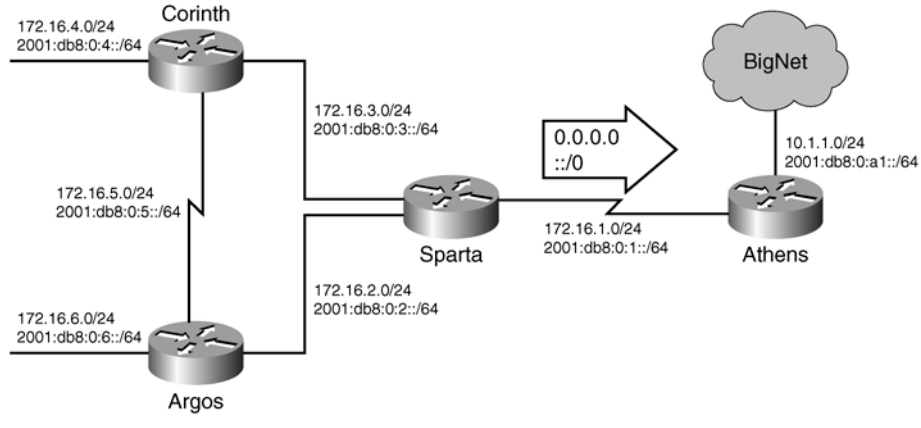


图 12-5 在 Athens 上使用命令 **default-network** 可以产生一个缺省网络通告

示例 12-10 RIP 使用命令 default-network 建立缺省路由

```
router rip
network 172.16.0.0
!
ip classless
ip default-network 10.0.0.0
```

如示例 12-11 所示，在 Athens 的路由表中网络 10.0.0.0 被标记为候选缺省路由，但是注意，没有指定最后可选网关，原因是 Athens 是到缺省网络的网关。即使在 RIP 配置中不存在有关网络 10.0.0.0 的语句（参见示例 12-12），**ip default-nework** 也将导致 Athens 通告一个缺省网络。当使用 RIP 时，Athens 上配置的命令 **ip default-nework** 会使 Athens 把 0.0.0.0 作为缺省路由进行通告，而不是命令 **ip default-nework** 指定的网络。

示例 12-11 在 Athens 的路由表中网络 10.0.0.0 被标记作为候选缺省路由

```
Athens#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
* 10.0.0.0/8 is subnetted, 1 subnets
C 10.1.1.0 is directly connected, Ethernet0
  172.16.0.0/16 is subnetted, 6 subnets
R 172.16.4.0 [120/2] via 172.16.1.2, 00:00:12, Serial0
R 172.16.5.0 [120/2] via 172.16.1.2, 00:00:12, Serial0
R 172.16.6.0 [120/2] via 172.16.1.2, 00:00:12, Serial0
C 172.16.1.0 is directly connected, Serial0
R 172.16.2.0 [120/1] via 172.16.1.2, 00:00:12, Serial0
R 172.16.3.0 [120/1] via 172.16.1.2, 00:00:12, Serial0
Athens#
```

示例 12-12 Sparta 的路由表显示出 Athens 正在通告一条缺省路由 0.0.0.0，而且 Athens 是 Spartade 的最后可选网关

```
Sparta#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is 172.16.1.1 to network 0.0.0.0
  172.16.0.0/24 is subnetted, 6 subnets
R   172.16.4.0 [120/1] via 172.16.3.2, 00:00:14, Ethernet1
R   172.16.5.0 [120/1] via 172.16.3.2, 00:00:14, Ethernet1
R   172.16.6.0 [120/1] via 172.16.2.2, 00:00:10, Ethernet0
C   172.16.1.0 is directly connected, Serial0
C   172.16.2.0 is directly connected, Ethernet0
C   172.16.3.0 is directly connected, Ethernet1
R* 0.0.0.0/0 [120/1] via 172.16.1.1, 00:00:17, Serial0
Sparta#
```

和 RIP 一样，如果静态路由 0.0.0.0 被配置，那么 EIGRP 将向邻居通告一条缺省路由，同时还会重新分配静态路由。正如第 7 章所讨论的，EIGRP 会把被重新分配的路由作为外部路由进行通告。

如果配置图 12-5 中的路由器运行 EIGRP，并且使用命令 **ip default-network**，那么 Athen 的配置见示例 12-13。

示例 12-13 命令 **default-network** 和 EIGRP 一起使用可以把一个网络标记为候选缺省路由

```
router eigrp 1
network 10.0.0.0
network 172.16.0.0
!
ip classless
ip default-network 10.0.0.0
```

命令 **ip default-network** 保持不变，但是注意，在 EIGRP 的配置中添加了关于网络 10.0.0.0 的语句。因为 EIGRP 使用真实网络地址作为缺省网络，所以必须对该地址进行配置，详见示例 12-14。比较示例 12-12 和示例 12-14 的路由表可以发现，RIP 把指向 0.0.0.0 的路由标记为缺省路由，而 EIGRP 则把指向 10.0.0.0/8 的路由标记为缺省路由。因为 Corinth 从 Aparta 那里学习到缺省路由，所以 Sparta 是 Corinth 的最后可选网关。如果到 Sparta 的链路发生故障，那么 Corinth 将使用 Argos 作为最后可选网关。

示例 12-14 EIGRP 使用真实网络地址，而不是 0.0.0.0，作为缺省网络。Corinth 的路由表显示出网络 10.0.0.0 被标记为缺省网络

```
Corinth#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is 172.16.3.1 to network 10.0.0.0
D*      10.0.0.0/8 [90/2195456] via 172.16.3.1, 00:02:32, Ethernet0
   172.16.0.0/16 is subnetted, 6 subnets
C     172.16.4.0 is directly connected, Ethernet1
C     172.16.5.0 is directly connected, Serial0

D       172.16.1.0 [90/1811456] via 172.16.3.1, 00:00:17, Ethernet0
D       172.16.6.0 [90/921600] via 172.16.3.1, 00:00:16, Ethernet0
D       172.16.2.0 [90/793600] via 172.16.3.1, 00:00:16, Ethernet0
C     172.16.3.0 is directly connected, Ethernet0
```

注意在 Athens 的配置中，命令 **ip default-network** 是一个全局命令，它不与特殊的路由选择协议相关联，也就是路由器上配置的任何路由选择协议都可以使用它，但前提是路由器要支持这条命令。如果路由器上配置了 RIP 和 EIGRP，那么它们都会通告一条缺省路由，这可以在示例 12-15 中 Conrinth 的路由表中看到。

示例 12-15 当 Athens 配置了 RIP 和 EIGRP，并且使用命令 ip default-netowrk，从 Corinth 的路由表可以看到有两条候选缺省路由

```
Corinth#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is 172.16.3.1 to network 10.0.0.0
D*      10.0.0.0/8 [90/2195456] via 172.16.3.1, 00:02:32, Ethernet0
   172.16.0.0/16 is subnetted, 6 subnets
C     172.16.4.0 is directly connected, Ethernet1
C     172.16.5.0 is directly connected, Serial0
```

（待续）

```
D       172.16.1.0 [90/1811456] via 172.16.3.1, 00:00:17, Ethernet0
D       172.16.6.0 [90/921600] via 172.16.3.1, 00:00:16, Ethernet0
D       172.16.2.0 [90/793600] via 172.16.3.1, 00:00:16, Ethernet0
C     172.16.3.0 is directly connected, Ethernet0
R* 0.0.0.0/0 [120/1] via 172.16.3.1, 00:00:17, Serial0
```

因为 EIGRP 的关联距离更小，所以来源于 EIGRP 的缺省网络成为最后可选网关。

使用这种方法通告缺省路由有一个固有的缺陷。如果路由器上配置了多个协议，例如 RIP 和 EIGRP，使用命令 **ip default-newwork**，那么就没有办法控制和限制由谁来通告缺省网络。在如图 12-5 中，如果 Athens 为 BigNet 运行 EIGRP，而为余下的网络运行 RIP，配置 **ip default-network** 的目的是向 RIP 通告缺省路由，同时 Athens 也向 EIGRP 通告缺省路由。这样做不仅会使从 RIP 域去往 Bignet 的流量的路由发生中断，而且还会影响 Bignet 内部的路由。

当向一个路由选择协议注入路由时，最好是选择一个可以提供最多控制的方法来最小化无目的路由的扩散。

12.3.3 案例研究：缺省信息始发命令

OSPF 的 ASBR 和 IS-IS 域间路由器不能自动地向它们的路由选择域通告缺省路由，即使在缺省路由存在的时候也一样。例如，假设图 12-5 中的 Athens 配置了 OSPF，并且设置了一条指向 BigNet 的缺省路由，Athens 的配置见示例 12-16。

示例 12-16 Athens 配置了 OSPF，并且还有一条静态缺省路由

```
router ospf 1
network 172.16.0.0 0.0.255.255 area 0
!
ip classless
ip route 0.0.0.0 0.0.0.0 10.1.1.2
```

示例 12-17 给出了 Athens 和 Sparta 的路由表。虽然静态路由使得在 Athen 上设置了最后可选网关，但是 Sparta 却不知道缺省路由。缺省路由必须以类型 5 的 LSA 方式被通告到 OSPF 域，这意味着 Athens 必须是一个 ASBR。然而到目前为止，Athens 的配置并没有告诉它执行这些功能。

示例 12-17 在 Athens 的 OSPF 进程不能向 OSPF 域自动通告缺省路由

```
Athens#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is 10.1.1.2 to network 0.0.0.0
    10.0.0.0 255.255.255.0 is subnetted, 1 subnets
C    10.1.1.0 is directly connected, Ethernet0
    172.16.0.0 is variably subnetted, 6 subnets, 2 masks
O    172.16.5.0 255.255.255.0 [110/138] via 172.16.1.2, 00:04:17, Serial0
O    172.16.4.1 255.255.255.0 [110/75] via 172.16.1.2, 00:04:17, Serial0
O    172.16.6.1 255.255.255.0 [110/75] via 172.16.1.2, 00:04:17, Serial0
C    172.16.1.0 255.255.255.0 is directly connected, Serial0
O    172.16.2.0 255.255.255.0 [110/74] via 172.16.1.2, 00:04:17, Serial0
O    172.16.3.0 255.255.255.0 [110/74] via 172.16.1.2, 00:04:17, Serial0
S* 0.0.0.0 0.0.0.0 [1/0] via 10.1.1.2
```

（待续）

```
Sparta#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
   172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
O     172.16.5.0/24 [110/74] via 172.16.2.2, 00:06:00, Ethernet1
                    [110/74] via 172.16.3.2, 00:06:00, Ethernet0
O     172.16.4.1/24 [110/11] via 172.16.3.2, 00:06:00, Ethernet0
O     172.16.6.1/24 [110/11] via 172.16.2.2, 00:06:00, Ethernet1
C     172.16.1.0/24 is directly connected, Serial0
C     172.16.2.0/24 is directly connected, Ethernet1
C     172.16.3.0/24 is directly connected, Ethernet0
```

命令 **default-informaiton originate** 是重新分配命令的一个特例，它将导致一条缺省路由被重新分配到 OSPF 或 IS-IS。同 **redistribute** 一样，命令 **default-informaiton originate** 通知 OSPF 路由器它成为一个 ASBR 或通知 IS-IS 路由器它成为一个域间路由器。而且还像 **redistribute** 一样指明被重新分配的缺省路由的度量，可以是 OSPF 外部度量类型，或者是 IS-IS 层级。为了向 OSPF 重新分配缺省路由，并且要求缺省路由的度量值为 10，外部度量类型为 E1，示例 12-18 给出了 Athens 的配置。

示例 12-18 在 Athens 上命令 **default-information originate** 用于产生一条缺省路由

```
router ospf 1
network 172.16.0.0 0.0.255.255 area 0
default-information originate metric 10 metric-type 1
!
ip classless
ip route 0.0.0.0 0.0.0.0 10.1.1.2
```

示例 12-19 给出的缺省路由正在被重新分配到 OSPF。而且在 Sparta 的 OSPF 数据库（参见示例 12-20）中也可以观察到这个路由。

示例 12-19 在 Athens 上配置 **default-information originate** 之后，缺省路由将被重新分配到 OSPF 域

```
Sparta#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is 172.16.1.1 to network 0.0.0.0
    172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
O      172.16.5.0/24 [110/74] via 172.16.2.2, 00:14:46, Ethernet0
O      172.16.4.1/32 [110/75] via 172.16.2.2, 00:14:46, Ethernet0
O      172.16.6.1/32 [110/11] via 172.16.2.2, 00:14:46, Ethernet0
C      172.16.1.0/24 is directly connected, Serial0
C      172.16.2.0/24 is directly connected, Ethernet0
C      172.16.3.0/24 is directly connected, Ethernet1
O* E1 0.0.0.0/0 [110/74] via 172.16.1.1, 00:02:55, Serial0
Sparta#
```

示例 12-20 像 ASBR 通告的其他外部路由一样，缺省路由以类型 5 的 LSA 方式被通告

```
Sparta#show ip ospf database external
       OSPF Router with ID (172.16.3.1) (Process ID 1)
               Type-5 AS External Link States
Routing Bit Set on this LSA
LS age: 422
Options: (No TOS-capability, No DC)
LS Type: AS External Link
Link State ID: 0.0.0.0 (External Network Number )
Advertising Router: 172.16.1.1
LS Seq Number: 80000002
Checksum: 0x5238
Length: 36
Network Mask: /0
       Metric Type: 1 (Comparable directly to link state metric)
       TOS: 0
       Metric: 10
       Forward Address: 0.0.0.0
       External Route Tag: 1
Sparta#
```

命令 **default-information originate** 还可以向 OSPF 和 IS-IS 重新分配被其他路由选择进程发现的缺省路由。在示例 12-21 中，指向网络 0.0.0.0 的静态路由被删除，而 Athens 与 BigNet 中的一台路由器使用 BGP 通信。

示例 12-21 配置 Athens 使用 BGP 获取路由，而不再使用静态路由

```
router ospf 1
network 172.16.0.0 0.0.255.255 area 0
default-information originate metric 10 metric-type 1
!
router bgp 65501
network 172.16.0.0
neighbor 10.1.1.2 remote-as 65502
!
ip classless
```

现在 Athens 从它的 BGP 邻居路由器那里学习到指向 0.0.0.0 的路由，并且使用类型 5 的 LSA 向 OSPF 域通告该路由（参见示例 12-22）。

示例 12-22 在 BigNet 中使用 BGP 的路由器向 Athens 通告缺省路由

```
Athens#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is 10.1.1.2 to network 0.0.0.0
    10.0.0.0/8 is subnetted, 1 subnets
C       10.1.1.0 is directly connected, Ethernet0
    172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
O IA    172.16.4.1/32 [110/139] via 172.16.1.2, 00:16:45, Serial0
O IA    172.16.5.0/24 [110/138] via 172.16.1.2, 00:16:45, Serial0
O IA    172.16.6.1/32 [110/75] via 172.16.1.2, 00:16:45, Serial0
C       172.16.1.0/24 is directly connected, Serial0
O IA    172.16.2.0/24 [110/74] via 172.16.1.2, 00:16:45, Serial0
O IA    172.16.3.0/24 [110/74] via 172.16.1.2, 00:16:45, Serial0
B* 0.0.0.0/0 [20/0] via 10.1.1.2, 00:12:02
Athens#
```

缺省路由或汇总路由的好处是提供网络的稳定性，但是如果缺省路由自身不稳定会发生

什么呢？例如在示例 12-19 中，假设通告给 Athens 的缺省路由在波动，也就是频繁地在可达与不可达之间变换。伴随着每一个变化，Athens 都必须向 OSPF 域发送一条新的类型 5 的 LSA，这个 LSA 将会被通告到所有非末梢区域。虽然这种泛洪和再泛洪对系统资源影响不大，但是，这不是网络管理员所期望的。解决办法是使用关键字 **always**。[1]采用示例 12-23 的配置方法，即使路由表中没有缺省路由，Athens 也总是会产生一条缺省路由。

示例 12-23 即使路由表中没有缺省路由，Athens 也将总是产生一条缺省路由

```
router ospf 1
network 172.16.0.0 0.0.255.255 area 0
default-information originate always metric 10 metric-type 1
!
router bgp 65501
network 172.16.0.0
neighbor 10.1.1.2 remote-as 65502
!
ip classless
```

使用这种配置方法，Athens 将始终通告一条缺省路由到 OSPF，不管它实际上是否有一条指向 0.0.0.0 的路由。如果在 OSPF 域内的一台路由器把 Athens 作为出口并转发了一个数据包，而 Athens 没有缺省路由，那么它将向源地址发送目标不可达的 ICMP 信息并且丢弃该数据包。

当 OSPF 域外仅有单一的缺省路由时，关键字 **always** 可以安全地被使用。如果不止一个 ASBR 通告了缺省路由，那么缺省出口应该是动态的——即缺省路由的丢失将会被通告。如果一个 ASBR 在它没有缺省路由时却声明有缺省路由，那么数据包将会被转发到它那里，而不是被转发到合理的 ASBR。

对于 IPv6，**defualt-informaitoin originate** 的工作方式很类似。在图 12-5 中，IPv6 由 IS-IS 进行路由。通告配置 Athens 将为 IPv6 产生一条缺省路由。

Athens 的配置见示例 12-24。

示例 12-24 Athens 为 IS-IS 协议产生一条 IPv6 的缺省路由

```
ipv6 unicast-routing
interface Ethernet0
 ip address 10.1.1.1 255.255.255.0
 ipv6 address 2001:DB8:0:A1::1/64
 ipv6 router isis
!
interface Serial0
 ip address 172.16.1.1 255.255.255.0
 ip router isis
 ipv6 address 2001:DB8:0:1::1/64
 ipv6 router isis
!
router isis
 net 01.0000.00ef.5678.00
 metric-style wide
 address-family ipv6
  multi-topology
  default-information originate
  exit-address-family
```

在向 IS-IS 数据库输入缺省路由并向其他邻居通告缺省路由之前，Athens 不需要从别的信息源那里获取缺省路由。其他路由器会把所有去往未知 IPv6 地址的数据包都转发给

[1] 这个关键字仅在 OSPF 下可用，在 IS-IS 下不支持。

Athens。如果 Athens 也没有路由，那么它就丢弃这些数据包。示例 12-25 是关于 Argos 上的 Athens 的第 2 层 IS-IS 数据库表项；示例 12-26 是 Argos 的 IPv6 路由表。

示例 12-25 IPv6 缺省路由被添加到 L2 的 IS-IS 数据库中

```
Argos#show isis database detail level-2 Athens.00-00
IS-IS Level-2 LSP Athens.00-00
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Athens.00-00          0x00000088   0xBD29        956               0/0/0
  Area Address: 01
  Topology:     IPv4 (0x0) IPv6 (0x2)
  NLPID:        0xCC 0x8E
  Hostname: Athens
  IP Address:   172.16.1.1
  IPv6 Address: 2001:DB8:0:A1::1
  Metric: 10         IS-Extended Athens.01
  Metric: 10         IS (MT-IPv6) Athens.01
  Metric: 10         IP 172.16.1.0/24
  Metric: 0          IPv6 (MT-IPv6) ::/0
  Metric: 10         IPv6 (MT-IPv6) 2001:DB8:0:1::/64
  Metric: 20         IPv6 (MT-IPv6) 2001:DB8:0:2::/64
  Metric: 20         IPv6 (MT-IPv6) 2001:DB8:0:3::/64
  Metric: 30         IPv6 (MT-IPv6) 2001:DB8:0:4::/64
  Metric: 30         IPv6 (MT-IPv6) 2001:DB8:0:5::/64
  Metric: 30         IPv6 (MT-IPv6) 2001:DB8:0:6::/64
  Metric: 30         IPv6 (MT-IPv6) 2001:DB8:0:20::/64
  Metric: 10         IPv6 (MT-IPv6) 2001:DB8:0:A1::/64
Argos#
```

示例 12-26 IPv6 缺省路由被作为 L2 的 IS-IS 添加到 IPv6 路由表中

```
Argos#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
I2  ::/0 [115/20]
     via FE80::204:C1FF:FE50:E700, FastEthernet0/1
I1  2001:DB8:0:1::/64 [115/20]
     via FE80::204:C1FF:FE50:E700, FastEthernet0/1
C   2001:DB8:0:2::/64 [0/0]
     via ::, FastEthernet0/1
L   2001:DB8:0:2::2/128 [0/0]
     via ::, FastEthernet0/1
I1  2001:DB8:0:3::/64 [115/20]
     via FE80::204:C1FF:FE50:E700, FastEthernet0/1
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.2
I1  2001:DB8:0:4::/64 [115/20]
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.2
C   2001:DB8:0:5::/64 [0/0]
     via ::, Serial0/0.2
L   2001:DB8:0:5::1/128 [0/0]
     via ::, Serial0/0.2
C   2001:DB8:0:6::/64 [0/0]
     via ::, FastEthernet0/0
L   2001:DB8:0:6::1/128 [0/0]
     via ::, FastEthernet0/0
I1  2001:DB8:0:20::/64 [115/20]
     via FE80::204:C1FF:FE50:F1C0, Serial0/0.2
```

（待续）

```
I1  2001:DB8:0:A1::/64 [115/30]
     via FE80::204:C1FF:FE50:E700, FastEthernet0/1
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
Argos#
```

12.3.4 案例研究：配置按需路由选择

使用命令 **router odr** 可以启用 ODR，不需要指明网络和其他参数。CDP 缺省情况下是被启用的，仅在因某种原因被关闭的情况下才需要被启用。在路由器上启用 CDP 进程的命令是 **cdp run**。为了在特定接口上启用 CDP，要使用命令 **cdp enable**。

图 12-6 给出了一个典型的星型拓扑结构。为了配置 ODR，中心路由器必须配置命令 **router odr**。只要所有路由器都运行 IOS 11.2 或更高版本，并且连接介质支持 SNAP（例如帧中继或 PVC），ODR 就可以运行，而且中心路由器将会学习到末梢网络。在末梢路由器上惟一需要配置的是一条指向中心路由器的静态缺省路由。

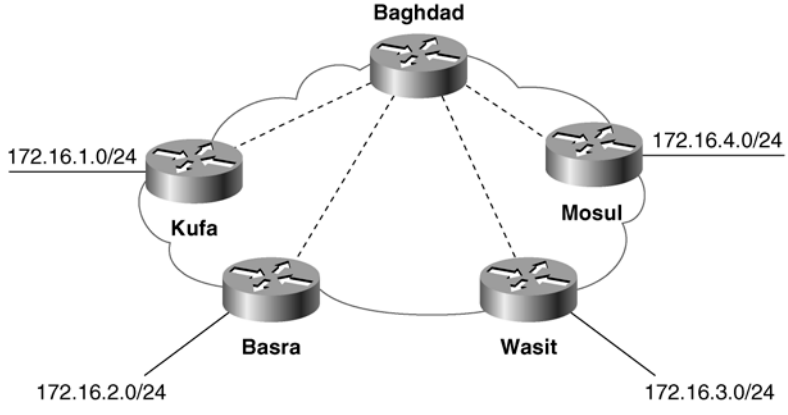


图 12-6 向这样的星型拓扑结构在帧中继网络中很普遍

ODR 还可以被重新分配。在图 12-6 中，如果 Baghdad 需要向 OSPF 通告 ODR 发现的路由，可以使用示例 12-27 的配置。

示例 12-27 ODR 发现的路由可以被重新分配到其他 IP 路由选择协议中

```
router odr
!
router ospf 1
redistribute odr metric 100
network 172.16.0.0 0.0.255.255 area 5
```

12.4 展 望

在简单且无环路网络中，本章讨论的配置和故障诊断缺省路由是一种微不足道的任务。当拓扑结构更加复杂，特别是包含环路时，由缺省路由和重新分配所带来的潜在问题将会增

加。第 13 章和第 14 章将讨论在复杂拓扑结构中控制路由行为的重要工具。

12.5 总结表：第 12 章命令总结

命令	描述
cdp enable	在接口上启用 CDP
cdp run	在路由器上全局启用 CDP
default-information originate [**always**][**metric** *metric-value*] [**metric-type** *type-value*]{**level-1**|**level-1-2** |**level-2**}[**route-map** *map-name*]	向 OSPF 和 IS-IS 路由选择域输入缺省路由
ip classless	启用无类别路由查询，以便路由器可以向直连网络的未知子网转发数据包
ip default-network *network-number*	在确定最后可选网关时指定一个网络作为候选路由
ip route *address mask* {*address*|*interface*}[*distance*][**tag** *tag*][**permanent**]	指定缺省路由表项
router odr	启用按需路由选择

12.6 复 习 题

1．开放协议使用的 IPv4 缺省路由的目标地址是什么？
2．IPv6 缺省路由的目标前缀/前缀长度是什么？
3．EIGRP 如何通告和标识缺省路由？
4．在运行 EIGRP 的路由器上可以使用指向 0.0.0.0 的静态路由作为缺省路由吗？
5．什么是末梢路由器？什么是末梢网络？
6．使用缺省路由代替完整的路由表的好处是什么？
7．使用完整的路由表代替缺省路由的好处是什么？
8．按需路由使用什么样的数据链路协议来发现路由？
9．在使用 ODR 时，对 IOS 有什么限制？
10．在使用 ODR 时，对介质有什么限制？

本章包括以下主题：

- 配置路由过滤器。

第 13 章

路由过滤

第 11 章中曾提到过在特殊路由器上，重新分配可能会在几种情况下导致不必要或不正确的路由。例如在图 11-3 及相关讨论中，一台或多台路由器选择了一条经过网络的非最佳路由，问题主要出在路由器更加信任 IGRP，因为 IGRP 的管理距离比 RIP 要小。更加普遍的是，任何时间指向相同目标网络的路由都会被多台路由器重新分配到路由选择域，其中可能会存在错误的路由选择。在某些情况下，可能会发生路由选择环路和黑洞。

示例 11-17 给出了另一个关于不必要路由或意外路由的例子。在这个案例中，不仅汇总路由 192.168.3.128/25 被通告到 OSPF，而且该路由还被重新分配到 EIGRP 域，然而 EIGRP 域即为被汇总子网的所在地。这种被通告路由沿错误方向穿过重新分配路由器的现象叫做路由回馈（route feedback）。

路由过滤可以使网络管理员对路由通告施加严格的控制。任何时刻路由器从一种协议向另一种协议重新分配路由，管理员可以使用路由过滤控制哪些路由被重新分配；同样的，路由器执行相互重新分配——在两个或多个路由选择协议之间相互共享路由——使用路由过滤器可以确保沿着惟一的方向通告路由。

图 13-1 给出了路由过滤器的另一种用途，在这里，一个路由选择域被分割为多个子域，每个子域包含多台路由器。

连接两个子域的路由器将对路由进行过滤，以便子域 B 中的路由器仅知道子域 A 中的部分路由。这种过滤可能是处于安全考虑，以便 B 域中的路由器仅知道已被授权的子网；或者仅仅是通过减少不必要的路由来维持 B 域内路由器的路由表和更新信息的大小。

此外，路由过滤器的另一个常见用途是建立路由防火墙。公司企业或政府机关常常需要被互连在一起，然而它们却处于独立的管理控制之下。如果你不能控制网络所有部分，

那么你很容易受到错误配置的影响，甚至恶意路由的攻击。如果在互连路由器上使用路由过滤，那么将确保路由器仅接收合法的路由。这种方法是一种安全的形式，但是在这种情况下，管制的是出站路由，而不是入站路由。

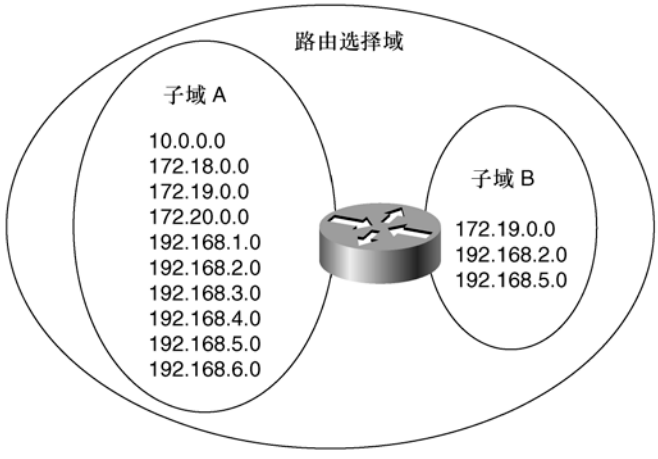


图 13-1 路由过滤器可以用于建立路由子域，以便仅向子域通告路由选择域内的部分地址

无论哪一种应用，路由过滤器都作为基本构建单元，被用于创建路由选择策略（routing policy）。路由选择策略是控制网络中数据包如何转发以及改变数据包缺省转发属性的一组规则。

外部的路由可以进入到路由表中，路由表中的路由也可以被通告出去，那么路由过滤器正是通过管制这些出入路由表的路由来工作的。路由过滤器对链路状态路由选择协议的影响和对距离矢量路由选择协议的影响稍微有点不同。运行距离矢量协议的路由器是基于自身路由表通告路由的，其结果是路由过滤器将会对路由器通告给其邻居路由器的路由产生影响。

另一方面，运行链路状态协议的路由器是基于自身链路状态数据库的信息来确定它们的路由，而不是基于被邻居路由器通告的路由条目。路由过滤器对链路状态的通告或链路状态数据库 [1]没有影响。所以路由过滤器会对配置了过滤器的路由器的路由表产生影响，但不会对邻居路由器的路由条目有任何影响。正因为这种特性，路由过滤器主要被用在进入链路状态域的重新分配点上，例如 OSPF 的 ASBR（自主系统边界路由器），在那里路由过滤器可以控制那些进入或离开该域的路由。在链路状态域内，路由过滤器的效用是有限的。

13.1 配置路由过滤器

使用下面所给出的任意一种方法都可以实现路由过滤：

- 使用命令 **distribute-list** 过滤特定路由；
- 使用命令 **distance** 操作路由的管理距离。

13.1.1 案例研究：过滤特定路由

图 13-2 显示出一部分网络运行 RIPv2 和 RIPng。Barkis 经 Traddles 提供了到网络其余部

[1] 请记住，链路状态协议的基本要求是，一个区域内的所有路由器必须具有一致的链路状态数据库。如果路由过滤器阻挡了一些 LSA，那么将违反上面的要求。

分的连接。除了 BigNet 内 700 个明确的 IPv4 路由之外，Traddles 还向 Barkis 通告了一条 IPv4 缺省路由。由于这条缺省路由，Barkis、Micawber、Peggotty 和 Heep 不需要知道 BigNet 中 700 条以外的路由。因此，在 Barkis 上配置过滤器的目的是从 Traddles 仅接收缺省路由，并拒绝其他所有路由。Barkis 的配置见示例 13-1。

示例 13-1 在 Barkis 上配置路由过滤器，允许缺省路由进入，拒绝所有其他地址

```
router rip
version 2
network 192.168.75.0
distribute-list 1 in Serial1
!
ip classless
access-list 1 permit 0.0.0.0
```

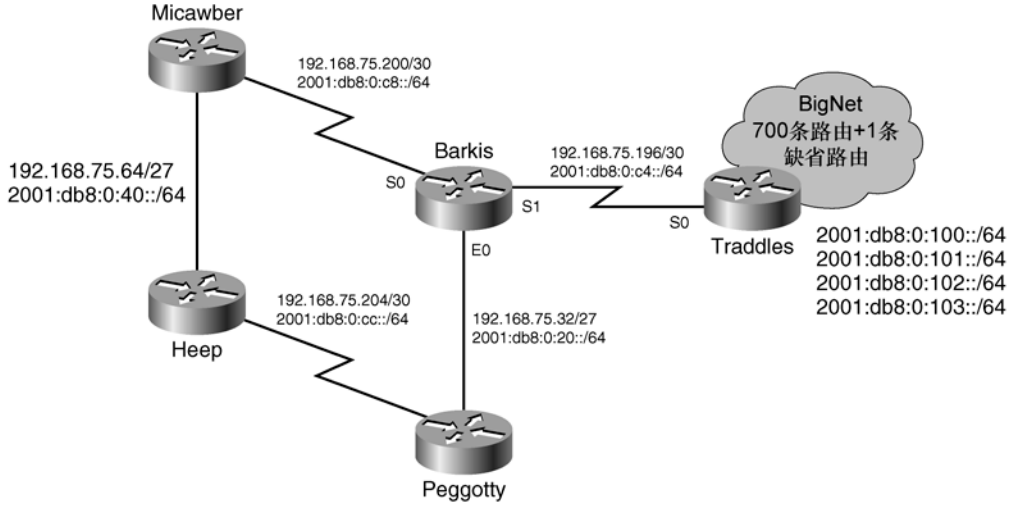


图 13-2 在 Barkis 上，路由过滤器仅接收来自 Traddles 的缺省路由，并拒绝 BigNet 所有其他路由

该路由过滤器检查从接口 S1 入站的路由，其中 S1 是连接 Traddles 的接口。路由过滤器指定 Barkis 的 RIP 进程仅接收那些被访问列表 1 许可的路由，其中访问列表 1 指明仅允许 0.0.0.0。[1] 访问列表隐含地拒绝了所有其他路由。示例 13-2 给出了 Barkis 的路由表。

示例 13-2 在来自 Traddles 的路由中，0.0.0.0 是惟一被接受的

```
Barkis#show ip route rip
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is 192.168.75.198 to network 0.0.0.0
    192.168.75.0/24 is variably subnetted, 5 subnets, 2 masks
C       192.168.75.32/27 is directly connected, Ethernet0
R     192.168.75.64/27 [120/1] via 192.168.75.201, 00:00:23, Serial0
C       192.168.75.196/30 is directly connected, Serial1
C       192.168.75.200/30 is directly connected, Serial0
R       192.168.75.204/30 [120/1] via 192.168.75.34, 00:00:13, Ethernet0
R*  0.0.0.0/0 [120/10] via 192.168.75.198, 00:00:03, Serial1
Barkis#
```

[1] 注意没有给出反码。访问列表的缺省反码是 0.0.0.0，这对于本配置是正确的。

IPv6 的路由也可以按照相同的办法进行过滤。如示例 13-3 的配置，Barkis 仅接受来自 Traddles 的缺省路由。

示例 13-3 Barkis 通过过滤 IPv6 路由仅接收缺省路由

```
ipv6 router rip emily
distribute-list prefix-list copperfield in Serial1
!
ipv6 prefix-list copperfield permit ::/0
```

IPv4 和 IPv6 配置惟一的不同是，IPv4 的命令 **deistribute-list** 参照访问列表，而 IPv6 则参照一个前缀列表。前缀列表可以允许和禁止 IPv6 前缀。

示例 13-4 给出了 Barkis 的 IPv6 路由表。

示例 13-4 ::/0 是从 Traddles 那里接受的惟一的 IPv6 RIP 路由

```
Barkis#show ipv6 route rip
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
R   ::/0 [120/2]
     via FE80::205:5EFF:FE6B:50A0, Serial1
R   2001:DB8:0:40::/64 [120/2]
     via FE80::2B0:64FF:FE30:1DE0, Serial0
R   2001:DB8:0:CC::/64 [120/2]
     via FE80::204:C1FF:FE50:F1C0, Ethernet0
Barkis#
```

在示例 13-5 中，对 Barkis 的配置进行了改动，使其从 2001:db8:0:103::/64 那里接受 2001:db8:0:100::/64，而不再接受缺省 IPv6 路由。修改后的前缀列表仅接受这个范围的地址。

示例 13-5 Barkis 仅接受一段 IPv6 地址进入 RIPng

```
ipv6 router rip emily
distribute-list prefix-list copperfield in Serial1
!
ipv6 prefix-list copperfield permit 2001:db8:0:100::/62 le 64
```

在 **prefix-list** 命令中，ge 和 le 可以指定 IPv6 前缀的范围，ge 表示大于或等于，le 表示小于或等于。Barkis 的前缀列表指定了前缀 2001:db8:0:100::/62，其中前 62 位相同，63 或 64 位可以是任意值。这个范围包括 2001:db8:0:100:/64、2001:db8:0:101::/64、2001:db8:0:102::/64 和 2001:db8:0:103::/64。前缀列表在附录 B 中将进一步讨论。

当然，沿串行链路通告的 700 条路由没想到会在链路的远端被全部丢弃，这对带宽而言，自然是一种浪费。因此更好的配置方法是将过滤器放在 Traddles 上，仅允许向 Barkis 通告缺省路由，相应的配置见示例 13-6。

示例 13-6 配置 Traddle 过滤出站的 RIP IPv4 路由

```
router rip
version 2
network 192.168.63.0
network 192.168.75.0
network 192.168.88.0
```

（待续）

```
distribute-list 1 out Serial0
!
ip classless
access-list 1 permit 0.0.0.0
```

对于 IPv6，Traddles 上的相应配置见示例 13-7。

示例 13-7 配置 Traddle 过滤出站 IPv6 RIPng 路由

```
ipv6 router rip emily
distribute-list prefix-list copperfield out Serial0
!
ipv6 prefix-list copperfield permit 2001:db8:0:100::/62 le 64
```

这里的过滤器配置看上去与原来的配置差不多相同，但它是过滤出站路由，而不是入站路由。示例 13-8 显示出从 Traddles 沿串行链路被通告的路由中仅包括缺省路由。

示例 13-8 在 Traddles 上的过滤器仅允许向 Barkis 通告缺省路由

```
Barkis#debug ip rip
RIP protocol debugging is on
Barkis#
RIP: received v2 update from 192.168.75.198 on Serial1
     0.0.0.0/0 -> 0.0.0.0 in 10 hops
RIP: sending v2 update to 224.0.0.9 via Ethernet0 (192.168.75.33)
     192.168.75.64/27 -> 0.0.0.0, metric 2, tag 0
     192.168.75.196/30 -> 0.0.0.0, metric 1, tag 0
     192.168.75.200/30 -> 0.0.0.0, metric 1, tag 0
     0.0.0.0/0 -> 0.0.0.0, metric 11, tag 0
RIP: sending v2 update to 224.0.0.9 via Serial0 (192.168.75.202)
     192.168.75.32/27 -> 0.0.0.0, metric 1, tag 0
     192.168.75.196/30 -> 0.0.0.0, metric 1, tag 0
     192.168.75.204/30 -> 0.0.0.0, metric 2, tag 0
     0.0.0.0/0 -> 0.0.0.0, metric 11, tag 0
RIP: sending v2 update to 224.0.0.9 via Serial1 (192.168.75.197)
     192.168.75.32/27 -> 0.0.0.0, metric 1, tag 0
     192.168.75.64/27 -> 0.0.0.0, metric 2, tag 0
     192.168.75.200/30 -> 0.0.0.0, metric 1, tag 0
     192.168.75.204/30 -> 0.0.0.0, metric 2, tag 0
RIP: received v2 update from 192.168.75.34 on Ethernet0
     192.168.75.64/27 -> 0.0.0.0 in 2 hops
     192.168.75.204/30 -> 0.0.0.0 in 1 hops
RIP: received v2 update from 192.168.75.201 on Serial0
     192.168.75.64/27 -> 0.0.0.0 in 1 hops
     192.168.75.204/30 -> 0.0.0.0 in 2 hops
```

在这两种配置中，Barkis 将向 Micawber 和 Peggotty 通告缺省路由，配置不会影响 Barkis 向 Traddles 通告的路由。

当在链路状态协议（例如 OSPF）下配置命令 **distribute-list** 时，关键字 **out** 不能与接口联合使用，[1] 因为不像距离矢量协议，链路状态协议不从自身的路由表中通告路由，没有更新信息被过滤。所以命令 **distribute-list 1 out Serial1** 在链路状态协议下是没有意义的。

在图 13-3 中，还连接了另一组路由器。这部分网络——ThemNet，在独立的管理控制之下，Creakle 也一样。因为 BigNet 的管理员不能访问和控制 ThemNet 内的路由器，所以应使用路由过滤器将来自 Creakle 发往 BigNet 的错误路由信息的可能性减到最少。例如，ThemNet 正在使用缺省路由（或许为了访问内部 Internet 连接）。如果这个缺省路由被通告到 BigNet，那么它将导致数据包被误转到 ThemNet 内，从而产生黑洞。

[1] 关键字 **out** 可以与一种路由选择协议联合使用，这将在下一个案例研究中讨论。

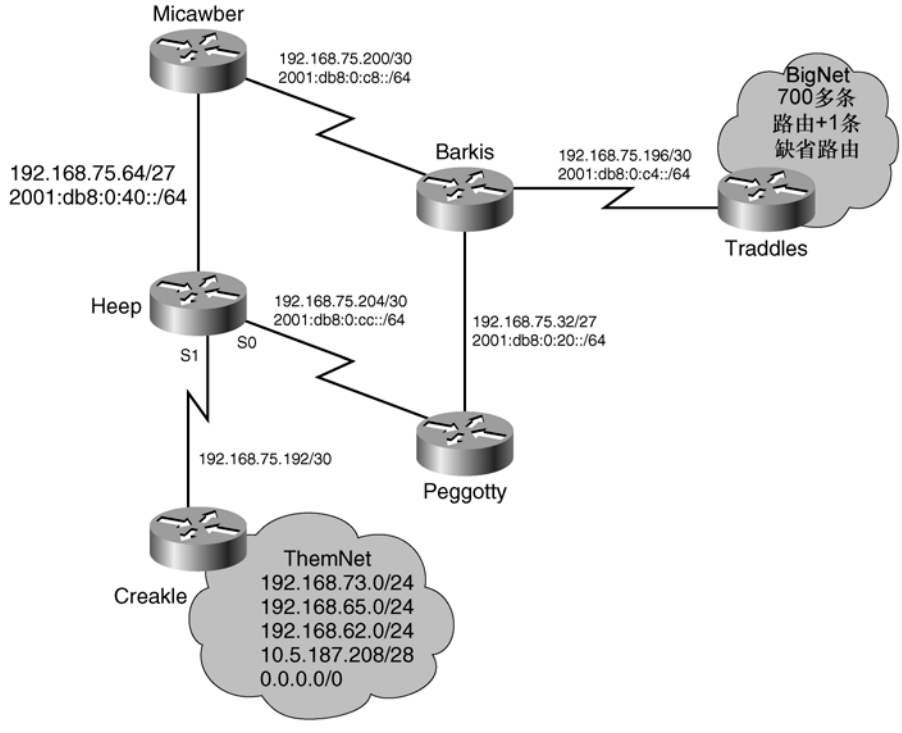


图 13-3 网络 ThemNet 不在 BigNet 管理员的控制之下

为了与 ThemNet 进行通信，Heep 仅允许必要的路由通过，相应的配置见示例 13-9。

示例 13-9 Heep 的配置对来自和去往 Creakle 的 RIP 路由进行过滤

```
router rip
version 2
network 192.168.75.0
distribute-list 2 out Serial1
distribute-list 1 in Serial1
!
ip classless
access-list 1 permit 192.168.73.0
access-list 1 permit 192.168.65.0
access-list 1 permit 192.168.62.0
access-list 1 permit 10.5.187.208
access-list 2 deny 0.0.0.0
access-list 2 permit any
```

分布列表 1 仅允许接受由访问列表 1 指定的，且来自 Creakle 的路由。分布列表阻挡了缺省路由和任何其他路由，这些路由可能会被不正确地插入到 ThemNet 路由表中。

分布列表 2 在适当的位置用于确保 BigNet 是一个正常的邻居；它阻挡了 BigNet 的缺省路由，否则该缺省路由将导致 ThemNet 内出现问题，但是它允许 BigNet 的所有其他路由通过。

13.1.2 案例研究：路由过滤和重新分配

路由器在任何时候执行相互重新分配时，路由回馈都可能会存在。例如，在图 13-4 中，来自 RIP 方的路由会被重新分配到 OSPF，并且再从 OSPF 重新分配回到 RIP。因此，使用路

由过滤器控制路由通告的方向将是一种明智的方法。

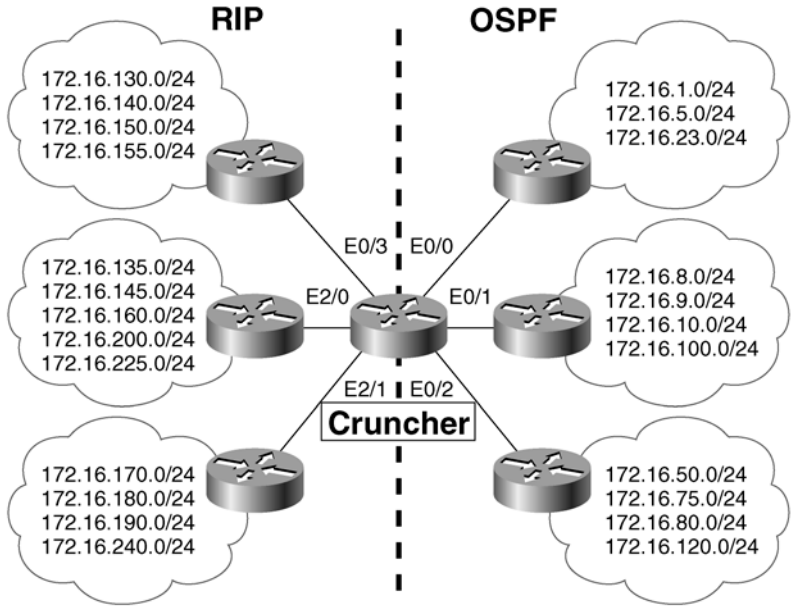


图 13-4 Cruncher 正在向 OSPF 重新分配 RIP 路由，而且还向 RIP 重新分配 OSPF 路由。路由过滤器将被用于阻止路由回馈

图 13-4 中的 Cruncher 可以在几个接口上同时使用 RIP 和 OSPF，Cruncher 的配置见示例 13-10。

示例 13-10 Cruncher 同时运行 OSPF 和 RIP，并且配置了过滤器用来控制通告到每一种协议的路由

```
router ospf 25
redistribute rip metric 100
network 172.16.8.254 0.0.0.0 area 25
network 172.16.50.254 0.0.0.0 area 25
distribute-list 3 in Ethernet0/0
distribute-list 3 in Ethernet0/1
distribute-list 3 in Ethernet0/2
!
router rip
version 2
redistribute ospf 25 metric 5
passive-interface Ethernet0/0
passive-interface Ethernet0/1
passive-interface Ethernet0/2
network 172.16.0.0
distribute-list 1 in Ethernet0/3
distribute-list 1 in Ethernet2/0
distribute-list 1 in Ethernet2/1
!
ip classless
access-list 1 permit 172.16.128.0 0.0.127.255
access-list 3 permit 172.16.0.0 0.0.127.255
```

