



Red Hat Enterprise Linux 7 联网指南

为 Red Hat Enterprise Linux 7 配置和管理联网

Stephen Wadeley

Christian Huffman

Red Hat Enterprise Linux 7 联网指南

为 Red Hat Enterprise Linux 7 配置和管理联网

Stephen Wadeley
Red Hat 出版部
swadeley@redhat.com

Christian Huffman
Red Hat 出版部
chuffman@redhat.com

法律通告

Copyright © 2010–2015 Red Hat, Inc.

This document is licensed by Red Hat under the [Creative Commons Attribution-ShareAlike 3.0 Unported License](https://creativecommons.org/licenses/by-sa/3.0/). If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

摘要

《Red Hat Enterprise Linux 7 联网指南》中记录了有关在 Red Hat Enterprise Linux 7 中配置和管理网络接口、网络及网络服务的信息。阅读对象为对 Linux 和联网有基本了解的系统管理员。本手册是在《Red Hat Enterprise Linux 6 部署指南》基础上编写。有关联网的章节时取自《部署指南》，并构成本手册的框架。

目录

部分 I. IP 联网	4
第 1 章 Red Hat Enterprise Linux 联网简介	5
1.1. 本手册结构	5
1.2. IP 网络 vs 非 IP 网络	5
1.3. NetworkManager 简介	5
1.4. 安装 NetworkManager	6
1.5. 使用文本用户界面 (nmtui) 进行网络配置	7
1.6. 使用 NetworkManager 的 CLI (nmcli) 进行网络配置	7
1.7. 使用命令行界面 (CLI) 进行网络配置	7
1.8. NetworkManager 及网络脚本	8
1.9. 使用 sysconfig 文件进行网络配置	9
1.10. 其他资料	10
第 2 章 配置 IP 联网	11
2.1. 静态和动态接口设置	11
2.2. 使用文本用户界面 nmtui	11
2.3. 使用 NetworkManager 命令行工具 nmcli	12
2.4. 使用命令行界面 (CLI)	22
2.5. 在 GNOME 图形用户界面中使用 NetworkManager	27
2.6. 其他资料	45
第 3 章 配置主机名	47
3.1. 了解主机名	47
3.2. 使用文本用户界面 nmtui 配置主机名	47
3.3. 使用 hostnamectl 配置主机名	48
3.4. 使用 nmcli 配置主机名	49
3.5. 其他资料	49
第 4 章 配置网络绑定	51
4.1. 了解主接口及从属接口的默认行为	51
4.2. 使用文本用户界面 nmtui 配置绑定	51
4.3. 使用 NetworkManager 命令行工具 nmcli	56
4.4. 使用命令行接口 (CLI)	57
4.5. 使用频道绑定	60
4.6. 使用 GUI 创建绑定连接	66
4.7. 其他资料	71
第 5 章 配置网络成组 (Network Teaming)	72
5.1. 了解网络成组	72
5.2. 了解主接口及从属接口的默认行为	72
5.3. 网络成组和绑定对比	73
5.4. 了解网络成组守护进程及“运行程序”	73
5.5. 安装网络成组守护进程	74
5.6. 将绑定转换为成组	74
5.7. 选择作为网络成组端口使用的接口	75
5.8. 选择网络成组配置方法	75
5.9. 使用文本用户界面 nmtui 配置网络成组	76
5.10. 使用命令行配置网络成组	80
5.11. 使用 teamnl 控制 teamd	87
5.12. 配置 teamd 运行程序	88
5.13. 使用 GUI 创建网络成组	95
5.14. 其他资料	99