在上面的配置中，访问列表逻辑允许某些路由但拒绝所有其他路由。该逻辑也可以被颠倒过来，拒绝某些路由但允许所有其他路由。关于 Cruncher 的配置改动见示例 13-11。

示例 13-11 修改 Cruncher 的访问控制，使其拒绝指定的路由但允许所有其他路由

```
access-list 1 deny 172.16.0.0 0.0.127.255
access-list 1 permit any
access-list 3 deny 172.16.128.0 0.0.127.255
access-list 3 permit any
```

第二个访问列表的作用同第一个访问列表相同。在这两种情况下，到 OSPF 域内目标网络的路由将不会从 RIP 向 OSPF 通告，同样到 RIP 域内目标网络的路由也不会从 OSPF 向 RIP 通告。然而，第二个访问列表的配置需要仔细地管理，以便最小化将来出现的问题。例如，一个新的地址被添加到 RIP 域，如果路由过滤器没有相应的做出修改，那么该路由将向 OSPF 通告并且还会被通告回来，因此可能会导致路由环路。这就需要在过滤器中添加一条拒绝语句来控制不发生环路。如果使用第二个访问列表替代第一个访问列表，并且向 RIP 域添加一个新地址，那么这个路由是不会被通告到 OSPF 的，除非访问列表被修改。这时就不会产生路由环路，而且 RIP 域内到该地址的路由也是正常的。因为在第二个访问列表的末尾存在 **permit any**，所以对列表的配置不像管理这样方便。为了向访问列表添加新的条目，整个列表首先必须被删除以便在 **permit any** 之前可以放置新的条目。[1]

为了方便地进行汇总操作，我们对图 13-4 中的子网地址作了精心的分配，产生了小型的访问列表，但却牺牲了精确性。对路由的控制越精确，意味着访问列表越大、越精确，这是以增加管理的注意力为代价的。

在重新分配点部署路由过滤器的另一种方法是借助路由进程进行过滤，而不是接口。例如，示例 13-12 的配置仅允许重新分配图 13-4 中的某些 IPv4 路由。

示例 13-12 Cruncher 在路由进程上过滤路由

```
router ospf 25
redistribute rip metric 100
network 172.16.1.254 0.0.0.0 area 25
network 172.16.8.254 0.0.0.0 area 25
network 172.16.50.254 0.0.0.0 area 25
distribute-list 10 out rip
!
router rip
version 2
redistribute ospf 25 metric 5
passive-interface Ethernet0/3
passive-interface Ethernet2/0
passive-interface Ethernet2/1
network 172.16.0.0
distribute-list 20 out ospf 25
!
ip classless
access-list 10 permit 172.16.130.0
access-list 10 permit 172.16.145.0
access-list 10 permit 172.16.240.0
access-list 20 permit 172.16.23.0
access-list 20 permit 172.16.9.0
access-list 20 permit 172.16.75.0
```

在 OSPF 配置下的路由过滤器允许 OSPF 通告 RIP 协议发现的路由，但这些路由必须是访问表 10 许可的路由。同样的，在 RIP 配置下的路由过滤器允许 RIP 通告 OSPF 25 发现的路由，但这些路由必须是访问列表 20 许可的路由。在两种情况下，路由过滤器对其他协议发

[1] 在某些 IOS 版本中，使用序列号可以把访问列表条目添加到指定的位置，使得访问列表条目非常容易管理。

现的路由没有影响。例如，如果 OSPF 重新分配 RIP 和 EIGRP 的路由，前面的分布列表将不会应用到 EIGRP 发现的路由上。

同样的配置也可以应用在 IPv6 前缀上。对于 RIPng 和 OSPFv3，分配列表会参照前缀列表，而不是访问列表，其中前缀列表被用来允许和禁止 IPv6 前缀，而不是 IPv4 地址。

当通过进程进行过滤时，仅允许使用关键字 **out**。在 OSPF 下使用 **distribute-list 10 in rip** 是没有意义的，因为路由已经通过 RIP 进入到路由表中了，OSPF 要么通告它（out），要么不通告。

注意，虽然通过路由选择协议进行过滤对于指定那些将要被重新分配的路由是很有用处的，但是它并不是防止路由回馈的好办法。例如，考虑一下在图 13-4 中 Cruncher 的配置（参见示例 13-13）。

示例 13-13 Cruncher 的配置对 OSPF 和 RIP 进程通告的路由进行过滤，但是不过滤哪些已经被添加到路由表中的路由

```
router ospf 25
redistribute rip metric 100
network 172.16.1.254 0.0.0.0 area 25
network 172.16.8.254 0.0.0.0 area 25
network 172.16.50.254 0.0.0.0 area 25
distribute-list 1 out rip
!
router rip
version 2
redistribute ospf 25 metric 5
passive-interface Ethernet0/3
passive-interface Ethernet2/0
passive-interface Ethernet2/1
network 172.16.0.0
distribute-list 3 out ospf 25
!
ip classless
access-list 1 permit 172.16.128.0 0.0.127.255
access-list 3 permit 172.16.0.0 0.0.127.255
```

假设一条来自 RIP 域的路由，如 172.16.190.0/24，被重新分配到 OSPF 域，然后又被通告回到 Cruncher，原因是路由器配置错误。虽然在 RIP 配置下的分布列表将会阻止路由被通告回 RIP 域，但是它却不能阻止路由以 OSPF 域发生的路由的身份进入 Cruncher 的路由表。事实上，过滤器认为路由已经通过 OSPF 进入路由表，因为 OSPF 的管理距离低于 RIP。Cruncher 将会优选 OSPF 路由，因此 Cruncher 把去往 172.16.190.0 的流量路由到 OSPF 的路由器，而不是 RIP 路由器。所以为了阻止路由回馈，必须在路由进入路由表之前，在路由入站时进行过滤。

13.1.3 案例研究：协议迁移

命令 **disntance** 可以为路由指定管理距离，这些路由是从一个特殊的 IPv4 或 IPv6 路由协议那里学习到的。命令在使用时不带任何可选参数。在最初考虑时，该操作看上去不像路由过滤功能，但是当运行多个路由选择协议时就不同了，这时将会基于路由的管理距离来确定是否接受或拒绝路由。

在图 13-5 中，网络运行 RIP，并且计划将路由选择协议转换为 EIGRP。有好几种方法能完成这样的路由迁移。一种方法是在每一台路由器上关闭老的协议，然后打开新的协议。虽

然这种方法对类似于图 13-5 的小型网络来说是合适的，但在更大的网络中，这种停工其是不切实际的。

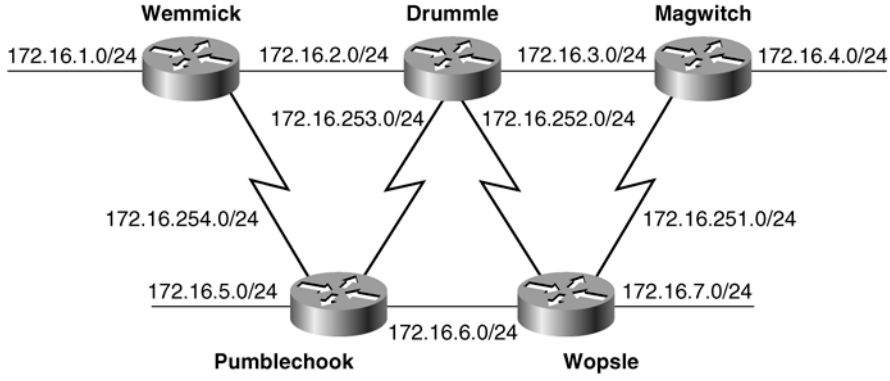


图 13-5 这些运行 RIP 的路由器将会被转为运行 EIGRP

另一种选择是不删除旧协议的同时添加新的协议。如果新协议的缺省管理距离小于旧协议，那么每台路由器将选择新协议通告的路由。随着路由器转换完毕，网络将收敛到新的协议上。在整个网络收敛到新的协议之后，则可以从所有路由器上删除旧的协议。

参照表 11-1，RIP 的缺省管理距离是 120，EIGRP 的缺省管理距离为 90。除了 RIP 之外，如果 EIGRP 被添加到每台路由器，那么路由器将随着邻居路由器开始使用 EIGRP 的同时也开始选择使用 EIGRP 路由。当所有 RIP 路由从路由表中消失时，网络将收敛到 EIGRP。RIP 进程接着会从路由器上被删除。

这种方法的问题是，在重新配置的时候可能会存在路由环路和黑洞。在图 13-5 中，大约几分钟就可以完成对 5 台路由器的重新配置和转换，所以这里不会像大型网络一样关注环路问题。

对这种双协议方法的改进是使用命令 **distance**，以确保禁止新协议的路由，一直到所有路由器为转换做好准备。这个过程中第一步是在所有路由器上降低 RIP 的管理距离，具体配置见示例 13-14。

示例 13-14 对图 13-5 网络中的每台路由器进行重新配置，减小 RIP 的管理距离

```
router rip
network 172.16.0.0
distance 70
```

注意，管理距离仅与单台路由器的路由进程相关。当 RIP 仍然是惟一正在运行的协议时，管理距离的改变不会影响到路由。

下一步是重新访问路由器，向每一台路由器添加 EIGRP 进程，EIGRP 的配置见示例 13-15。

示例 13-15 为图 13-5 中的每台路由器添加 EIGRP 配置

```
router eigrp 1
network 172.16.0.0
!
router rip
network 172.16.0.0
distance 70
```

由于 EIGRP 的缺省管理距离是 90，因此 RIP 的路由被优先选择（参见示例 13-16）。因

为没有路由器选择 EIGRP 路由，所以在配置期间不必确定停工时间表。这种方式使网络管理员在转换之前有时间重新检查每一台路由器的新配置。

示例 13-16 由于为 RIP 路由分配的管理距离为 70，所以优先选择的是 RIP 路由，而不是 EIGRP 路由

```
Drummle#show ip route
       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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
    172.16.0.0/24 is subnetted, 11 subnets
C      172.16.252.0 is directly connected, Serial0
C      172.16.253.0 is directly connected, Serial1
R      172.16.254.0 [70/1] via 172.16.2.253, 00:00:16, Ethernet0
                    [70/1] via 172.16.253.253, 00:00:05, Serial1
R      172.16.251.0 [70/1] via 172.16.252.253, 00:00:08, Serial0
                    [70/1] via 172.16.3.253, 00:00:01, Ethernet1
R      172.16.4.0 [70/1] via 172.16.3.253, 00:00:01, Ethernet1
R      172.16.5.0 [70/1] via 172.16.253.253, 00:00:05, Serial1
R     172.16.6.0 [70/1] via 172.16.252.253, 00:00:08, Serial0
                 [70/1] via 172.16.253.253, 00:00:05, Serial1
R      172.16.7.0 [70/1] via 172.16.252.253, 00:00:08, Serial0
R      172.16.1.0 [70/1] via 172.16.2.253, 00:00:17, Ethernet0
C      172.16.2.0 is directly connected, Ethernet0
C      172.16.3.0 is directly connected, Ethernet1
Drummle#
```

协议转换过程的下一步是再一次访问每台路由器，将 RIP 的距离改回到 120。这一步要计划停工时间。因为新的 RIP 更新路由的管理距离被指派为 120，所以管理距离为 70 的 RIP 路由将会超时（参见示例 13-17）。经过 210s 之后，宣布 RIP 路由失效（参见示例 13-18），最终 EIGRP 的路由将被优先选择（参见示例 13-19）。

示例 13-17 在 RIP 的管理距离被改回到 120 之后，管理距离为 70 的路由开始老化。这里所有 RIP 路由老化时间都已超过 2min

```
Drummle#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
    172.16.0.0/24 is subnetted, 11 subnets
C      172.16.252.0 is directly connected, Serial0
C      172.16.253.0 is directly connected, Serial1
R     172.16.254.0 [70/1] via 172.16.2.253, 00:02:31, Ethernet0
                   [70/1] via 172.16.253.253, 00:02:18, Serial1
R      172.16.251.0 [70/1] via 172.16.252.253, 00:02:27, Serial0
                    [70/1] via 172.16.3.253, 00:02:32, Ethernet1
R     172.16.4.0 [70/1] via 172.16.3.253, 00:02:32, Ethernet1
R     172.16.5.0 [70/1] via 172.16.253.253, 00:02:19, Serial1
R     172.16.6.0 [70/1] via 172.16.252.253, 00:02:27, Serial0
                 [70/1] via 172.16.253.253, 00:02:19, Serial1
R      172.16.7.0 [70/1] via 172.16.252.253, 00:02:27, Serial0
R      172.16.1.0 [70/1] via 172.16.2.253, 00:02:32, Ethernet0
C      172.16.2.0 is directly connected, Ethernet0
C      172.16.3.0 is directly connected, Ethernet1
```

示例 13-18 经过 3.5min，RIP 路由将被宣布失效

```
Drummle#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
    172.16.0.0/24 is subnetted, 11 subnets
C       172.16.252.0 is directly connected, Serial0
C       172.16.253.0 is directly connected, Serial1
R       172.16.254.0/24 is possibly down,
          routing via 172.16.253.253, Serial1
R       172.16.251.0/24 is possibly down,
          routing via 172.16.252.253, Serial0
R       172.16.4.0/24 is possibly down,
          routing via 172.16.3.253, Ethernet1
R       172.16.5.0/24 is possibly down,
          routing via 172.16.253.253, Serial1
R       172.16.6.0/24 is possibly down,
          routing via 172.16.253.253, Serial1
R       172.16.7.0/24 is possibly down,
          routing via 172.16.252.253, Serial0
R       172.16.1.0/24 is possibly down,
          routing via 172.16.2.253, Ethernet0
C       172.16.2.0 is directly connected, Ethernet0
C       172.16.3.0 is directly connected, Ethernet1
Drummle#
```

示例 13-19 缺省管理距离为 90 的 EIGRP 路由代替了 RIP 路由

```
Drummle#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
      U - per-user static route, o - ODR
Gateway of last resort is not set
    172.16.0.0/24 is subnetted, 11 subnets
C       172.16.252.0 is directly connected, Serial0
C       172.16.253.0 is directly connected, Serial1
D       172.16.254.0 [90/2195456] via 172.16.2.253, 00:01:11, Ethernet0
D       172.16.251.0 [90/2195456] via 172.16.3.253, 00:01:06, Ethernet1
D       172.16.4.0 [90/307200] via 172.16.3.253, 00:01:06, Ethernet1
D       172.16.5.0 [90/2195456] via 172.16.253.253, 00:01:11, Serial1
D       172.16.6.0 [90/2185984] via 172.16.252.253, 00:01:11, Serial0
                   [90/2185984] via 172.16.253.253, 00:01:11, Serial1
D       172.16.7.0 [90/2195456] via 172.16.252.253, 00:01:07, Serial0
D       172.16.1.0 [90/307200] via 172.16.2.253, 00:01:11, Ethernet0
C       172.16.2.0 is directly connected, Ethernet0
C       172.16.3.0 is directly connected, Ethernet1
Drummle#
```

虽然使用这种方法仍然有可能产生路由环路和黑洞，但是因为仅需要修改管理距离，所以转换速度会更快，人为的错误也越小。

这种方法的另一优点是，万一出现问题可以很容易地停止切换工作。在所有路由器中 RIP 进程仍然处于合适的位置，退回到 RIP 需要做的所有工作就是将管理距离改回 70。一旦新的 EIGRP 被测试并证明是稳定后，可以从所有路由器上删除 RIP 进程而不会有进一步的服务中断。

在使用双协议方法之前需要考虑一件事情，即同时在每台路由器上运行两个协议可能会

影响路由器的内存用量和处理速度。如果内存利用率、处理利用率或两者的平均值超过 50%或 60%，那么在提交转换工作之前应该进行仔细的实验室测试和仿真，以确保路由器可以处理这些额外的负载。如果路由器不能胜任，那么在配置新协议之前需要更加复杂的过程来删除旧协议，这可能是惟一的选择。

与上面过程不同之处在于，这里是增加新协议的管理距离，而不是降低旧协议的管理距离，接着在转换时再降低新协议的管理距离。但是，要确保在输入任何网络命令之前输入命令 **distance**，以便不会用新协议缺省的管理距离激活新协议。

回顾一下表 11-1，注意 EIGRP 有两个管理距离：对内部路由为 90，对外部路由为 170。因此，对 EIGRP 来说，命令 **distance** 还有些不同。例如，用提高 EIGRP 的管理距离代替降低 RIP 的管理距离，相应配置见示例 13-20。

示例 13-20　在图 13-5 中的所有路由器上提高 EIGRP 的内部管理距离，而不是降低 RIP 的管理距离

```
router eigrp 1
network 172.16.0.0
distance eigrp 130 170
!
router rip
network 172.16.0.0
```

添加关键字 **eigrp** 以说明指定的 EIGRP 管理距离。内部 EIGRP 路由的管理距离被改为 130，而外部路由的管理距离仍为 170。

使用双协议迁移到新的路由选择协议还需注意的一点是：确认你已经理解两种协议的行为。例如，某些协议（如 EIGRP）不会老化自身的路由条目。因此，如果要取代 EIGRP，需要增加额外的一步，就是在改变完管理距离之后使用命令 **clear ip route *** 清除路由表。

13.1.4　案例研究：多个重新分配点

图 13-6 给出的网络非常类似于图 11-3 给出的网络。回忆一下第 11 章中关于多个重新分配点问题的讨论，由于管理距离导致路由器选择了不满意路由的问题。在某些情况下，还会导致路由环路和黑洞。例如，在 Bumbel 的路由表（参见示例 13-21）中，指向网络 192.168.6.0 的路由的下一跳路由器是 Blathers，而不是 Monks。

示例 13-21　使用 Bumble 的路由可以经 Blathers（192.168.3.2）到达 192.168.6.0，这里包括两条串行链路和一个令牌环

```
Bumble#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
R     192.168.1.0/24 [120/1] via 192.168.2.1, 00:00:00, Ethernet0
C     192.168.2.0/24 is directly connected, Ethernet0
C     192.168.3.0/24 is directly connected, Serial0
```

（待续）

```
O     192.168.4.0/24 [110/70] via 192.168.3.2, 00:05:09, Serial0
O     192.168.5.0/24 [110/134] via 192.168.3.2, 00:05:09, Serial0
O E2 192.168.6.0/24 [110/100] via 192.168.3.2, 00:05:09, Serial0
Bumble#
```

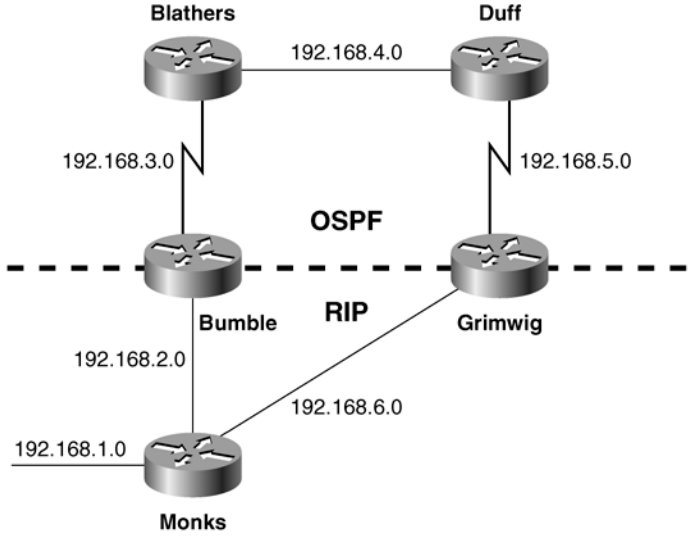


图 13-6 当在多个点进行互相重新分配时，管理距离会导致非最佳路径选择、路由环路和黑洞

解决这个问题的一种办法是，在重新分配点使用命令 **distribute-list** 来控制路由源点。Bumble 和 Grimwig 的配置分别见示例 13-22 和示例 13-23。

示例 13-22 Bumble 配置使 OSPF 进程仅接受 OSPF 域的地址，以及 RIP 进程仅接受 RIP 域的地址

```
router ospf 1
redistribute rip metric 100
network 192.168.3.1 0.0.0.0 area 0
distribute-list 1 in
!
router rip
redistribute ospf 1 metric 2
network 192.168.2.0
distribute-list 2 in
!
ip classless
access-list 1 permit 192.168.4.0
access-list 1 permit 192.168.5.0
access-list 2 permit 192.168.1.0
access-list 2 permit 192.168.6.0
```

示例 13-23 Grimwig 配置使 OSPF 进程仅接受 OSPF 域的地址，以及 RIP 进程仅接受 RIP 域的地址

```
router ospf 1
redistribute rip metric 100
network 192.168.5.1 0.0.0.0 area 0
distribute-list 1 in
!
```

（待续）

```
router rip
redistribute ospf 1 metric 2
network 192.168.6.0
distribute-list 2 in
!
ip classless
access-list 1 permit 192.168.3.0
access-list 1 permit 192.168.4.0
access-list 2 permit 192.168.1.0
access-list 2 permit 192.168.2.0
```

在上面两个配置中，访问列表 1 仅允许 OSPF 接受 OSPF 域内的地址，访问列表 2 仅允许 RTP 接受 RIP 域内的网络。在配置了路由过滤器之后，示例 13-24 给出了 Bumble 的路由表。

示例 13-24 在配置路由过滤之后，Bumble 将使用最佳路由到达网络 192.168.6.0

```
Bumble#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
R    192.168.1.0/24 [120/1] via 192.168.2.1, 00:00:12, Ethernet0
C    192.168.2.0/24 is directly connected, Ethernet0
C    192.168.3.0/24 is directly connected, Serial0
O    192.168.4.0/24 [110/70] via 192.168.3.2, 00:00:22, Serial0
O    192.168.5.0/24 [110/134] via 192.168.3.2, 00:00:22, Serial0
R    192.168.6.0/24 [120/1] via 192.168.2.1, 00:00:13, Ethernet0
Bumble#
```

使用这种配置方法的问题是，消除了多个重新分配点内在的冗余性。在示例 13-25 中，Bumble 的以太网链路被断开。由于在 OSPF 中过滤掉指向 RIP 网络的路由，因而所有路由现在都不可达。

示例 13-25 当 Bumble 的以太网链路发生故障后，RIP 网络变得不可达。路由过滤器可阻止 OSPF 向路由表中输入替代的路由

```
Bumble#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0, changed state to down
Bumble#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
C    192.168.3.0/24 is directly connected, Serial0
O    192.168.4.0/24 [110/70] via 192.168.3.2, 00:06:45, Serial0
O    192.168.5.0/24 [110/134] via 192.168.3.2, 00:06:45, Serial0
Bumble#
```

对 IPv4 来说，一种更好的方法是使用命令 **distance** 的两种形式来设置首选路由。[1] Bumble 和 Grimwig 的配置见示例 13-26 和示例 13-27。

[1] IPv6 路由协议也支持使用 **distance** 命令修改管理距离；但在 IPv6 的命令格式中仅有一个选项可以改变距离值。目前还没有选项用于指定邻居地址或前缀列表。

示例 13-26 Bumble 使用距离配置命令修改 IP 路由的管理距离

```
router ospf 1
redistribute rip metric 100
network 192.168.3.1 0.0.0.0 area 0
distance 130
distance 110 0.0.0.0 255.255.255.255 1
!
router rip
redistribute ospf 1 metric 2
network 192.168.2.0
distance 130
distance 120 192.168.2.1 0.0.0.0 2
!
ip classless
access-list 1 permit 192.168.4.0
access-list 1 permit 192.168.5.0
access-list 2 permit 192.168.1.0
access-list 2 permit 192.168.6.0
```

示例 13-27 Grimwig 使用距离配置命令修改 IP 路由的管理距离

```
router ospf 1
redistribute rip metric 100
network 192.168.5.1 0.0.0.0 area 0
distance 130
distance 110 0.0.0.0 255.255.255.255 1
!
router rip
redistribute ospf 1 metric 2
network 192.168.6.0
distance 130
distance 120 192.168.6.1 0.0.0.0 2
!
ip classless
access-list 1 permit 192.168.3.0
access-list 1 permit 192.168.4.0
access-list 2 permit 192.168.1.0
access-list 2 permit 192.168.2.0
```

在这两个配置中，第一个 **distance** 命令设置了 OSPF 和 RIP 的管理距离为 130。第二个 **distance** 命令根据被指定的通告路由器和参考访问列表来设定一个不同的管理距离。例如，Grimwig 的 RIP 进程为由 Monks（192.168.6.1）通告且被访问列表 2 许可的路由指定的管理距离为 120，所有其他路由的管理距离为 130。注意，这里与通告路由器地址一起使用的是反码。

在 OSPF 的配置中会存在更多的问题。通告路由器的地址不必是下一跳路由器的接口地址，而是产生 LSA 的路由器的 ID，其中路由就是根据 LSA 进行计算的。因此对命令 **distance** 来说，在 OSPF 配置下的地址和反码是 0.0.0.0 255.255.255.255，它们可以指定任意路由器。[1] 访问列表 1 许可的 OSPF 路由的管理距离被指派为 110，所有其他路由的管理距离为 130。

结果如示例 13-28 所示，第一个路由表显示 Grimwig 经过 Duff 到达 OSPF 域内的所有网络，Grimwig 经 Monks 到达 RIP 域内的所有网络。OSPF 正常的路由管理距离为 110，RIP 管理距离为 120。接着，Grimwig 的以太网链路发生故障。第二个路由表显示所有网络均经过 Duff 可达。指向 RIP 域内网络的路由的管理距离为 130。当以太网链路恢复正常后，来自 Monks 且管理距离为 120 的 RIP 通告将取代管理距离为 130 的 OSPF 通告。

[1] 相同的“任意”地址也可以和 RIP 一起使用，而使用一个特殊地址完全是为了示范目的。

示例 13-28 Grimwig 到 Monks 的以太网链路发生故障前后 Grimwig 的路由表

```
Grimwig#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
R    192.168.1.0/24 [120/1] via 192.168.6.1, 00:00:19, Ethernet0
R    192.168.2.0/24 [120/1] via 192.168.6.1, 00:00:19, Ethernet0
O    192.168.3.0/24 [110/134] via 192.168.5.2, 00:15:06, Serial0
O    192.168.4.0/24 [110/70] via 192.168.5.2, 00:15:06, Serial0
C    192.168.5.0/24 is directly connected, Serial0
C    192.168.6.0/24 is directly connected, Ethernet0
Grimwig#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0, changed state to down
Grimwig#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
O E2 192.168.1.0/24 [130/100] via 192.168.5.2, 00:00:08, Serial0
O E2 192.168.2.0/24 [130/100] via 192.168.5.2, 00:00:08, Serial0
O    192.168.3.0/24 [110/134] via 192.168.5.2, 00:16:23, Serial0
O    192.168.4.0/24 [110/70] via 192.168.5.2, 00:16:23, Serial0
C    192.168.5.0/24 is directly connected, Serial0
O E2 192.168.6.0/24 [130/100] via 192.168.5.2, 00:00:08, Serial0
Grimwig#
```

在图 13-6 中，如果其中一条串行链路发生故障，那么将会发生相反的事情。穿过 RIP 域将可以到达 OSPF 域内的网络，而且管理距离再次是 130（参见示例 13-29）。然而，不像 OSPF 可以快速地收敛，RIP 要花几分钟的时间才能收敛。这种慢收敛是由于在 Monks 处 RIP 的水平分割所导致的。Monks 将不会向 Bumble 和 Grimwig 通告 OSPF 路由，直到这两台路由器停止通告相同的路由并且现存的路由超时为止。

解决问题的办法是，关闭 Monks 两个以太网接口上的水平分割功能，这可以使用命令 **no ip split-horizon** 完成。虽然关闭水平分割缩短了收敛时间，但同时也失去了环路保护功能，不过还是值得的。在 Bumble 和 Grimwig 上基于管理距离的路由过滤可以阻止所有多跳环路，而且在两台路由器的相同以太网接口上的水平分割功能还可以打断单跳环路。

示例 13-29 到 Duff 串行链路发生故障之前和之后的 Grimwig 的路由表

```
Grimwig#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
R    192.168.1.0/24 [120/1] via 192.168.6.1, 00:00:04, Ethernet0
R    192.168.2.0/24 [120/1] via 192.168.6.1, 00:00:04, Ethernet0
O    192.168.3.0/24 [110/134] via 192.168.5.2, 00:00:12, Serial0
O    192.168.4.0/24 [110/70] via 192.168.5.2, 00:00:12, Serial0
```

（待续）

```
C    192.168.5.0/24 is directly connected, Serial0
C    192.168.6.0/24 is directly connected, Ethernet0
Grimwig#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to down
%LINK-3-UPDOWN: Interface Serial0, changed state to down
Grimwig#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route
Gateway of last resort is not set
R    192.168.1.0/24 [120/1] via 192.168.6.1, 00:00:07, Ethernet0
R    192.168.2.0/24 [120/1] via 192.168.6.1, 00:00:07, Ethernet0
R    192.168.3.0/24 [130/3] via 192.168.6.1, 00:00:07, Ethernet0
R    192.168.4.0/24 [130/3] via 192.168.6.1, 00:00:07, Ethernet0
R    192.168.5.0/24 is possibly down, routing via 192.168.6.1, Ethernet0
C    192.168.6.0/24 is directly connected, Ethernet0
Grimwig#
```

还有另一种在互相重新分配路由的路由选择协议之间帮助过滤地址的方法，就是路由标记（tag）。当路由被重新分配到某个路由选择协议时，这些路由可以被标记。标记是管理员设置的一个数字，可以惟一地标识重新分配到该路由器的一组路由集合。这个标记可以在另一台路由器上用于在重新分配回原来的路由选择协议时过滤掉这些路由。标记可以使用路由映射进行设置，这将在第 14 章中进一步讲述。

13.1.5 案例研究：使用管理距离设置路由器优先权

在图 13-6 中，假设策略规定把 Grimwig 作为到 OSPF 域的主路由器，仅当 Grimwig 不可达时才选择经过 Bumble 的路由。策略实施前， Monks 通过在 Grimwig 和 Bumble 之间执行等价负载均衡到达 OSPF 的网络（参见示例 13-30）。

示例 13-30 Monks 把从 Bumble 和 Grimwig 到 OSPF 域内所有网络都看成是等距离的

```
Monks#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
C    192.168.1.0/24 is directly connected, Ethernet2
C    192.168.2.0/24 is directly connected, Ethernet0
R    192.168.3.0/24 [120/2] via 192.168.2.2, 00:00:26, Ethernet0
                    [120/2] via 192.168.6.2, 00:00:23, Ethernet1
R    192.168.4.0/24 [120/2] via 192.168.2.2, 00:00:26, Ethernet0
                    [120/2] via 192.168.6.2, 00:00:23, Ethernet1
R    192.168.5.0/24 [120/2] via 192.168.2.2, 00:00:26, Ethernet0
                    [120/2] via 192.168.6.2, 00:00:23, Ethernet1
C    192.168.6.0/24 is directly connected, Ethernet1
Monks#
```

通过降低来自 Grimwig 的路由的管理距离，可以使得 Monks 优先选择 Grimwig，相应配置见示例 13-31。

示例 13-31　Monks 通告配置降低所有来自 Grimwig 的路由管理距离

```
router rip
network 192.168.1.0
network 192.168.2.0
network 192.168.6.0
distance 100 192.168.6.2 0.0.0.0
```

在这里，命令 **distance** 没有参考访问列表。所有 Grimwig（192.168.6.2）通告的路由的管理距离都将被指定为 100。所有其他路由（来自 Bumble）都被指定为 RIP 的缺省管理距离 120。因此 Grimwig 的路由被优先选择。

示例 13-32 给出了结果。第一个路由表显示了 Monks 选择 Grimwig 进行路由。当到 Grimwig 的连接发生故障时，路由表结尾显示 Monks 切换到 Bumble（192.168.2.2）。

示例 13-32　到 Grimwig 的以太网链路发生故障之前和之后的 Monks 的路由表

```
Monks#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
C    192.168.1.0/24 is directly connected, Ethernet2
C    192.168.2.0/24 is directly connected, Ethernet0
R    192.168.3.0/24 [100/2] via 192.168.6.2, 00:00:12, Ethernet1
R    192.168.4.0/24 [100/2] via 192.168.6.2, 00:00:12, Ethernet1
R    192.168.5.0/24 [100/2] via 192.168.6.2, 00:00:12, Ethernet1
C    192.168.6.0/24 is directly connected, Ethernet1
Monks#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet1, changed state to down
Monks#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
C    192.168.1.0/24 is directly connected, Ethernet2
C    192.168.2.0/24 is directly connected, Ethernet0
R    192.168.3.0/24 [120/2] via 192.168.2.2, 00:00:00, Ethernet0
R    192.168.4.0/24 [120/2] via 192.168.2.2, 00:00:00, Ethernet0
R    192.168.5.0/24 [120/2] via 192.168.2.2, 00:00:00, Ethernet0
R    192.168.6.0/24 [120/2] via 192.168.2.2, 00:00:00, Ethernet0
Monks#
```

13.2　展　望

对控制网络的行为来说，路由过滤器是一个非常有用的工具。在大型网络中，路由过滤器几乎是不可缺少的。但是路由过滤器的所有效用仅限于允许或不接受路由。第 14 章将介绍另一个强大的工具——路由映射，路由映射不仅可以标识路由，还可以主动地修改路由。

13.3 总结表：第 13 章命令总结

命令	描述
access-list *access-list-number* {**deny**|**permit**} *source* [*source-wildcard*]	定义标准 IP 访问列表表项
distance *weight* [*address mask* [*access-list-number*|*name*]]	定义管理距离，该值不同于缺省值
distance eigrp *internal-distance external-distance*	为 EIGRP 内部和外部路由定义管理距离，该值不同于缺省值
distance ospf {[**intra-area** *dist1*] [**inter-area** *dist2*] [**external** *dist3*]}	为域内、域间和外部路由定义管理距离，该值不同于缺省值
distribute-list{*access-list-number*|*name*} **in** [*interface-name*]	过滤入站 IPv4 更新路由
distribute-list {*access-list-number*|*name*} **out** [*interface-name*|*routing-process*|*autonomous-system-number*]	过滤出站 IPv4 更新路由
distribute-list prefix-list *list_name* [**in**]**out**]	过滤 IPv6 出站或入站更新路由
ipv6 prefix-list *list-name* [**seq** *num*] {**deny**|**permit**} *prefix/length* [**ge** *length* | **le** *length*]	定义 IPv6 前缀列表表项，允许或禁止一个或一组 IPv6 前缀
redistribute *protocol* [*process-id*]{**level-1**|**level-1-2**|**level-2**}[**metric** *metric-value*][**metric-type** *type-value*][**match**{**internal**|**external 1**|**external 2**}][**tag** *tag-value*] [**route-map** *map-tag*][**weight** *weight*][**subnets**]	配置向一路由选择协议重新分配路由，并指明被重新分配路由的源点

13.4 配置练习

1．在图 13-7 中路由器 A 的配置见示例 13-33。请在路由器 A 上配置路由过滤器，禁止除路由器 E 之外所有路由器知道子网 172.16.12.0/24。

示例 13-33 图 13-7 中路由器 A 的配置

```
router rip
redistribute eigrp 1 metric 3
passive-interface Ethernet0
passive-interface Ethernet1
network 172.16.0.0
!
router eigrp 1
redistribute rip metric 10000 1000 255 1 1500
passive-interface Ethernet2
passive-interface Ethernet3
network 172.16.0.0
```

2．图 13-7 中，在路由器 A 上配置路由过滤器，阻止路由器 D 学习到子网 172.16.10.0/24。

3．图 13-7 中，在路由器 A 上配置路由过滤器，仅允许向 RIP 域通告子网 172.16.2.0/24、172.16.8.0/24 和 172.16.9.0/24。

4．图 13-7 中，在路由器 A 上配置路由过滤器，禁止路由器 B 学习到 RIP 域内的任何子网。

5．路由器 A、B 和 C 上配置了 OSPFv3，路由器 A、D 和 E 配置了 RIPng。路由器 B 上连接了 IPv6 前缀 2001:db8:0:1::/64、2001:db8:0:2::/64 和 2001:db8:0:3::/64。路由器 E 连接了 2001:db8:0:a::/64、2001:db8:0:b::/64 和 2001:db8:0:c::/64。请在图 13-7 中路由器 A 上配置路由过滤器禁止把 2001:da8:0:a::/64 和 2001:db8:0:b::/64 通告给路由器 D。

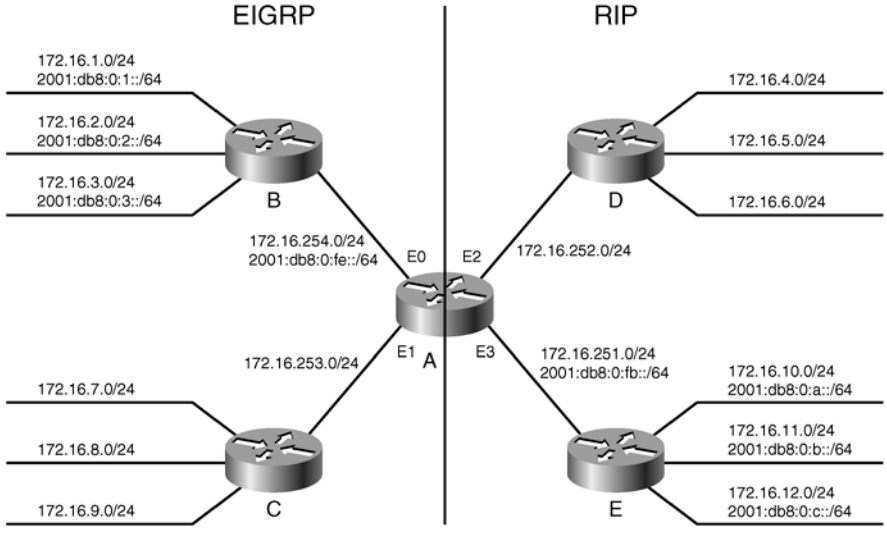


图 13-7 配置练习 1～4 中的网络

6．表 13-1 给出了图 13-8 中所有路由器的接口地址。路由器 A 和路由器 B 运行 EIGRP，路由器 E 和路由器 F 运行 IS-IS。路由器 C 和路由器 D 正在重新分配。为路由器 C 和路由器 D 配置命令 **disntance**，来阻止路由环路和路由回馈，但允许存在冗余路由。

表 13-1 图 13-8 中所有路由器的接口地址

路由器	接口	地址	掩码
A	E0	192.168.1.1	255.255.255.0
	S0	192.168.10.254	255.255.255.252
	S1	192.168.10.249	255.255.255.252
	S2	192.168.10.245	255.255.255.252
B	E0	192.168.2.1	255.255.255.0
	S0	192.168.10.246	255.255.255.252
	S1	192.168.10.241	255.255.255.252
C	S0	192.168.10.253	255.255.255.252
	S1	192.168.10.234	255.255.255.252
	S2	192.168.10.225	255.255.255.252
D	E0	192.168.10.250	255.255.255.252
	S1	192.168.10.242	255.255.255.252
	S2	192.168.10.237	255.255.255.252
E	E0	192.168.4.1	255.255.255.0
	S0	192.168.10.226	255.255.255.252
	S1	192.168.10.229	255.255.255.252
F	E0	192.168.3.1	255.255.255.0
	S0	192.168.10.230	255.255.255.252
	S1	192.168.10.233	255.255.255.252
	S2	192.168.10.238	255.255.255.252

7. 在图 13-8 中，使用命令 **distance** 配置路由器 D，使它仅接受来自路由器 A 的 EIGRP 路由。如果到路由器 A 的链路发生故障，那么路由器 D 将不接受来自路由器 B 的路由，虽然路由器 D 仍旧向路由器 B 通告路由。

8. 删除练习 7 中添加到路由器 D 的配置，配置图 13-8 中的路由器 C，使它通过路由器 A 可以到达所有目标网络，包括 IS-IS 域内的所有子网。仅在路由器 A 发生故障时，路由器 C 才选择经过路由器 E 和路由器 F 进行路由。

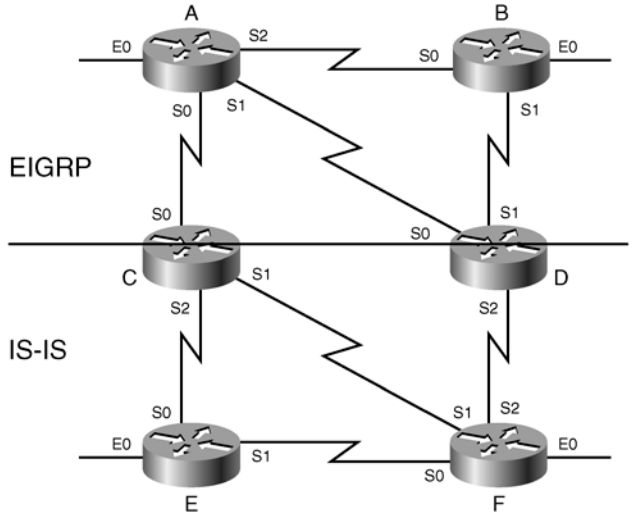


图 13-8 配置练习 6～8 的网络

13.5 故障诊断练习

1. 路由器的配置见示例 13-34。

示例 13-34 路由器试图滤掉缺省路由

```
router eigrp 1
network 10.0.0.0
distribute-list 1 in Ethernet5/1
!
access-list 1 deny 0.0.0.0 255.255.255.255
access-list 1 permit any
```

上面配置的目的是阻止进入接口 E5/1 的缺省路由，而允许进入该接口的所有其他路由。但是，在接口 E5/1 路由器却没有接受任何路由，错误在哪里？

2. 在图 13-6 中，Grimwig 的配置见示例 13-35。

示例 13-35 故障诊断练习 2 中 Grimwig 的配置

```
router ospf 1
redistribute rip metric 100
network 192.168.5.1 0.0.0.0 area 0
distance 255
distance 110 0.0.0.0 255.255.255.255 1
!
```

（待续）

```
router rip
redistribute ospf 1 metric 2
network 192.168.6.0
distance 255
distance 120 192.168.6.1 0.0.0.0 2
!
ip classless
access-list 1 permit 192.168.3.0
access-list 1 permit 192.168.4.0
access-list 2 permit 192.168.1.0
access-list 2 permit 192.168.2.0
```

该配置对 Grimwig 的路由有什么影响？

3．在图 13-9 中，路由器运行 OSPF，路由器 B 的配置见示例 13-36。

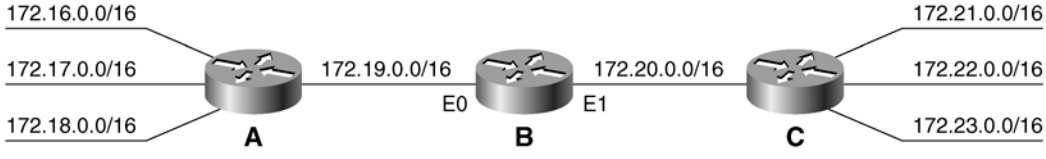


图 13-9 故障诊断练习 3 和 4 中的网络

示例 13-36 图 13-9 中路由器 B 的 OSPF 配置

```
router ospf 50
network 0.0.0.0 255.255.255.255 area 1
distribute-list 1 in
!
access-list 1 deny 172.17.0.0
access-list 1 permit any
```

上面配置的目的是禁止路由器 B 和路由器 C 具有关于网络 172.17.0.0 的路由表项。看上去该计划在路由器 B 上正在实施，但是路由器 C 还是具有关于 172.17.0.0 的路由表项，为什么？

4．在图 13-9 中，路由器运行 RIP，路由器 B 的配置见示例 13-37。

示例 13-37 路由器 B 的 RIP 配置

```
router rip
network 172.19.0.0
network 172.20.0.0
distribute-list 1 out Ethernet0
distribute-list 2 out Ethernet1
!
access-list 1 permit 172.18.0.0
access-list 2 permit 172.22.0.0
```

上面配置的目的是向路由器 A 仅通告网络 172.22.0.0，向路由器 C 仅通告网络 172.18.0.0。但是，在路由器 A 和路由器 C 的路由表中均没有 RIP 路由表项。错误在哪里？

本章包括以下主题：

- 路由映射的基本用途；
- 配置路由映射。

第 14 章

路由映射

路由映射与访问列表十分相似，它们都包含匹配确定数据包细节的准则、许可及拒绝这些数据包的操作。但是路由映射不像访问列表，它可以向匹配准则中加入设置准则，设置准则可以按照指定的方式真正地对数据包和路由信息进行修改，而且路由映射还有更多的选项用来匹配给定的数据包。总而言之，在你的网络中，路由映射是一个定制路由策略的强大工具。

14.1 路由映射的基本用途

路由映射可以用于路由重新分配和策略路由；虽然在前面几章对重新分配进行了广泛地讨论，但本章介绍的主题是策略路由。策略路由在大规模边界网关协议（BGP）的运行中， 是一个最必不可少的工具。对于使用路由映射实现 BGP 路由策略已超出了本书的范围，但在“TCP/IP 路由技术”第二卷中包括了这个内容（CCIE 专业开发）（1-57870-089-2）。

策略路由（policy route）只不过是复杂的静态路由。策略路由可以基于数据包源地址包头中的其他域向指定的下一跳路由器转发数据包，而静态路由是基于数据包的目的地址向指定的下一跳路由器转发数据包。策略路由还可以链接到扩展 IP 访问列表，使路由器可以基于协议类型和端口号进行路由选择。同静态路由一样，策略路由只会影响那些配置了策略路由的路由器。

图 14-1 给出了一个典型的策略路由应用的例子。AbnetNet 经路由器 Dogpath 连接到两个 Internet 服务提供商。AbnerNet 的公司策略规定一些用户的 Internet 流量经 ISP1 发送，而其他用户的 Internet 流量要经 ISP2 发送。如果其中一

个 ISP 不可达，那么流经该提供商的流量将被倒换到另一个提供商。在 Dogpatch 上的策略路由会根据本地策略对 Intenet 流量进行分配。流量的分配可以基于子网、特殊的用户甚至用户应用。

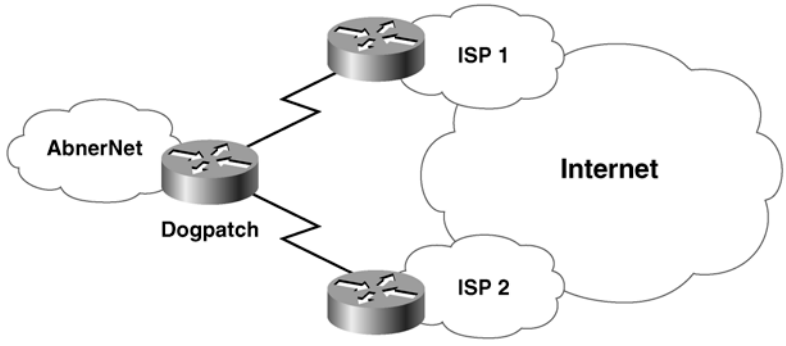


图 14-1 策略路由允许来自 AbnerNetde 的流量被路由到两个 Internet 服务提供商中的一个，流量的分配可以基于诸如源地址、源/目的地址组合、数据包大小或应用层端口等参数

图 14-2 给出了策略路由的另一种用法。右边的一个系统监测来自行星 Mongo 的入侵军队，而另一个系统保存了过去的 Dilbert 连环漫画。我们可以配置策略路由使得从 Mongo 系统到 Flash_G 的关键流量经过 FDDI（光纤分布式数据接口）链路，而优先级较低的 Dilbert 流量经过 56kbit/s 链路。反之亦然。

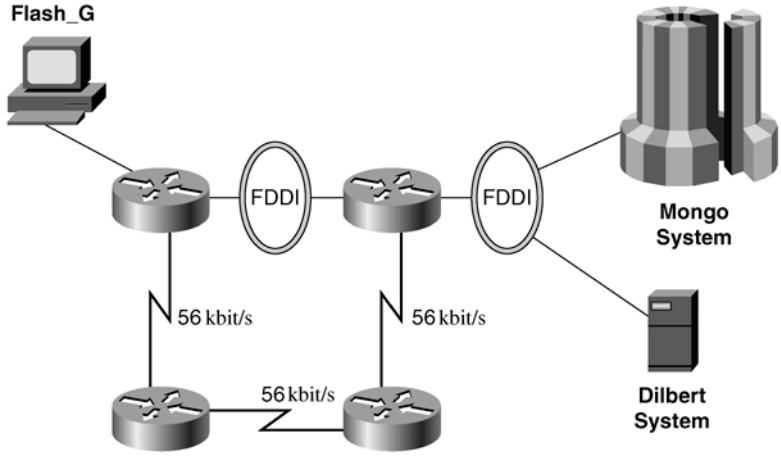


图 14-2 策略路由允许来自 Mongo 系统优先级较高的流量使用 FDDI 链路，而来自 Dilbert 系统优先级较低的流量使用 56kbit/s 链路

表 14-1 和表 14-2 给出了与重新分配一起使用的命令 **match** 和 **set**，表 14-3 和表 14-4 给出了与策略路由一起使用的命令 **match** 和 **set**。

表 14-1 **match 命令可以与重新分配一起使用**

命令	描述
match interface *type number* [*...type number*]	匹配被指定下一跳接口的路由
match ip address {*access-list-number*|*name*} [*...access-list- number*| *name*]	匹配被访问列表指明目标地址的路由

续表

命令	描述
match ip next-hop {*access-list-number*|*name*} [*...access-list-number*|*name*]	匹配被访问列表指明下一跳路由器地址的路由
match ip route-source {*access-list-number*|*name*}[*...access-list-number*|*name*]	匹配路由，其中通告该路由的路由器被访问列表指明
match metric *metric-value*	匹配带有指定度量的路由
match route-type {**internal**|**external**[**type-1**|**type-2**]|**level-1**|**level-2**}	匹配指定类型的 OSPF、EIGRP 或 IS-IS 路由
match tag *tag-value* [*...tag-value*]	匹配带有指定标记的路由

表 14-2 **set 命令可以与重新分配一起使用**

命令	描述
set level{level-1|level-2|level-1-2|stub-area|backbone}	设置 IS-IS 层或 OSPF 区域，其中匹配成功的路由将要被重新分配进入该区域
set metric {*metric-value*|*bandwidth delay reliability loading mtu*}	为匹配成功的路由设置度量值
set metric-type {internal|external|type-1|type-2}	为匹配成功的路由设置度量类型，该路由将要被重新分配进入 IS-IS 或 OSPF
set next-hop *next-hop*	为匹配成功的路由设置下一跳路由器的地址
set tag *tag-value*	为匹配成功的路由设置标记值

表 14-3 **match 命令可以与策略路由一起使用**

命令	描述
match ip address {*access-list-number*|*name*}[*...access-list-number*|*name*]	匹配带有标准或扩展访问列表中指定特征的数据包
match length *min max*	匹配数据包的第 3 层长度

表 14-4 **set 命令可以与策略路由一起使用**

命令	描述
set default interface *type number* [*...type number*]	当不存在指向目标网络的显式路由时，为匹配成功的数据包设置出站接口
set interface *type number* [*...type number*]	当存在指向目标网络的显式路由时，为匹配成功的数据包设置出站接口
set ip default next-hop *ip-address* [*...ip-address*]	当不存在指向目标网络的显式路由时，为匹配成功的数据包设置下一跳路由器地址
set ip next-hop *ip-address* [*...ip-address*]	当存在指向目标网络的显式路由时，为匹配成功的数据包设置下一跳路由器地址
set ip precedence *precedence*	为匹配成功的 IP 数据包设置服务类型字段的优先级位
set ip tos *type-of-service*	为匹配成功的数据包设置服务类型字段的 ToS 位

14.2 配置路由映射

与访问列表一样（参见附录 B），路由映射本身不会对任何东西产生影响，它们必须被某些命令所调用。这些命令最可能是策略路由选择命令或者是重新分配命令。策略路由选择将数据包发送到路由映射，而重新分配是将路由发送到路由映射。本节的案例研究将给出在重

新分配和策略路由选择中路由映射的用法。

路由映射是通过名字来标识的。例如，在示例 14-1 中的路由映射被命名为 Hagar。

示例 14-1 配置中定义了名为 Hagar 的路由映射

```
route-map Hagar permit 10
match ip address 110
set metric 100
```

每条路由映射语句都包含“许可”、“拒绝”操作及一个序列号。这个路由映射给出了一个许可操作和序列号 10。这些设置都是缺省的——也就是在配置路由映射时如果没有指定操作或序列号，那么路由映射的操作和序列号缺省值就是序列号 10。

序列号允许说明和编辑多个语句，考虑示例 14-2 中的配置步骤。

示例 14-2 修改路由映射 Hagar

```
route-map Hagar 20
 match ip address 111
 set metric 50
route-map Hagar 15
 match ip address 112
 set metric 80
```

这里，向路由映射 Hagar 添加了第二组和第三组路由映射语句，其中每组语句都包含自己的 **match** 和 **set** 设置语句。注意，第一次配置的序列号是 20，第二次配置的序列号是 15。如示例 14-3 所示，最终的配置中，尽管语句 15 是在后面被输入的，[1]但 IOS 还是把语句 15 放在了语句 20 的前面。

示例 14-3 IOS 按照顺序放置命令

```
route-map Hagar permit 10
 match ip address 110
 set metric 100
!
route-map Hagar permit 15
 match ip address 112
 set metric 80
!
route-map Hagar permit 20
 match ip address 111
 set metric 50
```

有了序列号以后，路由器还允许删除个别语句。例如下面的语句：

```
Linus(config)#no route-map Hagar 15
```

这里删除了语句 15，而其他语句被完整无缺地保留下来（参见示例 14-4）。

示例 14-4 删除与序列号 15 相关的匹配/设置语句后的路由映射 Hagar

```
route-map Hagar permit 10
 match ip address 110
 set metric 100
!
route-map Hagar permit 20
 match ip address 111
 set metric 50
```

[1] 还要注意一点，在配置中没有指定操作，所以缺省操作“许可”出现在最终的配置中。

编辑路由映射时必须谨慎。在这个例子中，如果输入了 **no route-map Hagar**，而没有指明序列号，那么整个路由映射将会被删除。同样的，如果在添加 **match** 和 **set** 语句时没有指定序列号，那么它们仅会修改语句 10。[1]

路由映射语句对数据包和路由的匹配是按序进行的。如果匹配成功，那么将执行所有 **set** 语句及许可或拒绝操作。如同使用访问列表一样，当匹配发生时，处理马上停止，指定的操作将被执行；此后路由或数据包不再传递给后继语句。下面考虑示例 14-5 的路由映射。

示例 14-5 路由映射 Sluggo

```
route-map Sluggo permit 10
 match ip route-source 1
 set next-hop 192.168.1.5
!
route-map Sluggo permit 20
 match ip route-source 2
 set next-hop 192.168.1.10
!
route-map Sluggo permit 30
 match ip route-source 3
 set next-hop 192.168.1.15
```

如果路由不能匹配到语句 10，那么它将被传递到语句 20。如果在语句 20 匹配发生，那么将执行设置命令，并且路由被允许。匹配成功的路由将不会被传递给语句 30。

拒绝操作的行为依赖于路由映射是用于策略路由选择还是重新分配。如果用于重新分配，那么路由将不会被重新分配。如果用于策略路由选择，则不对数据包进行策略路由，但是数据包会被传递给常规路由选择进行转发。

与访问列表一样，如果数据包或路由没有匹配到任何一个语句，那么对路由映射来说必须执行一个缺省操作。在每个路由映射的结尾都隐含了一个拒绝操作。经过重新分配路由映射的路由若没有发生匹配，则不会被重新分配。而经过策略路由映射而没有发生匹配的数据包将会按常规路由选择进程转发。

如果在路由映射语句中没有配置 **match** 语句，那么缺省操作是匹配所有数据包和路由。

在示例 14-6 中，每个路由映射语句可能有多个 **match** 和 **set** 语句。

示例 14-6 Garfield 包含多个与序列号 10 相关联的匹配和设置语句

```
route-map Garfield permit 10
match ip route-source 15
match interface Serial0
set metric-type type-1
set next-hop 10.1.2.3
```

在这个案例研究中若要执行 **set** 语句，每个 **match** 语句中都必须发生匹配。

14.2.1 案例研究：策略路由选择

使用命令 **ip policy route-map** 可以定义策略路由选择。该命令配置在接口且仅会对入站

[1] 在一些 IOS 版本中，如果你在修改路由映射的匹配和设置语句时没有指定序列号，那么将会产生一条 IOS 消息——“% Please specify the entry by its sequence”，而不是假定去修改与序号 10 相关的语句。

数据包有影响 。

在图 14-3 中，假设在 Linus 上实现了一个策略，使来自 172.16.6.0/24 的流量被转发到 Lucy，而把来自 172.16.7.0/24 的流量转发到 Pigpen。示例 14-7 给出了 Linus 的配置。

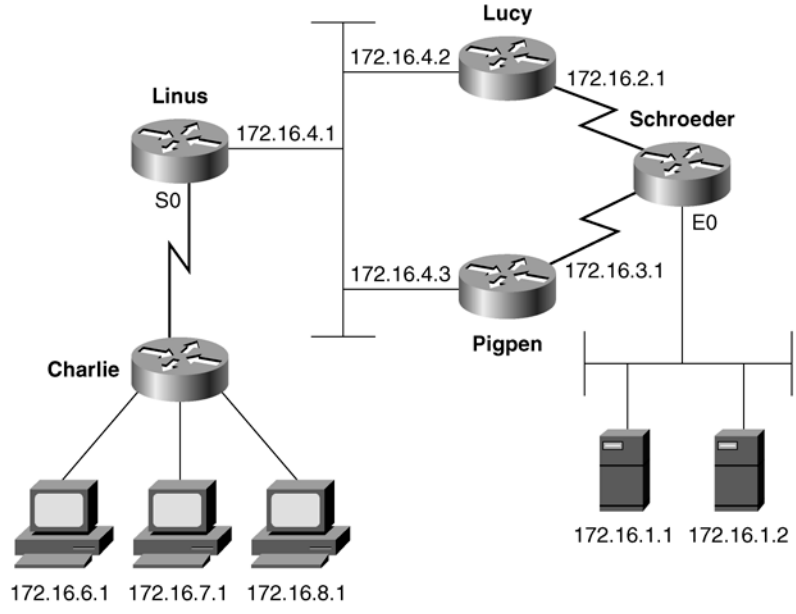


图 14-3 在 Linus 上配置策略路由，使某些数据包经过 Lucy，而其他数据包经过 Pigpen

示例 14-7 在 Linus 上的策略路由配置

```
interface Serial0
 ip address 172.16.5.1 255.255.255.0
 ip policy route-map Sally
!
access-list 1 permit 172.16.6.0 0.0.0.255
access-list 2 permit 172.16.7.0 0.0.0.255
!
route-map Sally permit 10
 match ip address 1
 set ip next-hop 172.16.4.2
!
route-map Sally permit 15
 match ip address 2
 set ip next-hop 172.16.4.3
```

S0 上的策略路由选择命令把入站数据包发送给路由映射 Sally。路由映射 Sally 的语句 10 使用访问列表 1 标识来自子网 172.16.6.0/24 的源地址。如果匹配成功，数据包将被转发到 Lucy，其中数据包的下一跳接口地址是 172.16.4.2。如果匹配不成功，数据包被发送给语句 15。该语句使用访问列表 2 匹配来自子网 172.16.7.0/24 的源地址。如果匹配成功，数据包被转发给 Pigpen（172.16.4.3）。任何没有匹配到语句 15 的数据包，例如来自子网 172.16.8.0/24 的数据包，将会被正常地路由。示例 14-8 给出了策略路由的结果。[1]

[1] 注意命令 **debug ip packet** 参考了访问列表 5。为了使调试功能对不感兴趣的流量不作显示，该访问列表仅允许连接在路由器 Charlie 的子网。

示例 14-8 配置在 Linus 接口 S0 上的策略路由把来自子网 172.16.6.0/24 的数据包路由到 Lucy（172.16.4.2），把来自子网 172.16.7.0/24 的数据包路由到 Pigpen（172.16.4.3）。来自子网 172.16.8.0/24 的数据包因没有匹配到策略路由，所以被正常地路由（在 Lucy 和 Pigpen 之间进行负载均衡）

```
Linus#debug ip packet 5
IP packet debugging is on for access list 5
Linus#
IP: s=172.16.7.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.6.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.6.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.6.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.6.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.8.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.8.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.8.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.8.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.3, len 60, forward
```

假设 Lucy 的以太网接口发生故障。Linus 将试图把来自 172.16.6.0 的数据包转发到 Lucy 的 IP 地址。因为在路由映射中数据包一旦匹配成功，即使指定的下一跳接口失效，也不会再次进行匹配，而且数据包也不能按常规被路由。在转发数据包之前为了强制 Linus 验证下一跳地址是否可用，可以使用命令 **set ip next-hop verify-availability**。Linus 将搜索 CDP 邻居表来验证下一跳地址是否在列表中；如果不在，则策略路由被拒绝，数据包将按常规被转发。示例 14-9 给出了当 Lucy 的以太网接口失效时 **debug ip policy** 和 **debug arp** 的输出结果。可以看出，数据包匹配成功并按照策略进行路由，但因路由器发出 ARP 请求后没有受到响应，所以数据包被丢弃。

示例 14-10 给出了在向 IP 的策略配置中添加命令 **set ip next-hop verify-availability** 后命令 **debug ip policy** 的输出。这时 Lucy 的以太网接口依然失效，数据包匹配成功，因为下一跳地址不在 Linus 的 CDP 表中，所以策略被拒绝，数据包最终按照常规方式被路由。

示例 14-9 在 Linus 上的 **debug ip policy** 和 **debug arp** 显示：即使策略指定的下一跳不可用，路由器依然根据配置的策略对数据包进行路由

```
Linus#debug arp
ARP packet debugging is on
Linus#debug ip policy
Policy routing debugging is on
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, FIB policy match
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, policy match
IP: route map Sally, item 10, permit
IP: s=172.16.6.1 (Serial0), d=172.16.2.1 (Ethernet0), len 100, policy routed
IP: Serial0 to Ethernet0 172.16.4.2
IP ARP: sent req src 172.16.4.1 0004.c150.e700,
                dst 172.16.4.2 0000.0000.0000 Ethernet0
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, FIB policy match
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, policy match
IP: route map Sally, item 10, permit
IP: s=172.16.6.1 (Serial0), d=172.16.2.1 (Ethernet0), len 100, policy routed
IP: Serial0 to Ethernet0 172.16.4.2
IP ARP: sent req src 172.16.4.1 0004.c150.e700,
```

（待续）

```
                dst 172.16.4.2 0000.0000.0000 Ethernet0
IP ARP: creating incomplete entry for IP address: 172.16.4.2 interface Ethernet0
IP ARP: sent req src 172.16.4.1 0004.c150.e700,
                dst 172.16.4.2 0000.0000.0000 Ethernet0
IP ARP throttled out the ARP Request for 172.16.4.2
```

示例 14-10 Linus 上的 **debug ip** 显示：在配置了 **set ip next-hop verify-availavbility** 后，因为下一跳不在 Linus 邻居表中，策略被拒绝，因而数据包按照常规方式被路由

```
Linus#debug ip policy
Policy routing debugging is on

IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, FIB policy match
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, FIB policy rejected - normal
forwarding
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, FIB policy match
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, FIB policy rejected - normal
forwarding
```

当 Lucy 的以太网接口恢复正常后，示例 14-11 给出了 **debug ip policy** 的输出，可以看出数据包按照策略成功地被路由。

示例 14-11 在配置了 **set ip next-hop verify-availavbility** 的 Linus 上使用命令 **debug ip policy**，从结果可以看出：当对下一跳的验证成功后，数据包即可以被策略路由

```
Linus#
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, FIB policy match
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, g=172.16.4.2, len 100, FIB policy routed
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, len 100, FIB policy match
IP: s=172.16.6.1 (Serial0), d=172.16.2.1, g=172.16.4.2, len 100, FIB policy routed
```

当仅按照源地址进行策略路由选择时可以使用标准 IP 访问列表。如果需要借助源地址和目标地址进行路由，那么要使用扩展访问列表。如果需要把从任意子网到主机 172.16.1.1 的数据包转发到 Lucy，其次把从主机 172.16.7.1 到 172.16.1.2 的数据包转发给 Pigpen，其他所有数据包被正常地路由，那么示例 14-12 给出的配置可以实现这个要求。

示例 14-12 策略路由映射可以参考扩展 IP 访问列表对指定的源目地址对进行匹配

```
interface Serial0
ip address 172.16.5.1 255.255.255.0
ip policy route-map Sally
!
access-list 101 permit ip any host 172.16.1.1
access-list 102 permit ip host 172.16.7.1 host 172.16.1.2
!
route-map Sally permit 10
match ip address 101
set ip next-hop 172.16.4.2
!
route-map Sally permit 15
match ip address 102
set ip next-hop 172.16.4.3
```

这里再次使用路由映射 Sally，现在除了 **match** 语句参照访问列表 101 和 102 外，其他没有变化。产生的结果见示例 14-13。

示例 14-13 从主机 172.16.7.1 到主机 172.16.1.1 的数据包被路由映射 Sally 中的语句 10 匹配成功，因而被转发给 Lucy。从相同主机到主机 172.16.1.2 的数据包则被转发到 Pigpen。从子网 172.16.7.0/24 上的另一个地址到主机 172.16.1.2 的数据包没有被 Sally 匹配到，因而被正常地路由

```
Linus#debug ip packet 5
IP packet debugging is on for access list 5
Linus#
IP: s=172.16.7.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.1 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.2 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.2 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.2 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.7.1 (Serial0), d=172.16.1.2 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.7.254 (Serial0), d=172.16.1.2 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.7.254 (Serial0), d=172.16.1.2 (Ethernet0), g=172.16.4.2, len 60, forward
IP: s=172.16.7.254 (Serial0), d=172.16.1.2 (Ethernet0), g=172.16.4.3, len 60, forward
IP: s=172.16.7.254 (Serial0), d=172.16.1.2 (Ethernet0), g=172.16.4.2, len 60, forward
```

其次，假设你的策略规定：把从子网 172.16.1.0/24 上的服务器发出的 FTP 流量转发到 Lucy，而把从相同服务器发出的 Telnet 流量转发到 Pigpen。这个计划使大容量 FTP 流量和突发、活跃的 Telnet 在 Schroeder 处被分离到两条串行链路上。示例 14-14 给出了 Schroeder 的相应配置。

示例 14-14 Schroeder 的策略路由配置用于转发 FTP 和 Telnet 流量

```
interface Ethernet0
ip address 172.16.1.4 255.255.255.0
ip policy route-map Rerun
!
access-list 105 permit tcp 172.16.1.0 0.0.0.255 eq ftp any
access-list 105 permit tcp 172.16.1.0 0.0.0.255 eq ftp-data any
access-list 106 permit tcp 172.16.1.0 0.0.0.255 eq telnet any
!
route-map Rerun permit 10
match ip address 105
set ip next-hop 172.16.2.1
!
route-map Rerun permit 20
match ip address 106
set ip next-hop 172.16.3.1
```

访问列表 105 和 106 不仅检查源地址和目的地址，而且还检查源端口。在示例 14-15 中，使用 **debug ip packet** 和选项 **detail**，可以观察到 Schroeder 所转发的数据包类型。访问列表 10 对所要显示的数据包作出了限制，仅允许显示从 172.16.1.1 到 172.16.6.1 的数据包。

示例 14-15 FTP 数据包（TCP 端口 20 和 21）被转发到 Lucy，而源地址和目地地址相同的 Telnet 数据包（TCP 端口 23）则被转发到 Pigpen。回应应答数据包（ICMP 类型 0）在策略路由中没有找到匹配语句，将被正常地路由

```
Schroeder#debug ip packet detail 10
IP packet debugging is on (detailed) for access list 10
Schroeder#
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial0), g=172.16.2.1, len 1064, forward
    TCP src=20, dst=1047, seq=3702770065, ack=591246297, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial0), g=172.16.2.1, len 64, forward
    TCP src=21, dst=1046, seq=3662108731, ack=591205663, win=14335 ACK PSH
```

（待续）

```
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial0), g=172.16.2.1, len 1476, forward
    TCP src=20, dst=1047, seq=3702771089, ack=591246297, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 40, forward
    TCP src=23, dst=1048, seq=3734385279, ack=591277873, win=14332 ACK
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 52, forward
    TCP src=23, dst=1048, seq=3734385279, ack=591277873, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 40, forward
    TCP src=23, dst=1048, seq=3734385291, ack=591277876, win=14332 ACK
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial0), g=172.16.2.1, len 60, forward
    ICMP type=0, code=0
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 60, forward
    ICMP type=0, code=0
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial0), g=172.16.2.1, len 60, forward
    ICMP type=0, code=0
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 60, forward
    ICMP type=0, code=0
```

分割批量和交互式流量的目的是使小数据包特性的交互式流量不受大数据包特性的批量流量影响而增大时延。最后一个例子也是这样的。在本例中该方法的缺点是，如果需要隔离的流量类型很多，那么通过目标端口标识流量会使访问列表变得惊人的庞大。

如果策略的目标是将小数据包从大数据包中分离出来，那么可以对数据包长度进行匹配，相应配置见示例 14-16。

示例 14-16 Schroeder 的策略路由配置将基于数据包长度转发流量

```
interface Ethernet0
ip address 172.16.1.4 255.255.255.0 ip policy route-map Woodstock !
route-map Woodstock permit 20
match length 1000 1600
set ip next-hop 172.16.2.1
!
route-map Woodstock permit 30
match length 0 400
set ip next-hop 172.16.3.1
```

在这里，**match length** 语句指明了数据包长度的最小值和最大值。路由映射的语句 20 使所有长度在 1000～1600 字节之间的数据包经过串行链路到达 Lucy。语句 30 使所有长度大于 400 字节的数据包经过串行链路到达 Pigpen。长度在 400～1000 字节之间的数据包则被正常地路由。

示例 14-17 给出了新路由映射的结果。从 172.16.1.2 到 172.16.6.1 的 FTP、Telnet 和回应应答数据包现在将根据它们的大小被路由，而不是按照数据包的地址和端口。

示例 14-17 长度大于等于 1000 字节的数据包被路由到 Lucy，而长度小于等于 400 字节的数据包被路由到 Pigpen。所有长度在 400～1000 字节之间的数据包将被正常地路由

```
Schroeder#debug ip packet detail 10
IP packet debugging is on (detailed) for access list 10
Schroeder#
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial0), g=172.16.2.1, len 1476, forward
    TCP src=20, dst=1063, seq=1528444161, ack=601956937, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial0), g=172.16.2.1, len 1476, forward
    TCP src=20, dst=1063, seq=1528442725, ack=601956937, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial0), g=172.16.2.1, len 1476, forward
    TCP src=20, dst=1063, seq=1528444161, ack=601956937, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 840, forward
    TCP src=20, dst=1063, seq=1528445597, ack=601956937, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 40, forward
    TCP src=21, dst=1062, seq=1469372904, ack=601897901, win=14329 ACK
```

（待续）

```
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 54, forward
    TCP src=21, dst=1062, seq=1469372904, ack=601897901, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 40, forward
    TCP src=21, dst=1062, seq=1469372918, ack=601897901, win=14335 ACK FIN
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 44, forward
    TCP src=23, dst=1064, seq=1712116521, ack=602140570, win=14335 ACK SYN
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 43, forward
    TCP src=23, dst=1064, seq=1712116522, ack=602140570, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 40, forward
    TCP src=23, dst=1064, seq=1712116525, ack=602140573, win=14332 ACK
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 52, forward
    TCP src=23, dst=1064, seq=1712116525, ack=602140573, win=14335 ACK PSH
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 60, forward
    ICMP type=0, code=0
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 60, forward
    ICMP type=0, code=0
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 60, forward
    ICMP type=0, code=0
IP: s=172.16.1.2 (Ethernet0), d=172.16.6.1 (Serial1), g=172.16.3.1, len 60, forward
    ICMP type=0, code=0
```

到目前为止，所示例的策略路由都是对从一个特殊接口进入路由器的数据包产生影响，但是路由器自己产生的数据包又会怎样？使用命令 **ip local policy route-map** 可以使这些数据包也按策略进行路由。与 **ip policy route-map** 不同，**ip policy route-map** 被配置在接口上，而命令 **ip local policy route-map** 则是被全局地配置在路由器上。

为了对 Schroeder 产生的数据包应用前面示范的策略，示例 14-18 给出了相应的配置。

示例 14-18　针对 Schroeder 产生数据包的策略路由配置

```
interface Ethernet0
 ip address 172.16.1.4 255.255.255.0
 ip policy route-map Woodstock
!
ip local policy route-map Woodstock
!
access-list 120 permit ip any 172.16.1.0 0.0.0.255
access-list 120 permit ospf any any
!
route-map Woodstock permit 10
 match ip address 120
!
route-map Woodstock permit 20
 match length 1000 1600
 set ip next-hop 172.16.2.1
!
route-map Woodstock permit 30
 match length 0 400
 set ip next-hop 172.16.3.1
```

这里特别让人感兴趣的是语句 10，该语句没有 **set** 语句，而仅是允许访问列表 120 匹配到的数据包。访问列表 120 依次允许 OSPF 数据包和所有到子网 172.16.1.0/24 的数据包。如果没有访问列表的第一行，那么由 Schroeder 产出的数据包和到子网 172.16.1.0/24 的数据包将会被语句 20 和 30 转发到错误的接口。图 14-4 说明了为什么有必要配置第二行。Schroeder 的 OSPF Hello 数据包长度为 44 字节，如果没有包括语句 10，那么 OSPF Hello 数据包将匹配到语句 30 并被转发到 Pigpen，这将切断 Lucy 与 Schroeder 之间的邻接关系。如果匹配到语句 10，OSPF 数据包将被允许且按正常路由转发，邻接关系不会发生改变。

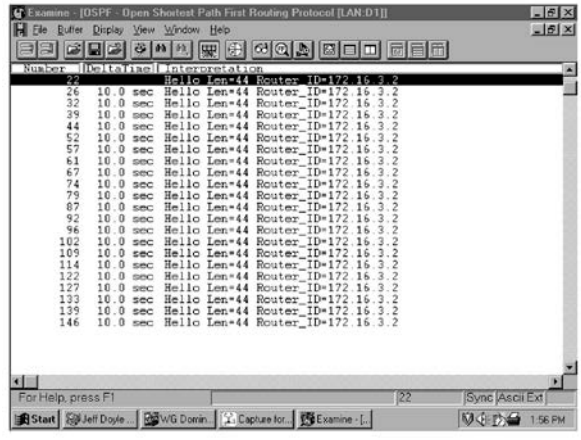


图 14-4 可以在分析仪中看到 OSPF Hello 数据包的长度

14.2.2 案例研究：策略路由选择和服务质量路由选择

虽然服务质量（QoS）路由选择超出了本卷的范围，但是这里必须注意，策略路由选择可以是 QoS 的一个重要组成部分。带有 QoS 的策略路由选择可以在数据包进入路由器接口时，通过设置数据包 IP 头内字段中的优先级和服务类型位来实现。图 14-5 给出了 ToS 字段的位信息。虽然在现代网络中很少使用 ToS 位，但是在 QoS 应用中优先级位焕发出新的生命力。ToS 位可用于改变路由器为数据包选择的路由。而优先级位则用于区分路由器中数据包的优先次序。

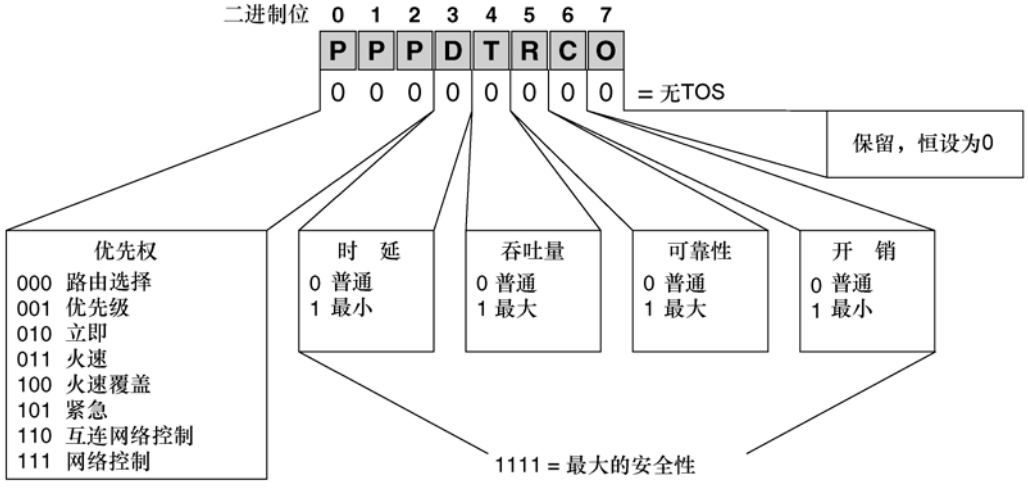


图 14-5 IP 报头服务类型字段中的优先级和 ToS 位

在路由映射中使用 **set ip precedence** 语句可以设置优先级位。通过指定 3 个优先级位的十进制等价数或关键字可以对优先级进行设置。表 14-5 给出了会用到的十进制数和关键字。

表 14-5 命令 **set ip precedence** 使用的优先级值和关键字

比特	数字	关键字
000	0	路由

续表

比特	数字	关键字
001	1	优先级
010	2	立即
011	3	火速
100	4	火速－覆盖
101	5	紧急
110	6	Internet
111	7	网络

ToS 位可以通过语句 **set ip tos** 来设置；同优先级语句一样，语句的参数可以是数字和关键字，如表 14-6 所示。与优先级不同的是，你可以使用组合 ToS 值。例如，指定 ToS 为 12（1100b）表明最小带宽和最大吞吐量。由于仅能使用一个关键字，因此为了设置组合 ToS 值，必须使用数字。

表 14-6 命令 set ip tos 用到的 ToS 值和关键字

比特	数字（0～15）	关键字
0000	0	正常
0001	1	最小开销
0010	2	最大可靠性
0100	4	最大吞吐量
1000	8	最小时延

图 14-6 给出了在 QoS 路由选择中怎样使用策略路由的例子。这里，路由器 Pogo 在网络 OkefenokeeNet 的边界。通过在 Pogo 的串行链路上配置策略路由，可以改变入站数据包的优先级位或 ToS 位，把 IP 流量区别为几种流量类型。具体配置见示例 14-19。

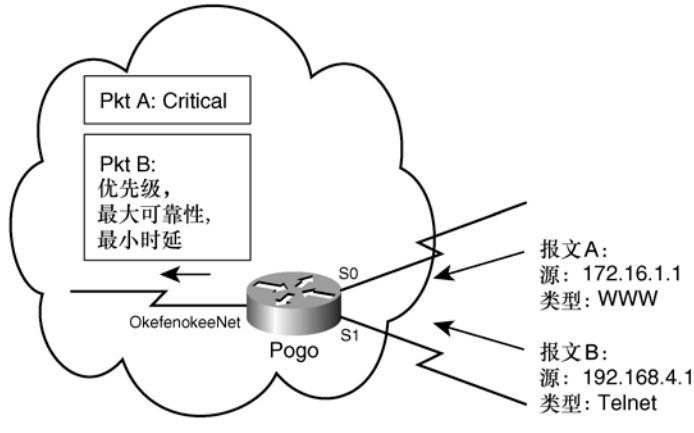


图 14-6 策略路由可以对进入网络数据包的优先级位和 ToS 位进行设置，网络中的路由器将基于对这些位的设置进行 QoS 决策

示例 14-19 在 Pogo 设置优先级和 ToS 位

```
interface Serial0
ip address 10.1.18.67 255.255.255.252
ip policy route-map Albert
!
interface Serial1
ip address 10.34.16.83 255.255.255.252
ip policy route-map Albert
!
access-list 1 permit 172.16.0.0 0.0.255.255
access-list 110 permit tcp any eq www any
!
route-map Albert permit 10
match ip address 1 110
set ip precedence critical
!
route-map Albert permit 20
set ip tos 10
set ip precedence priority
```

语句 10 说明如果数据包匹配到访问列表 1 和 110，那么优先级被设置为紧急。注意语句 20 没有 **match** 语句，那么该语句将匹配所有在语句 10 没有匹配成功的数据包。在语句 20 中还有两个 **set** 语句，这些语句将设置 ToS 位为最小延迟和最大可靠性，并且设置优先级为优先。图 14-7 给出了在 OkefenokeeNet 中捕获到的数据包，该数据包已经被 Pogo 上的路由映射修改过了。

在设置好进入网络数据包的优先级和 ToS 位之后，网络内的路由器将基于这些位所定义的部分或全部服务类别进行 QoS 决策。例如，为了对流量划分优先等级，可以根据优先级和 ToS 位配置优先级、自定义或加权公平队列。在某些实现中，优先级可被用于拥塞控制机制，例如加权随机早期检测（Weighted Random Early Detection，WRED）。或者通过配置访问列表，使其可以基于优先级或 ToS 位允许和拒绝数据包经过某条链路，以便实现粗服务类别路由选择（Class of Service, CoS）。

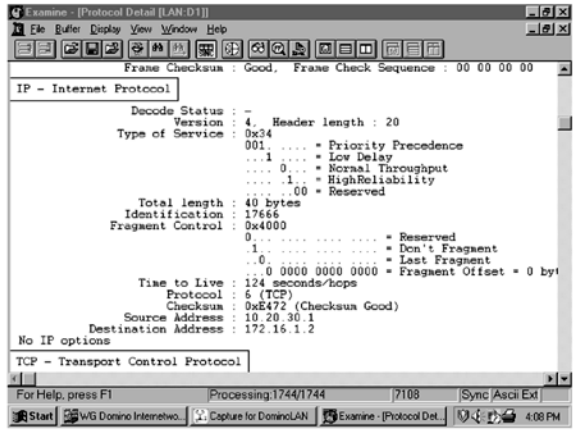


图 14-7 Pogo 的策略路由将这个数据包的优先级位设置为 001b，ToS 位设置为最小延迟和最大可靠性（1010b）

14.2.3 案例研究：路由映射和重新分配

在 IPv4 和 IPv6 协议下，通过在 **redistribute** 命令中添加对路由映射的调用就可以使路由

映射和重新分配一起使用。在图 14-8 给出的网络中，在路由器 Zippy 上，IS-IS 的 IPv4 路由和 OSPF 路由正在进行相互的路由重新分配。在图示中列出的网络和子网的地址中，仅第 3 个八位组字节为奇数的子网被重新分配。

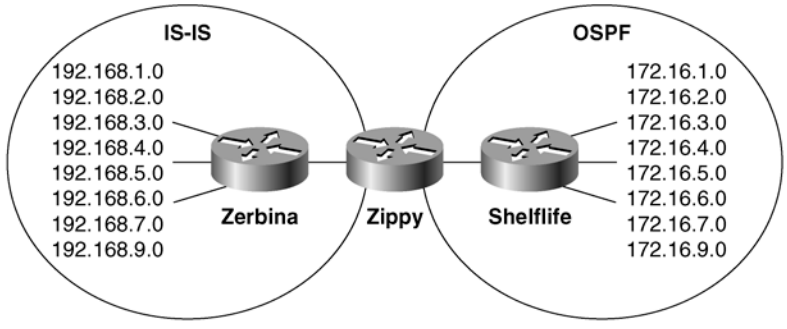


图 14-8 OSPF 和 IS-IS 正在进行相互的路由重新分配。**redistribute** 命令和路由映射一起被用作简单的路由过滤，或者说它们可以用于修改被重新分配路由的属性

Zippy 的配置见示例 14-20。

示例 14-20 Zippy 仅对第 3 个八位组字节为奇数的地址进行重新分配

```
router ospf 1
 redistribute isis level-1 metric 20 subnets route-map Griffy
 network 172.16.10.2 0.0.0.0 area 5
!
router isis
 redistribute ospf 1 metric 25 route-map Toad metric-type internal level-2
 net 47.0001.1234.5678.9056.00
!
access-list 1 permit 192.168.2.0
access-list 1 permit 192.168.4.0
access-list 1 permit 192.168.6.0
access-list 2 permit 172.16.1.0
access-list 2 permit 172.16.3.0
access-list 2 permit 172.16.5.0
access-list 2 permit 172.16.7.0
access-list 2 permit 172.16.9.0
!
route-map Griffy deny 10
 match ip address 1
!
route-map Griffy permit 20
!
route-map Toad permit 10
 match ip address 2
```

路由映射 Griffy 和 Toad 执行相同的功能，但是它们的逻辑却不相同。Griffy 使用否定逻辑，它标识那些不被重新分配的路由，而 Toad 使用肯定逻辑，它标识将要被重新分配的路由。

Griffy 的语句 10 拒绝访问列表 1 许可的任何路由（第 3 个八位组字节为偶数的地址）。因为语句 10 不能匹配到第 3 个八位组字节为奇数的地址，因此它们被传递到语句 20。语句 20 没有 **match** 命令，因而缺省匹配所有地址。语句 20 具有允许操作，所以许可奇数地址的路由。结果如示例 14-21 所示。

示例 14-21 Shelflife 路由表包含的 IS-IS 域内的目标网络，第 3 个八位组字节都为奇数

```
Shelflife#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
Gateway of last resort is not set
O  E2  192.168.9.0 [110/20] via 172.16.10.2, 00:24:46, Ethernet0
O  E2  192.168.1.0 [110/20] via 172.16.10.2, 00:24:46, Ethernet0
O  E2  192.168.3.0 [110/20] via 172.16.10.2, 00:24:46, Ethernet0
O  E2  192.168.5.0 [110/20] via 172.16.10.2, 00:24:47, Ethernet0
O  E2  192.168.7.0 [110/20] via 172.16.10.2, 00:24:47, Ethernet0
       172.16.0.0 255.255.255.0 is subnetted, 9 subnets
C         172.16.9.0 is directly connected, Serial0
C         172.16.10.0 is directly connected, Ethernet0
O         172.16.4.0 [110/159] via 172.16.9.2, 14:05:33, Serial0
O         172.16.5.0 [110/159] via 172.16.9.2, 14:05:33, Serial0
O         172.16.6.0 [110/159] via 172.16.9.2, 14:05:33, Serial0
O         172.16.7.0 [110/159] via 172.16.9.2, 14:05:33, Serial0
O         172.16.1.0 [110/159] via 172.16.9.2, 14:05:33, Serial0
O         172.16.2.0 [110/159] via 172.16.9.2, 14:05:33, Serial0
O         172.16.3.0 [110/159] via 172.16.9.2, 14:05:33, Serial0
Shelflife#
```

路由映射有一条单一语句允许访问列表 2 许可的路由（第 3 个八位组字节为奇数）。第 3 个八位组字节为偶数的地址在访问列表 2 中寻找不到匹配。当重新分配时，缺省的路由映射语句是拒绝所有路由，所以不能被访问列表 2 匹配的地址将不会被重新分配。示例 14-22 给出了路由映射 Toad 的结果。

示例 14-22 Zerbina 路由表中包含的 OSPF 域内的目标网络，第 3 个八位组字节为奇数

```
Zerbina#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
Gateway of last resort is not set
C    192.168.9.0/24 is directly connected, Serial0
C    192.168.10.0/24 is directly connected, Ethernet0
i L1 192.168.1.0/24 [115/15] via 192.168.9.2, Serial0
i L1 192.168.2.0/24 [115/15] via 192.168.9.2, Serial0
i L1 192.168.3.0/24 [115/15] via 192.168.9.2, Serial0
i L1 192.168.4.0/24 [115/15] via 192.168.9.2, Serial0
i L1 192.168.5.0/24 [115/15] via 192.168.9.2, Serial0
i L1 192.168.6.0/24 [115/15] via 192.168.9.2, Serial0
i L1 192.168.7.0/24 [115/15] via 192.168.9.2, Serial0
     172.16.0.0/24 is subnetted, 5 subnets
i L2    172.16.9.0 [115/35] via 192.168.10.2, Ethernet0
i L2    172.16.5.0 [115/35] via 192.168.10.2, Ethernet0
i L2    172.16.7.0 [115/35] via 192.168.10.2, Ethernet0
i L2    172.16.1.0 [115/35] via 192.168.10.2, Ethernet0
i L2    172.16.3.0 [115/35] via 192.168.10.2, Ethernet0
Zerbina#
```

另一个配置将实现相同的目的，例如路由映射 Toad 使用示例 14-23 的访问列表将起到同样的作用。

示例 14-23 Zippy 上路由映射 Toad 的另一种配置

```
access-list 2 deny 172.16.2.0
access-list 2 deny 172.16.4.0
access-list 2 deny 172.16.6.0
access-list 2 permit any
```

虽然路由映射可以像简单的路由过滤一样工作得很好，但是它的能力体现在可以按照多种方式改变路由。考虑示例 14-24 中 Zippy 的配置。

示例 14-24 Zippy 的路由映射配置设置了度量类型、度量值和被重新分配路由的层级

```
router ospf 1
 redistribute isis level-1 metric 20 subnets route-map Griffy
 network 172.16.10.2 0.0.0.0 area 5
!
router isis
 redistribute ospf 1 metric 25 route-map Toad metric-type internal level-2
 net 47.0001.1234.5678.9056.00
!
ip classless
access-list 1 permit 192.168.2.0
access-list 1 permit 192.168.4.0
access-list 1 permit 192.168.6.0
access-list 2 permit 172.16.9.0
access-list 2 permit 172.16.5.0
access-list 2 permit 172.16.7.0
access-list 2 permit 172.16.1.0
access-list 2 permit 172.16.3.0
!
route-map Griffy permit 10
 match ip address 1
 set metric-type type-1
!
route-map Griffy permit 20
!
route-map Toad permit 10
 match ip address 2
 set metric 15
 set level level-1
!
route-map Toad permit 20
```

路由映射 Griffy 中语句 10 允许访问列表 1 中地址的路由，并且将它们作为外部类型 1 被重新分配到 OSPF。语句 20 允许所有其他路由，并把它们作为外部类型 2 重新分配。结果见示例 14-25。

路由映射 Toad 中语句 10 允许指向访问列表 2 所定义的地址的路由，并且这些路由作为 L1 路由被重新分配到 IS-IS，度量值为 15。语句 20 允许所有其他路由，在 IS-IS 配置下的 **redistribute** 命令把这些路由作为 L2 路由重新分配，度量值为 25（参见示例 14-26）。

示例 14-25 如果路由的目标网络在 IS-IS 域，且地址的第 3 个八位组字节为偶数，则路由为 E1，如果为奇数，则为 E2

```
Shelflife#show ip route
Codes:C -connected,S -static,I -IGRP,R -RIP,M -mobile,B -BGP
      D -EIGRP,EX -EIGRP external,O -OSPF,IA -OSPF inter area
      E1 -OSPF external type 1,E2 -OSPF external type 2,E -EGP
      i -IS-IS,L1 -IS-IS level-1,L2 -IS-IS level-2,*-candidate default
```

（待续）

```
Gateway of last resort is not set
O E2 192.168.9.0 [110/20 ] via 172.16.10.2,,00:13:43,Ethernet0
O E2 192.168.1.0 [110/20 ] via 172.16.10.2,,00:13:43,Ethernet0
O E1 192.168.2.0 [110/30 ] via 172.16.10.2,,00:13:43,Ethernet0
O E2 192.168.3.0 [110/20 ] via 172.16.10.2,,00:13:44,Ethernet0
O E1 192.168.4.0 [110/30 ] via 172.16.10.2,,00:13:44,Ethernet0
O E2 192.168.5.0 [110/20 ] via 172.16.10.2,,00:13:44,Ethernet0
O E1 192.168.6.0 [110/30 ] via 172.16.10.2,,00:13:44,Ethernet0
O E2 192.168.7.0 [110/20 ] via 172.16.10.2,,00:13:44,Ethernet0
     172.16.0.0 255.255.255.0 is subnetted,9 subnets
C       172.16.9.0 is directly connected,Serial0
C       172.16.10.0 is directly connected,Ethernet0
O       172.16.4.0 [110/159 ] via 172.16.9.2,,15:49:29,Serial0
O       172.16.5.0 [110/159 ] via 172.16.9.2,,15:49:30,Serial0
O       172.16.6.0 [110/159 ] via 172.16.9.2,,15:49:30,Serial0
O       172.16.7.0 [110/159 ] via 172.16.9.2,,15:49:30,Serial0
O       172.16.1.0 [110/159 ] via 172.16.9.2,,15:49:30,Serial0
O       172.16.2.0 [110/159 ] via 172.16.9.2,,15:49:30,Serial0
O       172.16.3.0 [110/159 ] via 172.16.9.2,,15:49:30,Serial0
Shelflife#
```

示例 14-26 如果路由的目标网络在 OSPF 域且地址的第 3 个八位组字节为奇数，则路由为 L1，路由度量为 15；如果为偶数，则为 L2。路由度量为 25（加 10 是因为从 Zippy 到 Zebbina 的跳数为 10）

```
Zerbina#show ip route
Codes:C -connected,S -static,I -IGRP,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,E -EGP
      i -IS-IS,L1 -IS-IS level-1,L2 -IS-IS level-2,*-candidate default
      U -per-user static route,o -ODR
Gateway of last resort is not set
C    192.168.9.0/24 is directly connected,Serial0
C    192.168.10.0/24 is directly connected,Ethernet0
i L1 192.168.1.0/24 [115/15 ] via 192.168.9.2,,Serial0
i L1 192.168.2.0/24 [115/15 ] via 192.168.9.2,,Serial0
i L1 192.168.3.0/24 [115/15 ] via 192.168.9.2,,Serial0
i L1 192.168.4.0/24 [115/15 ] via 192.168.9.2,,Serial0
i L1 192.168.5.0/24 [115/15 ] via 192.168.9.2,,Serial0
i L1 192.168.6.0/24 [115/15 ] via 192.168.9.2,,Serial0
i L1 192.168.7.0/24 [115/15 ] via 192.168.9.2,,Serial0
     172.16.0.0/24 is subnetted,8 subnets
i L1    172.16.9.0 [115/25 ] via 192.168.10.2,,Ethernet0
i L2    172.16.4.0 [115/35 ] via 192.168.10.2,,Ethernet0
i L1    172.16.5.0 [115/25 ] via 192.168.10.2,,Ethernet0
i L2    172.16.6.0 [115/35 ] via 192.168.10.2,,Ethernet0
i L1    172.16.7.0 [115/25 ] via 192.168.10.2,,Ethernet0
i L1    172.16.1.0 [115/25 ] via 192.168.10.2,,Ethernet0
i L2    172.16.2.0 [115/35 ] via 192.168.10.2,,Ethernet0
i L1    172.16.3.0 [115/25 ] via 192.168.10.2,,Ethernet0
Zerbina#
```

14.2.4 案例研究：路由标记

图 14-9 表明，来自多个路由选择域的路由被重新分配到一个运行 OSPF 的传输域，其中每个域都运行单独的路由选择协议。在 OSPF 域的另一边，路由必须被重新分配回到它们各自的域。在从 OSPF 域进入每个域的出口点，可以使用路由过滤器以允许仅属于该域的路由通过。然而，如果每个域的路由很多或变动很大，那么路由过滤器也可能会很难管理。

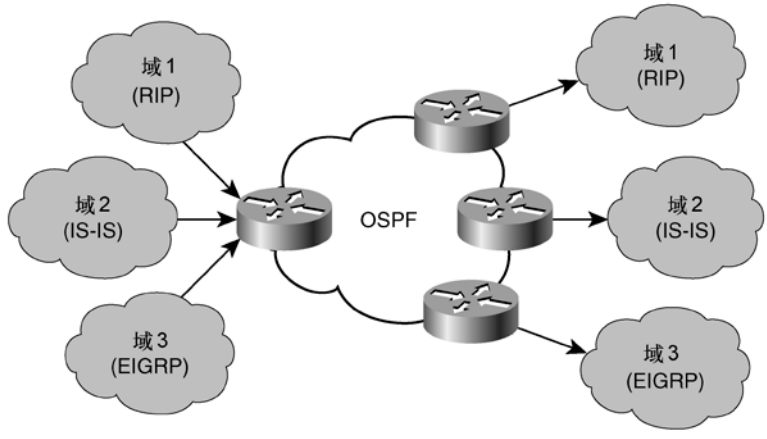


图 14-9 图左边 3 个域的路由被重新分配到一个运行 OSPF 的传输域。图右边域的路由必须被重新分配回产生它们的域

处理这个问题的另一种方法是，在 OSPF 传输域的入口对路由进行标记（tag），该标记在每个域内均是惟一的。在出口处，我们可以借助标记重新分配路由，而不通过明确的地址。传输网络的路由选择协议没有必要使用该标记，而仅仅是向外部网络来回传送它们。RIPv2、EIGRP、集成 IS-IS 和 OSPF 都支持路由标记。BGP 也支持路由标记。RIPv1 不支持标记。本节中有一个案例研究会讨论运行 OSPF 的传输网络如何使用路由标记。

回顾一下第 6 章、第 7 章、第 8 章和第 10 章的数据包格式，可以看出 RIPv2 消息支持 16 位标记，IS-IS 域间路由协议信息 TLV 也支持 16 为标记，而 EIGRP 外部路由 TLV 和 OSPF 5 型 LSA 支持 32 位标记。这些标记可以用十进制数表示，因此 RIPv2 所携带的标记值在 0～65 535 之间，EIGRP 和 OSPF 携带的标记值在 0～4 294 967 295 之间。

在图 14-10 中，路由器 Dagwood 正在接受来自 3 个不同路由域的路由，并且把它们重新分配到 OSPF 域。目标是标记来自每个域的路由以便在 OSPF 域内可以标识它们的源点域。来自域 1 的路由标记为 1，域 2 的标记为 2，等等。

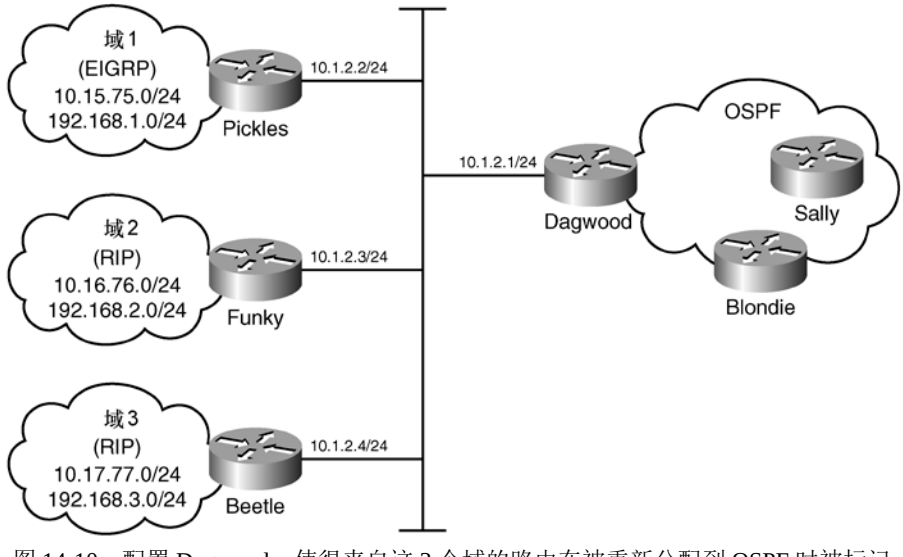


图 14-10 配置 Dagwood，使得来自这 3 个域的路由在被重新分配到 OSPF 时被标记

Dagwood 配置如示例 14-27 所示。

示例 14-27 当路由 从 RIP 和 EIGRP 被重新分配到 OSPF 时，Dagwood 对这些路由进行标记

```
router ospf 1
  redistribute eigrp 1 metric 10 subnets tag 1
  redistribute rip metric 10 subnets route-map Dithers
  network 10.100.200.1 0.0.0.0 area 0
!
router rip
  network 10.0.0.0
!
router eigrp 1
  network 10.0.0.0
!
access-list 1 permit 10.1.2.3
access-list 2 permit 10.1.2.4
!
route-map Dithers permit 10
  match ip route-source 1
  set tag 2
!
route-map Dithers permit 20
  match ip route-source 2
  set tag 3
```

首先，注意 OSPF 配置下的命令 **redistribute igrp**。Dagwood 接受的 EIGRP 路由仅来自一个 EIGRP 域，所以直接在 **redistribute** 命令上设置标记为 1。然而，RIP 路由是来自两个 RIP 域的，因此这里需要路由映射。路由映射 Dithers 设置 RIP 路由的标记为 2 或 3，具体依赖于路由是学自 Funky（10.1.2.3）还是 Beetle（10.1.2.4）。图 14-11 给出了一个 LSA 通告，被通告的路由是从 RIP 那里学到的路由之一，标记为 2。

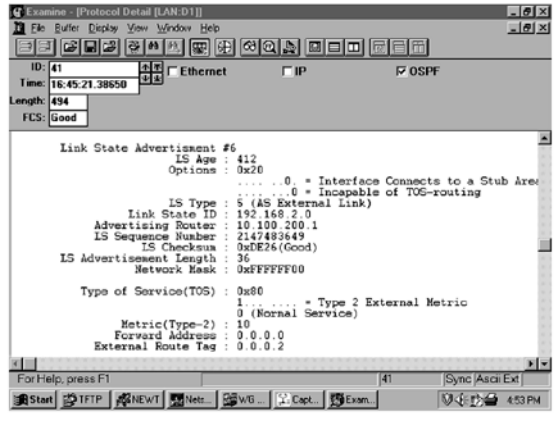


图 14-11 这个类型 5 的 LSA 正在通告域 2 内的网络 192.168.2.0，域 2 在 OSPF 域内。路由标记见最后一行

在 OSPF 的链路状态数据库中也可以观察到路由标记（参见示例 14-28）。

示例 14-28 OSPF 链路状态数据库指明了每个外部路由的标记，该标记是由 Dagwood 的重新分配进程设置的

```
Blondie#show ip ospf database
       OSPF Router with ID (10.100.200.2)(Process ID 1)
```

（待续）

```
                Router Link States (Area 0)
Link ID         ADV Router      Age     Seq#       Checksum  Link count
10.100.200.3    10.100.200.3    671     0x80000003  0x00A137  4
10.100.200.2    10.100.200.2    39      0x80000002  0x6FF5    3
10.100.200.1    10.100.200.1    40      0x80000033  0x33E1    3
                Net Link States (Area 0)
Link ID         ADV Router      Age     Seq#       Checksum
10.100.200.1    10.100.200.1    40      0x80000001  0xB0A7
                AS External Link States
Link ID         ADV Router      Age     Seq#       Checksum  Tag
192.168.2.0     10.100.200.1    641     0x80000028  0x904D    2
10.17.77.0      10.100.200.1    642     0x80000028  0xC817    3
192.168.3.0     10.100.200.1    642     0x80000028  0x9744    3
10.15.75.0      10.100.200.1    642     0x80000028  0xD213    1
10.1.2.0        10.100.200.1    642     0x80000028  0xA19B    1
10.16.76.0      10.100.200.1    642     0x80000028  0xCD15    2
192.168.1.0     10.100.200.1    644     0x80000028  0x8956    1
10.100.200.0    10.100.200.1    644     0x80000028  0x6EA4    1
Blondie#
```

在图 14-12 中，Blondie 仅可以向 Alley 重新分配域 2 的路由，向 Oop 重新分配域 1 的路由。因为这些路由在进入 OSPF 传输域时已经被标记过，按示例 14-29 的方式很容易实现这些功能。

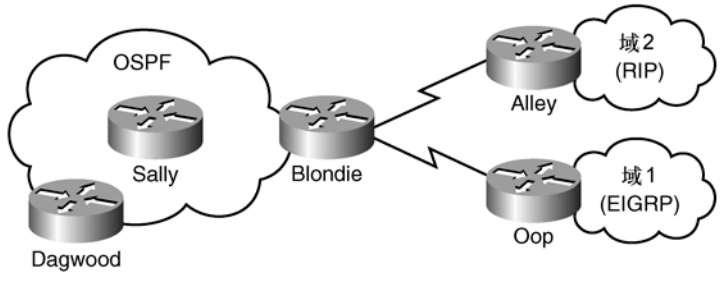


图 14-12 Blondie 使用路由映射并根据路由标记重新分配路由

示例 14-29 Blondie 使用路由映射向 EIGRP 重新分配标记为 1 的路由，向 RIP 重新分配标记为 2 的路由

```
router ospf 1
  network 10.100.200.2 0.0.0.0 area 0
!
router rip
  redistribute ospf 1 match external 2 route-map Daisy
  passive-interface Ethernet0
  passive-interface Serial1
  network 10.0.0.0
  default-metric 5
!
router eigrp 1
  redistribute ospf 1 match external 2 route-map Herb
  passive-interface Ethernet0
  passive-interface Serial0
  network 10.0.0.0
  default-metric 10000 1000 255 1 1500
!
route-map Daisy permit 10
  match tag 2
!
route-map Herb permit 10
  match tag 1
```

示例 14-30 给出了在 Alley 和 Oop 上的路由。使用路由标记过滤路由的一个缺点是不能通过接口过滤路由。例如，如果 Blondie 必须向域 2 和域 3 发送路由，而且这两个域都运行 RIP。则不可能通过配置使路由映射向一个 RIP 进程发送某些路由，而向另一个 RIP 进程发送其他的路由。这些路由必须使用命令 **distribute-list** 借助地址来过滤。

示例 14-30 在图 14-12 中，Alley 和 Oop 的路由表显示了 Blondie 的配置结果

```
Alley#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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

     10.0.0.0/8 is variably subnetted, 5 subnets, 3 masks
C       10.1.3.0/24 is directly connected, Serial0
R       10.1.5.4/30 [120/1] via 10.1.3.1, 00:00:25, Serial0
R       10.1.4.0/24 [120/1] via 10.1.3.1, 00:00:25, Serial0
R       10.16.76.0/24 [120/5] via 10.1.3.1, 00:00:25, Serial0
R       10.100.200.2/32 [120/1] via 10.1.3.1, 00:00:25, Serial0
R    192.168.2.0/24 [120/5] via 10.1.3.1, 00:00:25, Serial0
Alley#
```

```
Oop#show ip route
Codes: C - connected, S - static, I - IGRP, 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, E - EGP
       i - IS-IS, 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

     10.0.0.0/8 is variably subnetted, 6 subnets, 3 masks
D       10.1.3.0/24 [90/2681856] via 10.1.4.1, 00:21:36, Serial0
D EX    10.1.2.0/24 [170/2425856] via 10.1.4.1, 00:08:22, Serial0
D       10.1.5.4/30 [90/2681856] via 10.1.4.1, 00:22:40, Serial0
C       10.1.4.0/24 is directly connected, Serial0
D EX    10.15.75.0/24 [170/2425856] via 10.1.4.1, 00:04:56, Serial0
D       10.100.200.2/32 [90/2297856] via 10.1.4.1, 00:22:40, Serial0
D EX 192.168.1.0/24 [170/2425856] via 10.1.4.1, 00:04:56, Serial0
Oop#
```

14.2.5 案例研究：从 OSPF 路由表中滤掉被标记的路由

图 14-10 和图 14-12 中运行 OPSF 协议的网络都是一个传输网络。如果传输区中设备不需要向任何其他域发送数据包，那么在 OSPF 的路由表中就不必维护这些域的网络。因此 OSPF 路由器可以使用标记路由、分布列表和路由映射来阻止这些网络被添加到路由表中，而又不会影响到链路状态数据库中的表项。

按照示例 14-31 对 OSPF 域内的路由器 Sally 进行修改。

示例 14-31 Sally 使用标记把路由从 OSPF 路由表中过滤掉

```
router ospf 1
 network 10.100.200.1 0.0.0.0 area 0
 network 10.1.5.0 0.0.0.255 area 0
 distribute-list route-map Charlie in
!
route-map Charlie deny 10
 match tag  1 2 3
!
route-map Charlie permit 20
```

路由映射 Charlie 禁止标记为 1、2 和 3 的地址络，因此命令 **distribute-list in** 会把这些地址从路由表中删去。被路由映射语句 20 允许的所有其他地址将被添加到路由表中。在执行重新分配的边界路由器 Blondie 和 Dagwood 上没有应用这个分布列表。如果一个地址不在路由表中，即使它存在于 OSPF 的 LSA 数据库中，路由器也不会把它重新分配到其他路由选择协议。示例 14-32 给出了 Sally 的路由表和 OSPF 的 LSA 数据库。

示例 14-32 带有标记的 OSPF 地址被从路由表中过滤掉，然而这些地址仍存在于 OSPF 的 LSA 的数据库中

```
Sally#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

     10.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
C       10.1.5.4/30 is directly connected, Serial0/0.2
C       10.1.5.0/30 is directly connected, Serial0/0.1
O       10.100.200.2/32 [110/65] via 10.1.5.6, 00:17:24, Serial0/0.2
C       10.100.200.3/32 is directly connected, Loopback0
O       10.100.200.1/32 [110/65] via 10.1.5.1, 00:17:24, Serial0/0.1
```

```
Sally#show ip ospf database

            OSPF Router with ID (10.100.200.3) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#     Checksum Link count
10.100.200.1    10.100.200.1    1183        0x80000004 0x003756 3
10.100.200.2    10.100.200.2    1181        0x80000004 0x00E1A1 3
10.100.200.3    10.100.200.3    1177        0x8000000A 0x00933E 4

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
10.1.2.0        10.100.200.1    94          0x80000003 0x00EB76 1
10.15.75.0      10.100.200.1    603         0x80000002 0x001FEC 1
10.16.76.0      10.100.200.1    346         0x80000002 0x001AEE 2
10.16.77.0      10.100.200.1    353         0x80000002 0x001FEE 2
192.168.1.0     10.100.200.1    603         0x80000002 0x00D530 1
192.168.2.0     10.100.200.1    346         0x80000002 0x00DC27 2
192.168.3.0     10.100.200.1    353         0x80000002 0x00E427 2
Sally#
```

Sally 的 IP 路由表中不再包含标记为 1、2 和 3 的路由。

14.2.6 案例研究：使用路由映射重新分配 IPv6 路由

IPv6 路由选择协议同样支持使用路由映射重新分配路由。配置方法与 IPv4 基本相同。

如图 14-13 所示，在图 14-10 中的网络上添加了 IPv6 地址和路由选择协议。Funky 和 Beetle 都运行了 RIPng。Dagwood 在 RIPng 和 IS-IS 之间重新分配 IPv6 前缀。仅仅是来自 Beetel 的 IPv6 前缀被重新分配到 IS-IS 中，其中前缀 2001:db8:0:77::/64 的度量值为 10，2001:db8:0:200::/64 的度量值为 100。

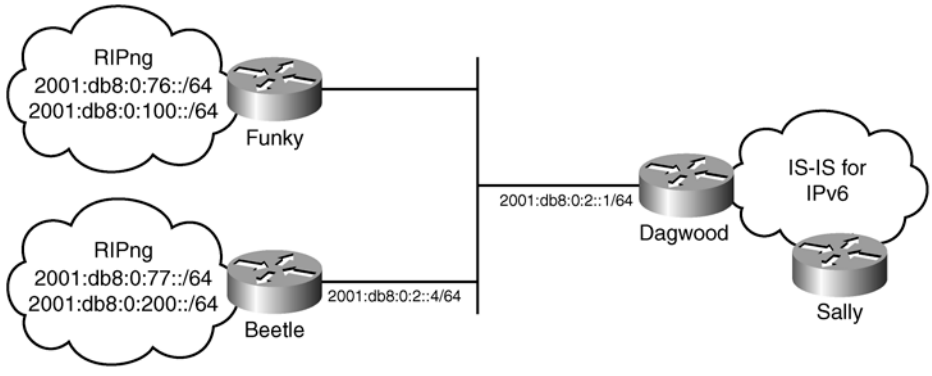


图 14-13 在图 14-10 的网络中添加了 IPv6，并且 IPv6 前缀在 RIPng 和 IS-IS 之间被重新分配

Dagwood 的配置见示例 14-33。

示例 14-33 Dagwood 的 IPv6 配置

```
Interface ethernet 0/0
 ipv6 address 2001:db8:0:2::1/64
 ipv6 rip domain3 enable
!
interface serial 0/0.1 point-to-point
 ipv6 address 2001:db8:0:5::1/64
 ipv6 router isis
!
ipv6 router rip domain3
!
router isis
 net 00.0001.0000.5678.ef01.00
 Address-family ipv6
  Redistribute rip domain3 route-map Beetlefilter
!
route-map Beetlefilter permit 10
 match ipv6 route-source prefix-list 1
 match ipv6 address prefix-list 3
 set metric 10
!
route-map Beetlefilter permit 20
 match ipv6 route-source prefix-list 1
 match ipv6 address prefix-list 2
 set metric 100
!
ipv6 prefix-list 1 permit 2001:db8:0:2::4/128
ipv6 prefix-list 2 permit 2001:db8:0:200::/64
ipv6 prefix-list 3 permit 2001:db8:0:77::/64
```

我们可以使用 IPv6 的前缀列表来匹配路由信息的源点或为重新分配指定的地址。Beetlefilter 的语句 10 指明路由的源地址必须是 Beetle 的 IPv6 地址，而且前缀地址必须是 2001:db8:0:77::/64，一旦匹配成功，度量值将被设置为 10。如果匹配失败，执行语句 20。如果源地址是 Beetel，前缀为 2001:db8:0:200::/64，那么度量值设置为 100。如果源地址不是 Beetel 或前缀也不匹配，那么该路由将不会被重新分配。

Sally 的 IS-IS（参见示例 14-34）数据库给出了被重新分配的前缀。

示例 14-34 可以从 Sally 的 IS-IS 数据库中看到被重新分配的路由

```
Sally#show isis database detail Dagwood-00.00

IS-IS Level-1 LSP Dagwood.00-00
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Dagwood.00-00         0x00000004   0xCA8B        938               0/0/0
  Area Address: 00.0001
  NLPID:        0x8E
  Hostname: Dagwood
  IPv6 Address: 2001:DB8:0:5::1
  Metric: 10         IPv6 2001:DB8:0:5::/64
  Metric: 10         IS Sally.00

IS-IS Level-2 LSP Dagwood.00-00
LSPID                 LSP Seq Num  LSP Checksum  LSP Holdtime      ATT/P/OL
Dagwood.00-00         0x00000008   0x28D7        1089              0/0/0
  Area Address: 00.0001
  NLPID:        0x8E
  Hostname: Dagwood
  IPv6 Address: 2001:DB8:0:5::1
  Metric: 10         IS Sally.00
  Metric: 10         IPv6 2001:DB8:0:5::/64
  Metric: 10         IPv6 2001:DB8:0:77::/64
  Metric: 100        IPv6 2001:DB8:0:200::/64
Sally#
```

14.3 展　　望

本章结束了本书对 TCP/IP 路由技术中内部网关协议的深入讨论，如果你正准备成为 CCIE，那么你当然想在考试之前知道本书的主题。你可以使用每章后面的复习题和配置练习测试一下你的理解能力和准备水平。如果你没有学习过有关外部网关协议的 TCP/IP 路由技术，那么在你下一步安排中，学习这部分内容是较为合理的。

14.4 总结表：第 14 章命令总结

命令	描述
access-list *access-list-number* {**deny**|**permit**} *source* [*source-wildcard*]	定义标准 IP 访问列表表项
access-list *access-list-number* {**deny**|**permit**} *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*] [**tos** *tos*] [**log**]	定义扩展 IP 访问列表表项
ip local policy route-map *map-tag*	为路由器产生的数据包定义策略路由

续表

命令	描述
ip policy route-map map-tag	为经过路由器的数据包定义策略路由
match interface *type number* [*...type number*]	匹配路由，指定接口中的一个接口去往该路由的下一跳
match ip address {*access-list-number*|*name*} [*...access-list-number*|*name*]	匹配路由，其中该路由的目标地址被访问列表指明
match ip next-hop{*access-list-number*|*name*} [*...access-list-numbe r*|*name*]	匹配路由，其中该路由的下一跳路由器地址被访问列表指明
match ip route-source{*access-list-number*|*name*} [*...access-list-numbe r*|*name*]	匹配路由，其中通告该路由的路由器被访问列表指明
match ipv6 address prefix-list {*name***}**	匹配路由，其中该路由的目标地址被前缀列表指明
match ipv6 next-hop prefix-list {*name***}**	匹配下一跳路由器地址，其中该路由的目标地址被前缀列表指明
match ipv6 route-source prefix-list {*name***}**	匹配被路由器通告的路由，其中该路由的目标地址被前缀列表指明
match length *min max*	匹配数据包的第 3 层的长度
match metric *metric-value*	匹配指定度量的路由
match route-type{internal|external[type-1| type-2]|level-1|level-2}	匹配指定类型的 OSPF、EIGRP 或 IS-IS 路由
match tag *tag-value*[*...tag-value*]	匹配指定标记的路由
ipv6 prefix-list *list-name* **[seq** *seq-number***]{deny** *ipv6 – prefix / prefix-length* **| permit** *ipv6-prefix / prefix-length* **| description** *text***} [ge** *ge-value***] [le** *le-value***]**	定义 IPv6 前缀列表
redistribute *protocol* [*process-id*]{**level-1**|**level-1-2**| **level-2**}[**metric** *metric-value*][**metric-type** *type-value*][**match**{**internal**|**external 1**|**external 2**}] [**tag** *tag-value*] [**route-map** *map-tag*][**weight** *weight*][**subnets**]	配置进入路由选择协议的重新分配，并且指明被重新分配路由的源点
set level {level-1|level-2|level-1-2|stub-area|backbone}	设置 IS-IS 层或 OSPF 区域，其中匹配成功的路由将被重新分配进入该区域
set default interface *type number* [*...type number*]	当不存在到达目标网络的显式路由时，为匹配成功的数据包设置出站接口
set interface *type number* [*...type number*]	当存在到达目标网络的显式路由时，为匹配成功的数据包设置出站接口
set ip default next-hop *ip-address* [*...ip-address*]	当不存在到达目标网络的显式路由时，为匹配成功的数据包设置下一跳路由器地址
set ip next-hop *ip-address* [*...ip-address*]	当存在到达目标网络的显式路由时，为匹配成功的数据包设置下一跳路由器地址
set ip precedence *precedence*	设置被匹配 IP 数据包服务类型字段中的优先级位
set ip tos *type-of-service*	设置被匹配数据包服务类型字段中 ToS 位
set metric{*metric-value*|*bandwidth delay reliability loading mtu*}	为被匹配路由设置度量值
set metric-type{internal|external|type-1|type-2}	为将要被重新分配到 IS-IS 或 OSPF 的被匹配路由设置度量类型
set next-hop *next-hop*	为被匹配路由设置下一跳路由器地址
set tag *tag-value*	为被匹配路由设置标记值

14.5 复 习 题

1. 路由映射有哪些方面类似于访问列表？两者有什么不同？
2. 策略路由是什么？
3. 路由标记是什么？

4．路由标记以什么样的方式影响路由选择协议？

14.6 配置练习

1．在图 14-14 中，请为路由器 A 配置策略路由，使得从子网 172.168.1.0/28 出发经过 172.16.1.112/28 到达路由器 A 的数据包被转发到路由器 D，而从子网 172.16.1.128/28 出发经 172.16.1.240/28 到达路由器 A 的报文被转发到路由器 E。

2．在图 14-14 中，为路由器 A 配置策略路由，使得从子网 172.16.1.64/28 出发经过 172.16.1.112/28 到达路由器 C 的数据包被转发到路由器 D；而相同的数据包如果到达路由器 B，则被转发到路由器 E。所有其他数据包将被正常地转发。

3．在图 14-14 中，为路由器 A 配置策略路由，使得所有经 172.16.1.240/28 去往子网 172.16.1.0/28 且源端口为 SMTP 的数据包被转发到路由器 C，任何其他去往相同子网的 UDP 数据包被转发到路由器 B。通过策略路由或常规路由选择协议没有向路由器 C 或路由器 B 转发其他的数据包。

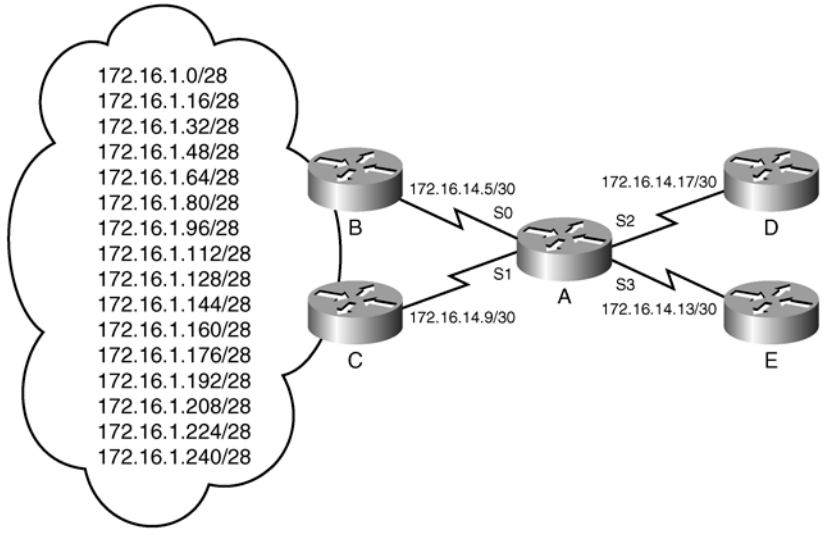


图 14-14 配置练习 1～3 中的网络

4．在图 14-15 中，路由器的 OSPF 和 EIGRP 的配置如下：

```
router eigrp 1
network 192.168.100.0
!
router ospf 1
network 192.168.1.0 0.0.0.255 area 16
```

配置路由器把内部 EIGRP 路由作为度量为 10 的 E1 路由向 OSPF 重新分配，把外部 EIGRP 路由作为度量为 50 的 E2 路由向 OSPF 重新分配。EIGRP 域内的所有网络和子网除了 10.201.100.0/24 外，都将被重新分配。

5．配置图 14-15 中的路由器，向 EIGRP 重新分配内部 OSPF 路由，其中内部 OSPF 路由的时延要小于外部 OSPF 路由的时延。仅允许 OSPF 域内的 3 个 C 类网络被重新分配。

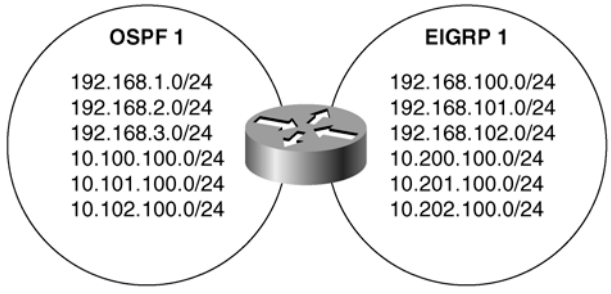


图 14-15 练习 4 和练习 5 的路由器配置

14.7 故障诊断练习

已知下面配置：

```
interface TokenRing1
ip address 192.168.15.254 255.255.255.0
ip policy route-map Ex1
!
access-list 1 permit 192.168.0.0 0.0.255.255
access-list 101 permit host 192.168.10.5 any eq telnet
!
route-map Ex1 permit 5
match ip address 1
set ip next-hop 192.168.16.254
!
route-map Ex1 permit 10
match ip address 101
set ip next-hop 192.168.17.254
```

上面配置的目的是对所有源地址前缀在 192.168.0.0 ～ 192.168.255.255 中的数据包进行策略路由。除了来自主机 192.168.10.5 的 Telnet 数据包外，所有其他数据包将被转发到 192.168.17.254。在配置中有两个错误导致策略路由工作不正常，请问错误是什么？



第四部分

附　　录

附录 A　教程：二进制和十六进制

附录 B　教程：访问列表

附录 C　CCIE 备考提示

附录 D　复习题答案

附录 E　配置练习答案

附录 F　故障诊断练习答案



附录 A

教程：二进制和十六进制

理解二进制和十六进制的最佳方法是先透彻领悟十进制计数系统。十进制（decimal）系统是基于 10 的计数系统（词根 deci-表示 10）。“基于 10”指的是由 10 个数位（digit）0 到 9 来表示数。通常我们使用十进制，这是因为我们的祖先用他们的手指（finger）来计算牛、孩子、敌人（事实上 digit 的意思就是 finger）。

使用“位值（place value）”，可以用不多的几个数位（如 10 个十进制数位）来表示很大的数。所有计数系统的位值从最右边开始，是基数的 0 次幂。从右往左，基数的幂依次增大 1：

$B^4B^3B^2B^1B^0$

基数是 10 时，前 5 个位值是：

$10^4 10^3 10^2 10^1 10^0$

对任何基数，前两个位值是最容易计算的。任何数的 0 次幂是 1，所以 $10^0=1$。任何数的 1 次幂就是它本身，所以 $10^1=10$。第三个位值也是容易计算的，只要简单地用第二个位值乘以基数就可以。事实上，每一个位值都可以用它前边一个位值乘以基数计算出来。所以上面 5 个位值是：

$10^0 = 1$

$10^1 = 1 \times 10 = 10$

$10^2 = 10 \times 10 = 100$

$10^3 = 100 \times 10 = 1\ 000$

$10^4 = 1\ 000 \times 10 = 10\ 000$

所以，对于基数是 10 的计数系统，前 5 个位值是：

10 000　1 000　100　10　1

根据位值来读一个数，比如 57 258，指的是有 5 个 10 000，7 个 1 000，2 个 100，5 个 10，以及 8 个 1。就是说：

5×10 000 = 50 000

7×1 000 = 7 000

2×100 = 200

5×10 = 50

8×1 = 8

将这些结果相加，结果是 50 000 +7 000 + 200 + 50 + 8 = 57 258。

我们对十进制都非常熟悉，所以，我们很少会去考虑将一个数分解成位值。但是，这种方法对于阐明其他进制的数是非常至关重要的。

A.1 二进制数

计算机从最底层来看，只不过是电子开关的集合而已。而数字和字符是由这些开关的状态来表示的。由于一个开关仅有两种状态——开或者关，所以它使用二进制（binary），或者说基数为 2 的计数系统（词根 *bi* 表示 2）。一个基数为 2 的系统仅仅有两个数位：0 和 1。计算机通常将这两个数位集合成 8 个位值，即一个字节（byte）或八位组字节（octet）。这 8 个位值是：

$2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0$

位值这样计算：

$2^0 = 1$

$2^1 = 1 \times 2 = 2$

$2^2 = 2 \times 2 = 4$

$2^3 = 4 \times 2 = 8$

$2^4 = 8 \times 2 = 16$

$2^5 = 16 \times 2 = 32$

$2^6 = 32 \times 2 = 64$

$2^7 = 64 \times 2 = 128$

所以一个二进制八位组的位值是： 128 64 32 16 8 4 2 1

因此，二进制八位组 10010111 可以这样理解：

1×128 = 128

0×64 = 0

0×32 = 0

1×16 = 16

0×8 = 0

1×4 = 4

1×2 = 2

1×1 = 1

或者 128 + 16 + 4 + 2 + 1 = 151

对于二进制数，因为每一个位值要么就是该值本身，要么就没有，所以比较简单。另外一个例子：11101001 = 128 + 64 + 32 + 8 + 1 = 233。就是说，将二进制转为十进制仅仅是一个

将位值相加的过程，将十进制转为二进制仅仅是将位值相减的过程。例如，要将十进制数 178 转为二进制，首先把 178 减去最高的位值：

1．178 大于 128，所以我们就知道在该位值上有一个 1：178−128＝50。
2．50 比 64 小，该位值上有一个 0。
3．50 比 32 大，所以该位值上有一个 1：50−32＝18。
4．18 比 16 大，该位值上有一个 1：18−16＝2。
5．2 比 8 小，该位值上有一个 0。
6．2 比 4 小，该位值上有一个 0。
7．2 等于 2，该位值上有一个 1：2−2＝0。
8．0 小于 1，该位值上有一个 0。

把这些步骤的结果综合起来，用二进制表示 178 就是 10110010。

另外一个例子可能会有帮助。给出 110：

1．110 比 128 小，所以在位值上有一个 0。
2．110 比 64 大，所以在位值上有一个 1：110−64 = 46。
3．46 比 32 大，所以在位值上有一个 1：46−32 = 14。
4．14 比 16 小，所以在位值上有一个 0。
5．14 比 8 大，所以在位值上有一个 1：14−8 = 6。
6．6 比 4 大，所以在位值上有一个 1：6−4=2。
7．第 2 个位值上有一个 1：2−2 = 0。
8．0 比 1 小，所以在位值上有一个 0。

所以，110 用二进制表示就是 01101110。

A.2 十六进制数

写一个二进制八位组并不有趣。对于经常要使用这些数字的人来说，受欢迎的是更简洁的表示法。一个可能的表示法是为每一个可能的八位组分配一个单独的字符。但是，8 位有 2^8=256 种不同的组合，所以，用单独的字符表示所有八位组需要 256 个数位，或者说一个基数为 256 的计数系统。

将一个八位组看作是两个各 4 位的组合或许会更简单一些。例如，11010011 可以看作是 1101 和 0011。对 4 个位来说，有 2^4=16 种不同的组合，所以有基数 16，或者说十六进制（hexadecimal）计数系统，一个八位组可以用两位来表示（词根 *hex* 的意思是“six”，*deci* 的意思是“ten”）。表 A.1 列出了十六进制数以及相应的十进制数和二进制数。

表 A-1　　十六、十和二进制数

十六进制	十进制	二进制
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100

续表

十六进制	十进制	二进制
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

因为十六进制和十进制的前 10 个数字是一样的，所以我们有意在一个十六进制数前面加 0x，或者在后面加一个 h，以便和十进制数区分开。例如，十六进制数 25 应该写成 0x25 或者 25h。本书使用 0x 表示法。

刚才学过二进制的表示法，很容易写出一个 4 位二进制数的十进制表达形式。同样也很容易将一个十进制数转为十六进制。于是，我们可以很容易地通过 3 个步骤将一个二进制八位组转为十六进制：

1．将八位组分成 2 个 4 位的二进制数。

2．将每个 4 位二进制数转为十进制。

3．把每个十进制数用十六进制来表示。

例如：把 11010011 转为十六进制：

1．11010011 变成 1101 和 0011。

2．1101=8+4+1=13，0011=2+1=3。

3．13=0xD，3=0x3。

所以，11010011 用十六进制表示就是 0xD3。

把十六进制转为二进制是上述 3 步的简单逆序。例如，把 0x7B 转为二进制：

1．0x7=7，0xB=11。

2．7=0111，11=1011。

3．把 2 个 4 位二进制数写在一起就是 0x7B=01111011，十进制为 123。



附录 B

教程：访问列表

目前对访问列表的命名可能并不是很恰当，访问列表最初的目的是许可或拒绝访问进入、离开或经过路由器的数据包。而现在访问列表已成为控制帧和数据包行为的强大工具，它们的用途可分为 3 类（如图 B-1 所示）。

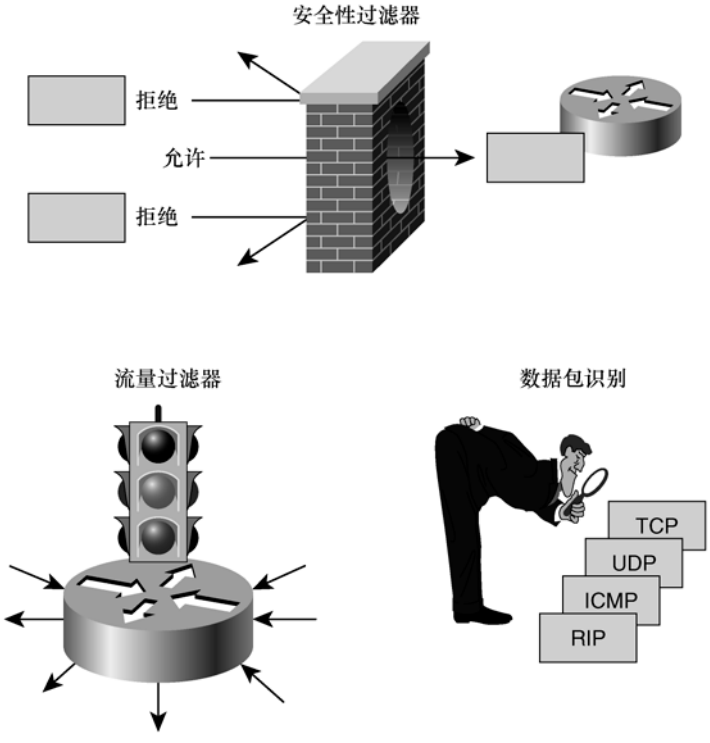


图 B-1　访问列表可以用做安全过滤器、流量过滤器或用于数据包标识

- 安全过滤器可以保护路由器及路由器传递流量所到达网络的完整性。一台安全过滤器许可少数清晰的数据包通过，而拒绝其他任何数据包通过。
- 流量过滤器可以阻止不必要的数据包通过带宽有限的链路。这些过滤器的外形和行为同安全过滤器很

相似，但是逻辑一般是颠倒的：流量过滤器拒绝少数不必要的数据包而许可其他任何数据包通过。

- 在 Cisco 路由器上的许多工具要求数据包验证，例如拨号列表、路由过滤器、路由映射和队列列表，都必须能够标识确定的数据包。访问列表可以链接到这些或其他工具上，并且提供数据包标识功能。

B.1 访问列表基础知识

一个访问列表是一组按顺序排列的过滤器。每台过滤器是由匹配标准和一个过滤动作构成的。过滤动作不是许可就是拒绝；而过滤标准既可以用像源地址一样简单的参数，也可以复杂到使用诸如源和目的地址、协议类型、端口号或套接字（Socket）和某些标记状态（如 TCP ACK 位）等参数的组合。

一个数据包从栈顶进入过滤器（如图 B-2 所示），在每台过滤器中匹配的标准将被应用，如果发生匹配，那么指定的许可或拒绝动作将被执行。如果匹配没有发生，数据包将向下移动到栈中下一台过滤器并再次重复匹配过程。

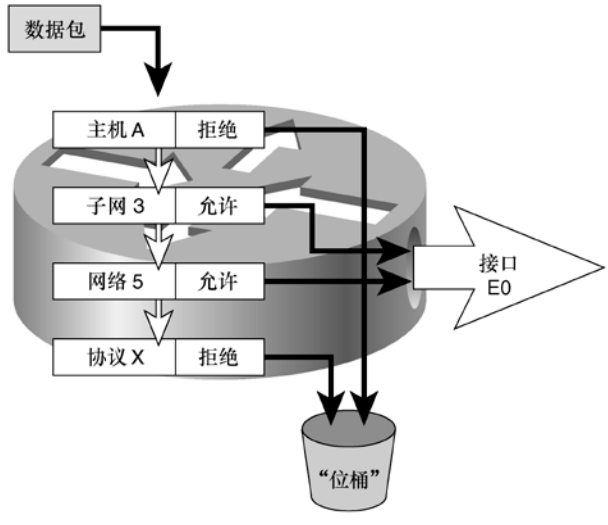


图 B-2 一个访问列表是一组按顺序排列的过滤器，每台过滤器定义了一种匹配标准和一个过滤动作

在图 B-2 中，许可意味着数据包将被允许离开接口 E0；拒绝意味着数据包将被丢弃。例如，源地址为主机 A 的数据包将在第一台过滤器中被丢弃。假设数据包的源地址是网络 5 中子网 2 上的主机 D，而第一个过滤器指明了对主机 A 的匹配标准，因此没有匹配发生，数据包进入第二层。在第二台过滤器中又指明了子网 3，因此还是没有匹配发生。数据包进入第三台过滤器，该过滤器指明了网络 5，因此匹配成功。由于本层的动作是许可，所以数据包被允许离开接口 E0。

B.1.1 隐式拒绝一切

如果数据包经过所有过滤器都没有发生匹配，那会出现什么情况？在这种情况下路由器必须知道该如何处理数据包，也就是必须有一个缺省动作。缺省动作可以是允许所有没有匹

配的数据包通过或者是拒绝它们通过。Cisco 选择了拒绝它们通过：对任何经过访问列表的数据包，如果没有发生匹配将会被自动丢弃。

这种方法是一个正确的工程选择，特别是当访问列表用于安全控制时。丢弃那些本不应该被丢弃的数据包，比起许可那些你因疏忽而没有进行过滤的数据包应该更好。

最后一台过滤器被称为隐式地拒绝一切（implicit deny any）（如图 B-3 所示）。正如其名字所暗示的，在你创建的任何一个访问列表中都不会显示该过滤器。它仅仅是一种缺省动作，并且它在所有访问列表中都处于最后。

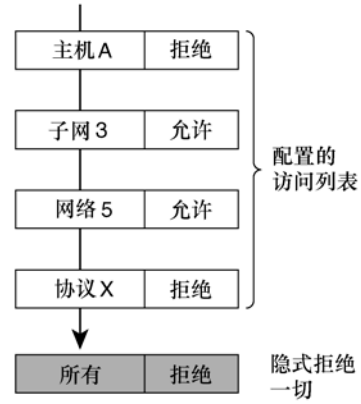


图 B-3 所有以隐式拒绝一切结尾的访问列表将丢弃那些在表中没有匹配的数据包

通过在最后一行建立显式地许可一切（explicit permit any）可以覆盖这个缺省动作。这里隐含了一点，即经过其他所有过滤器的数据包在到达缺省的拒绝一切动作之前，会匹配到许可一切动作，因而没有匹配到任何动作的数据包将被许可——不会有数据包到达隐式地拒绝动作。

B.1.2 顺序性

访问列表是从上到下按顺序执行的。这一概念很重要：或许访问列表发生故障的最普遍原因就是，过滤器的放置顺序出现错误。按顺序的访问列表的第一个匹配项总是被执行的。在第一个匹配项匹配成功后，访问列表其余的匹配项将被忽略。

在图 B-4 中，子网 10.23.147.0/24 应该被拒绝，而网络 10.0.0.0 其余的子网应该被许可。左边的访问列表顺序有错误，网络 10.0.0.0（包括它的子网 10.23.147.0）将匹配到第 1 行并且被许可。而被拒绝的子网数据包根本不会到达第 2 行。

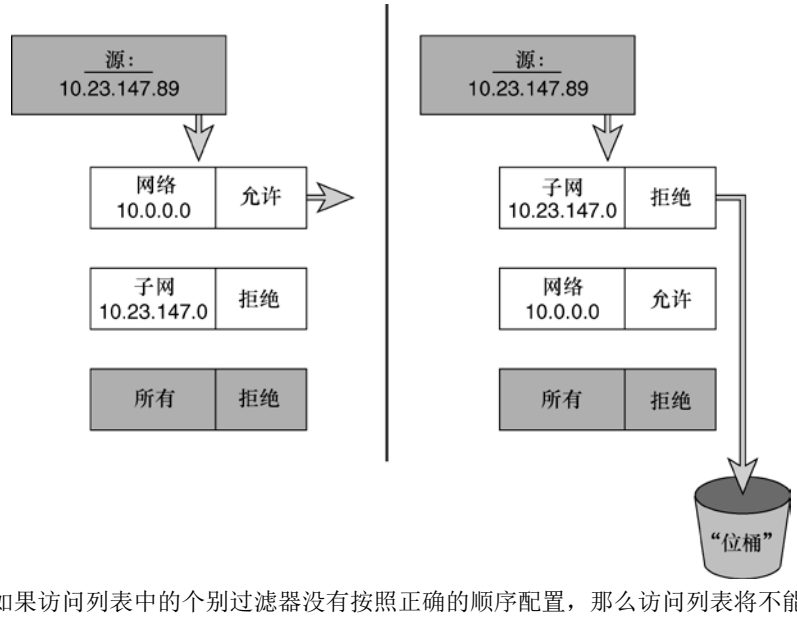


图 B-4 如果访问列表中的个别过滤器没有按照正确的顺序配置，那么访问列表将不能正常工作

右边的访问列表是准确的。子网 10.23.147.0 匹配到第 1 行并且被拒绝，而 10.0.0.0 的其余子网将会匹配到第 2 行并且被许可。

B.1.3 访问列表类型

图 B-4 中右边的访问列表的实际配置参见示例 B-1

示例 B-1 图 B-4 的第二个访问列表显示，示例中的每一行都代表了每过滤器层

```
access-list 9 deny 10.23.147.0 0.0.0.255
access-list 9 permit 10.0.0.0 0.255.255.255
```

一个配置行表示访问列表的每过滤器层。这里将简略地讨论一下访问列表中的各种组件，但请先注意在两行中都出现的数字 9 表示是访问列表编号，它主要有两个用途：

- 将访问列表中的所有行都连接在一些，使得该访问列表区别于路由器配置中的其他访问列表（通常在一个路由器中存在多个访问列表）。
- 路由器必须有区分访问列表类型的方法。Cisco IOS 软件有 IP、IPX、AppleTalk、DEC、NetBIOS、桥接和许多其他协议的访问列表，而且其中许多协议都有多种访问列表类型。访问列表编号可以表明路由器访问列表的类型。

访问列表类型可以通过数字和名字来标识。表 B-1 给出了被编号的访问列表类型和每种类型的访问列表可用的编号范围。例如，因为编号 1010 在 1000~1099 之间，所以 **access list 1010** 标识了 IPX SAP。

表 B-1 Cisco 访问列表编号

访问列表类型	范围
标准 IP	1～99，1300～1999
扩展 IP	100～199，2000～2699
以太网类型代码	200～299
以太网地址	700～799
透明桥接（协议类型）	200～299
透明桥接（厂商代码）	700～799
扩展的透明桥接	1100～1199
DECnet 和扩展的 DECnet	300～399
XNS	400～499
扩展 XNS	500～599
AppleTalk	600～699
源路由桥接（协议类型）	200～299
源路由桥接（厂商代码）	700～799
标准 IPX	800～899
扩展 IPX	900～999
IPX SAP	1000～1099
NLSP 路由汇总	1200～1299
标准 VINES	1～99
扩展 VINES	100～199
简单 VINES	200～299

在一个范围内，访问列表编号不必按照任何特殊顺序，也就是说，在路由器中不必按第一个标准 IP 列表为 1，第二个为 2 等进行编号。它们可以使用 1～99，或者 1300～1399 中任意一个数字，只要保证在一台路由器中每个访问列表编号惟一就可以。

此外，请注意，不同协议的一些编号范围是相同的，例如以太网类型代码、源路由桥接和简单 VINES。在这种情况下，路由器将会通过访问列表自身的格式来区分访问列表类型。

可以使用名字代替数字来标识下面的访问列表类型：

- Apollo 域；
- 标准 IP；
- 扩展 IP；
- ISO CLNS；
- 源路由桥接 NetBIOS；
- 标准 IPX；
- 扩展 IPX；
- IPX Sap；
- IPX NetBIOS；
- NLSP 路由汇总。

在示例 B-2 中，名字为 Boo 的访问列表用来标识 IPX NetBIOS。

示例 B-2 名字为 Boo 的访问列表拒绝了各种 NetBIOS 设备

```
netbios access-list host Boo deny Atticus
netbios access-list host Boo deny Scout
netbios access-list host Boo deny Jem
netbios access-list host Boo permit *
```

注意，虽然标准和扩展 IP 访问列表通常使用编号，但是它们也可以使用名字。在 IOS 11.2 及更高的版本中支持这一协定。在某些环境中，可能会使用大量 IP 列表配置路由器。用名字代替数字，可以更加容易地标识单独的访问列表。

命名 IP 访问列表当前仅可以与数据包和路由过滤器一起使用。更详细的信息请参考 Cisco 配置指南。

B.1.4 编辑访问列表

任何一位曾经从控制台上编辑过多行访问列表的人，都会告诉你这是一种在挫折中经受锻炼的过程。在 IOS 12.2(14)版本之前，没有办法能向列表的中间添加一行。所有新输入的行都被添加到列表尾部。如果你发现输入错误并且试图删去其中特定的一行，例如，

```
no access-list 101 permit tcp 10.2.5.4 0.0.0.255 192.168.3.0 0.0.0.255 eq 25
```

那么访问列表 101 的所有行将与这一行一起被删除。

更方便的办法是将访问列表剪切和粘贴到你 PC 的记事本，或者上载配置到 TFTP 服务器上，然后在那里编辑访问列表。当编辑结束后，新的访问列表将被载入路由器。这里提醒一下，所有新输入的行都会被添加到访问列表的尾部。记住，始终将 **no access-list #**加在被编辑访问列表的开始，其中 #是你正在编辑的访问列表编号。示例 B-3 显示了一个例子。

示例 B-3 在 PC 或服务器上开始创建一个访问列表之前，先添加一条命令 **no access-list**，这样加载到路由器上的访问列表每一次都是重新创建的

```
no access-list 5
access-list 5 permit 172.16.5.4 0.0.0.0
access-list 5 permit 172.16.12.0 0.0.0.255
access-list 5 deny 172.16.0.0 0.0.255.255
access-list 5 permit any
```

no access-list 5 将会在添加新访问列表之前，从配置文件中删除旧的访问列表 5。如果你省略了这一步，那么新列表仅仅会被添加到旧列表的结尾。

利用命令 **show access-list** 可以显示当前配置的列表，参见示例 B-4 所示。

示例 B-4 利用命令 **show access-list** 可以显示路由器上当前配置的访问列表

```
Router#show access-list 5
Standard IP access list 5
    10 permit 172.16.5.4
    20 permit 172.16.12.0, wildcard bits 0.0.0.255
    30 deny   172.16.0.0, wildcard bits 0.0.255.255
    40 permit any
Router#
```

这里请注意每一个访问列表条目前的数字。它们是序列号。在 IOS 12.2(14)S 版本后，序列号是自动增加到访问列表条目中的。这个序列号允许你在某个列表的顶端或中间插入一个条目。如果你不指定序列号，第一个条目将指定为 10，后面跟着的每一个序列号都将增加 10。当一台路由器重启时，序列号会重置为 10、20、30 等。序列号也允许我们在某个访问列表中删除指定的条目。

参见示例 B-5 中的配置，在访问列表 5 中，允许子网 172.16.20.0 中所有的主机通过的条目替代允许子网 172.16.12.0 中所有的主机通过的条目。

示例 B-5 访问列表的更新可以通过使用序列号替换或增加条目实现

```
ip access-list standard 5
no 20
20 permit 172.16.20.0 0.0.0.255
```

在增加一个序列号为 20 的新条目之前必须首先删除原有的序列号为 20 的条目，否则将会出现序列号冲突的错误。

B.2 标准 IP 访问列表

有两种进入访问列表的方式。其中之一是：

access-list *access-list-number*{**deny**|**permit**} *source[source-wildcard]*

另一个配置访问列表的方法是输入全局模式下的访问列表命令，也可以进入访问列表的配置模式。在访问列表的配置模式下，可以允许或拒绝数据包的通过，指定序列号和注释：

ip access-list standard {*access-list-number* | *name*}

采用上面这条命令可以进入访问列表配置模式。有关标准 IP 访问列表的进一步配置选项是：

```
[sequence-number] {{{deny|permit} source [source-wildcard]} | {remark
  up-to-100-characters-of-a-remark}}
```

这条命令根据表 B-1 在 1～99 和 1300～1999 之间指定了访问列表的编号、动作（许可和拒绝）、源 IP 地址、通配符（或反向）掩码。示例 B-6 中显示了一个标准 IP 访问列表的例子。

示例 B-6 标准访问列表 1 允许和拒绝多个主机和子网地址

```
access-list 1 permit 172.22.30.6 0.0.0.0
access-list 1 permit 172.22.30.95 0.0.0.0
access-list 1 deny 172.22.30.0 0.0.0.255
access-list 1 permit 172.22.0.0 0.0.31.255
access-list 1 deny 172.22.0.0 0.0.255.255
access-list 1 permit 0.0.0.0 255.255.255.255
```

例子中的前两行允许源地址属于指定主机 172.22.30.6 和 172.22.30.96 的数据包通过。这看似很恰当，尽管反码 0.0.0.0 可能没有意义。第 3 行拒绝子网 172.22.30.0 上所有其他主机，这也是相当直观的。但第 4 行的目的就不是很明显了，它允许地址范围是 172.22.0.1~172.22.31.255 的主机通过，反码指定了本行的地址范围。第 5 行拒绝 B 类网络 172.22.0.0 的所有子网，最后一行允许所有其他地址。

另一种方法也可以配置相同的列表，参见示例 B-7 所示。

示例 B-7 在这里使用访问列表配置模式也定义了示例 B-6 中显示的相同的标准 IP 访问列表

```
ip access-list standard 1
 10 permit 172.22.30.6 0.0.0.0
 15 permit 172.22.30.95 0.0.0.0
 20 deny 172.22.30.0 0.0.0.255
 permit 172.22.0.0 0.0.31.255
 deny 172.22.0.0 0.0.255.255
 permit 0.0.0.0 255.255.255.255
```

开始 3 个列表条目的序列号是指定的，第 4、第 5 和第 6 个的条目是分别由其前面的条目加 10 自动分配的，也就是 30、40 和 50。通过在访问列表中期望位置的上面和下面条目的序列号之间指定一个序列号，就可以简单地在两个条目之间增加一条新的语句。示例 B-8 就显示了这样的一个例子。

示例 B-8 利用序列号在一个标准的 IP 访问列表之间增加一条新的语句

```
ip access-list standard 1
 17 permit 172.22.30.100 0.0.0.0
```

参见示例 B-8，在允许 172.22.30.95 通过的条目之后和拒绝子网其他地址的条目（deny 172.22.30.0 0.0.0.255）之前增加一条新的条目。

这个条目也可以被简单地删除。示例 B-9 中删除序列号为 17 的条目。

示例 B-9 利用序列号，在一个标准的 IP 访问列表中间删除一个条目

```
ip access-list standard 1
 no 17
```

为了使访问列表在以后更加容易理解，可以在任何条目之前或之后增加一个注释。在示

例 B-10 和示例 B-11 中的访问列表配置中包括了注释。

示例 B-10 在一个标准的 IP 访问列表中增加注释

```
access-list 1 remark permit the 2 management hosts
access-list 1 permit 172.22.30.6 0.0.0.0
access-list 1 permit 172.22.30.95 0.0.0.0
access-list 1 remark deny everyone else on the subnet
access-list 1 deny 172.22.30.0 0.0.0.255
access-list 1 permit 172.22.0.0 0.0.31.255
access-list 1 deny 172.22.0.0 0.0.255.255
access-list 1 permit 0.0.0.0 255.255.255.255
```

示例 B-11 使用路由器的访问列表配置模式可以在一个标准的 IP 访问列表中增加注释

```
ip access-list standard 1
 remark permit the 2 management hosts
 10 permit 172.22.30.6 0.0.0.0
 15 permit 172.22.30.95 0.0.0.0
 remark deny everyone else on the subnet
 20 deny 172.22.30.0 0.0.0.255
 permit 172.22.0.0 0.0.31.255
 deny 172.22.0.0 0.0.255.255
 permit 0.0.0.0 255.255.255.255
```

示例 B-10 和示例 B-11 中演示了两种配置同一个列表的方法。其中，注释对表的功能没有任何影响，但它们可以使一个复杂的访问列表对以后的阅读者变得更加友好一些。

为了完全理解访问列表的功能，读者需要理解反向掩码

回忆一下 IP 地址掩码的作用：为了从主机地址导出网络地址或子网地址，掩码中的 1 对应着网络位，0 对应着主机位。在每一位上执行布尔与（Boolean AND）操作可以得到网络或子网号。图 B-5(a)包括了“与”（AND）函数的真值表和用英文表达的函数状态。

比较两位，当且仅当两位都为 1 时结果为 1。

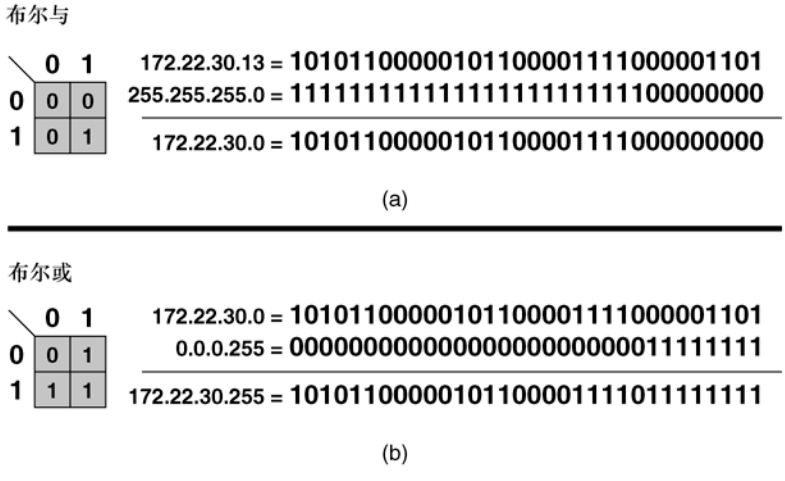


图 B-5 真值表、布尔与和布尔或的例子

布尔或（Boolean OR）是布尔与（Boolean AND）的反函数，真值表如图 B-5(b)所示。

比较两位，当且仅当两位都为 0 时结果为 0。

一个反码（inverse mask）（Cisco 更喜欢用术语通配掩码（wildcard mask））把对应于地址位中将要被准确匹配的位设置为 0，其他位设置为 1——值为 1 的位经常叫做不关心位。接

着反码将同地址进行或操作。

注意，图 B-5(b)中的或操作结果是 172.22.30.255。在 IP 术语中这一结果意指子网 172.22.30.0 上的所有主机地址。来自 172.22.30.0 的任何指定地址都将匹配到该地址/反码组合上。

图 B-6 给出了两种书写标准 IP 访问列表的快捷方法。图 B-6(a)给出的全 0 反码指明被讨论地址的所有 32 位都必须准确匹配到 172.22.30.6。对标准 IP 访问列表来说，缺省掩码是 0.0.0.0。所以给出的替换语句除了不带指定掩码外，同第一个语句完全相同。注意，这个缺省掩码不能应用到下一节将要讨论的扩展 IP 访问列表。

图 B-6(b)给出了允许一切的地址/反码组合。地址 0.0.0.0 实际上仅是一个占位符，真正起作用的是掩码 255.255.255.255。通过将每一位都设置为 1，该掩码将匹配任何地址。可以替换该语句的方法是使用关键字 **any**，它与第一个语句的意思相同。

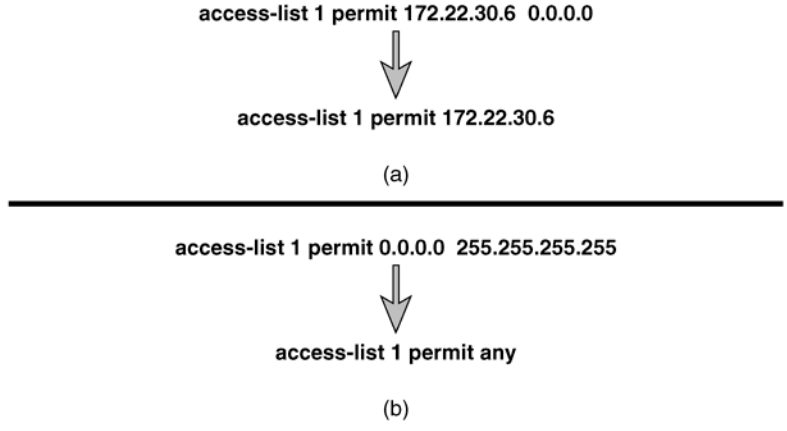


图 B-6 在配置标准 IP 访问列表时可以使用的两种快捷方法

B.3 扩展 IP 访问列表

扩展 IP 访问列表在指定过滤内容方面提供了更多的灵活性。扩展 IP 访问列表的基本格式如下：

```
access-list access-list-number [dynamic dynamic-name [timeout minutes]]
  {deny|permit} protocol source source-wildcard
  destination destination-wildcard [precedence precedence][tostos]
  [log|log-input][time-range time-range-name][fragments]
```

读者可以使用和配置标准列表相同的方法，在全局访问列表配置模式下配置扩展的访问列表。

在扩展的访问列表中也可以使用序列号。用法和标准列表中的用法相同。自反访问列表（reflexive access list）只可以在全局访问列表配置模式下使用，并且只能配置在命名的 IP 访问列表中。自反访问列表将在这个附录的后面章节讲述。

这里有一些特性已经很熟悉了，但还有一些新的特性：

- 对于扩展 IP 访问列表来说，*access-list-umber* 范围在 100~199，或 2000～2699 之间。
- **dynamic** 表示这个列表是一个动态的访问列表。动态访问列表用于“Lock-and-Key”

的安全特性。某个用户可以通过 Telnet 访问一台路由器，路由器使用像 TACACS+ 或 RADIUS 这样的认证服务器进行认证，并且在动态入口处基于源和目的信息允许或拒绝该用户访问。

- **timeout** 定义了一个临时条目在一个动态列表中保留的最大时间，以分钟计。缺省情况下条目没有超时配置，它永远保留在列表中。
- *protocol* 是一个新变量，它可以在 IP 包头的协议字段寻找匹配，可选择的关键字是 **eigrp**、**gre**、**icmp**、**igmp**、**igrp**、**ip**、**ipinip**、**nos**、**ospf**、**tcp** 和 **udp**。还可以使用 0~255 中的一个整数表示 IP 协议号。**ip** 是一个通用关键字，它可以匹配任意和所有的 IP 协议，同样地，反码 255.255.255.255 将可以匹配所有地址。
- 注意为了进行匹配，数据包的 *source* 地址和 *destination* 地址都将被检查；它们有各自的反码。
- **precedence** 和 **tos** 是可选变量，它们可以在 IP 包头中的优先级字段和服务类型字段寻找匹配。优先级取值范围在 0~7 之间，TOS 在 0~15 之间，或者用关键字表示，可参见 Cisco 可用关键字列表文档。
- **log** 是一个可选项，它指定打开信息日志功能。路由器试图记录的信息可能包括匹配的列表号与名字、源和目的地址、上层端口号和匹配记录的数据包数目。
- **log-input** 增加输入接口和源 MAC 地址或虚电路号到日志输出。
- **time-range** 可以创建一个临时的访问列表。Time-range 定义了访问列表有效的时间间隔。扩展访问列表中的 **time-range** 参数参考全局 **time-range** 命令。全局命令 **time-range** 定义了实际的时间参数。
- **fragments** 定义了如何通过访问列表条目处理分段的数据包。根据是否在访问列表中指定第 3 层或第 3 层和第 4 层信息，以及列表的条目是否允许数据包通过来以不同的方式处理分段。缺省情况下（不指定关键字 **fragments**），包含第 3 层信息（IP 地址，IP 端口号）的条目应用于所有未分段的数据包、初始分段和非初始分段的数据包。对于包含第 3 层和第 4 层（除了 IP 地址还有 TCP 或 UDP 端口号）信息的条目，则应用于未分段和初始分段的数据包。这个条目也可以在以下情况应用于非初始分段的数据包：如果非初始分段数据包的第 3 层信息匹配某个条目的第 3 层信息（IP 地址，IP 端口号），并且是匹配允许通过语句，那么这个分段就被允许通过。如果该条目是匹配拒绝通过的语句，那么将处理下一个访问列表。如果指定了 **fragment**，条目将仅仅应用于非初始分段数据包。**fragment** 关键字不能配置到包含第 4 层信息的条目，例如 TCP 或 UDP 端口号。

示例 B-12 演示了一个扩展 IP 访问列表的例子。

示例 B-12 扩展 IP 访问列表可以通过多种方式允许和拒绝数据包

```
access-list 101 permit ip 172.22.30.6 0.0.0.0 10.0.0.0 0.255.255.255 time-range
morning
access-list 101 permit ip 172.22.30.95 0.0.0.0 10.11.12.0 0.0.0.255
access-list 101 deny ip 172.22.30.0 0.0.0.255 192.168.18.27 0.0.0.0
access-list 101 permit ip 172.22.0.0 0.0.31.255 192.168.18.0 0.0.0.255
access-list 101 deny ip 172.22.0.0 0.0.255.255 192.168.18.64 0.0.0.63
access-list 101 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
time-range morning
periodic weekdays 08:00 to 11:59
```

以下对示例 B-12 中每一行进行说明：

- 第 1 行：在一个指定的时间范围内，源地址是 172.22.30.6 且目的地址属于网络 10.0.0.0 的数据包被允许通过。第 7 行和第 8 行定义了时间范围。在其他的时间，这个访问列表条目是没有激活的。没有激活的条目意味着这个条目就像根本没有在访问列表中一样被忽略掉。
- 第 2 行：源地址是 172.22.30.95 且目的地址属于子网 10.11.12.0/24 的数据包被允许通过。
- 第 3 行：源地址属于子网 172.22.30.0/24 且目的地址是 192.168.18.27 的数据包被拒绝通过。
- 第 4 行：源地址在 172.22. 0.0~172.22.31.255 之间且目的地址属于网络 192.168.18.0 的数据包被允许通过。
- 第 5 行：源地址属于网络 172.22.0.0 且目的地址前 26 位是 192.168.18.64 的数据包被拒绝通过。
- 第 6 行：从任意源地址到任意目的地址的 IP 数据包被允许通过。
- 第 7 行和第 8 行：命名为“morning”的时间范围定义为每周日早上 8:00～11:59，由列表的第 1 行引用。

图 B-7 给出了两种书写扩展 IP 访问列表的快捷方法。回想一下标准 IP 访问列表曾使用过缺省掩码 0.0.0.0，但是这个缺省掩码不能用于扩展 IP 访问列表，因为路由器无法正确地进行解释。但是对扩展列表也存在一个替代表述。在图 B-7(a)中，如果数据包的源地址是主机 172.22.30.6 且目的地址为主机 10.20.30.40，那么数据包被允许通过。在扩展 IP 访问列表中出现的掩码 0.0.0.0，可以通过在地址之前添加关键字 **host** 来代替。

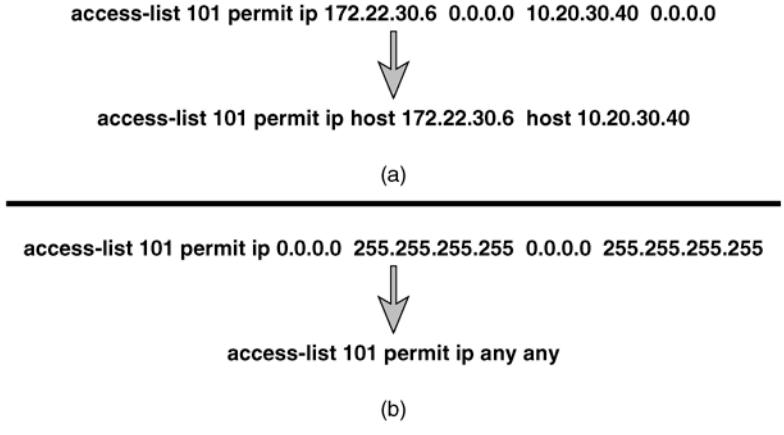


图 B-7 书写扩展 IP 访问列表的两种快捷方法

在图 B-7(b)的例子中，允许从任意源点到任意目标的数据包通过。正像标准访问列表一样，对源地址、目的地址或源和目的地址都可以使用关键字 **any** 代替地址/反码组合 0.0.0.0 255.255.255.255。

扩展访问列表比标准访问列表更强大，因为前者不仅检查数据包的源地址，而且检查所有有价值的内容，但是任何事情都是有代价的。使用扩展列表所要付出的代价是增加处理负担（如图 B-8 所示）。因为访问列表中每一行都需要检查数据包中的多个字段，因而会发生多次 CPU 中断。如果访问列表非常庞大或路由器很忙，那么这个要求可能会对性能产生不利的影响。

尽可能保持访问列表的长度较小，这样可以减轻路由器的处理负担。而且还要注意，在

匹配发生时，指定的动作会被调用，这时处理会停止。因此，如果你能够使大多数匹配都发生在你所给出访问列表的前几行，那么性能将会被提升。虽然这种方式并不总是可行的，但是在设计访问列表时需要记住这一点。

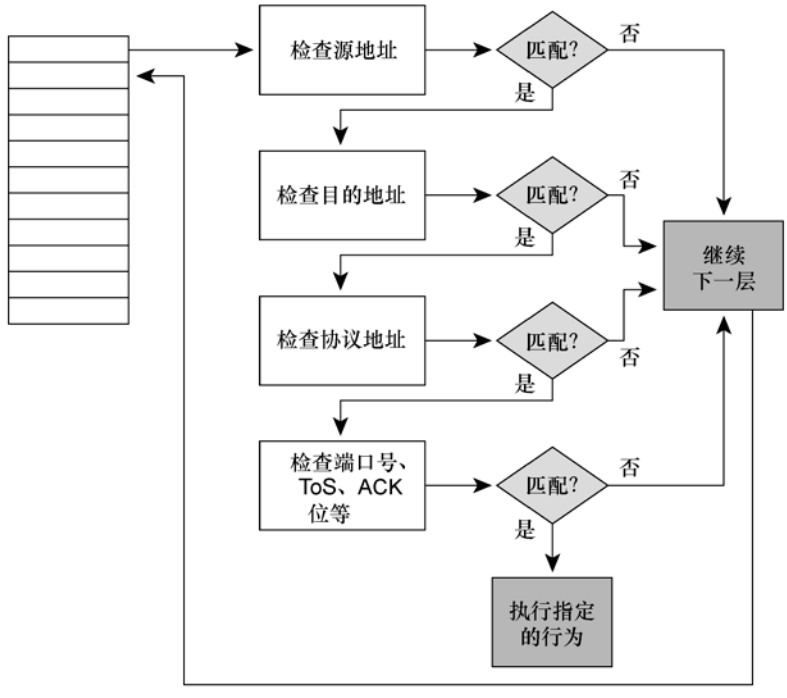


图 B-8 访问列表的决策流程图

某些路由器平台支持一个叫做“Turbo ACL”的功能特性，它可以对访问列表进行编译。路由器可以把所配置的访问列表编译到一个查询表中。条目的顺序保持不变，查询时间和查询占用的 CPU 可以大大降低。像时间范围等一些条目是不能包含在一个编译的列表中的。输入命令 **access-list compiled** 可以在路由器上进行增强的访问列表配置。

作为练习，请根据示例 B-12 给出一个更精彩的访问列表配置。也就是说，用尽可能少的行数重写访问列表，但又不损失任何功能（提示：一个相同功能的访问列表仅需要 4 行，不包括末尾的两个时间命令）。下面一段是答案，在进一步阅读之前请读者尝试重写这个列表。

第 1 行可以删除。第 1 行在周日的早晨允许主机 172.22.30.6 访问地址 10.0.0.0/8。没有这一行，通过第 6 行仍然可以允许从这台主机到地址 10.0.0.0 的访问，第 6 行允许所有没有在它之前被拒绝的数据包通过。

第 2 行也可以删除。在第 6 行也允许主机 172.22.30.95 到 10.11.12.0/24 的访问。

读者可能在想第 4 行是否也可以删除，但是请注意第 5 行拒绝了比包含第 4 行允许的地址范围更大的地址段。因此，在第 5 行指定的其余地址被丢弃前，利用第 4 行允许一个较小的子网地址通过是必要的。

B.3.1 TCP 访问列表

检查 TCP 段的扩展访问列表的行语法如下：

```
access-list access-list-number [dynamic dynamic-name [timeout minutes]]
  {deny|permit} tcp source source-wildcard[operator[port]] destination
  destination-wildcard [operator[port]] [established]
  [precedence precedence][tos tos] [log|log-input]
  [time-range time-range-name][fragments]
```

注意，协议变量是 **tcp**。这里最重要的特性是，访问列表可以检查 TCP 段头中的源和目的端口号。其结果是，过滤数据包的选项不仅是去往或来自一个特殊地址，而且还可以是去往或来自一个特殊套接字（一个 IP 地址/应用端口的组合）。

还没有解释的 TCP 访问列表的特性是 *operator* 和 *port*：

- *operator*——指定逻辑操作。选项可以是 **eq**（等于）、**neq**（不等于）、**gt**（大于）、**lt**（小于）和 **range**（指明包括的端口范围）。如果使用 **range** 运算符，那么要指定两个端口号。
- *port*——指明被匹配的应用层端口号。几个常用的端口号是 Telnet（23）、FTP（20 和 21）、SMTP（25）和 SNMP（169）。完整的 TCP 端口号列表参见 RFC 1700。

假设你实现了一个访问列表可以阻止外部发起的 TCP 会话进入到你的网络中，但是你又想让内部发起的 TCP 会话的响应通过，那应该怎么办？通过检查 TCP 段头内的 ACK 和 RST 标记，关键字 **established** 可以实现这一点。如果这两个标记都没有被设置，表明源点正在向目标建立 TCP 连接，那么匹配不会发生。最终数据包将会在访问列表的后继行中被拒绝。

示例 B-13 显示了一个 TCP 访问列表的例子。

示例 B-13 这个 TCP 访问列表允许已经建立的连接和允许访问某些地址的 SMTP 和 Telnet 端口

```
access-list 110 permit tcp any 172.22.0.0 0.0.255.255 established
access-list 110 permit tcp any host 172.22.15.83 eq 25
access-list 110 permit tcp 10.0.0.0 0.255.255.255 172.22.114.0 0.0.0.255 eq 23
```

以下是对示例 B-13 中的每一行的解释：

第 1 行：如果连接是从网络 172.22.0.0 发起的，那么允许从任意源点到该网络的 TCP 数据包通过。

第 2 行：允许来自任意源点，且目标端口号是主机 172.22.16.83 的端口 25（SMTP）的 TCP 数据包通过。

第 3 行：允许来自网络 10.0.0.0，去往网络 172.22.114.0/24 且目标端口为 23（telnet）的 TCP 数据包通过。

隐式拒绝一切的所有数据包被丢弃。

B.3.2 UDP 访问列表

检查 UDP 段的扩展访问列表的行语法如下：

```
access-list access-list-number [dynamic dynamic-name [timeout minutes]]
  {deny|permit} udp source source-wildcard[operator[port]] destination
  destination-wildcard [operator[port]]
  [precedence precedence][tos tos] [log|log-input]
  [time-range time-range-name][fragments]
```

除了协议变量是 **udp** 外，该格式与 TCP 的格式非常相似。另一个不同之处是没有关键字 **established**。 原因是 UDP 提供无连接传输服务，在主机之间没有建立连接。

通过向前面例子中添加额外的 3 行可以得到示例 B-14 中的例子。

示例 B-14 这个访问列表允许 TCP 和 UDP 数据包通过

```
access-list 110 permit tcp any 172.22.0.0 0.0.255.255 established
access-list 110 permit tcp any host 172.22.15.83 eq 25
access-list 110 permit tcp 10.0.0.0 0.255.255.255 172.22.114.0 0.0.0.255 eq 23
access-list 110 permit udp 10.64.32.0 0.0.0.255 host 172.22.15.87 eq 69
access-list 110 permit udp any host 172.22.15.85 eq 53
access-list 110 permit udp any any eq 161
```

第 4 行：允许从子网 10.64.32.0/24 到主机 172.22.15.87 且目标端口为 69（TFTP）的 UDP 数据包通过。

第 5 行：允许从任意源点到主机 172.22.15.85 且目标端口为 53（域名服务器）的 UDP 数据包通过。

第 6 行：允许从任意源点到任意目标的 SNMP 数据包通过。

在列表中没有发现匹配，隐式拒绝一切停止丢弃所有数据包。

B.3.3 ICMP 访问列表

检查 ICMP 数据包的扩展访问列表的行语法如下：

```
access-list access-list-number [dynamic dynamic-name [timeout minutes]]
  {deny|permit} icmp source source-wildcard destination destination-wildcard
  [icmp-type[icmp-code]|icmp-message] [precedence precedence][tos tos]
  [log|log-input][time-range time-range-name][fragments]
```

icmp 目前在协议字段，注意这里没有源端口或目的端口，因为 ICMP 是一种网络层协议。该行可以用于过滤所有 ICMP 消息，或者读者可以使用下面的选项过滤指定的 ICMP 信息：

- *icmp-type* 编码范围是 0~255。所有 ICMP 类型编号见 RFC 1700。
- 过滤粒度可以通过指定 *icmp-code* 进行增加。一个 ICMP 代码指定了 ICMP 数据包类型的一个子集；这些代码是一个在 0～255 之间的数字，也在 RFC 1700 中描述。
- 可以输入 ICMP 消息的名字替代 ICMP 类型和 ICMP 代码。

示例 B-15 中显示了一个 ICMP 访问列表的例子。

示例 B-15 这个 ICMP 访问列表拒绝指定的数据包和允许其他所有的数据包

```
access-list 111 deny icmp 172.22.0.0 0.0.255.255 any 0
access-list 111 deny icmp 172.22.0.0 0.0.255.255 any 3 9
access-list 111 deny icmp 172.22.0.0 0.0.255.255 any 3 10
access-list 111 deny icmp 172.22.0.0 0.0.255.255 any traceroute
access-list 111 permit ip any any
```

以下是对示例 B-15 中每一行的解释：

第 1 行：拒绝从网络 172.22.0.0 到任意目标的 ICMP ping 响应（回应应答，ICMP 类型 0）通过。

第 2 行：拒绝从网络 172.22.0.0 到任意目标的 ICMP 目的网络不可达数据包通过，其中代码号为 9。

第 3 行：拒绝从网络 172.22.0.0 到任意目标的 ICMP 目的网络不可达数据包通过，其中代码号为 10。

第 4 行：拒绝从网络 172.22.0.0 到任意目的地的 ICMP traceroute。

第 5 行：允许所有其他 IP 数据包通过。

B.4 调用访问列表

如果不通过调用命令使数据包发送到访问列表，访问列表是不进行任何处理的；这里所调用的命令定义了如何使用访问列表。命令如下所示：

```
ip access-group access-list-number {in|out}
```

在接口上配置这条命令可以建立安全过滤器或流量过滤器，并且可以应用于进、出流量。如果 **in** 或 **out** 关键字都没有被指定，那么缺省值是出站。当然，访问列表编号指定了接收该命令所发送数据包的访问列表。图 B-9 给出了该命令的两种配置。

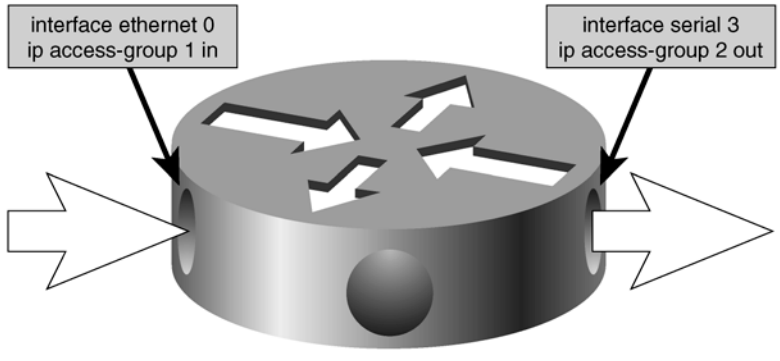


图 B-9 命令 **ip access-group** 使用指定的访问列表在接口上针对进出流量建立过滤器

图 B-9 中的访问列表 1 过滤进入接口 E0 的 IP 数据包，它对于出站数据包和其他协议（如 IPX）产生的数据包不起作用。访问列表 2 过滤离开接口 S3 的 IP 数据包，它对于入站数据包和其他协议产生的数据包不起作用。在入站访问列表中尽可能的执行拒绝语句来代替出站列表，可以减少路由器处理那些将要被丢弃的数据包所花费的系统资源。

多个接口可以调用相同的访问列表，但是在任意一个接口上，对每一种协议仅能有一个进入和离开的访问列表。

在图 B-10 中，前面例子中给出的 TCP、UDP 和 ICMP 访问列表被用作过滤器。源自前面两个例子中的访问列表 110 被应用到以太网接口 0 上，用来检查入站流量。应用到相同接口的访问列表 111 检查出站流量。仔细分析一下两个访问列表，包括它们之间的相互关系，再考虑下面问题：

- 从 172.23.12.5 到 10.64.32.7 的 ping 响应想从接口 Ethernet0 出站，是否允许通过？
- 想在主机 172.22.67.4 上 ping 网络 10.64.32.20 上的一台设备，从接口 Ethernet0 出站可以 ping 通吗？

来自 172.23.12.5 的 ping 响应包允许从接口 Ethernet0 出去。来自 172.22.0.0/16 的 ping 响应包会被拒绝，而不是 172.23.0.0/16。从 172.22.67.4 到 10.64.32.20 的 ping 包，从接口 Ethernet0 出去将不被允许。这个 ping 请求可以成功地从该接口出去，但它的响应包将被入站访问列表拒绝。

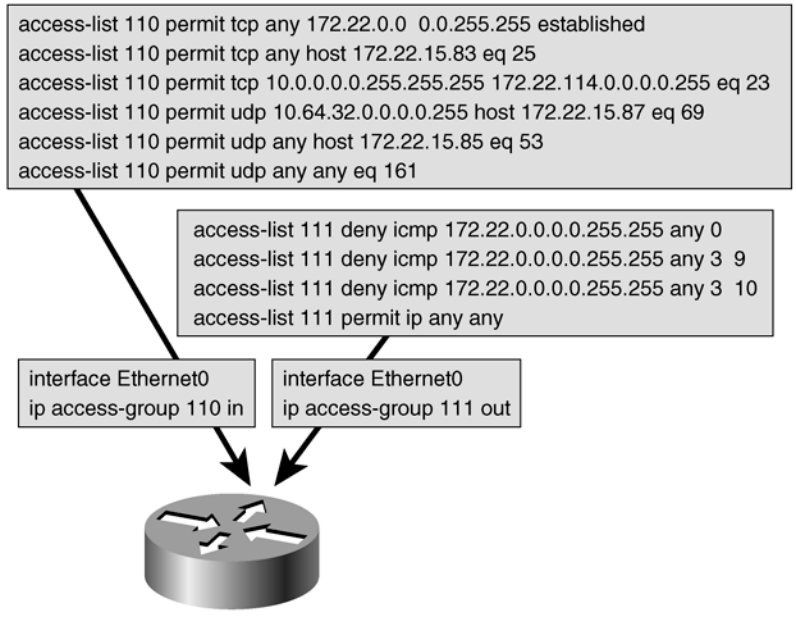


图 B-10 这里访问列表 110 用于过滤以太网接口上的入站数据包。访问列表 111 用于过滤该接口上的出站数据包

另一种调用访问列表的命令是 **access-class**。该命令用于控制到达路由器或由路由器虚拟终端线路发起的 telnet 会话，而不进行数据包过滤。命令格式如下：

```
access-class access-list-number {in|out}
```

图 B-11 给出了使用命令 **access-class** 的例子，访问列表 3 控制路由器 VTY 线路将要接受的 telnet 会话的源点地址。访问列表 4 控制路由器 VTY 线路可以连接的目标地址。

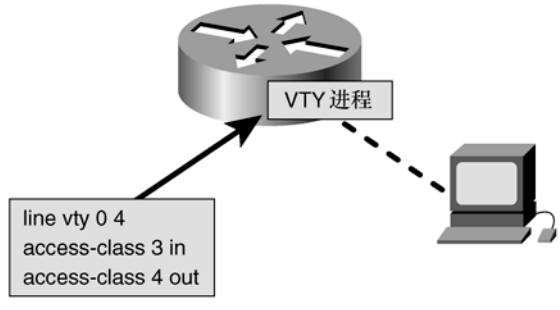


图 B-11 命令 **access-class** 使用访问列表控制到达和由路由器虚拟终端线路发起的 telent 流量

命令 **access-class** 对于路由器传输的 telnet 流量不起作用，它仅影响到达路由器以及由路由器发起的 telnet 会话。

B.5 自反访问列表

自反访问列表是自动驻留的、暂时的、基于会话的过滤器。如果某台路由器允许通过网

络内部向外部的主机初始发起一个会话，那么自反列表就允许返回的会话数据流。自反列表和被扩展命名的 IPv4 访问列表一起使用。使用自反列表的会话过滤器可以与使用关键字 **established** 的 TCP 过滤器进行比较。使用 **established** 关键字，TCP 会话是从网络内部初始发起的。如果返回的数据流具有 ACK 或 RST 的设置，那么该数据包就是以前建立连接的会话的一部分，并且该数据包被允许通过。带有 **established** 关键字的条目是访问列表中的永久条目。

自反访问列表使用不同的参数来确定数据包是否是以前建立的会话的一部分。对于 TCP 或 UDP 数据包来说，自反访问列表使用源和目的 IP 地址，以及源和目的 TCP 或 UDP 端口号。

当存在一个从网络内部初始发起的会话时，自反访问列表就会保留从初始的数据包中收集的会话信息。反转并添加源和目的 IP 地址以及源和目的端口号，连同上层协议类型（例如 TCP 和 UDP），作为临时自反列表的允许语句。该条目在出现以下几种情况之前都是保持活动的：不再有任何有关该会话的数据流和超时值；收到两个 FIN 标记的数据包；或者在 TCP 数据包中设置了 RST 标记。

示例 B-16 显示了一个自反访问列表配置的例子。

示例 B-16　示例中的自反访问列表命名为 infilter

```
interface Serial0/0.1 point-to-point
 ip address 172.25.150.65 255.255.255.192
 ip access-group infilter in
 ip access-group outfilter out
!
ip access-list extended infilter
 permit eigrp any any
 permit udp any any eq rip
 evaluate sessiontraffic
ip access-list extended outfilter
 permit tcp any any reflect sessiontraffic
 permit icmp any any echo time-range morning reflect sessiontraffic
!
time-range morning
 periodic weekdays 9:00 to 12:30
```

在这个示例中，在与外部网络连接的接口上应用了过滤器。outfilter 列表只允许在每周日早晨 9:00～12:30 之间，从内部网络初始发起的所有 TCP 数据包和 ICMP echo 请求数据包通过。在串行接口 Serial0/0.1 的出站方向应用了 outfilter 列表。在 permit 语句中使用了关键字 **reflect**，它创建了一个名为“sessiontraffic”的自反访问列表。在出现匹配这个带有 **reflect** 关键字的 permit 条目时，就会保留这个自反访问列表。

即将进入接口 Serial0/0.1 的数据包需要通过 infilter 访问列表进行过滤。这些数据包是源自某个外部网络的。在这个实例中，infilter 列表允许 EIGRP 和 RIP 的数据包通过。当进入的数据包匹配 EIGRP 和 RIP 条目后，自反访问列表 sessiontraffic 将依次被判定。自反访问列表在末尾没有隐含的 deny-all 语句，但包含自反访问列表的扩展访问列表末尾具有该语句。

示例 B-17 显示了在 TCP 和 ICMP 流量从接口 Serial0/0.1 出去之前的访问列表。

示例 B-18 显示了在 TCP 和 ICMP 流量从接口 Serial0/0.1 出去之后的访问列表。

示例 B-17 命令 show ip access-list 显示了路由器上配置的所有永久与临时的 IP 访问列表

```
Router#show access-lists
Extended IP access list infilters
  10 permit eigrp any any
  20 permit udp any any eq rip (1074 matches)
  30 evaluate sessiontraffic
Extended IP access list outfilter
  10 permit tcp any any reflect sessiontraffic (45 matches)
  20 permit icmp any any echo time-range morning (active) reflect sessiontraffic
Reflexive IP access list sessiontraffic
```

从内部网络向外部网络发起 ping 和 telnet 操作。

示例 B-18 中显示了发起上述流量后的访问列表信息。

示例 B-18 命令 show ip access-list 显示了路由器上动态创建的自反访问列表的条目

```
Router#show ip access-list
Extended IP access list infilters
    10 permit eigrp any any
    20 permit udp any any eq rip (1101 matches)
    30 permit udp any any eq 521
    40 evaluate sessiontraffic
Extended IP access list outfilter
    10 permit tcp any any reflect sessiontraffic (188 matches)
    20 permit icmp any any echo time-range morning (active) reflect sessiontraffic
(9 matches)
Reflexive IP access list sessiontraffic
     permit tcp host 192.168.16.225 eq telnet host 192.168.50.130
     eq 11002 (55 matches) (time left 293)
    permit icmp host 192.168.16.225 host 192.168.50.130  (19 matches) (time left 270)
```

示例 B-17 的输出信息显示了访问列表入站和出站过滤，以及它们配置的参数。请注意 ICMP 条目在出站过滤中是 active 的，这意味着现在路由器的时间和这周的日期落在所配置的时间范围之内。输出信息也显示了没有驻留的自反访问列表 sessiontraffic。

在发起 ICMP 的 ping 和 telnet 会话，数据包通过 serial0/0.1 出去后，在示例 B-18 中再次显示了访问列表信息。这次就有了自反访问列表的条目。这些条目是匹配所有从外部网络到达 serial0/0.1 的数据包的，一直持续到计时器超时或会话被关闭。从外部网络发起 telnet 会话和 ICMP echo 不会成功。

自反访问列表不支持那些在会话期间改变端口号的协议，例如 FTP。

B.6 可供选择的关键字

大多数网络专业人员都知道一些较常用的 TCP 端口号和一些 UDP 端口号。很少有人可以说出有关 ping 或目标不可达的 ICMP 类型是什么，知道目标不可达类型的 ICMP 代码的人就更少了。从 IOS 10.3 开始，配置访问列表可以使用关键字来代替端口、类型或代码编号。使用关键字，访问列表 110 和 111 显示在示例 B-19 中。

示例 B-19 在访问列表中关键字替代了端口号

```
access-list 110 permit tcp any 172.22.0.0 0.0.255.255 established
access-list 110 permit tcp any host 172.22.15.83 eq smtp
access-list 110 permit tcp 10.0.0.0 0.255.255.255 172.22.114.0 0.0.0.255 eq telnet
access-list 110 permit udp 10.64.32.0 0.0.0.255 host 172.22.15.87 eq tftp
access-list 110 permit udp any host 172.22.15.85 eq domain
access-list 110 permit udp any any eq snmp
!
access-list 111 deny icmp 172.22.0.0 0.0.255.255 any echo-reply
access-list 111 deny icmp 172.22.0.0 0.0.255.255 any net-unreachable
administratively-prohibited
access-list 111 deny icmp 172.22.0.0 0.0.255.255 any host-unreachable
administratively-prohibited
access-list 111 permit ip any any
```

注意，如果你将路由器从 10.3 以前的版本升级到新版本，紧接着重启路由器，那么路由器将会使用新的语法重写配置文件中的访问列表，包括关键字。如果你随后又需要重载最初 10.3 之前的镜像文件，那么路由器将无法理解被修改的访问列表。记住在任何情况下，升级之前要将原来的配置文件上载到 TFTP 服务器上。

B.7 命名访问列表

对于每台路由器，798 个标准访问列表或 799 个扩展访问列表的限制看上去已经够用了，但是有些情况（比如动态访问列表[1]）可能就不够了。从 IOS 11.2 开始提供的命名访问列表打破了这种限制，它的另一个优点是描述名可以使数量庞大的访问列表更加便于管理。

为了使用命名访问列表，访问列表的第 1 行应采用以下格式：

```
ip access-list {standard|extended} name
```

因为没有编号区分访问列表类型，所以该行可以将列表指定为标准 IP 列表或扩展 IP 列表。下面紧接着可以使用许可或拒绝语句，标准列表的语法如下：

```
{deny|permit} source [source-wildcard]
```

基础扩展列表的语法如下：

```
{deny|permit} protocol source source-wildcard destination destination-wildcard
    [precedence precedence][tos tos][log]
```

在这两种情况下，都没有出现命令的 **access-list** *access-list-number* 部分，但是其他部分完全相同。在相同路由器上标准和扩展访问列表不能同名。除了在接口上建立命名访问列表的命令涉及到用名字替代编号外，在其他方式中命令均保持不变。图 B-12 使用命名格式对图 B-10 中的访问列表进行了转换。

[1] 本教程中没有涉及动态访问列表。详细内容可参见 Cisco IOS 安全配置手册——配置 Lock-and-Key Security（动态访问列表），查询更多的信息。

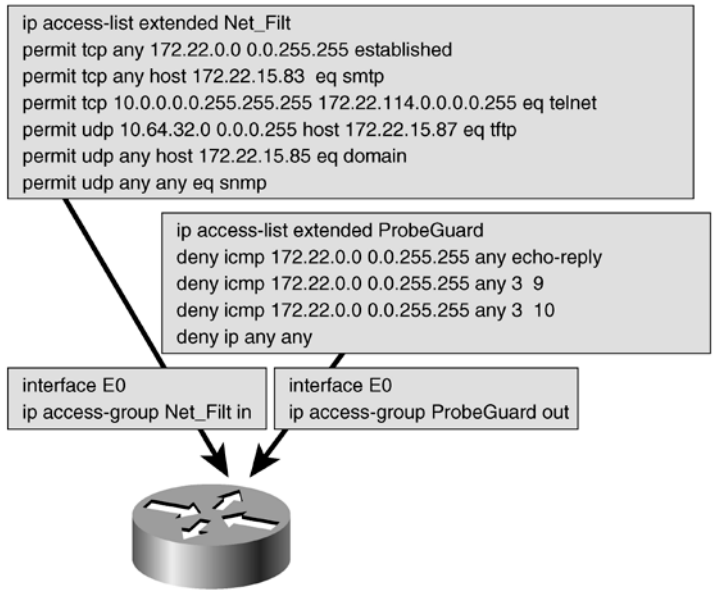


图 B-12 图 B-10 中的访问列表现在被配置为命名访问列表

B.8 前缀列表

前缀列表（prefix list）用于在路由更新时允许或拒绝指定的某个地址或地址范围。BGP 路由选择协议在 IPv4 中使用前缀列表。在 IPv6 的协议之间交换 IPv6 地址或过滤更新时，所有 IPv6 的路由选择协议都可以使用前缀列表。

前缀列表可以被命名。表中的条目允许或拒绝通过一个地址或地址范围，参见示例 B-20。

示例 B-20 前缀列表允许和拒绝 IPv6 地址通过

```
ipv6 prefix-list v6_addr_filt permit 2001:db8:0:1::/64
ipv6 prefix-list v6_addr_filt permit 2001:db8:0:10::/60 le 64
ipv6 prefix-list v6_addr_filt permit ::/0 ge 62 le 64
```

第 1 条允许长度精确为 64 位的前缀 2001:db8:0:1::通过。第 2 条和第 3 条允许通过一段地址范围。关键字 **le** 表示一个前缀长度范围，它所匹配的前缀长度大于等于该语句中的前缀后面指定的长度，小于关键字 **le** 后面指定的长度。前缀列表 v6_addr_filt 的第 2 条允许匹配 2001:db8:0:10::和长度在 60～64 之间的前缀。关键字 **ge** 指定了某个地址范围内的前缀最小长度。如果没有包含关键字 **le**，它就假定所匹配的前缀范围的最大长度为 128 位，即 IPv6 前缀的最大位数。当包括关键字 **le** 时，地址范围的最大匹配长度就是 **le** 后面指定的长度。前缀列表 v6_addr_filt 的第 3 条允许所有长度在 62～64 之间的前缀通过。

B.9 对放置过滤器的考虑

为了达到最佳的性能，你不仅需要考虑访问列表自身的有效设计，而且还要考虑如何在

路由器和网络中去布置过滤器。

凭借经验，安全过滤器通常是作为入站过滤器。在不必要或不被信任的数据包到达路由选择进程之前将它们过滤掉，阻止欺骗攻击——数据包欺骗路由选择进程，使其认为它来自某处，但它实际并不是来自那里。另外一方面，流量过滤器通常是出站过滤器。当你考虑在某一点的流量过滤器可以阻止不必要的数据包占用某条数据链路时，这种方法将很有意义。

除了这两种经验外，要考虑的另一个因素是访问列表和路由选择进程将要使用的 CPU 周期数。入站过滤器是在路由选择进程之前被调用，而出站过滤器是在路由选择进程之后被调用（如图 B-13 所示）。如果大多数经过路由选择进程的数据包被访问列表拒绝，那么入站过滤器可以节省一些处理周期。

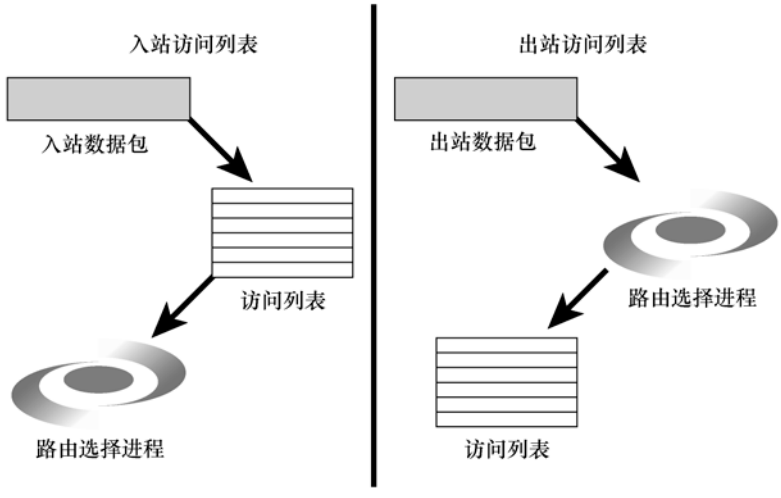


图 B-13 入站过滤器是在路由选择进程之前被调用，而出站过滤器是在路由选择进程之后被调用

标准 IP 访问列表仅能过滤源地址。因而，使用标准列表的过滤器必须被放置在尽可能靠近目标的地方，以便源点还可以访问到其他没有被过滤的目标（如图 B-14（a）所示）。其结果是带宽和 CPU 周期被浪费在最终将要被丢弃的数据包上。

因为扩展 IP 访问列表可以非常准确地标识数据包特性，所以它们应该被放置在尽可能靠近源点的地方，这样可以防止带宽和 CPU 浪费在传输无用的数据包上（如图 B-14（b）所示）。另一方面，扩展列表的复杂性意味着更多的处理负担。当决定在那里放置过滤器时，需要权衡考虑。

你还必须理解访问列表将怎样影响路由器上的交换。例如，使用扩展访问列表的接口不能被独立地交换；动态访问列表不能被硅交换，而且可能影响硅交换的性能。在 IOS 11.2 版要之前根本不支持命名访问列表。

在骨干或核心路由器上，访问列表对交换的影响可能是很重要的。通过阅读 Cisco 关于 IOS 的配置指南可以确保全面地研究和理解访问列表所产生的影响。在某些情况下，一个数据包过滤路由器——一个专门用于数据包过滤的较小的路由器——可能被用来减轻重要路由器的负担。

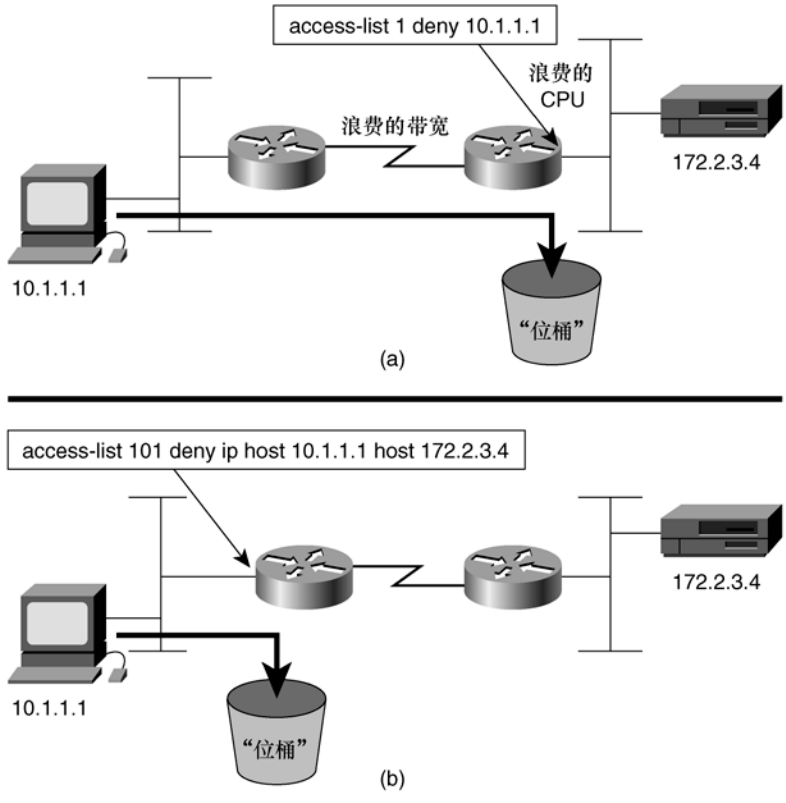


图 B-14 使用标准列表的过滤器必须被放置在尽可能靠近目标的地方(a)，而扩展访问列表可以放置在靠近源点的地方(b)

B.10 访问列表的监视和计费

在不必显示路由器的完整配置的前提下，能够检查一个甚至所有访问列表是非常有用的。命令 **show ip access-list** 可以显示路由器上所有 IP 访问列表的简化语法。如果要观察一个特定的访问列表，可以通过指定名字和标号（参见示例 B-21）。如果没有输入关键字 **ip**（**show access-list**），那么所有访问列表都将被显示出来。

示例 B-21 命令 show ip access-list 可以显示访问列表的简化语法

```
Woody#show ip access-list 110
Extended IP access list 110
    10 permit tcp any 172.22.0.0 0.0.255.255 established
    20 permit tcp any host 172.22.15.83 eq smtp
    30 permit tcp 10.0.0.0 0.255.255.255 172.22.114.0 0.0.0.255 eq telnet
    40 permit udp 10.64.32.0 0.0.0.255 host 172.22.15.87 eq tftp
    50 permit udp any host 172.22.15.85 eq domain
    60 permit udp any any eq snmp
Woody#
```

跟踪被访问列表拒绝的数据包作为安全计划或编制策略能力的一部分，这是很有用的。命令 **ip accounting access-violations** 可以配置在独立的接口上，用来创建所有在该接口被访问列表拒绝的数据包的数据库。可以使用命令 **show ip accounting access-violations** 来检查数

据库，结果可以显示出匹配该地址的源和目标地址、数据包数、字节数以及拒绝该数据包的访问列表编号（参见示例 B-22）。命令 **clear ip accounting** 可以清除计费数据库。

示例 B-22 访问列表计费数据库可以使用命令 show ip accounting access-violations 来查看

```
Woody#show ip accounting access-violations
   Source          Destination             Packets               Bytes    ACL
  10.1.4.1       255.255.255.255                13                 936    110
  10.1.4.1       172.22.1.1                     12                1088    110

Accounting data age is 10
Woody#
```

计费功能将关闭接口上的独立交换功能和硅交换功能。在需要这些交换模式的接口上不能使用计费功能。

作为最后一种技巧，你应该知道被列表尾部隐含地拒绝一切所丢弃的数据包，它们是不能被计费跟踪的。为了跟踪这些数据包，仅需在列表尾部配置一条拒绝一切的条目，参见示例 B-23。

示例 B-23 在访问列表的尾部增加一条 **deny any** 条目，用来跟踪因为不匹配列表中所有其他条目的而被丢弃的数据包

```
access-list 110 permit tcp any 172.22.0.0 0.0.255.255 established
access-list 110 permit tcp any host 172.22.15.83 eq smtp
access-list 110 permit tcp 10.0.0.0 0.255.255.255 172.22.114.0 0.0.0.255 eq telnet
access-list 110 permit udp 10.64.32.0 0.0.0.255 host 172.22.15.87 eq tftp
access-list 110 permit udp any host 172.22.15.85 eq domain
access-list 110 permit udp any any eq snmp
access-list 110 deny ip any any 1
```



附录 C

CCIE 备考提示

要成为 Cisco 认证互连网络专家（CCIE），远远不像业界其他一些认证，只是“读一本书，参加一次考试”就行了。在通过了相对简单的笔试后，你需要在一场极其艰难的实验室实践考试中展示你的专业才能。虽然你要对 Cisco 的配置命令必须非常熟悉，但在实验室考试中最大的挑战是与 Cisco IOS 软件无关，他们测试的是你对交换机、路由器以及路由选择协议的理解程度。正因为如此，CCIE 们才被认为是千里挑一的互连网络专家。

有层次地构建网络包括 4 个步骤；这 4 个步骤同样也适用在准备 CCIE 实验室考试的过程中。

- **计划**：冷静而正确地评估你现在的经验和不足之处。计算一下你每天能用来学习的时间。检查你的资源，包括实验室设备、资金、书籍、训练的时间以及能请教问题的熟人、老师、专家。衡量你自身的长处和短处：你擅于考试吗？你在压力下能很好地工作吗？你对挫折和失落的反应是什么？你学习习惯好吗？你能很好地通过阅读来学习吗？通过利用以上的答案，编写一张计划表，按要求去执行你的计划。
- **设计**：尽可能多地与 CCIE 交流，询问他们的准备方案，并按你的要求，扬长避短地设计你的准备方案。你的方案应该能准确地反映出在一定的时段内将你现有水平提升至 CCIE Lab 的水平。你还可以通过一系列具有明确目标的小工程来架构一个大的项目。综合你各方面的因素，来制订你的计划，否则你的计划会受到很大影响。
- **实现**：有很多失败的例子是因为在设计完成之前就忙于实施。应该清楚地编写你的准备计划，规划好方案的每一步，而不要草率实施。一旦你开始实施

了，就要坚持不懈。不要放弃，也不要气馁，更不要懒惰。检查核对你要完成的每一步。

- **优化**：你的准备计划应该是一个生动的文档。当你每前进一步时，都会遇到比你预计的要么困难要么简单的问题。你可以采用灵活的态度，增加额外的任务来帮助你掌握每一个主题。

只有你才能设计最适合自己的准备计划。在下面的几节里，我给出的建议并不是叫你死板地遵循，而是给你一些点子，让你创建自己的学习计划。一些小的技巧来源于我作为一个 CCIE 和 Cisco 系统讲师的经验，也来源于我的那些成功通过 CCIE 考试的同事们的经验。

C.1 牢固的基础

如果你是一名初学者，或者你的连网经验有限，你的第一步就是牢牢掌握网络互连和 Cisco 路由器的基础部分。这个过程包括课堂训练和自学。我建议参考 Cisco.com，可以在“Learning and Events”中查找有关职业认证的有用信息。

Cisco 通过它的培训伙伴提供了很多实践培训课程。只要你的时间和资源允许，尽可能多地参加这些培训。其中一些培训特别重要：

- Cisco 网络设备互连（ICND）；
- 组建可扩展的 Cisco 互连网络（BSCI）；
- 组建 Cisco 远程访问网络（BCRAN）；
- 组建 Cisco 多层交换网络（BCMSN）；
- Cisco 网络故障诊断（CIT）。

充分利用你参加的每一节课，多问老师问题，多和同学们一起讨论。最重要的是，好好利用你接触设备的机会。不要光满足于表面，一定要充分理解“为什么”和“如何做”。当你做完一个实验，不要停止，多掌握一些设备方面的知识，看看有哪些配置和调试命令，多尝试使用它们。如果有时间，多配置几次直到熟练。

上课会让你找出你自身知识的不足。多读书，填补你对网络互连协议和技术方面知识的空白。Internet 上有很多好的技术指南，当你开始学习一种新知识时，一定要记得在 Web 上进行搜索。

C.2 认证途径

在撰写本书第一版的时候，CCIE 还是 Cisco 公司提供的惟一认证。目前，已经具有很多级别的认证了，甚至在 CCIE 认证项目上也具有了很多专门的认证。制定这些分级认证的好处是可以给你鼓励（虽然它们也不是必须要预先通过的），在你通向 CCIE 的道路上实现阶段性的目标，也可以在这个过程中被授予具有价值的认证。

认证的第一个目标是 Cisco 认证网络工程师（CCNA）。这个级别的认证可以证明你掌握了基本的网络互连和网络协议方面的知识，能够安装和实现小型的 LAN、WAN 和接入网络，同时理解基本的 Cisco 硬件和 IOS 软件功能。

Cisco 认证的第二个目标是 Cisco 认证资深网络工程师（CCNP）。这个高级别的认证表明

你掌握了大型企业网络运行和故障诊断方面的技术。拥有了 CCNP 认证资格，你就做好了向专家级别进军的准备，可以开始你的 CCIE 认证之旅了。

C.3 实践经验

几乎所有的 CCIE 都会告诉你，实践经验对于准备实验室考试来说是最有价值的。决不要放过任何一个配置和调试路由器的机会。如果你现在的工作无法接触到路由器或者交换机，那么和你们公司的网络工程师打好交道，向他们讲述你的目标，尽可能多地与他们取得联系。

如果你能接触实验室设备，就一定要充分利用这些设备。在实验室取得的经验是不可替代的，在实验室，你可以随心所欲地进行配置，引入各种各样的问题，而不用担心实际环境中的网络崩溃。

有些在线的公司提供远程访问的付费实验室。所付的费用差别很大，这根据你能使用的设备情况而定。

还有一种选择，就是建立自己的实验室。虽然代价昂贵，但作为一名 CCIE，你最终所拿的薪水会让你感到自己的投资非常划算。许多旧 Cisco 设备的价格是很合理的。请订阅 Cisco 新闻组，comp.dcom.sys.cisco，或 groupstudy.com 这样的学习组；经常有人在上面卖旧路由器，你或许能与他们进行一次划算的交易。在本书案例中使用的大多数我自己的实验室设备都是在 eBay 上购买的。虽然两台路由器也可以用，但最好你至少拥有 4 台路由器，其中一台应该有 4 个或者更多的串口，这样你可以将它配置为帧中继交换机。记住你不需要顶级设备，过时的路由器对你而言就已经很好了，因为没有人会在一个商用的网络中使用它们。

C.4 深入学习

掌握了一定的基础知识，同时已具有实际动手能力，你应该开始深入学习网络协议了。至少你应该阅读本书中列出的 RFC，相关的 RFC 读得越多越好。www.ietf.org 是最好的站点之一。

当然，不是所有的网络协议在 RFC 中都有描述。寻找更高级的非 IP 协议方面的教程、白皮书和指南，这些可能会包含在考试中。你也应该学习 Ethernet 和 WAN 协议，比如 T-1、ISDN、帧中继和 ATM。你可以在 www.cisco.com 上找到非常有价值的公开资料。当然，除了本书外，Cisco Press 还提供了大量书籍，如 CCNA、CCNP 和 CCIE 方面的书籍，读者可以在 Ciscopress.com 上去寻找你所需要的书籍。

一个很好地将理论和实践知识结合在一起的学习工具是 Cisco 新闻组 http://comp.dcom.sys.cisco。在上面可以提交你遇到的具有挑战性的难题，你先进行自我解答，然后等待由 CCIE 们以及 Cisco 工程师们给出答案，看看你的答案是不是正确，如果不正确，找出问题所在。你也可以将问题发送到 Cisco 的新闻组，大多数参与者都比较友好，并愿意分享他们的经验。

还有一些网络上的学习组是专门针对准备 CCNP 和 CCIE 考试的。其实历史最久和最著名的一个学习组是 groupstudy.com。参加这样的小组是获取资源、分享经验、加入讨论，以及获取疑难问题答案的最佳途径。我强烈推荐读者加入到在线的学习组中去。

最后，如果你找到一些志同道合者，组成一个学习小组吧。我知道某些公司的 CCIE 学

习小组非常有效，从中诞生了很多 CCIE。

C.5 最后 6 个月

具有坚实的理论和实践背景，你最后 6 个月的准备应该包括仔细阅读“Cisco IOS 配置指南”。阅读每一章时，回顾“Cisco IOS 命令参考”中的相关章节，确保对于每一个协议，你已经熟悉 IOS 中的所有配置命令；然后在你的实验室中用不同的方法来配置各章节中所覆盖的协议。尝试不同的假设，并使协议运行在不常见的环境下。你会发现，当一种配置如你所料运行时，你并不能得到最好的经验，在与之相反的情况下，你却往往能学到更多的东西。

为你的配置和想法作笔记，记下它们是如何工作或者不工作的。记得去发掘记录所有与协议有关的调试工具，譬如 **debug** 和 **show** 命令。

在配置手册每一章的最后，你的目标是应该至少配置协议的最本质部分。当你参加 CCIE 实验室考试时，不用动太多的脑筋去配置相对简单的部分，抽出时间去思考困难的问题，这是很重要的。你也应该了解一个协议是怎样运作的，协议之间是如何相互影响的，存在哪些配置选项，以及怎样进行协议的故障诊断，等等。当你达到每一章的目标之后，进入下一章。

最后 6 个月的开始，你应该参加 CCIE 的笔试。不要被这个笔试所迷惑，它只是想淘汰那些完全没有准备的人，好让他们仅仅损失参加笔试所花的钱，而不是浪费参加实验室考试所需的更大的费用。如果你好好准备了，你会发现其实笔试并不是很难。

一些 Cisco 培训伙伴提供 CCIE 的准备课程和“CCIE boot camps”，给你一些在类似 CCIE 实验室考试的条件下做实验的经验。不要以这些课程取代勤奋的学习和实践。如果你选择参加这些课程，你首先应该认真准备了 CCIE 考试。从这些商业培训中得到的最大好处就是“在枪口下”的感觉——在紧迫的时间限制中解决困难问题。

C.6 参加考试

CCIE 实验室考试不仅仅测试你的实践和理论知识，而且测试你在压力下运用这些知识的能力。既然要参加了这样紧张的一天考试，就不要无谓地为自己再增加压力。

- 大多数人第一次参加 CCIE 实验室考试都会失败，因此要有参加第二次考试的准备。这能帮助你在参加第一次考试时保持冷静，或许，你就可能一次成功。
- 安排好你的行程，提前到达考场。考试之后再安排回去的行程。你显然不愿意考试期间考虑行程安排。
- 考试的前一天晚上确定考场的位置，不要因为迷了路慌慌张张出现在考场上。
- 考试前一天晚上最多做做小复习，如果你想填鸭式的恶补，只能使自己焦虑不安、失眠。
- 吃一顿好的晚餐，不要喝酒，美美睡一觉。
- 考试当天，吃一顿好的早餐。吃得好有助于你表现更出色，这是事实。
- 穿舒服一点，穿得好不会给你加分。

考试之前，会要求你在线在一张表格上签名，表明你不会泄漏实验室考试题目的细节。

考试长达 8 小时，有一次午餐休息。虽然其他的考生在同一个房间里考试，但你得独立完成。通常，其他考生的题目和你的是不同的。考官会分配给你一个网络拓扑和一组问题，你必须在 8 小时内完成所要求的配置。

在考试的任何一个部分，如果你不明白某个特定的要求，不要犹豫，去问监考官，他会帮助你。最重要的是，放松并集中注意力。

完成考试就会进行实验室考试评分，结果通常会在考试后的下一个工作日发送给你。你会收到一个通知你考试结果的电子邮件，并可以访问在线报考页面查看你的成绩。如果通过了考试，你将会看到你的 CCIE 编号；如果没有，你将会看到一个得分报告。

当你通过 CCIE 考试，你就完成了值得你万分骄傲的一件事。如果本书或者附录中的一些建议曾经帮助你达成你的目标，请给我发电子邮件，让我分享你的骄傲。



附录 D

复习题答案

第 1 章

1．TCP/IP 协议簇的 5 个层是：

- 物理层；
- 数据链路层；
- IP 层；
- 主机到主机层；
- 应用层。

物理层——包含关于物理介质的协议。

数据链路层——包含了如何控制物理层的一些协议：介质是怎样存取和共享的，介质上的设备是怎样标识的，数据在介质上传播之前是怎样成帧的。

IP 层——包含一些将数据链路逻辑分组到一个网络以及跨网络进行通信的协议。

主机到主机层——包含一些定义并控制中逻辑的、端到端的路径的协议。

应用层——对应于 OSI 中的会话层、表示层以及应用层。

2．现在使用最普遍的 IP 版本是版本 4。

3．当数据包的长度超过它所要去的那个数据链路的 MTU（Maximum Transmission Unit）时，路由器要将它分段。数据包中的数据将被分成小段，每一段被封装在独立的数据包中。接收端使用标识符、分段偏移域以及标记域的 MF 位来进行重组。

4．TTL 域防止丢失的数据包在 IP 网络中无休止地传播。该域包含一个 8 位整数，此数由产生数据包的主机设定。数据包每经过一台路由器，TTL 值将被减 1。如果一台路由器将 TTL 减至 0，它将丢弃该数据包并发送一个 ICMP 超时消息给数据包的源地址。

5．IP 地址的分类如下：

- A 类：第一个八位组字节的第 1 位是 0；
- B 类：第一个八位组字节的前 2 位是 10；
- C 类：第一个八位组字节的前 3 位是 110；
- D 类：第一个八位组字节的前 4 位是 1110；
- E 类：第一个八位组字节的前 4 位是 1111。

6．A、B、C 类 IP 地址用点分十进制以及二进制表示如下：

类	第一个八位组字节 二进制范围	第一个八位组字节 十进制范围
A	00000001～01111110	1～126
B	10000000～10111111	128～191
C	11000000～11011111	192～223

7．IP 地址掩码标识了 IP 地址的网络部分。32 位掩码中的 1 标识了 IP 地址中相应的网络位，0 标识了主机位。将 IP 地址和掩码进行布尔与（AND）运算，结果是，IP 地址中对应于掩码网络部分的那一段不变，而对应于主机部分的全变成 0。

8．子网化是对 A、B、C 类地址进行子分组。如果没有子网化，A、B、C 类的主 IP 地址的网络部分将只能标识一个数据链路。子网化使用主 IP 地址中的一些主机位作为网络位，允许一个单独的主地址被划分为多个网络地址。

9．有类别路由选择协议不能区分全 0 子网和主 IP 地址，也不能区分全 1 子网和主 IP 地址的全主机、全子网广播地址。

10．ARP（地址解析协议）的作用是将数据链路上接口的 IP 地址映射到相应的 MAC 地址。

11．代理 ARP 是 IP 路由器的功能之一。如果路由器收到一个 ARP 请求，并且

- 目标网络或子网在路由器的路由表中；
- 路由表指出目标可以通过某一个接口可达，该接口不同于接受到 ARP 请求的那个接口；
- 路由器将用自己的 MAC 地址对该 ARP 请求进行回应。

12．重定向是 IP 路由器的功能之一。如果一台设备向原设备发送给路由器一个数据包，而且该路由器必须转发该数据包至同一数据链路上的下一跳路由器，那么该路由器将向原设备发送一条其可以直接到达下一跳路由器的重定向消息。

13．TCP 在无连接的 IP 层之上提供了面向连接的服务。UDP 则提供了无连接服务。

14．序列号确保了准确的排序；校验、确认、计时器以及重传机制确保了可靠性；滑动窗口机制确保了流量控制。

15．MAC 地址是固定长度的二进制整数。如果用 MAC 地址作为 IP 地址的主机部分，那么子网化将不能得以实现。因为不可能灵活地使用一些主机位作为网络位。

16．UDP 报头的惟一目的是增加源端口及目的端口号。

第 2 章

1．IPv6 地址的长度是 128 位。

2．IPv6 地址可以表示为通过冒号分开的 8 个 16 位以十六进制表示的分段。

3．简化 IPv6 地址的两条规则是：

- 任何一个 16 位分段的前导 0 都可以省略；
- 由全 0 构成的一个或多个 16 位分段的任何单个连续的字符串都可以表示为一个双冒号。

4．使用多个双冒号会使地址变得含混不清，这样不能准确地确定每一个含 0 字符串的长度。

5．两个地址都是全 0。::/0 表示一个缺省地址，而::/128 表示一个未指定的地址。

6．单播 IPv6 地址的主机部分是接口 ID，长度通常是 64 位。

7．单播 IPv6 地址的子网 ID 是 16 位长。

8．起始于 FF80::/10 的 IPv6 地址是一个链路本地地址。

9．这是一个全球单播地址，开始 3 位以 001 标识。

10．任意播地址是表示一个服务的地址而不是表示一台设备的地址，因此它可以代表多台设备。

11．多播地址是表示一组设备的地址，不是表示单台设备的地址。

12．IPv6 报头的长度为 40 字节。

13．流标记字段通过在报头中标记各自不同的流（具有相同的源地址与目的地址和相同的源与目的端口的数据包），允许高颗粒度的负载分担，而不用因为检查数据包的负荷而降低性能。

14．IPv6 下一报头字段相当于 IPv4 中的协议号字段。命名之所以不同是因为，这个字段的值指定的可能是随后的协议报头，也可能是一个 IPv6 扩展报头。

15．跳数限制字段对应于 IPv4 中的生存时间（TTL）字段。命名改变是因为路由器从来没有根据传送时间来递减这个字段的值，而是在每经过一台传送的路由器时将该字段递减 1，实际上就是用跳数替代了传送时间。

16．IPv6 下一报头字段和 IPv4 中的协议号字段一样，也是一个 8 位字段；如果下一报头是一个上层协议报头，那么它指的就是协议号。但是它也可以指定为与协议号字段不同的字段；如果下一报头是一个 IPv6 扩展报头，那么它指的就是报头的类型号。

17．扩展报头使 IPv6 报头显得更具有效率，它可以专门指定专用功能，并只在这个专用功能使用的时候才被包含。

18．ICMPv6（对应于协议号）的下一报头值是 58。

19．除了分段扩展报头外，IPv6 分段与 IPv4 分段的重要不同是，IPv6 路由器不对数据包进行分段。它会告诉始发主机要么对数据包进行分段，要么确保不发起太大的数据包。

20．NDP 使用的 5 个 ICMPv6 消息是路由器请求（RS）、路由器通告（RA）、邻居请求（NS）、邻居通告（NA）和重定向。

21．设置 M 标记是告诉主机使用 DHCPv6 配置它的地址。O 标记告诉主机使用 DHCPv6 去查找其他链路参数。

22．可达时间字段指定了某个节点在确认邻居可达后应该假定它的邻居是可达的时间，以毫秒为单位。

23．重传计时器字段指定了连续传送的邻居请求之间节点应该等待的时间，以毫秒为单位。

24．在 RA 中路由器生存时间的值为 0，表示始发路由器不应该增加到一台主机的缺省路由器列表中去。

25．设置了 S 标记，表示 NA 是用来响应某个 NS 的。只有在 NA 用来响应一个请求时，才能表示双向可达已经确认，并且在邻居缓存中将邻居地址更改为可达状态；如果收到的是清除了 S 位的 NA，则表示它是未被请求的，并不在邻居缓存中改变它的状态。

26．全状态地址自动配置依靠 DHCPv6 给主机分配地址。无状态地址自动配置使用 RA 确定一个比链路本地更大范围的前缀，加上 MAC-to-EUI64 转换来确定主机地址。

27．MAC-to-EUI64 转换在 MAC 地址中间插入一个值 0xFFFE，接着反转 U/L 位为 1，从而通过一个 48 位的 MAC 地址创建了一个 64 位接口 ID。

28．对于任意播地址从来不需要执行地址冲突检测。

29．前缀 FF02:0:0:0:0:1:FF00::/104 用作被请求节点的多播地址。它加在被请求的地址的最后 24 位之前。

30．IPv6 使用 NDP 的邻居地址解析功能代替了 ARP，同时邻居缓存也代替了 ARP 缓存。

31．私有地址是随机生成的接口 ID，并在某些正常的周期或主机获取一个新的前缀时改变。它用来和一个自动配置的公共地址连在一起，确保主机的匿名性。公共地址用于可达性，而私有地址用于某台主机始发的所有数据包的源地址。

32．不完全状态表示该条目的邻居地址解析正在处理。

33．Probe 状态表示已经发送了一个 NS 去校验某个 Stale 状态的条目的双向可达性，但响应的 NA 还没有收到。

34．邻居不可达性检测校验了一个邻居的双向可达性，要么通过来自收到发送消息的确认的上层协议的“提示（Hints）”；要么通过主动探测带有 NS 的邻居。

第 3 章

1．路由表中的每一个表项至少要包括目标地址和下一跳的路由器地址，或者表明目标地址是直接相连的。

2．这意味着路由器知道，对于相同的主 IP 地址的不同子网，有多个子网掩码。

3．非连续子网指的是被一个不同的主 IP 地址分隔开的一个主 IP 网络地址的两个或多个子网。

4．**show ip route** 命令用来查看 Cisco 路由器的路由表。

5．命令 **show ipv6 route** 可以显示 IPv6 的路由表。显示的信息有前缀、前缀长度，以及下一跳地址或出站接口，同时还有管理距离和路由度量。

6．第一个数字是学习到该路由的路由选择协议的管理距离，第二个数字是该路由的度量值。

7．在静态路由表项中，如果使用本地接口来代替下一跳的地址，目标地址将作为直接连接的地址进入路由表。

8．汇总路由是一个单独的路由表项，指向多个子网或主 IP 地址。对于静态路由，汇总路由能减少需要配置的静态路由表项。

9．管理距离是一个路由选择协议或者静态路由的优先等级。每一个路由选择协议和静

态路由都有管理距离值。当一台路由器从多个路由选择协议得知到达同一目标地址的多个路由选择表项，它将使用管理距离最小的那条路由。

10．浮动静态路由是到达目标地址的备用路由。它的管理距离被设得很高，这样只有当别的优先级高的路由均不可用时，它才被派上用场。

11．等价负载均衡把流量分布在具有相同度量值的多条路径上；非等价负载均衡把流量分布在具有不同度量值的多条路径上。流量将根据路由代价分配，代价高的分配得少，代价低的分配得多。

12．如果在一个入站接口上配置了 CEF，那么数据包将使用 CEF 进行交换，并使用 CEF 负载均衡规则：根据所做的配置，对于 IPv4 是每目的地或每数据包的，对于 IPv6 是每目的地的。如果 CEF 没有在入站接口上配置，那么出站接口将确定交换模式，以及负载均衡的方法。如果一个接口是快速交换，将执行每目标负载均衡；如果一个接口是处理交换，将执行每数据包负载均衡。

13．当路由器需要转发数据包，但通过单一路由表查找却得不到它所需的所有信息，则会发生递归路由表查找。例如，路由器执行一次查找得到去往一个目标的路由，下一跳是路由器 A，但它却不知道该如何到达路由器 A，于是执行另外一次查找得到去往路由器 A 的路由。

第 4 章

1．路由选择协议是路由器之间所讲的一种“语言”，用来共享网络目标地址的信息。

2．一个路由选择协议至少要定义以下过程：

- 将网络的可达信息传递给其他路由器；
- 从其他路由器接收可达信息；
- 根据它所拥有的可达信息决定最佳路由，在路由表中记录该信息；
- 反应、补偿、宣告网络中的拓扑变化。

3．路由的度量值，也叫做路由代价或者路由距离，用来决定到达一个目的地的最佳路径。*最佳*由所使用的度量值类型定义。

4．收敛时间：一组路由器所花费用来完成路由选择信息交换的时间。

5．负载均衡是通过多条路径向同一目标地址传送数据包的过程。有 4 种类型的负载均衡：

- 等价，每数据包；
- 等价，每目的；
- 非等价，每数据包；
- 非等价，每目的。

6．距离矢量协议：每台路由器依据它邻居的路由来计算路由，然后传递给其他的邻居。

7．距离矢量协议存在的一些问题是：

- 因为它依靠邻居得到正确路由选择信息，所以易产生不正确的信息；
- 收敛慢；
- 路由环路；

- 计数无限。

8．邻居指的是连接至同一数据链路的路由器。

9．当路由超出特定期限之后，路由无效计时器将它们从路由表中删除。

10．使用简单的水平分隔，将不会发送路由信息到产生该路由信息的源点；具有毒性逆转的水平分隔虽然发送至源点，但把度量值设成不可达。

11．当路由环路更新路由时会产生计数到天穷大的问题；每台路由器增加路由的度量值直到度量值达到无穷大。可以把“无穷大”定义为合理的较低的度量值，这样可以很快达到无穷大，路由也就宣布为不可达，由此可以控制计数到无穷大的影响。

12．抑制计时器可用来预防路由选择环路。如果一条路由宣布为不可达或者度量值超过特定的阈值，路由器将停止接受有关该路由的其他信息，直到抑制计时器超时。这样，可以防止路由器在网络收敛时接受错误的路由选择信息。

13．距离矢量路由器发送它整个的路由表，但它仅仅发送给直接连接的邻居。链路状态路由器仅仅发送有关它直接相连链路的信息，但它广播该信息至整个网络区域。距离矢量协议通常使用不同的 Bellman－Ford 算法来计算路由，链路状态协议通常使用不同的 Dijkstra 算法来计算路由。

14．拓扑数据库保存了路由选择域中所有路由器产生的链路状态信息。

15．每台路由器广播链路状态信息通告，该通告描述其自身的链路、自身链路的状态，以及整个网络区域中连接至这些链路的所有邻居。所有的路由器在链路状态数据库中保存所有收到的这些链路状态通告。每台路由器根据拓扑数据库中的信息计算最短路径树，并且基于此树往路由表中添加路由。

16．序列号帮助路由器区分同一个链路状态通告的多个拷贝，并防止泛洪的链路状态通告在整个网络中无限循环。

17．Aging 防止老的、可能过期的链路状态信息在拓扑数据库中停留时间过长，或被一台路由器接受。

18．构建最短路径树时，路由器首先将它自己作为根，然后使用拓扑数据库中的信息，创建所有与它直连的邻居列表。到一个邻居的代价最小的路径将成为树的一个分枝，该路由器的所有邻居都被加入列表。检查该列表，看是否有重复路径；如果有，代价高的路径将从列表中删除，代价低的路由器将被加入树；该路由器的邻居也加入列表，再次检查该列表是否有重复路径。此过程不断重复，直到列表中没有路由器为止。

19．在一个路由选择域中，区域是子域。由于限制了区域中每台路由器的链路状态数据库的大小，区域使得链路状态路由更加高效。

20．根据使用时的需要，一个自主系统可被定义为一个在相同管理域中的网络，或者是一个单独的路由选择域。

21．内部网关协议是在自主系统内部进行路由的路由选择协议；外部网关协议是在自主系统间进行路由的路由选择协议。

第 5 章

1．RIP 使用 UDP 端口 520。

2．RIP 的度量值是跳数。不可达网络的跳数是 16，RIP 将它视为无限距离。

3．每隔 30s 减去一个小的随机变量，RIP 周期性地发送更新。这样可以防止和邻居的更新同步。

4．如果 6 次更新都未收到，则一个路由表项被标记为不可达。

5．当一条路由宣布为不可达时，启动垃圾收集计时器，或者称为刷新计时器。当计时器超时，该路由才从路由表中删除。这个过程使得一条不可达的路由在路由表中停留足够长的时间，以便邻居都能够被通知到。

6．随机计时器，定时范围 1～5s，防止在拓扑改变时触发更新“风暴”。

7．请求消息要求路由器进行更新。响应消息就是一个更新。

8．请求消息可能要求一个完全的更新，或者在一些特殊情况下，它会要求特定的路由。

9．当更新计时器超时，或者当接受到一条请求消息时，发出一条响应消息。

10．RIP 更新不包括目标地址的子网掩码，因此 RIP 路由器靠它自己接口的子网掩码来判断主网络地址如何被子网化。如果不配置特定的主网络地址，路由器将不知道主网络是被子网化的；因此，不会将主网络地址的子网通告给其他的主网络。

第 6 章

1．路由标记字段、子网掩码字段和下一跳字段是 RIPv2 的扩展，而在 RIPv1 消息中没有。RIP 消息的基本格式并没有改变；版本 2 仅仅添加了一些域。

2．除了使用一些新的域，RIPv2 支持身份验证和组播更新。

3．RIPv2 使用组播地址 224.0.0.9。组播路由选择消息比广播好，因为主机和非 RIPv2 路由器将忽略组播消息。

4．当另外的路由选择协议将 RIPv2 域作为它的传输域时，该协议便可以使用 RIPv2 的路由标记字段与 RIPv2 域另一端的对等体进行通信。

5．下一跳字段用来把下一跳的地址通知给在相同多址网络上的其他路由器，该下一跳地址到达目标地址在度量值上要比原路由器近。

6．RIPv2 和 RIPv1 使用相同的 UDP 端口，端口号 520。

7．RIPng 使用 UDP 端口号为 521。

8．无类别路由选择协议不考虑在路由发现过程中的主网络地址，仅仅寻找最长匹配。

9．为支持 VLSM，路由选择协议在其更新中必须包括每个目标地址的子网掩码。

10．RIPv2 的 Cisco 实现支持明文验证和 MD5 验证。RFC 2453 仅仅定义了明文验证。

第 7 章

1．EIGRP 是一种距离矢量协议。

2．缺省时，EIGRP 使用不超过 50%的链路带宽，带宽配置在路由器的接口上。这个百分比可以通过命令 **ip bandwidth-percent eigrp** 来改变。

3．EIGRP 和 IGRP 使用相同的公式来计算复合度量值。但是 EIGRP 通过一个因子 256

来扩展度量值。

EIGRP 的 4 个基本部分是：

- 依赖于协议的模块；
- 可靠传输协议；
- 邻居发现和恢复模块；
- 扩散更新算法。

4．可靠传输意味着 EIGRP 数据包可以有保证、按次序地传输。RTP 使用可靠组播，收到的数据包均被确认，以保证传输；使用序列号保证它们被有次序地传输。

5．序列号保证路由器接收的是最新的路由表项。

6．EIGRP 使用组播地址 224.0.0.10。

7．EIGRP 使用的数据包类型是：

- Hello 数据包；
- 确认数据包；
- 更新数据包；
- 查询数据包；
- 请求数据包；
- SIA-Queries；
- SIA-Replies。

8．缺省的 EIGRP Hello 间隔是 5s，而在一些较慢的接口（T-1 或更低）上，这个时间是 60s。

9．EIGRP 的缺省抑制时间是 Hello 间隔的 3 倍。

10．邻居表存储了有关 EIGRP-speaking 邻居的信息；拓扑表列出了所有具有可行后继的已知路由。

11．到一个目标网络的可行距离是，路由器计算出的到达目标的最小距离。

12．对一个目标网络来说，可行后继是通过可行条件选出的。如果一个邻居通告的到达一个目标网络的距离比收到该通告的路由器到此目标网络的可行距离要小，那么就满足可行条件。也就是说，如果一台路由器的邻居到达目标网络的距离比该路由器要近，那么这个邻居就是满足可行条件的。还有一种说法，即相对于目标网络而言，邻居是位于下游的。

13．可行后继指的是一个满足可行条件的邻居。

14．后继指的是目前正用来作为下一跳的可行后继。

15．在特定路由器上，如果它已经查询了它的邻居来寻找可行后继，但是还没有收到任何一个邻居的回应时，我们说，这台路由器上的路由是活跃的；当查询全部完成，路由则是非活跃的。

16．当拓扑表里没有可行后继时，路由变为活跃的。

17．从被查询的邻居收到一条回应时，路由从活跃变为非活跃。

18．当路由器在 active time（缺省 3min）内没有收到被查询邻居的回应，路由被宣布为 stuck-in-active。收到一条代表邻居的具有无限度量值的回应，以满足 DUAL，然后该邻居从邻居表中删除。

19．从一个 IP 网络地址产生一组子网地址，称为子网化。地址聚合指的是从一组网络或

子网地址汇总出一个 IP 网络地址。

第 8 章

1．从 OSPF 路由器的角度来说，邻居指的是直接与该 OSPF 路由器相邻的其他 OSPF 路由器。

2．OSPF 邻接是指到一个邻居的概念上的链路，可以通过该链路传送 LSA。

3．有 5 种 OSPF 数据包类型，它们的作用如下：

- Hello，用来发现邻居，建立和保持邻接；
- Update，用来在邻居间发送 LSA；
- 数据库描述数据包，被路由器用来在数据库同步的过程中向邻居描述它的链路状态数据库；
- 链路状态请求数据包，被路由器用来向邻居的链路状态数据库请求 LSA。
- 链路状态确认数据包，用来保证可靠的 LSA 传输。

4．路由器产生一个链路状态通告来描述一个或多个目标网络。OSPF 的 Update 数据包将 LSA 从一个邻居传送至另外一个邻居。虽然 LSA 在整个 OSPF 域或区域（area）内广播，但 Update 数据包不会离开单个的数据链路。

5．最常用的 LSA 类型和作用如下：

- 类型 1（Router LSA）由每台路由器产生，用来描述产生它的路由器、该路由器的直连链路和状态，以及该路由器的邻居。
- 类型 2（Network LSA）由多点访问链路上的指定路由器（Designated Router）产生，描述了该链路以及所有相连的邻居。
- 类型 3（Netrwork Summary LSA）由区域边界路由器（area border router）产生，描述了区域间的目标网络。
- 类型 4（ASBR Summary LSA）由区域边界路由器产生，描述了区域外的自主系统边界路由器（autonomous system boundary router）。
- 类型 5（AS External LSA）由自主系统边界路由器产生，描述 OSPF 域之外的目标网络。
- 类型 7（NSSA External LSA）由非末梢区域内的自主系统边界路由器产生。

6．路由器使用链路状态数据库来存储所有它知道的 OSPF LSA，包括它自己的。数据库同步指的是保证一个区域内所有的路由器都有一致的链路状态数据库的过程。

7．缺省的 OSPF HelloInterval 是 10s。

8．缺省的 OSPF RouterDeadInterval 是 HelloInterval 的 4 倍。

9．路由器 ID 是一个 OSPF 路由器用来标识它自己的地址。它或者是路由器所有回环（loopback）接口的最高地址；或者如果没有配置 loopback 接口，就是路由器所有 LAN 接口的最高地址。它也可以手工配置。

10．一个区域是一个 OSPF 子域，在区域内，所有的路由器都有一致的链路状态数据库。

11．区域 0 是骨干区域。其他所有的区域必须通过主干区域来发送它们的区域内流量。

12．MaxAge，1h，是 LSA 被认为过期的时限。

13．4 种 OSPF 路由器类型是：

- 内部路由器（IR）：它所有的 OSPF 接口属于同一个区域；
- 骨干路由器（BR）：是区域 0 内的内部路由器；
- 区域边界路由器（ABR）：在多个区域内有 OSPF 接口；
- 自主系统边界路由器（ASBR）：宣告外部路由至 OSPF 域。

14．4 种 OSPF 路径类型是：

- 域内路径；
- 域间路径；
- 类型 1 外部路径；
- 类型 2 外部路径。

15．5 种 OSPF 网络类型为：

- 点到点网络；
- 广播型网络；
- 非广播型多址网络；
- 点到多点网络；
- 虚链路。

16．指定路由器（DR）：代表了一个多路访问网络以及连接至这个网络和 OSPF 域的其余部分的路由器。

17．Cisco IOS 这样计算一个接口的出站代价：10^8/BW，其中 BW 是该接口上配置的带宽。10^8 可以通过 OSPF 命令 **auto-cost reference-bandwidth** 改变。

18．如果一个区域内的一台或者多台路由器不通过向区域外发送数据包就不能向区域内的其他路由器发送数据包，那么这个区域应该划分子区域。

19．虚链路指的是通过一个非骨干区域扩展 OSPF 骨干连接的通道。

20．末梢区域（stub area）是指类型 5 LSA 不能广播到的区域。完全末梢区域指的是类型 3、类型 4 或类型 5 LSA 不能广播到的区域（除了用来通告缺省路由的类型 3 LSA）。通过非末梢区域，外部的目标网络能够被通告至 OSPF 域内，但是类型 5 LSA 不能被 ABR 送至非末梢区域。

21．OSPF 网络表项是路由表中的表项，描述了 IP 目标网络。OSPF 路由器表项是在一个单独的路由表中的表项，该路由表只记录了到达 ABR 和 ASBR 的路由。

22．类型 2 的认证使用 MD5 加密，类型 1 认证使用明文密码。

23．LSA 报头中区别不同 LSA 的 3 个字段分别是类型、通告路由器、链路状态 ID。LSA 报头中区别相同 LSA 的不同实例的 3 个字段分别是序列号、老化时间及校验和。

第 9 章

1．在编写本书的时候，OSPFv3 还不能支持 IPv4。为了能够在 IPv4 和 IPv6 之间进行路由选择，我们必须同时运行 OSPFv2 和 OSPFv3。

2．每条链路上多个实例意味着，在连接到同一条广播链路上的不同路由器之间可以形成各自不同的邻接；因此不同的 OSPFv3 路由选择域能够使用相同的共享链路，相互之间互

不干扰。OSPFv3 报头中的 Instance ID 字段实现了这个特性。

3．OSPFv3 数据包可以使用内嵌的 IPv6 认证进行认证（依赖 IPv6 认证扩展报头）。OSPFv3 不需要像 OSPFv2 那样有自己的认证机制。

4．OSPFv3 下一报头号和 OSPFv2 的协议号相同，都是 89。

5．OSPFv3 使用保留的多播地址 FF02::5（AllSPFRouters）和 FF02::6（AllDRouters）。

6．不是。OSPFv3 使用和 OSPFv2 相同的 5 个消息数据包类型。

7．第 1 位是 U 位，用来指出如果接收路由器收到未知类型的 LSA 应该如何处理。第 2 位和第 3 位是 S 位，用来表示 LSA 的泛洪扩散范围。

8．OSPFv3 支持链路本地扩散范围，而 OSPFv2 不支持。链路 LSA 使用这个扩散范围。

9．OSPFv3 路由器和网络 LSA 不通告前缀，而 OSPFv2 的路由器和网络 LSA 通告前缀。

10．区域内前缀 LSA 携带的是与始发路由器相连的 IPv6 前缀。

11．链路 LSA 携带的仅仅是两个直接连接的邻居之间的信息。

第 10 章

1．中间系统是 ISO 称呼路由器的术语。

2．网络协议数据单元是 ISO 称呼数据包的术语。

3．L1 路由器与其他的区域没有直接连接。L2 路由器与 L1 路由器之间没有邻接关系。L1/L2 路由器路由区域间和区域内的流量，并且为 L1 路由器充当区域间网关。

4．Cisco 路由器的缺省配置是 L1/L2 类型。

5．IS-IS 区域的边界在路由器之间，在链路之上。OSPF 区域的边界由路由器自己定义。

6．两台具有相同 AID 的 L1/L2 路由器将同时形成 L1 和 L2 类型的邻接关系。两台具有不同 AID 的 L1/L2 路由器将只形成 L2 类型的邻接关系。

7．两台 L2 类型的路由器将形成一个 L2 邻接关系，而 AID 可以相同或不同。

8．网络实体标题是路由器标识自己和它所在区域的一个地址。

9．在一个 NET 内 NSAP 选择符应该设成 0x00。

10．系统 ID 在 IS-IS 域内惟一地标识了一台路由器。

11．NET 最后 7 个八位组字节前面的部分是区域地址。

12．IS-IS 不选取 BDR。

13．伪节点 ID 是 LAN ID 的最后一个八位组字节。它的作用是为了区别由一台路由器产生的多个 LAN ID，该路由器在多个 LAN 中是 DR。

14．IS-IS LSP 的 MaxAge 是 1200s（20min）。MaxAgee（或开始的剩余生存时间）可以最大配置为 65 535s。

15．OSPF 增加 age 至 MaxAge；IS-IS 减小 age 至 0。一个新的 OSPF LSA 有 age 值 0，而一个新的 IS-IS LSP 有 age 值 MaxAge。

16．IS-IS 路由器的刷新率是 900s（15min）。

17．一个完全序列号数据包（CSNP）列出了一个数据库中所有的 LSP。在一个广播网络上，指定路由器周期性地发送 CSNP 来保持数据库的同步。

18．一个部分序列号数据包列出了一个或多个 LSP。有两个用途：在点到点网络上，它

用来确认 LSP 的接收；在广播网络上，它用来请求 LSP。

19．IS-IS 路由器使用超载位通知它的邻居其内存过载了，并且不能存储整个链路状态数据库。

20．L1/L2 路由器使用 Attached 位通知 L1 路由器它与 L2 骨干有连接。

21．Up/down 位用来区分某个地址是区域内部始发的，还是泄漏到该区域的地址。

22．ISO 指定 4 个度量值：缺省度量（Default）、开销度量（Expense）、时延度量（Delay）、差错度量（Error）。Cisco 只支持 Default。

23．两种度量类型是普通度量值和扩展度量值，普通度量最大值是 63，扩展度量的最大值为 16 777 214。

24．一个 IS-IS 路由的最大度量值，在普通度量类型下是 1023，在扩展度量类型下是 4 261 412 864。

25．L1 IS-IS 度量值作用于区域内路由，L2 IS-IS 度量值作用于区域间路由。

26．内部度量值作用于目标网络在 IS-IS 域内的路由；外部度量值作用于目标网络在 IS-IS 域外的路由。

27．即使在多拓扑模式下同时配置 IPv4 和 IPv6，两台路由器之间也是形成单个邻接关系。

28．在单台路由器上可以配置多个 L1 区域，只配置一个 L2 区域。

29．两个活动的 mesh 组模式是 Blocked 和 Set（Numbered）模式。Blocked 模式可以大大减少泛洪扩散，但可能减少冗余性，增加收敛时间。Set 模式或编号 mesh 组减少的泛洪扩散负载不会像 Blocked 模式那样多，但是也降低了冗余性和收敛时间的潜在影响。

第 11 章

1．从其他的路由选择协议、同一路由选择协议的两个进程之间、静态路由或者直连到目的网络学来的路由可以被重分配到一个路由选择域。路由也可以在 IS-IS 第 1 层和第 2 层之间重新分配。

2．度量值用来在同一个路由选择协议产生的、到达同一目标网络的多条路由中间决定最佳路径；而管理距离用来在不同路由选择协议产生的、到达同一目标网络的多条路由中间决定最佳路径。

3．在一个具有较高管理距离值的路由选择域中，一条路由可以被重新分配到具有较低管理距离值的路由选择域。如果该路由被分配回高管理距离域，数据包可能会错误地路由至低管理距离域。

4．从无类别域到有类别域重新分配可变子网化的目标网络地址可能会有问题。有类别域可能不能识别来自无类别域重新分配的所有子网。

5．OSPF 和 IS-IS 能理解缺省度量值，而 RIP、IGRP 和 EIGRP 则不能。

6．**metric** 命令为特定的重新分配语句引入了度量值。**default-metric** 语句为所有不包括 **metric** 命令的重新分配语句引入度量值。

7．如果没有 **subnets** 关键字，只有不与路由器直连的主网络地址将被重新分配。

8．产生汇总路由的路由器应该使用 null 接口作为汇总路由的下一跳。如果有一个数据包，只能匹配汇总路由，但不能匹配更加具体的、能够到达目的地址的路由，那么这个数据

包将被丢弃。这防止了路由器转发丢失的数据包。

第 12 章

1．IPv4 的缺省路由地址是 0.0.0.0。

2．IPv6 的缺省前缀/前缀长度是::/0。

3．EIGRP 通告一个缺省地址作为外部地址类型。

4．是的。

5．末梢路由器是指只有一条链路连至其他路由器的路由器。末梢网络是指只连至一台路由器的网络。

6．使用缺省路由（而不是完整的路由表）使得路由表很小，这样能够节省路由器内存，并且能限制必须被处理的路由选择信息，节省路由器处理周期。

7．使用完整的路由表（而不是缺省路由）能使路由匹配更加精确。

8．ODR 使用 Cisco 发现协议（CDP）来发现路由。

9．ODR 在 IOS 11.2 和以后版本中可用。

10．ODR 运行的介质必须支持 SNAP。

第 14 章

1．路由映射和访问列表相似，它们都定义了匹配的标准，以及匹配后采取的动作。它们的不同点在于路由映射不光定义了匹配标准，还定义了设置（set）标准。设置标准能修改一条路由或者根据数据包的参数路由一个数据包。

2．策略路由是一种静态路由，它使用路由映射来决定哪些数据包应该转发，应该发往哪里。

3．路由标记是路由选择信息数据包中的一些域，它们允许在路由选择域内部能够携带外部信息。

4．路由标记不会影响携带它们的路由选择协议。



附录 E

配置练习答案

第 1 章

1．如果 D 类地址的前 4 位是 1110，那么第一个八位组字节最小是 11100000，最大是 11101111。用十进制表示，分别为 224 和 239。所以，D 类地址的第一个八位组字节取值从 224 到 239。

2．（a）需要足够的子网位数 *n*，使得 2*n*−2>=16 000；需要足够的主机位数 *h*，使得 2*h*−2>=700。子网掩码 255.255.252.0 为一个 A 类地址提供 16 382 个子网，为其中每一个子网提供 1022 个主机地址。此掩码是惟一的答案。如果多一个子网位（255.255.254.0），将没有足够的主机地址。如果少一个子网位（255.255.248.0），将没有足够的子网。

（b）需要足够的子网位数 *n*，使得 2*n*−2>=500；需要足够的主机位数 *h*，使得 2*h*−2>=100。子网掩码 255.255.255.128 为一个 B 类地址提供 510 个子网，为其中每一个子网提供 126 个主机地址。同样，此掩码是惟一的答案。

3．用 6 位来子网化，一个 C 类地址将有 $2^6-2=62$ 个子网，每个子网有 $2^2-2=2$ 个主机地址。以这种模式，一个 C 类地址可以被 62 个点到点链路所使用。一个点到点链路只需要两个主机地址——链路的每一端一个。

4．有 28 位掩码的 C 类地址可以有 14 个子网，每个子网有 14 个主机地址。首先给出子网。

这些子网是：

11111111111111111111111**1111**0000= 255.255.255.240（掩码）

11000000101010001001001**0001**0000= 192.168.147.16

11000000101010001001001**0010**0000= 192.168.147.32

11000000101010001001001**0011**0000 = 192.168.147.48
11000000101010001001001**0100**0000 = 192.168.147.64
11000000101010001001001**0101**0000 = 192.168.147.80
11000000101010001001001**0110**0000 = 192.168.147.96
11000000101010001001001**0111**0000 = 192.168.147.112
11000000101010001001001**1000**0000 = 192.168.147.128
11000000101010001001001**1001**0000 = 192.168.147.144
11000000101010001001001**1010**0000 = 192.168.147.160
11000000101010001001001**1011**0000 = 192.168.147.176
11000000101010001001001**1100**0000 = 192.168.147.192
11000000101010001001001**1101**0000 = 192.168.147.208
11000000101010001001001**1110**0000 = 192.168.147.224

下面给出每个子网的主机。每个子网的广播地址同样给出。

每个子网的主机地址是：

11000000101010001001001**0001**0000 = 192.168.147.16 (子网)
11000000101010001001001**0001**0001 = 192.168.147.17
11000000101010001001001**0001**0010 = 192.168.147.18
11000000101010001001001**0001**0011 = 192.168.147.19
11000000101010001001001**0001**0100 = 192.168.147.20
11000000101010001001001**0001**0101 = 192.168.147.21
11000000101010001001001**0001**0110 = 192.168.147.22
11000000101010001001001**0001**0111 = 192.168.147.23
11000000101010001001001**0001**1000 = 192.168.147.24
11000000101010001001001**0001**1001 = 192.168.147.25
11000000101010001001001**0001**1010 = 192.168.147.26
11000000101010001001001**0001**1011 = 192.168.147.27
11000000101010001001001**0001**1100 = 192.168.147.28
11000000101010001001001**0001**1101 = 192.168.147.29
11000000101010001001001**0001**1110 = 192.168.147.30
11000000101010001001001**0001**1111 = 192.168.147.31(广播)
11000000101010001001001**0010**0000 = 192.168.147.32(子网)
11000000101010001001001**0010**0001 = 192.168.147.33
11000000101010001001001**0010**0010 = 192.168.147.34
11000000101010001001001**0010**0011 = 192.168.147.35
11000000101010001001001**0010**0100 = 192.168.147.36
11000000101010001001001**0010**0101 = 192.168.147.37
11000000101010001001001**0010**0110 = 192.168.147.38
11000000101010001001001**0010**0111 = 192.168.147.39
11000000101010001001001**0010**1000 = 192.168.147.40
11000000101010001001001**0010**1001 = 192.168.147.41

11000000101010001001001100101010 = 192.168.147.42
11000000101010001001001100101011 = 192.168.147.43
11000000101010001001001100101100 = 192.168.147.44
11000000101010001001001100101101 = 192.168.147.45
11000000101010001001001100101110 = 192.168.147.46
11000000101010001001001100101111 = 192.168.147.47 (广播)
11000000101010001001001100110000 = 192.168.147.48 (子网)
11000000101010001001001100110001 = 192.168.147.49
11000000101010001001001100110010 = 192.168.147.50
11000000101010001001001100110011 = 192.168.147.51
11000000101010001001001100110100 = 192.168.147.52
11000000101010001001001100110101 = 192.168.147.53
11000000101010001001001100110110 = 192.168.147.54
11000000101010001001001100110111 = 192.168.147.55
11000000101010001001001100111000 = 192.168.147.56
11000000101010001001001100111001 = 192.168.147.57
11000000101010001001001100111010 = 192.168.147.58
11000000101010001001001100111011 = 192.168.147.59
11000000101010001001001100111100 = 192.168.147.60
11000000101010001001001100111101 = 192.168.147.61
11000000101010001001001100111110 = 192.168.147.62
11000000101010001001001100111111 = 192.168.147.63 (广播)
11000000101010001001001101000000 = 192.168.147.64 (子网)
11000000101010001001001101000001 = 192.168.147.65
11000000101010001001001101000010 = 192.168.147.66
11000000101010001001001101000011 = 192.168.147.67
11000000101010001001001101000100 = 192.168.147.68
11000000101010001001001101000101 = 192.168.147.69
11000000101010001001001101000110 = 192.168.147.70
11000000101010001001001101000111 = 192.168.147.71
11000000101010001001001101001000 = 192.168.147.72
11000000101010001001001101001001 = 192.168.147.73
11000000101010001001001101001010 = 192.168.147.74
11000000101010001001001101001011 = 192.168.147.75
11000000101010001001001101001100 = 192.168.147.76
11000000101010001001001101001101 = 192.168.147.77
11000000101010001001001101001110 = 192.168.147.78
11000000101010001001001101001111 = 192.168.147.79 (广播)
11000000101010001001001101010000 = 192.168.147.80 (子网)
11000000101010001001001101010001 = 192.168.147.81

11000000101010001001001101010010 = 192.168.147.82
11000000101010001001001101010011 = 192.168.147.83
11000000101010001001001101010100 = 192.168.147.84
11000000101010001001001101010101 = 192.168.147.85
11000000101010001001001101010110 = 192.168.147.86
11000000101010001001001101010111 = 192.168.147.87
11000000101010001001001101011000 = 192.168.147.88
11000000101010001001001101011001 = 192.168.147.89
11000000101010001001001101011010 = 192.168.147.90
11000000101010001001001101011011 = 192.168.147.91
11000000101010001001001101011100 = 192.168.147.92
11000000101010001001001101011101 = 192.168.147.93
11000000101010001001001101011110 = 192.168.147.94
11000000101010001001001101011111 = 192.168.147.95 (广播)
11000000101010001001001101100000 = 192.168.147.96 (子网)
11000000101010001001001101100001 = 192.168.147.97
11000000101010001001001101100010 = 192.168.147.98
11000000101010001001001101100011 = 192.168.147.99
11000000101010001001001101100100 = 192.168.147.100
11000000101010001001001101100101 = 192.168.147.101
11000000101010001001001101100110 = 192.168.147.102
11000000101010001001001101100111 = 192.168.147.103
11000000101010001001001101101000 = 192.168.147.104
11000000101010001001001101101001 = 192.168.147.105
11000000101010001001001101101010 = 192.168.147.106
11000000101010001001001101101011 = 192.168.147.107
11000000101010001001001101101100 = 192.168.147.108
11000000101010001001001101101101 = 192.168.147.109
11000000101010001001001101101110 = 192.168.147.110
11000000101010001001001101101111 = 192.168.147.111 (广播)
11000000101010001001001101110000 = 192.168.147.112 (子网)
11000000101010001001001101110001 = 192.168.147.113
11000000101010001001001101110010 = 192.168.147.114
11000000101010001001001101110011 = 192.168.147.115
11000000101010001001001101110100 = 192.168.147.116
11000000101010001001001101110101 = 192.168.147.117
11000000101010001001001101110110 = 192.168.147.118
11000000101010001001001101110111 = 192.168.147.119
11000000101010001001001101111000 = 192.168.147.120
11000000101010001001001101111001 = 192.168.147.121

11000000101010001001001101111010 = 192.168.147.122
11000000101010001001001101111011 = 192.168.147.123
11000000101010001001001101111100 = 192.168.147.124
11000000101010001001001101111101 = 192.168.147.125
11000000101010001001001101111110 = 192.168.147.126
11000000101010001001001101111111 = 192.168.147.127 (广播)
11000000101010001001001110000000 = 192.168.147.128 (子网)
11000000101010001001001110000001 = 192.168.147.129
11000000101010001001001110000010 = 192.168.147.130
11000000101010001001001110000011 = 192.168.147.131
11000000101010001001001110000100 = 192.168.147.132
11000000101010001001001110000101 = 192.168.147.133
11000000101010001001001110000110 = 192.168.147.134
11000000101010001001001110000111 = 192.168.147.135
11000000101010001001001110001000 = 192.168.147.136
11000000101010001001001110001001 = 192.168.147.137
11000000101010001001001110001010 = 192.168.147.138
11000000101010001001001110001011 = 192.168.147.139
11000000101010001001001110001100 = 192.168.147.140
11000000101010001001001110001101 = 192.168.147.141
11000000101010001001001110001110 = 192.168.147.142
11000000101010001001001110001111 = 192.168.147.143 (广播)
11000000101010001001001110010000 = 192.168.147.144 (子网)
11000000101010001001001110010001 = 192.168.147.145
11000000101010001001001110010010 = 192.168.147.146
11000000101010001001001110010011 = 192.168.147.147
11000000101010001001001110010100 = 192.168.147.148
11000000101010001001001110010101 = 192.168.147.149
11000000101010001001001110010110 = 192.168.147.150
11000000101010001001001110010111 = 192.168.147.151
11000000101010001001001110011000 = 192.168.147.152
11000000101010001001001110011001 = 192.168.147.153
11000000101010001001001110011010 = 192.168.147.154
11000000101010001001001110011011 = 192.168.147.155
11000000101010001001001110011100 = 192.168.147.156
11000000101010001001001110011101 = 192.168.147.157
11000000101010001001001110011110 = 192.168.147.158
11000000101010001001001110011111 = 192.168.147.159 (广播)
11000000101010001001001110100000 = 192.168.147.160 (子网)
11000000101010001001001110100001 = 192.168.147.161

11000000101010001001001**1010**0010 = 192.168.147.162
11000000101010001001001**1010**0011 = 192.168.147.163
11000000101010001001001**1010**0100 = 192.168.147.164
11000000101010001001001**1010**0101 = 192.168.147.165
11000000101010001001001**1010**0110 = 192.168.147.166
11000000101010001001001**1010**0111 = 192.168.147.167
11000000101010001001001**1010**1000 = 192.168.147.168
11000000101010001001001**1010**1001 = 192.168.147.169
11000000101010001001001**1010**1010 = 192.168.147.170
11000000101010001001001**1010**1011 = 192.168.147.171
11000000101010001001001**1010**1100 = 192.168.147.172
11000000101010001001001**1010**1101 = 192.168.147.173
11000000101010001001001**1010**1110 = 192.168.147.174
11000000101010001001001**1010**1111 = 192.168.147.175 (广播)
11000000101010001001001**1011**0000 = 192.168.147.176 (子网)
11000000101010001001001**1011**0001 = 192.168.147.177
11000000101010001001001**1011**0010 = 192.168.147.178
11000000101010001001001**1011**0011 = 192.168.147.179
11000000101010001001001**1011**0100 = 192.168.147.180
11000000101010001001001**1011**0101 = 192.168.147.181
11000000101010001001001**1011**0110 = 192.168.147.182
11000000101010001001001**1011**0111 = 192.168.147.183
11000000101010001001001**1011**1000 = 192.168.147.184
11000000101010001001001**1011**1001 = 192.168.147.185
11000000101010001001001**1011**1010 = 192.168.147.186
11000000101010001001001**1011**1011 = 192.168.147.187
11000000101010001001001**1011**1100 = 192.168.147.188
11000000101010001001001**1011**1101 = 192.168.147.189
11000000101010001001001**1011**1110 = 192.168.147.190
11000000101010001001001**1011**1111 = 192.168.147.191 (广播)
11000000101010001001001**1100**0000 = 192.168.147.192 (子网)
11000000101010001001001**1100**0001 = 192.168.147.193
11000000101010001001001**1100**0010 = 192.168.147.194
11000000101010001001001**1100**0011 = 192.168.147.195
11000000101010001001001**1100**0100 = 192.168.147.196
11000000101010001001001**1100**0101 = 192.168.147.197
11000000101010001001001**1100**0110 = 192.168.147.198
11000000101010001001001**1100**0111 = 192.168.147.199
11000000101010001001001**1100**1000 = 192.168.147.200
11000000101010001001001**1100**1001 = 192.168.147.201

11000000101010001001001**1100**1010 = 192.168.147.202
11000000101010001001001**1100**1011 = 192.168.147.203
11000000101010001001001**1100**1100 = 192.168.147.204
11000000101010001001001**1100**1101 = 192.168.147.205
11000000101010001001001**1100**1110 = 192.168.147.206
11000000101010001001001**1100**1111 = 192.168.147.207 (广播)
11000000101010001001001**1101**0000 = 192.168.147.208 (子网)
11000000101010001001001**1101**0001 = 192.168.147.209
11000000101010001001001**1101**0010 = 192.168.147.210
11000000101010001001001**1101**0011 = 192.168.147.211
11000000101010001001001**1101**0100 = 192.168.147.212
11000000101010001001001**1101**0101 = 192.168.147.213
11000000101010001001001**1101**0110 = 192.168.147.214
11000000101010001001001**1101**0111 = 192.168.147.215
11000000101010001001001**1101**1000 = 192.168.147.216
11000000101010001001001**1101**1001 = 192.168.147.217
11000000101010001001001**1101**1010 = 192.168.147.218
11000000101010001001001**1101**1011 = 192.168.147.219
11000000101010001001001**1101**1100 = 192.168.147.220
11000000101010001001001**1101**1101 = 192.168.147.221
11000000101010001001001**1101**1110 = 192.168.147.222
11000000101010001001001**1101**1111 = 192.168.147.223 (广播)
11000000101010001001001**1110**0000 = 192.168.147.224 (子网)
11000000101010001001001**1110**0001 = 192.168.147.225
11000000101010001001001**1110**0010 = 192.168.147.226
11000000101010001001001**1110**0011 = 192.168.147.227
11000000101010001001001**1110**0100 = 192.168.147.228
11000000101010001001001**1110**0101 = 192.168.147.229
11000000101010001001001**1110**0110 = 192.168.147.230
11000000101010001001001**1110**0111 = 192.168.147.231
11000000101010001001001**1110**1000 = 192.168.147.232
11000000101010001001001**1110**1001 = 192.168.147.233
11000000101010001001001**1110**1010 = 192.168.147.234
11000000101010001001001**1110**1011 = 192.168.147.235
11000000101010001001001**1110**1100 = 192.168.147.236
11000000101010001001001**1110**1101 = 192.168.147.237
11000000101010001001001**1110**1110 = 192.168.147.238
11000000101010001001001**1110**1111 = 192.168.147.239 (广播)

5. 这里给出此题的答案，可以看出比上一题短，因为没有写出每一个子网的每一台主机地址。有 29 位掩码的 C 类地址意味着将有 30 个子网，每个子网有 6 个主机地址。

这些子网是：

11111111111111111111111**11111**000 = 255.255.255.248 (mask)

11000000101010001001001**100001**000 = 192.168.147.8

11000000101010001001001**100010**000 = 192.168.147.16

11000000101010001001001**100011**000 = 192.168.147.24

11000000101010001001001**100100**000 = 192.168.147.32

11000000101010001001001**100101**000 = 192.168.147.40

11000000101010001001001**100110**000 = 192.168.147.48

11000000101010001001001**100111**000 = 192.168.147.56

11000000101010001001001**101000**000 = 192.168.147.64

11000000101010001001001**101001**000 = 192.168.147.72

11000000101010001001001**101010**000 = 192.168.147.80

11000000101010001001001**101011**000 = 192.168.147.88

11000000101010001001001**101100**000 = 192.168.147.96

11000000101010001001001**101101**000 = 192.168.147.104

11000000101010001001001**101100**000 = 192.168.147.112

11000000101010001001001**101111**000 = 192.168.147.120

11000000101010001001001**110000**000 = 192.168.147.128

11000000101010001001001**110001**000 = 192.168.147.136

11000000101010001001001**110010**000 = 192.168.147.144

11000000101010001001001**110011**000 = 192.168.147.152

11000000101010001001001**110100**000 = 192.168.147.160

11000000101010001001001**110101**000 = 192.168.147.168

11000000101010001001001**110110**000 = 192.168.147.176

11000000101010001001001**110111**000 = 192.168.147.184

11000000101010001001001**111000**000 = 192.168.147.192

11000000101010001001001**111001**000 = 192.168.147.200

11000000101010001001001**111010**000 = 192.168.147.208

11000000101010001001001**111011**000 = 192.168.147.216

11000000101010001001001**111100**000 = 192.168.147.224

11000000101010001001001**111101**000 = 192.168.147.232

11000000101010001001001**111110**000 = 192.168.147.240

下面给出每个子网的广播地址（将每个子网的主机位全置为 1）。

子网广播地址是：

11000000101010001001001**100001**111 = 192.168.147.15

11000000101010001001001**100010**111 = 192.168.147.23

11000000101010001001001**100011**111 = 192.168.147.31

11000000101010001001001**100100**111 = 192.168.147.39

11000000101010001001001**100101**111 = 192.168.147.47

11000000101010001001001**100110**111 = 192.168.147.55

11000000101010001001001100111111 = 192.168.147.63
11000000101010001001001101000111 = 192.168.147.71
11000000101010001001001101001111 = 192.168.147.79
11000000101010001001001101010111 = 192.168.147.87
11000000101010001001001101011111 = 192.168.147.95
11000000101010001001001101100111 = 192.168.147.103
11000000101010001001001101101111 = 192.168.147.111
11000000101010001001001101100111 = 192.168.147.119
11000000101010001001001101111111 = 192.168.147.127
11000000101010001001001110000111 = 192.168.147.135
11000000101010001001001110001111 = 192.168.147.143
11000000101010001001001110010111 = 192.168.147.151
11000000101010001001001110011111 = 192.168.147.159
11000000101010001001001110100111 = 192.168.147.167
11000000101010001001001110101111 = 192.168.147.175
11000000101010001001001110110111 = 192.168.147.183
11000000101010001001001110111111 = 192.168.147.191
11000000101010001001001111000111 = 192.168.147.199
11000000101010001001001111001111 = 192.168.147.207
11000000101010001001001111010111 = 192.168.147.215
11000000101010001001001111011111 = 192.168.147.223
11000000101010001001001111100111 = 192.168.147.231
11000000101010001001001111101111 = 192.168.147.239
11000000101010001001001111110111 = 192.168.147.247

最后，给出每个子网的主机地址，它们是介于子网地址和子网广播地址之间的地址。

子　网	广　播	主 机 地 址
192.168.147.8	192.168.147.15	192.168.147.9 - 192.168.147.14
192.168.147.16	192.168.147.23	192.168.147.17 - 192.168.147.22
192.168.147.24	192.168.147.31	192.168.147.25 - 192.168.147.30
192.168.147.32	192.168.147.39	192.168.147.33 - 192.168.147.38
192.168.147.40	192.168.147.47	192.168.147.41 - 192.168.147.46
192.168.147.48	192.168.147.55	192.168.147.49 - 192.168.147.54
192.168.147.56	192.168.147.63	192.168.147.57 - 192.168.147.62
192.168.147.64	192.168.147.71	192.168.147.65 - 192.168.147.70
192.168.147.72	192.168.147.79	192.168.147.73 - 192.168.147.78
192.168.147.80	192.168.147.87	192.168.147.81 - 192.168.147.86
192.168.147.88	192.168.147.95	192.168.147.89 - 192.168.147.94
192.168.147.96	192.168.147.103	192.168.147.97 - 192.168.147.102
192.168.147.104	192.168.147.111	192.168.147.105 - 1 92.168.147.110
192.168.147.112	192.168.147.119	192.168.147.113 - 192.168.147.118
192.168.147.120	192.168.147.127	192.168.147.121 - 192.168.147.126
192.168.147.128	192.168.147.135	192.168.147.129 - 192.168.147.134

续表

子 网	广 播	主 机 地 址
192.168.147.136	192.168.147.143	192.168.147.137 - 192.168.147.142
192.168.147.144	192.168.147.151	192.168.147.145 - 192.168.147.150
192.168.147.152	192.168.147.159	192.168.147.153 - 192.168.147.158
192.168.147.160	192.168.147.167	192.168.147.161 - 192.168.147.166
192.168.147.168	192.168.147.175	192.168.147.169 - 192.168.147.174
192.168.147.176	192.168.147.183	192.168.147.177 - 192.168.147.182
192.168.147.184	192.168.147.191	192.168.147.185 - 192.168.147.190
192.168.147.192	192.168.147.199	192.168.147.193 - 192.168.147.198
192.168.147.200	192.168.147.207	192.168.147.201 - 192.168.147.206
192.168.147.208	192.168.147.215	192.168.147.209 - 192.168.147.214
192.168.147.216	192.168.147.223	192.168.147.217 - 192.168.147.222
192.168.147.224	192.168.147.231	192.168.147.225 - 192.168.147.230
192.168.147.232	192.168.147.239	192.168.147.233 - 192.168.147.238
192.168.147.240	192.168.147.247	192.168.147.241 - 192.168.147.246

6．一个有 20 位掩码的 B 类地址将有 14 个子网，每个子网有 4 094 个主机地址。

这些子网是：

11111111111111111111000000000000 = 255.255.240.0 （掩码）

10101100000100000001000000000000 = 172.16.16.0

10101100000100000010000000000000 = 172.16.32.0

10101100000100000011000000000000 = 172.16.48.0

10101100000100000100000000000000 = 172.16.64.0

10101100000100000101000000000000 = 172.16.80.0

10101100000100000110000000000000 = 172.16.96.0

10101100000100000111000000000000 = 172.16.112.0

10101100000100001000000000000000 = 172.16.128.0

10101100000100001001000000000000 = 172.16.144.0

10101100000100001010000000000000 = 172.16.160.0

10101100000100001011000000000000 = 172.16.176.0

10101100000100001100000000000000 = 172.16.192.0

10101100000100001101000000000000 = 172.16.208.0

10101100000100001110000000000000 = 172.16.224.0

子网广播地址是：

10101100000100000001111111111111 = 172.16.31.255

10101100000100000010111111111111 = 172.16.47.255

10101100000100000011111111111111 = 172.16.63.255

10101100000100000100111111111111 = 172.16.79.255

10101100000100000101111111111111 = 172.16.95.255

10101100000100000110111111111111 = 172.16.111.255

10101100000100000111111111111111 = 172.16.127.255

10101100000100001000111111111111 = 172.16.143.255

10101100000100001001111111111111 = 172.16.159.255
10101100000100001010111111111111 = 172.16.175.255
10101100000100001011111111111111 = 172.16.191.255
10101100000100001100111111111111 = 172.16.207.255
10101100000100001101111111111111 = 172.16.223.255
10101100000100001110111111111111 = 172.16.239.255

使用以上的子网和广播地址，主机地址是：

子　网	广　播	主机地址
172.16.16.0	172.16.31.255	172.16.16.1-172.16.31.254
172.16.32.0	172.16.47.255	172.16.32.1-172.16.47.254
172.16.48.0	172.16.63.255	172.16.48.1-172.16.63.254
172.16.64.0	172.16.79.255	172.16.64.1-172.16.79.254
172.16.80.0	172.16.95.255	172.16.80.1-172.16.95.254
172.16.96.0	172.16.111.255	172.16.96.1-172.16.111.254
172.16.112.0	172.16.127.255	172.16.112.1-172.16.127.254
172.16.128.0	172.16.143.255	172.16.128.1-172.16.143.254
172.16.144.0	172.16.159.255	172.16.144.1-172.16.159.254
172.16.160.0	172.16.175.255	172.16.160.1-172.16.175.254
172.16.176.0	172.16.191.255	172.16.176.1-172.16.191.254
172.16.192.0	172.16.207.255	172.16.192.1-172.16.207.254
172.16.208.0	172.16.223.255	172.16.208.1-172.16.223.254
172.16.224.0	172.16.239.255	172.224.16.1-172.16.239.254

第 3 章

1．首先确定每个链路的子网地址，然后写出静态路由。记住路由器的路由表中将缺省包含直接相连的子网。静态路由是：

RTA

```
ip route 192.168.2.64 255.255.255.224 192.168.2.131
ip route 192.168.2.160 255.255.255.224 192.168.2.131
ip route 192.168.1.128 255.255.255.240 192.168.2.131
ip route 192.168.1.16 255.255.255.240 192.168.2.131
ip route 192.168.2.32 255.255.255.224 192.168.2.131
ip route 192.168.1.160 255.255.255.240 192.168.2.131
ip route 10.1.1.0 255.255.255.0 192.168.2.131
ip route 10.1.3.0 255.255.255.0 192.168.2.131
ip route 10.1.2.0 255.255.255.0 192.168.2.131
```

RTB

```
ip route 10.1.4.0 255.255.255.0 192.168.2.132
ip route 192.168.1.128 255.255.255.240 192.168.2.174
ip route 192.168.1.16 255.255.255.240 192.168.2.174
ip route 192.168.2.32 255.255.255.224 192.168.2.174
ip route 192.168.1.160 255.255.255.240 192.168.2.174
ip route 10.1.1.0 255.255.255.0 192.168.2.174
```

```
ip route 10.1.3.0 255.255.255.0 192.168.2.174
ip route 10.1.2.0 255.255.255.0 192.168.2.174
```

RTC

```
ip route 10.1.4.0 255.255.255.0 192.168.2.185
ip route 192.168.2.128 255.255.255.224 192.168.2.185
ip route 192.168.2.64 255.255.255.224 192.168.2.185
ip route 192.168.2.32 255.255.255.224 192.168.1.20
ip route 10.1.1.0 255.255.255.0 192.168.1.173
ip route 10.1.3.0 255.255.255.0 192.168.1.173
ip route 10.1.2.0 255.255.255.0 192.168.1.173
```

RTD

```
ip route 10.1.4.0 255.255.255.0 192.168.1.29
ip route 192.168.2.128 255.255.255.224 192.168.1.29
ip route 192.168.2.64 255.255.255.224 192.168.1.29
ip route 192.168.2.160 255.255.255.224 192.168.1.29
ip route 192.168.1.128 255.255.255.240 192.168.1.29
ip route 192.168.1.160 255.255.255.240 192.168.1.29
ip route 10.1.1.0 255.255.255.0 192.168.1.29
ip route 10.1.3.0 255.255.255.0 192.168.1.29
ip route 10.1.2.0 255.255.255.0 192.168.1.29
```

RTE

```
ip route 10.1.4.0 255.255.255.0 192.168.1.163
ip route 192.168.2.128 255.255.255.224 192.168.1.163
ip route 192.168.2.64 255.255.255.224 192.168.1.163
ip route 192.168.2.160 255.255.255.224 192.168.1.163
ip route 192.168.1.128 255.255.255.240 192.168.1.163
ip route 192.168.1.16 255.255.255.240 192.168.1.163
ip route 192.168.2.32 255.255.255.224 192.168.1.163
ip route 10.1.2.0 255.255.255.0 10.1.3.2
```

RTF

```
ip route 10.1.4.0 255.255.255.0 10.1.3.1
ip route 192.168.2.128 255.255.255.224 10.1.3.1
ip route 192.168.2.64 255.255.255.224 10.1.3.1
ip route 192.168.2.160 255.255.255.224 10.1.3.1
ip route 192.168.1.128 255.255.255.240 10.1.3.1
ip route 192.168.1.16 255.255.255.240 10.1.3.1
ip route 192.168.2.32 255.255.255.224 10.1.3.1
ip route 192.168.1.160 255.255.255.240 10.1.3.1
ip route 10.1.1.0 255.255.255.0 10.1.3.1
```

2．静态路由是：

RTA

```
ip route 192.168.0.0 255.255.0.0 192.168.2.131
ip route 10.1.0.0 255.255.0.0 192.168.2.131
```

RTB

```
ip route 10.1.4.0 255.255.255.0 192.168.2.132
ip route 192.168.0.0 255.255.0.0 192.168.2.174
ip route 10.1.0.0 255.255.0.0 192.168.2.174
```

RTC

```
ip route 10.1.4.0 255.255.255.0 192.168.2.185
ip route 192.168.2.32 255.255.255.224 192.168.1.20
ip route 10.1.0.0 255.255.0.0 192.168.1.173
ip route 192.168.2.64 255.255.255.224 192.168.2.185
ip route 192.168.2.128 255.255.255.224 192.168.2.185
```

RTD

```
ip route 10.1.0.0 255.255.0.0 192.168.1.29
ip route 192.168.0.0 255.255.05.0 192.168.1.29
```

RTE

```
ip route 10.1.4.0 255.255.255.0 192.168.1.163
ip route 192.168.0.0 255.255.0.0 192.168.1.163
ip route 10.1.2.0 255.255.255.0 10.1.3.2
```

RTF

```
ip route 10.1.0.0 255.255.0.0 10.1.3.1
ip route 192.168.0.0 255.255.0.0 10.1.3.1
```

3．静态路由是：

RTA

```
ip route 172.16.7.0 255.255.255.0 172.16.2.2
ip route 172.16.7.0 255.255.255.0 172.16.4.2 50
ip route 172.16.6.0 255.255.255.0 172.16.2.2
ip route 172.16.6.0 255.255.255.0 172.16.4.2 50
ip route 172.16.8.0 255.255.255.0 172.16.4.2
ip route 172.16.8.0 255.255.255.0 172.16.2.2 50
ip route 172.16.5.0 255.255.255.0 172.16.4.2
ip route 172.16.5.0 255.255.255.0 172.16.2.2 50
ip route 172.16.9.0 255.255.255.0 172.16.2.2
ip route 172.16.9.0 255.255.255.0 172.16.4.2
```

RTB

```
ip route 172.16.1.0 255.255.255.0 172.16.2.1
ip route 172.16.1.0 255.255.255.0 172.16.6.1 50
ip route 172.16.4.0 255.255.255.0 172.16.2.1
ip route 172.16.4.0 255.255.255.0 172.16.6.1 50
ip route 172.16.9.0 255.255.255.0 172.16.6.1
ip route 172.16.9.0 255.255.255.0 172.16.2.1 50
ip route 172.16.5.0 255.255.255.0 172.16.6.1
ip route 172.16.5.0 255.255.255.0 172.16.2.1 50
ip route 172.16.8.0 255.255.255.0 172.16.6.1
ip route 172.16.8.0 255.255.255.0 172.16.2.1
```

RTC

```
ip route 172.16.1.0 255.255.255.0 172.16.6.2
ip route 172.16.1.0 255.255.255.0 172.16.5.1
ip route 172.16.4.0 255.255.255.0 172.16.5.1
ip route 172.16.4.0 255.255.255.0 172.16.6.2 50
ip route 172.16.2.0 255.255.255.0 172.16.6.2
ip route 172.16.2.0 255.255.255.0 172.16.5.1 50
```

```
ip route 172.16.7.0 255.255.255.0 172.16.6.2
ip route 172.16.7.0 255.255.255.0 172.16.5.1 50
ip route 172.16.8.0 255.255.255.0 172.16.5.1
ip route 172.16.8.0 255.255.255.0 172.16.6.2 50
```

RTD

```
ip route 172.16.1.0 255.255.255.0 172.16.4.1
ip route 172.16.1.0 255.255.255.0 172.16.5.2 50
ip route 172.16.2.0 255.255.255.0 172.16.4.1
ip route 172.16.2.0 255.255.255.0 172.16.5.2 50
ip route 172.16.9.0 255.255.255.0 172.16.5.2
ip route 172.16.9.0 255.255.255.0 172.16.4.1 50
ip route 172.16.6.0 255.255.255.0 172.16.5.2
ip route 172.16.6.0 255.255.255.0 172.16.4.1 50
ip route 172.16.7.0 255.255.255.0 172.16.5.2
ip route 172.16.7.0 255.255.255.0 172.16.4.1
```

第 5 章

1．除了这里给出的 RIP 配置，在 RTE 和 RTF 之间还必须使用 2 级地址配置 192.168.5.0 的一个子网。否则子网 192.168.5.192/27 和 192.168.5.96/27 不连续。RIP 配置如下：

RTA

```
router rip
  network 192.168.2.0
```

RTB

```
router rip
  network 192.168.2.0
```

RTC

```
router rip
  network 192.168.2.0
  network 192.168.3.0
```

RTD

```
interface ethernet 0
  ip address 192.168.4.3 255.255.255.0
  ip address 192.168.5.3 255.255.255.224 secondary
router rip
  network 192.168.3.0
  network 192.168.4.0
```

RTE

```
interface ethernet 0
  ip address 192.168.4.1 255.255.255.0
  ip address 192.168.5.1 255.255.255.224 secondary
router rip
  network 192.168.4.0
  network 192.168.5.0
```

RTF

```
interface ethernet 0
  ip address 192.168.4.2 255.255.255.0
  ip address 192.168.5.2 255.255.255.224 secondary
router rip
  network 192.168.4.0
  network 192.168.6.0
```

2．为了在 RTC 和 RTD 之间单播 RIP 更新，配置为：

RTC

```
router rip
  network 192.168.2.0
  neighbor 192.168.3.2
```

RTD

```
router rip
  network 192.168.3.0
  neighbor 192.168.3.1
```

3．更新时间作用于整个 RIP 进程。如果串行链路的更新时间变了，路由器其他链路的更新时间也要改变。这就意味着邻居路由器的计时器要发生变化，邻居路由器的邻居的计时器也要顺次变化，等等。在一台路由器上改变更新计时器的级联效应导致 RIP 域中每台路由器的计时器都要变化。在每台路由器上增加 RIP 更新周期至 2min 的命令为：

```
timers basic 120 360 360 480
```

因为更新计时器改变了，所以无效计时器、抑制计时器和刷新计时器也必须改变。像缺省的那样，把无效计时器和抑制计时器设为更新计时器的 6 倍，将使得这个网络的转换时间很长。所以，把无效计时器和抑制计时器设为更新计时器的 3 倍。刷新计时器必须比抑制计时器长，所以这里把它设成长 60s。

4．从 RTA 到网络 192.168.4.0 有两跳，所以给度量值增加 14 将会使路由的度量值达到 16（不可达）。记得在配置练习 1 中，192.168.5.0 的一个子网必须在和 192.168.4.0 相同的链路上使用 2 级地址来配置，这样 192.168.5.0 的子网才连续。因此，从 RTB 到 192.168.5.0 也是两跳。假设 RTA 和 RTB 上连接到 RTC 的接口都是 E0，配置为：

RTA

```
router rip
offset-list 1 in 14 Ethernet0
network 192.168.2.0
!
access-list 1 permit 192.168.4.0 0.0.0.0
```

RTB

```
router rip
offset-list 1 in 14 Ethernet0
network 192.168.2.0
!
access-list 1 permit 192.168.5.0 0.0.0.0
```

5．RTB 有更长一点的掩码，能正确解释所有子网。问题在 RTA 上。RTA 有 27 位掩码，它把 RTB 的子网 192.168.20.40/29 和 192.168.20.49/29 解释为 192.168.20.32/27——与它的直连链路相同的子网。因此，RTA 没有整个网络的正确视图。

但是，如果代理 ARP 启用了，数据包仍然能被路由。例如，假设 RTA 要路由一个目的

地址是 192.168.20.50 的数据包。RTA 错误地把这个地址当作是它的子网 192.168.20.32/27 的成员，并且在该子网上发出 ARP 请求，要求得到 192.168.20.50 的 MAC 地址。RTB 听到 ARP 请求，它正确地解释这个地址为它的子网 192.168.20.48/29 的一个成员，并且回应它在 192.168.20.32/29 上的接口的 MAC 地址。RTA 然后向 RTB 转发数据包，RTB 再将这个数据包转发至正确目的地。如果代理 ARP 没有启用，数据包将不会被正确地从 RTA 转发至 RTB。

第 6 章

1．Taos 的 RIP 配置中可增加 **neighbor 172.25.150.206** 语句，使 Taos 单播 RIP 更新至该地址。该措施仅当下面情况满足时有效。Pojoaque 的 RIPv1 进程符合这样的规则：版本号高于 1 的 RIP 消息的未用域被忽略，剩下的数据包被处理。

2．首先计算有最大主机数的子网。然后利用未用的子网位，计算有次最大主机数的子网，等等。记住，当用位组来表示子网时，没有连续的子网能以相同的位组开始。例如，如果第一个子网开始于 00，所有后续的子网必须开始于 01、10 或 11。如果第二个子网开始于 010，后续的子网不能开始于 010。

一种解答如下（子网位用黑体表示）：

00000000	192.168.100.0/26 (62 hosts)
01000000	192.168.100.64/27 (30 hosts)
01100000	192.168.100.96/28 (14 hosts)
01110000	192.168.100.112/28 (14 hosts)
10000000	192.168.100.128/28 (14 hosts)
10010000	192.168.100.144/28 (14 hosts)
10100000	192.168.100.160/28 (14 hosts)
10110000	192.168.100.176/29 (8 hosts)
10111000	192.168.100.184/29 (8 hosts)
11000000	192.168.100.192/29 (8 hosts)
11001000	192.168.100.200/29 (8 hosts)
11010000	192.168.100.208/29 (8 hosts)
11011000	192.168.100.216/30 (2 hosts)
11011100	192.168.100.220/30 (2 hosts)
11100000	192.168.100.224/30 (2 hosts)
11100100	192.168.100.228/30 (2 hosts)
11101000	192.168.100.232/30 (2 hosts)
11101100	192.168.100.236/30 (2 hosts)
11110000	192.168.100.240/30 (2 hosts)
11110100	192.168.100.244/30 (2 hosts)
11111000	192.168.100.248/30 (2 hosts)
11111100	192.168.100.252/30 (2 hosts)

3．RTA、RTB 和 RTD 的 RIP 配置中有 **version 2** 语句。另外，RTA 和 RTB 的 RIP 配置中包含 **no auto-summary** 语句。RTA 和 RTB 上连接到子网 192.168.2.64/28 的接口配置了语

句 **ip rip send version 1 2** 和 **ip rip receive version 1 2**。RTD 上连接到子网 192.168.2.128/28 的接口配置了语句 **ip rip send version 1** 和 **ip rip receive version 1**。

4．下面的解答在 RTB 和 RTD 上使用一个密钥链 *CCIE* 以及一个密钥串 *exercise4*。假设两台路由器的接口都是 S0，RTB 和 RTD 的配置如下：

```
key chain CCIE
  key 1
    key-string exercise4
!
interface Serial0
  ip address 192.168.1.15X 255.255.255.252
  ip rip authentication mode md5
  ip rip authentication key-chain CCIE
```

5．这里的解答假设配置练习 4 中的验证密钥在 2004 年 10 月 31 日午夜启动生效。这里使用的第二个密钥串是 *exercise5a*，第三个是 *exercise5b*。

```
key chain CCIE
  key 1
    key-string exercise4
    accept-lifetime 00:00:00 Oct 31 2004 00:00:00 Nov 3 2004
    send-lifetime 00:00:00 Oct 31 2004 00:30:00 Nov 3 2004
  key 2
    key-string exercise5a
    accept-lifetime 00:00:00 Nov 3 2004 duration 36000
    send-lifetime 00:00:00 Nov 3 2004 duration 36000
  key 3
    key-string exercise5b
    accept-lifetime 10:00:00 Nov 3 2004 infinite
    send-lifetime 10:00:00 Nov 3 2004 infinite
!
interface Serial0
  ip address 192.168.1.15X 255.255.255.252
  ip rip authentication mode md5
  ip rip authentication key-chain CCIE
```

6．全局配置 **ipv6 unicast-routing**。在 LAN 端口上配置 IPv6 地址。在路由器 RTA 和 RTB 共享的 LAN 上配置地址 2001:DB8:0:2::/64。在所有的 IPv6 地址接口上配置 RIPng 路由选择进程。

第 7 章

1．EIGRP 配置如下：

RTA

```
router eigrp 5
network 172.16.0.0
network 172.18.0.0
```

RTB

```
router eigrp 5
network 172.16.0.0
```

RTC

```
router eigrp 5
network 172.16.0.0
network 172.17.0.0
```

2．下面的解答使用了一个密钥链 *CCIE* 以及密钥串 *exercise2a* 和 *exercise2b*。假设今天的日期是 2004 年 11 月 30 日，第一个密钥 8:30AM 开始启用，串口的配置为：

```
key chain CCIE
key 1
key-string exercise2a
accept-lifetime 08:30:00 Dec 2 2004 08:30:00 Jan 1 2005
send-lifetime 08:30:00 Dec 2 2004 08:30:00 Jan 1 2005
key 2
key-string exercise2b
accept-lifetime 08:30:00 Jan 1 2005 infinite
send-lifetime 08:30:00 Jan 1 2005 infinite
!
interface Serial0
ip address 172.16.3.19X 255.255.255.252
ip authentication key-chain eigrp 5 CCIE
ip authentication mode eigrp 5 md5
```

3．RTD 的 EIGRP 配置为：

```
router eigrp 5
network 172.16.0.0
network 172.17.0.0
no auto-summary
```

在 RTC 上自动汇总必须被关掉。

4．RTF 上的 EIGRP 配置为：

```
router eigrp 5
network 172.16.0.0
network 172.18.0.0
no auto-summary
```

RTA 在子网 172.18.10.96/27 上的接口上加入语句 **ip summary-address eigrp 5 172.18.10.192 255.255.255.224**。而且，注意到自动汇总在 RTA 上必须关掉，因为 RTF 上出现了网络 172.16.0.0。

5．RTA 能送给 RTF 172.16.3.128/25 的汇总地址，能送给 RTB 172.16.3.0/25 的汇总地址。所有其他的汇总都是自动执行。

第 8 章

1．OSPF 配置为：

RTA

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
```

RTB

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.5.0.0 0.0.255.255 area 5
area 5 virtual-link 10.100.100.9
```

RTC

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.10.0.0 0.0.255.255 area 10
network 10.30.0.0 0.0.255.255 area 30
```

RTD

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.20.0.0 0.0.255.255 area 20
```

RTE

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.15.0.0 0.0.255.255 area 15
```

RTF

```
router ospf 1
network 10.5.0.0 0.0.255.255 area 5
```

RTG

```
router ospf 1
network 10.10.1.58 0.0.0.0 area 10
```

RTH

```
router ospf 1
network 10.20.100.100 0.0.0.0 area 20
network 10.20.1.0 0.0.0.255 area 20
```

RTI

```
router ospf 1
network 10.5.0.0 0.0.255.255 area 5
network 10.35.0.0 0.0.255.255 area 35
area 5 virtual-link 10.100.100.2
```

RTJ

```
router ospf 1
network 10.15.0.0 0.0.255.255 area 15
```

RTK 到 RTN 有帧中继接口。到帧中继网络的 4 个接口都在同一个子网上，因此 OSPF 网络类型必须是广播或是点到多点的：

RTK

```
interface Serial0
encapsulation frame-relay
ip address 10.30.254.193 255.255.255.192
ip ospf network point-to-multipoint
!
router ospf 1
network 10.30.0.0 0.0.255.255 area 30
```

RTL

```
encapsulation frame-relay
ip address 10.30.254.194 255.255.255.192
ip ospf network point-to-multipoint
!
router ospf 1
network 10.30.0.0 0.0.255.255 area 30
```

RTM

```
encapsulation frame-relay
ip address 10.30.254.195 255.255.255.192
ip ospf network point-to-multipoint
!
router ospf 1
network 10.30.0.0 0.0.255.255 area 30
```

RTN

```
encapsulation frame-relay
ip address 10.30.254.196 255.255.255.192
ip ospf network point-to-multipoint
!
router ospf 1
network 10.30.0.0 0.0.255.255 area 30
```

2．ABR 配置为：

RTB

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.5.0.0 0.0.255.255 area 5
area 5 virtual-link 10.100.100.9
area 0 range 10.0.0.0 255.255.0.0
area 5 range 10.5.0.0 255.255.0.0
```

RTC

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.10.0.0 0.0.255.255 area 10
network 10.30.0.0 0.0.255.255 area 30
area 0 range 10.0.0.0 255.255.0.0
area 10 range 10.10.0.0 255.255.0.0
area 30 range 10.30.0.0 255.255.0.0
```

RTD

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.20.0.0 0.0.255.255 area 20
area 0 range 10.0.0.0 255.255.0.0
area 20 range 10.20.0.0 255.255.0.0
```

RTE

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.15.0.0 0.0.255.255 area 15
area 0 range 10.0.0.0 255.255.0.0
area 15 range 10.15.0.0 255.255.0.0
```

RTI

```
router ospf 1
network 10.5.0.0 0.0.255.255 area 5
network 10.35.0.0 0.0.255.255 area 35
area 5 virtual-link 10.100.100.2
area 0 range 10.0.0.0 255.255.0.0
area 5 range 10.5.0.0 255.255.0.0
area 35 range 10.35.0.0 255.255.0.0
```

3．配置为：

RTE

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.15.0.0 0.0.255.255 area 15
area 15 stub
```

RTJ

```
router ospf 1
network 10.15.0.0 0.0.255.255 area 15
area 15 stub
```

4．配置为：

RTC

```
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.10.0.0 0.0.255.255 area 10
network 10.30.0.0 0.0.255.255 area 30
area 0 range 10.0.0.0 255.255.0.0
area 10 range 10.10.0.0 255.255.0.0
area 30 stub no-summary
area 30 range 10.30.0.0 255.255.0.0
```

RTK

```
router ospf 1
network 10.30.0.0 0.0.255.255 area 30
area 30 stub
```

RTL

```
router ospf 1
network 10.30.0.0 0.0.255.255 area 30
area 30 stub
```

RTM

```
router ospf 1
network 10.30.0.0 0.0.255.255 area 30
area 30 stub
```

RTN

```
router ospf 1
network 10.30.0.0 0.0.255.255 area 30
area 30 stub
```

5．配置为：

RTD

```
router ospf 1
```

```
network 10.0.0.0 0.0.255.255 area 0
network 10.20.0.0 0.0.255.255 area 20
area 20 nssa
```

RTH

```
router ospf 1
network 10.20.100.100 0.0.0.0 area 20
area 20 nssa
```

6．点到点线路只有一个端点需要被配置为按需电路。在本题解答中，RTC 被选择：

```
interface Serial0
ip address 10.30.255.249 255.255.255.252
ip ospf demand-circuit
!
router ospf 1
network 10.0.0.0 0.0.255.255 area 0
network 10.10.0.0 0.0.255.255 area 10
network 10.30.0.0 0.0.255.255 area 30
area 0 range 10.0.0.0 255.255.0.0
area 10 range 10.10.0.0 255.255.0.0
area 30 range 10.30.0.0 255.255.0.0
```

第 9 章

1．对于已经配置到某个运行 OSPFv3 进程的接口上的辅助 IPv6 前缀地址，不需要再配置另外的 OSPFv3 命令。也就是说，如果在某个接口上配置了 OSPFv3，在这个接口上增加一个辅助 IPv6 地址，那么这个辅助地址就会被自动地加入到 OSPFv3 进程中。

2．是的，两台路由器如果配置了不同的前缀也能够形成邻接关系。在 Hello 数据包中不包含前缀信息，Hello 数据包的源地址是链路本地地址。配置在接口上的其他 IPv6 地址和这两台路由器是否形成邻接关系无关。

3．路由器的基本配置如下：

RtrA:

```
ipv6 unicast-routing
int e 0/0
 ipv6 address 2001:db8:0:4::3/64
 ipv6 ospf 1 area 1
int e 0/1
 ipv6 address 2001:db8:0:5::1/64
 ipv6 ospf 1 area 1
```

RtrB:

```
ipv6 unicast-routing
int e 0/0
 ipv6 address 2001:db8:0:4::2/64
 ipv6 ospf 1 area 1
int e 0/1
 ipv6 address 2001:db8:0:6::1/64
 ipv6 ospf 1 area 1
```

RtrC:

```
ipv6 unicast-routing
int e 0/0
 ipv6 address 2001:db8:0:4::1/64
 ipv6 ospf 1 area 1

int serial 0/0
 ipv6 address 2001:db8:0:8::1/64
 ipv6 ospf 1 area 0
```

RtrD:

```
ipv6 unicast-routing
int s 0/0
 ipv6 address 2001:db8:0:8::2/64
 ipv6 ospf 1 area 0
```

4．以下新的路由器配置将区域 1 变为一个完全末梢区域，并汇总到区域 0：

RtrA:

```
ipv6 unicast-routing
int e 0/0
 ipv6 address 2001:db8:0:5::3/64
 ipv6 ospf 1 area 1
ipv6 router ospf 1
 area 1 stub
```

RtrB:

```
ipv6 unicast-routing
int e 0/0
 ipv6 address 2001:db8:0:4::2/64
 ipv6 ospf 1 area 1
ipv6 router ospf 1
 area 1 stub
```

RtrC:

```
ipv6 unicast-routing
int e 0/0
 ipv6 address 2001:db8:0:4::1/64
 ipv6 ospf 1 area 1
int serial 0/0
 ipv6 address 2001:db8:0:8::1/64
 ipv6 ospf 1 area 0
ipv6 router ospf 1
 area 1 stub no-summary
 area 1 range 2001:db8:0:10::/61
```

第 10 章

1．本题答案是使用系统 ID 0000.1234.abcX，X 使得路由器在 IS-IS 域中惟一。系统 ID 有 6 个八位组字节，是 Cisco IOS 所要求的。最小的 NET 长度是 8 个八位组字节，其中一个是 SEL，所以区域地址是一个八位组字节的数字，对应于表 10.6 中的区域号。配置为：

RTA

```
interface Ethernet0
ip address 192.168.1.17 255.255.255.240
ip router isis
!
interface Ethernet1
ip address 192.168.1.50 255.255.255.240
ip router isis
!
router isis
net 00.0000.1234.abc1.00
is-type level-1
```

RTB

```
interface Ethernet0
ip address 192.168.1.33 255.255.255.240
ip router isis
!
interface Ethernet1
ip address 192.168.1.51 255.255.255.240
ip router isis
!
router isis
net 00.0000.1234.abc2.00
is-type level-1
```

RTC

```
interface Ethernet0
ip address 192.168.1.49 255.255.255.240
ip router isis
!
interface Serial0
ip address 192.168.1.133 255.255.255.252
ip router isis
!
router isis
net 00.0000.1234.abc3.00
```

RTD

```
interface Serial0
ip address 192.168.1.134 255.255.255.252
ip router isis
!
interface Serial1
ip address 192.168.1.137 255.255.255.252
ip router isis
!
router isis
net 02.0000.1234.abc4.00
is-type level-2-only
```

RTE

```
interface Serial0
ip address 192.168.1.142 255.255.255.252
```

```
ip router isis
!
interface Serial1
ip address 192.168.1.145 255.255.255.252
ip router isis
!
interface Serial2
ip address 192.168.1.138 255.255.255.252
ip router isis
!
router isis
net 02.0000.1234.abc5.00
is-type level-2-only
```

RTF

```
interface Serial0
ip address 192.168.1.141 255.255.255.252
ip router isis
!
interface Serial1
ip address 192.168.1.158 255.255.255.252
ip router isis
!
router isis
net 02.0000.1234.abc6.00
is-type level-2-only
```

RTG

```
interface Ethernet0
ip address 192.168.1.111 255.255.255.224
ip router isis
!
interface Serial0
ip address 192.168.1.157 255.255.255.252
ip router isis
!
router isis
net 01.0000.1234.abc7.00
```

RTH

```
interface Ethernet0
ip address 192.168.1.73 255.255.255.224
ip router isis
!
interface Ethernet1
ip address 192.168.1.97 255.255.255.224
ip router isis
!
router isis
net 01.0000.1234.abc8.00
is-type level-1
```

RTI

```
interface Ethernet0
ip address 192.168.1.225 255.255.255.248
```

```
ip router isis
!
interface Ethernet1
ip address 192.168.1.221 255.255.255.248
ip router isis
!
interface Serial0
ip address 192.168.1.249 255.255.255.252
ip router isis
!
interface Serial1
ip address 192.168.1.146 255.255.255.252
ip router isis
!
router isis
net 03.0000.1234.abc9.00
```

RTJ

```
interface Ethernet0
ip address 192.168.1.201 255.255.255.248
ip router isis
!
interface Ethernet1
ip address 192.168.1.217 255.255.255.248
ip router isis
!
router isis
net 03.0000.1234.abca.00
is-type level-1
```

RTK

```
interface Ethernet0
ip address 192.168.1.209 255.255.255.248
ip router isis
!
interface Serial0
ip address 192.168.1.250 255.255.255.252
ip router isis
!
router isis
net 03.0000.1234.abcb.00
is-type level-1
```

2．配置为：

RTA

```
ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.17 255.255.255.240
ip router isis
ipv6 address 2001:db8:0:10::1/64
ipv6 router isis
!
interface Ethernet1
ip address 192.168.1.50 255.255.255.240
```

```
ip router isis
ipv6 address 2001:db8:0:30::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc1.00
is-type level-1
metric-style wide
address-family ipv6
multi-topology
```

RTB

```
ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.33 255.255.255.240
ip router isis
ipv6 address 2001:db8:0:20::1/64
ipv6 router isis
!
interface Ethernet1
ip address 192.168.1.51 255.255.255.240
ip router isis
ipv6 address 2001:db8:0:30::3/64
ipv6 router isis

!
router isis
net 00.0000.1234.abc2.00
is-type level-1
metric-style wide
address-family ipv6
multi-topology
```

RTC

```
ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.49 255.255.255.240
ip router isis
ipv6 address 2001:db8:0:30::1/64
ipv6 router isis
!
interface Serial0
ip address 192.168.1.133 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:84::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc3.00
metric-style wide
address-family ipv6
multi-topology
```

RTD

```
ipv6 unicast-routing
```

```
interface Serial0
ip address 192.168.1.134 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:84::2/64
ipv6 router isis
!
interface Serial1
ip address 192.168.1.137 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:88::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc4.00
is-type level-2-only
metric-style wide
address-family ipv6
multi-topology
```

RTE

```
ipv6 unicast-routing
interface Serial0
ip address 192.168.1.142 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:8c::2/64
ipv6 router isis
!
interface Serial1
ip address 192.168.1.145 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:90::1/64
ipv6 router isis
!
interface Serial2
ip address 192.168.1.138 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:88::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc5.00
is-type level-2-only
metric-style wide
address-family ipv6
multi-topology
```

RTF

```
ipv6 unicast-routing
interface Serial0
ip address 192.168.1.141 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:8c::1/64
ipv6 router isis
!
```

```
interface Serial1
ip address 192.168.1.158 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:9c::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc6.00
is-type level-2-only
metric-style wide
address-family ipv6
multi-topology
```

RTG

```
ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.111 255.255.255.224
ip router isis
ip router clns
ipv6 address 2001:db8:0:60::f/64
ipv6 router isis
!
interface Serial0
ip address 192.168.1.157 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:9c::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc7.00
metric-style wide
address-family ipv6
multi-topology
```

RTH

```
ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.73 255.255.255.224
ip router isis
ipv6 address 2001:db8:0:40::9/64
ipv6 router isis
!
interface Ethernet1
ip address 192.168.1.97 255.255.255.224
ip router isis
ip router clns
ipv6 address 2001:db8:0:60::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc8.00
is-type level-1
metric-style wide
address-family ipv6
multi-topology
```

RTI

```
ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.225 255.255.255.248
ip router isis
ipv6 address 2001:db8:0:e0::1/64
ipv6 router isis
!
interface Ethernet1
ip address 192.168.1.221 255.255.255.248
ip router isis
ipv6 address 2001:db8:0:d8::5/64
ipv6 router isis
!
interface Serial0
ip address 192.168.1.249 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:f8::1/64
ipv6 router isis
!
interface Serial1
ip address 192.168.1.146 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:90::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc9.00
metric-style wide
address-family ipv6
multi-topology
```

RTJ

```
ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.201 255.255.255.248
ip router isis
ipv6 address 2001:db8:0:c8::1/64
ipv6 router isis
!
interface Ethernet1
ip address 192.168.1.217 255.255.255.248
ip router isis
ipv6 address 2001:db8:0:d8::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abca.00
is-type level-1
metric-style wide
address-family ipv6
multi-topology
```

RTK

```
ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.209 255.255.255.248
ip router isis
ipv6 address 2001:db8:0:d0::1/64
ipv6 router isis
!
interface Serial0
ip address 192.168.1.250 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:f8::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abcb.00
is-type level-1
metric-style wide
address-family ipv6
multi-topology
```

3．配置为：

RTD

```
Ipv6 unicast-routing
Key chain Fair
 Key 1
 Key-string Eiffel

interface Serial0
ip address 192.168.1.134 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:84::2/64
ipv6 router isis
!
interface Serial1
ip address 192.168.1.137 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:88::1/64
ipv6 router isis
authentication key-chain Fair level-2
authentication mode md5 level-2

!
router isis
net 00.0000.1234.abc4.00
is-type level-2-only
metric-style wide
address-family ipv6
multi-topology
```

RTE

```
Ipv6 unicast-routing
Key chain Fair
 Key 1
 Key-string Eiffel
```

```
Key chain Paris
Key 1
 Key-string Tower
interface Serial0
ip address 192.168.1.142 255.255.255.252
ip router isis
ipv6 address 2001:db7:0:8c::2/64
ipv6 router isis
authentication key-chain Paris level-2
authentication mode md5 level-2
!
interface Serial1
ip address 192.168.1.145 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:90::1/64
ipv6 router isis
!
interface Serial2
ip address 192.168.1.138 255.255.255.252
ip router isis ipv6 address 2001:db8:0:88::2/64
ipv6 router isis
authentication key-chain Fair level-2
authentication mode md5 level-2
!
router isis
net 00.0000.1234.abc5.00
is-type level-2-only
metric-style wide
address-family ipv6
multi-topology
```

RTF

```
Ipv6 unicast-routing
Key chain Paris
Key 1
 Key-string Tower
interface Serial0
ip address 192.168.1.141 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:8c::1/64
ipv6 router isis
authentication key-chain Paris level-2
authentication mode md5 level-2
!
interface Serial1
ip address 192.168.1.158 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:9c::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc6.00
is-type level-2-only
metric-style wide
```

```
address-family ipv6
multi-topology
```

4．配置为：

RTG

```
Ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.111 255.255.255.224
ip router isis
ipv6 address 2001:db8:0:96::f/64
ipv6 router isis
ip router clns
!
interface Serial0
ip address 192.168.1.157 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:9c::1/64
ipv6 router isis!
router isis
net 00.0000.1234.abc7.00
area-password Scotland
metric-style wide
address-family ipv6
multi-topology
```

RTH

```
Ipv6 unicast-routing
interface Ethernet0
ip address 192.168.1.73 255.255.255.224
ip router isis
ipv6 address 2001:db8:0:40::9/64
ipv6 router isis
!
interface Ethernet1
ip address 192.168.1.97 255.255.255.224
ip router isis
ip router clns
ipv6 address 2001:db8:0:60::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc8.00
is-type level-1
area-password Scotland
metric-style wide
address-family ipv6
multi-topology
```

5．配置为：

RTC

```
Ipv6 unicast-routing
Key chain Austria
Key 1
 Key-string Vienna
```

```
interface Ethernet0
ip address 192.168.1.49 255.255.255.240
ip router isis
ipv6 address 2001:db8:0:30::1/64
ipv6 router isis
!
interface Serial0
ip address 192.168.1.133 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:84::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc3.00

authentication key-chain Austria level-2
authentication mode text
metric-style wide
address-family ipv6
multi-topology
```

RTD

```
Ipv6 unicast-routing
Key chain Fair
 Key 1
 Key-string Eiffel

Key chain Austria
Key 1
 Key-string Vienna

interface Serial0
ip address 192.168.1.134 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:84::2/64
ipv6 router isis
!
interface Serial1
ip address 192.168.1.137 255.255.255.252
ip router isis
authentication key-chain Fair level-2
authentication mode md5 level-2
ipv6 address 2001:db8:0:88::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc4.00
is-type level-2-only
authentication key-chain Austria level-2
authentication mode text
metric-style wide
address-family ipv6
multi-topology
```

RTE

```
Ipv6 unicast-routing
Key chain Fair
 Key 1
 Key-string Eiffel

Key chain Paris
Key 1
 Key-string Tower

Key chain Austria
Key 1
 Key-string Vienna
interface Serial0
ip address 192.168.1.142 255.255.255.252
ip router isis
authentication key-chain Paris level-2
authentication mode md5 level-2
 ipv6 address 2001:db8:0:8c::2
 ipv6 routing isis
!
interface Serial1
ip address 192.168.1.145 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:90::1/64
ipv6 router isis
!
interface Serial2
ip address 192.168.1.138 255.255.255.252
ip router isis
authentication key-chain Fair level-2
authentication mode md5 level-2
ipv6 address 2001:db8:0:88::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc5.00
is-type level-2-only
authentication key-chain Austria level-2
authentication mode text
metric-style wide
address-family ipv6
multi-topology
```

RTF

```
Ipv6 unicast-routing
Key chain Paris
Key 1
 Key-string Tower

Key chain Austria
Key 1
 Key-string Vienna
```

```
interface Serial0
ip address 192.168.1.141 255.255.255.252
ip router isis
authentication key-chain Paris level-2
authentication mode md5 level-2
ipv6 address 2001:db8:0:8c::1/64
ipv6 router isis
!
interface Serial1
ip address 192.168.1.158 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:9c::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc6.00
is-type level-2-only
authentication key-chain Austria level-2
authentication mode text
metric-style wide
address-family ipv6
multi-topology
```

RTG

```
Ipv6 unicast-routing
Key chain Austria
Key 1
 Key-string Vienna
interface Ethernet0
ip address 192.168.1.111 255.255.255.224
ip router isis
ip router clns
ipv6 address 2001:db8:0:96::f/64
ipv6 router isis
!
interface Serial0
ip address 192.168.1.157 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:9c::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc7.00
area-password Scotland
authentication key-chain Austria level-2
authentication mode text
metric-style wide
address-family ipv6
multi-topology
```

RTI

```
Ipv6 unicast-routing
Key chain Austria
```

```
Key 1
 Key-string Vienna
interface Ethernet0
ip address 192.168.1.225 255.255.255.248
ip router isis
ipv6 address 2001:db8:0:e0::1/64
ipv6 router isis
!
interface Ethernet1
ip address 192.168.1.221 255.255.255.248
ip router isis
ipv6 address 2001:db8:0:d8::5/64
ipv6 router isis
!
interface Serial0
ip address 192.168.1.249 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:f8::1/64
ipv6 router isis
!
interface Serial1
ip address 192.168.1.146 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:90::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc9.00
authentication key-chain Austria level-2
authentication mode text
metric-style wide
address-family ipv6
multi-topology
```

6. 配置为：

RTC

```
Ipv6 unicast-routing
Key chain Austria
Key 1
 Key-string Vienna
interface Ethernet0
ip address 192.168.1.49 255.255.255.240
ip router isis
ipv6 address 2001:db8:0:30::1/64
ipv6 router isis
!
interface Serial0
ip address 192.168.1.133 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:84::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc3.00
```

```
summary-address 192.168.1.0 255.255.255.192
authentication key-chain Austria level-2
authentication mode text
metric-style wide
address-family ipv6
multi-topology
summary-prefix 2001:db8::/58
```

RTG

```
Ipv6 unicast-routing
Key chain Austria
Key 1
 Key-string Vienna
interface Ethernet0
ip address 192.168.1.111 255.255.255.224
ip router isis
ip router clns
ipv6 address 2001:db8:0:96::f/64
ipv6 router isis
!
interface Serial0
ip address 192.168.1.157 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:9c::1/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc7.00
area-password Scotland
authentication key-chain Austria level-2
authentication mode text
metric-style wide
summary-address 192.168.1.64 255.255.255.192
address-family ipv6
multi-topology
summary-prefix 2001:db8:0:40::/58
```

RTI

```
Ipv6 unicast-routing
Key chain Austria
Key 1
 Key-string Vienna
interface Ethernet0
ip address 192.168.1.225 255.255.255.248
ip router isis
ipv6 address 2001:db8:0:e0::1/64
ipv6 router isis
!
interface Ethernet1
ip address 192.168.1.221 255.255.255.248
ip router isis
ipv6 address 2001:db8:0:d8::5/64
ipv6 router isis
!
interface Serial0
```

```
ip address 192.168.1.249 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:f8::1/64
ipv6 router isis
!
interface Serial1
ip address 192.168.1.146 255.255.255.252
ip router isis
ipv6 address 2001:db8:0:90::2/64
ipv6 router isis
!
router isis
net 00.0000.1234.abc9.00
authentication key-chain Austria level-2
authentication mode text
metric-style wide
summary-address 192.168.1.192 255.255.255.192
address-family ipv6
multi-topology
summary-prefix 2001:db8:0:c0::/58
```

第 11 章

1．RIP 不宣告子网 172.16.1.144/28。为了纠正这个问题，对这个子网加一条带 27 位掩码的静态路由。因为此掩码涉及 RTB 的 E0 接口，所以该路由被自动分配至 RIP 中。

```
interface Ethernet0
ip address 172.16.1.146 255.255.255.240
ipv6 address 2001:db8:0:190::2/64
ipv6 ospf 1 area 0
!
interface Ethernet1
ip address 172.16.1.98 255.255.255.224
ipv6 address 2001:db8:0:160::2/64
ipv6 rip ripdomain enable
!
router ospf 1
redistribute rip metric 50 subnets
network 172.16.1.0 0.0.0.255 area 0
!
router rip
redistribute ospf 1 metric 2
passive-interface Ethernet0
network 172.16.0.0
!
ipv6 router rip ripdomain
redistribute ospf 1 metric 2
ipv6 router ospf
redistribute rip ripdomain metric 50

ip classless
ip route 172.16.1.128 255.255.255.224 Ethernet0
```

2．RTB 的配置为：

```
ipv6 router rip ripdomain
redistribute ospf 1 metric 2

ipv6 router ospf
redistribute rip ripdomain metric 50
summary-prefix 2001:db8:0:200::/48
```

3．RTB 的配置为：

```
interface Ethernet0
ip address 172.16.1.146 255.255.255.240
ip router isis
ipv6 address 2001:db8:0:190::2/64
ipv6 router isis
!
interface Ethernet1
ip address 172.16.1.98 255.255.255.224
ip summary-address eigrp 1 172.16.1.128 255.255.255.128
ipv6 address 2001:db8:0:160::2/64
ipv6 rip ripdomain enable
!
router eigrp 1
redistribute isis level-1 metric 10000 1000 255 1 1500
passive-interface Ethernet0
network 172.16.0.0
!
router isis
net 00.0000.1234.abc1.00
summary-address 172.16.2.0 255.255.255.0
redistribute eigrp 1 metric 20 metric-type external level-1
address-family ipv6
summary-prefix 2001:db8:0:200::/48
redistribute rip metric 20 metric-type external level-1
!
ipv6 router rip ripdomain
redistribute isis metric 5
```

第 13 章

1．RTA 的配置为：

```
router rip
redistribute eigrp 1 metric 3
passive-interface Ethernet0
passive-interface Ethernet1
network 172.16.0.0
distribute-list 1 in Ethernet3
!
router eigrp 1
redistribute rip metric 10000 1000 255 1 1500
passive-interface Ethernet2
```

```
passive-interface Ethernet3
network 172.16.0.0!
access-list 1 deny 172.16.12.0
access-list 1 permit any
```

2．RTA 的配置为：

```
router rip
redistribute eigrp 1 metric 3
passive-interface Ethernet0
passive-interface Ethernet1
network 172.16.0.0
distribute-list 2 out Ethernet2
!
router eigrp 1
redistribute rip metric 10000 1000 255 1 1500
passive-interface Ethernet2
passive-interface Ethernet3
network 172.16.0.0
!
access-list 2 deny 172.16.10.0
access-list 2 permit any
```

3．RTA 的配置为：

```
router rip
redistribute eigrp 1 metric 3
passive-interface Ethernet0
passive-interface Ethernet1
network 172.16.0.0
distribute-list 3 out eigrp 1
!
router eigrp 1
redistribute rip metric 10000 1000 255 1 1500
passive-interface Ethernet2
passive-interface Ethernet3
network 172.16.0.0
!
access-list 3 permit 172.16.2.0
access-list 3 permit 172.16.8.0
access-list 3 permit 172.16.9.0
```

4．RTA 的配置为：

```
router rip
redistribute eigrp 1 metric 3
passive-interface Ethernet0
passive-interface Ethernet1
network 172.16.0.0
!
router eigrp 1
redistribute rip metric 10000 1000 255 1 1500
passive-interface Ethernet2
passive-interface Ethernet3
network 172.16.0.0
distribute-list 4 out Ethernet0
!
access-list 4 permit 172.16.1.0
```

```
access-list 4 permit 172.16.2.0
access-list 4 permit 172.16.3.0
access-list 4 permit 172.16.7.0
access-list 4 permit 172.16.8.0
access-list 4 permit 172.16.9.0
```

5．在 RIPng 进程下配置的分配列表为：

```
ipv6 router rip ripname
distribute-list prefix-list listname out ethernet2
!
ipv6 prefix-list listname seq 5 deny 2001:DB8:0:A::/63 le 64
ipv6 prefix-list listname seq 10 permit ::/0 le 128
```

6．EIGRP 为外部路由分配高距离值 170，给内部路由分配低距离值 90。如果 EIGRP 域内的任何目标地址从 IS-IS 被重新分配至 EIGRP，将会被忽略，除非内部路由失效。所以，不要求手动操作 EIGRP 距离。RTC 和 RTD 的 IS-IS 配置中的 **distance** 语句为：

RTC

```
distance 115
distance 170 192.168.10.254 0.0.0.0 1
!
access-list 1 permit any
```

RTD

```
distance 115
distance 170 192.168.10.249 0.0.0.0 1
distance 170 192.168.10.241 0.0.0.0 1
!
access-list 1 permit any
```

7．RTD 中 IS-IS 配置的 **distance** 语句是：

```
distance 115
distance 255 192.168.10.241 0.0.0.0 1
!
access-list 1 permit any
```

8．RTC 中 EIGRP 配置的 **distance** 语句是：

```
distance eigrp 90 90
```

第 14 章

1．RTA 的配置为：

```
interface Serial0
ip address 172.16.14.6 255.255.255.252
ip policy route-map Exercise1
!
interface Serial1
ip address 172.16.14.10 255.255.255.252
ip policy route-map Exercise1
!
access-list 1 permit 172.16.1.0 0.0.0.127
access-list 2 permit 172.16.1.128 0.0.0.127
```

```
!
route-map Exercise1 permit 10
match ip address 1
set ip next-hop 172.16.14.17
!
route-map Exercise1 permit 20
match ip address 2
set ip next-hop 172.16.14.13
```

2．RTA 的配置为：

```
interface Serial0
ip address 172.16.14.6 255.255.255.252
ip policy route-map Exercise2A
!
interface Serial1
ip address 172.16.14.10 255.255.255.252
ip policy route-map Exercise2B
!
access-list 1 permit 172.16.1.64 0.0.0.63
!
route-map Exercise2 permit 10
match ip address 1
set ip next-hop 172.16.14.13
!
route-map Exercise2B permit 10
match ip address 1
set ip next-hop 172.16.14.17
```

3．RTA 的配置为：

```
interface Serial2
ip address 172.16.14.18 255.255.255.252
ip access-group 101 in
ip policy route-map Exercise3
!
interface Serial3
ip address 172.16.14.14 255.255.255.252
ip access-group 101 in
ip policy route-map Exercise3
!
access-list 101 permit udp any 172.168.1.0 0.0.0.255
access-list 101 permit tcp any eq smtp 172.16.1.0 0.0.0.255
access-list 102 permit tcp any eq smtp 172.16.1.0 0.0.0.255
access-list 103 permit udp any 172.168.1.0 0.0.0.255
!
route-map Exercise3 permit 10
match ip address 102
set ip next-hop 172.16.14.9
!
route-map Exercise3 permit 20
match ip address 103
set ip next-hop 172.16.14.5
```

4．OSPF 配置为：

```
router ospf 1
redistribute eigrp 1 route-map Exercise4
```

```
network 192.168.1.0 0.0.0.255 area 16
!
access-list 1 permit 10.201.100.0

!
route-map Exercise4 deny 10
match ip address 1
!
route-map Exercise4 permit 20
match route-type internal
set metric 10
set metric-type type-1
!
route-map Exercise4 permit 30
match route-type external
set metric 50
set metric-type type-2
```

5. EIGRP 配置为：

```
router eigrp 1
redistribute ospf 1 route-map Exercise5
network 192.168.100.0
!
access-list 1 permit 192.168.1.0
access-list 1 permit 192.168.2.0
access-list 1 permit 192.168.3.0
!
route-map Exercise5 permit 10
match ip address 1
match route-type internal
set metric 10000 100 255 1 1500
!
route-map Exercise5 permit 30
match ip address 1
match route-type external
set metric 10000 10000 255 1 1500
```

附录 F

故障诊断练习答案

第 1 章

1．子网：10.14.64.0
主机地址：10.14.64.1～10.14.95.254
广播地址：10.14.95.255
子网：172.25.0.224
主机地址：172.25.0.225～175.25.0.254
广播地址：172.25.0.255
子网：172.25.16.0
主机地址：172.25.16.1～172.25.16.126
广播地址：172.25.16.127

2．192.168.13.175/28 是子网 192.168.13.160/28 的广播地址。

第 3 章

1．从 Piglet 到子网 192.168.1.64/27 不再可达，从 Piglet 到子网 10.4.6.0/24 和 10.4.7.0/24 也不再可达。

2．错误发生在：

- RTA，第 2 个表项，应该是 **ip route 172.20.80.0 255.255.240.0 172.20.20.1**。
- RTB，第 3 个表项，应该是 **ip route 172.20.208.0 255.255.240.0 172.20.16.50**。
- RTC，第 2 个表项，应该是 **ip route 172.20.208.0 255.255.240.0 172.20.16.50** 以及第 5 个表项，应该是 **ip route 172.20.80.0 255.255.240.0 172.20.20.1**。

3．错误是：

- RTC：到 10.5.8.0/24 的路由指向了错误的下一跳地址。
- RTC：到 10.1.1.0/24 的路由应该是 10.5.1.0/24（网络中没有 10.1.1.0/24）。
- RTC：到 10.5.4.0/24 没有路由。
- RTD：到 10.4.5.0/24 的路由应该是 10.5.4.0/24。

第 5 章

1．新的访问列表给每一条路由（除了 10.33.32.0）添加了两跳。

2．RTB 根据其错误配置的掩码解释 172.16.0.0 的所有子网。结果是，正如其路由表中的 4 个表项所示：

- 表项 1：正确。因为 172.16.24.0 掩码既可以是 22 位也可以是 23 位。
- 表项 2：RTC 宣告子网 172.16.26.0。因为该地址的第 23 位是 1 且 RTB 使用 22 位的掩码，从 RTB 的角度来看这个 1 出现在主机地址部分。所以，RTB 把 172.16.26.0 的宣告看作是一条主机路由，并且在路由表里将其掩码设为 32 位。
- 表项 3：RTB 将其接口地址 172.16.22.5 解释为子网 172.16.20.0/22 的一部分，而不是 172.16.22.0/23 的一部分。当 RTB 收到 RTA 关于子网 172.16.20.0/23 的宣告时，由于 RTB 认为自己与该子网直接相连，于是忽略了该通告。注意， RTA 和 RTC 的路由表里没有子网 172.16.22.0，这是由于同一个原因：RTB 宣告它为 172.16.20.0。
- RTB 将其接口地址 172.16.18.4 解释为子网 172.16.16.0/22 的一个成员，而不是 172.16.18.0/23。

3．答案在示例 5-30 中 RTC 的路由表里。注意到 172.16.26.0/23 的路由在 2 分 42 秒内没有更新。RTC 的无效计时器在它听到来自 RTD 的一条新的更新之前就超时了，于是它宣布到 172.16.26.0/23 的路由无效。因为没有从 RTA 或 RTB 来的路由被宣布无效，所以问题出在 RTD 的 update 计时器上——更新周期太长。当 RTD 最终送出一条更新，它再次出现在 RTC 的路由表里，并且保留到 RTC 的无效计时器再次超时。

第 6 章

1．RTA 和 RTB 只发送和接收 RIPv2 消息。RTC 接受 RIPv1 和 RIPv2 消息，但只发送 RIPv1 消息。结果是，RTA 和 RTB 的路由表里没有子网 192.168.13.75/27。虽然 RTC 接收来自 RTA 和 RTB 的更新，但它的路由表里没有子网 192.168.13.90/28 和 192.168.13.86/29，因为 RTC 把它们解释成 192.168.13.64/27，这是与 RTC 的 E0 接口直接相连的。

2．是的。RTA 和 RTB 的路由表中将会加入子网 192.168.13.64/27，因为它们现在能接收来自 RTC 的 RIPv1 更新。

第 7 章

1．在到子网 A 的路径上 RTG 是 RTF 的后继。

2．RTC 到子网 A 的可行距离是 309 760。

3．RTG 到子网 A 的可行距离是 2 214 319。

4．RTG 的拓扑表显示，RTD 和 RTE 是它到子网 A 的可行后继。

5．RTA 到子网 B 的可行距离是 2 198 016。

第 8 章

1．在一个邻居接口上，IP 地址或者其掩码配置出错。

2．一台路由器被配置为末梢区域路由器，另一台不是。

3．接收端路由器被配置为 MD5（类型 2）验证，其邻居没有配置认证（类型 0）。

4．两台路由器上配置的密码不一致。

5．邻居接口没有被配置相同的区域 ID。

6．RTA 的 **network** 语句顺序错误。第一个语句匹配 IP 地址 192.168.50.242，并把它放入区域 192.168.50.0。

7．链路 ID 10.8.5.1 最可疑，因为它的序列号比其他链路的序列号要高很多。

第 10 章

1．虽然 IS-IS 形成了邻接关系，但是它的类型是 IS 而不是 L1、L2 或者 L1/L2，这表明存在问题。IP 地址并不在同一个子网里。

2．路由器只发送 L1 Hello，表明它是一台 L1 路由器。它只从 0000.3090.c7df 接收 L2 Hello，表明它是一台 L2 路由器。

第 11 章

1．水平分割将阻止 192.168.10.0/24 从 IGRP 域到 RIP 域的宣告。

2．不。因为不是所有 10.0.0.0 的子网都在 RIP 端与 Mantle 直接相连。

3．Robinson 的 EIGRP1 配置中的 **network** 语句匹配接口 192.168.3.32，即使没有 EIGRP 的 Hello 在这个接口被转发出去。因此，这个子网在 EIGRP1 域内被声明为内部的。

4．EIGRP 自动加入这条汇总路由，因为子网 192.168.3.0 是从 OSPF 重新分配到 EIGRP 中的。

5．区域 1 中的第 1 层路由器将不知道汇总路由。RIP 被重新分配到第 2 层中（这是缺省

情况进行分配的层）。命令 **summary-prefix** 正在汇总被通告到第 1 层中的路由。

第 13 章

1．访问列表第一行中的反码错误。由于此反码，所有路由将被匹配。这一行应该是 **access-list 1 deny 0.0.0.0 0.0.0.0**。

2．所有没有被访问列表匹配的路由将赋予距离值 255（不可到达）。如果一个优先路由失败，Grimwig 将不使用备用路径。

3．OSPF 通过它从 LSA 学来的信息计算路由。路由过滤器不影响 LSA，所以过滤器仅仅影响配置它的路由器。

4．这两台路由过滤器涉及错误的接口。**distribute-list 1** 应该对应 E1，**distribute-list 2** 应该对应 E0。

第 14 章

两个错误都是顺序错误。首先，访问列表 101 中的 *telnet* 关键字与目标端口关联；正确的应该是源端口。其次，路由映射语句顺序错误。从 192.168.10.5 来的 telnet 数据包匹配第一个语句，并且被转发至 192.168.16.254。