第 6 章 配置网络桥接	100
6.1. 使用文本用户界面 nmtui 配置桥接	100
6.2. 使用 NetworkManager 命令行工具 nmcli	103
6.3. 使用命令行界面 (CLI)	105
6.4. 使用 GUI 配置网络桥接	108
6.5. 其他资源	113
第 7 章 配置 802.1Q VLAN 标记	114
7.1. 选择 VLAN 接口配置方法	114
7.2. 使用文本用户界面 nmtui 配置 802.1Q VLAN 标记	114
7.3. 使用命令行工具 nmcli 配置 802.1Q VLAN 标记	116
7.4. 使用命令行配置 802.1Q VLAN 标记	118
7.5. 使用 GUI 配置 802.1Q VLAN 标记	120
7.6. 其他资料	122
第 8 章 一致网络设备命名	123
8.1. 命名方案层级结构	123
8.2. 了解设备重命名过程	123
8.3. 了解可预期网络接口设备名称	124
8.4. 在 System z 中用于 Linux 系统的网络设备命名规则	124
8.5. VLAN 接口命名方案	125
8.6. 使用 biosdvnname 保持网络设备命名一致	125
8.7. 管理员备注	126
8.8. 控制网络设备名称选择	126
8.9. 禁用一致网络设备命名	126
8.10. 网络设备命名故障排除	127
8.11. 其他资料	128
部分 II. InfiniBand 和 RDMA 联网	129
第 9 章 配置 InfiniBand 和 RDMA 网络	130
9.1. 了解 InfiniBand 和 RDMA 技术	130
9.2. 与 InfiniBand 及 RDMA 相关的软件包	131
9.3. 配置基础 RDMA 子系统	131
9.4. 配置子网管理器	133
9.5. 测试早期 InfiniBand RDMA 操作	135
9.6. 配置 IPoIB	138
9.7. 使用文本用户界面 nmtui 配置 InfiniBand	139
9.8. 使用命令行工具 nmcli 配置 IPoIB	141
9.9. 使用命令行配置 IPoIB	142
9.10. 配置 IPoIB 后测试 RDMA 网络	143
9.11. 使用 GUI 配置 IPoIB	143
9.12. 其他资料	145
部分 III. 服务器	146
第 10 章 DHCP 服务器	147
10.1. 为什么使用 DHCP	147
10.2. 配置 DHCP 服务器	147
10.3. DHCP 中继代理程序	153
10.4. 配置 DHCP 服务器	154
10.5. 用于 IPv6 的 DHCP (DHCPv6)	157
10.6. 其他资料	157
第 11 章 DNS 服务器	158

第 11 章 DNS 服务器	150
11.1. DNS 简介	158
11.2. BIND	159
附录 A. 修订历史	182
索引	182

部分 I. IP 联网

这部分论述了如何在 Red Hat Enterprise Linux 中配置网络。

第 1 章 Red Hat Enterprise Linux 联网简介

1.1. 本手册结构

本手册中的所有新资料都采用可与介绍材料有明显区别的方法编写及组合，比如根据配置任务解释概念及使用示例。Red Hat 工程内容服务希望您可以根据需要迅速找到配置说明，同时仍提供一些相关解释和概念性材料，以帮助您理解并决定您需要的正确任务。已对《Red Hat Enterprise Linux 6 部署指南》中重复使用的部分进行检查，并根据需要进行更改，以满足将概念与任务分离的方式。

这些资料是根据目标而非方法分组。将使用不同方法完成某项具体任务的说明分为一组。这样做是为更方便您找到如何完成具体任务或目标的信息，同时可让您了解可用的不同方法。

每一章中会按照以下顺序排列配置方法：

- ✱ 文本用户界面工具 **nmtui**,
- ✱ **NetworkManager** 的命令行工具 **nmcli**,
- ✱ 其他命令行方法及配置文件使用,
- ✱ 图形用户界面 (GUI)，比如 **nm-connection-editor** 或 **control-network** 指向 **NetworkManager**。

可使用这个命令行工具运行命令，即 **命令行界面 (CLI)**，但命令行也可以启动编辑器以编写或编辑配置文件。因此，**ip** 命令及配置文件（比如 **ifcfg** 文件）的用法会在放在一起。

1.2. IP 网络 vs 非 IP 网络

Most modern networks fall into one of two very broad categories: IP based networks. These are all networks that communicate via Internet Protocol addresses, which is the standard for the Internet and for most internal networks today. This generally includes Ethernet, Cable Modems, DSL Modems, dial up modems, Wi-Fi, VPN connections and more.

Then there are non-IP based networks. These are usually very specific niche networks, but one in particular has grown in usage enough to warrant mention here and that is InfiniBand. Because InfiniBand is not an IP network, many features and configurations normally used on IP networks are not applicable to InfiniBand. [第 9 章 配置 InfiniBand 和 RDMA 网络](#) in this guide covers the specific requirements of configuring and administering an InfiniBand network and also the broader class of RDMA capable devices.

1.3. NetworkManager 简介

在 Red Hat Enterprise Linux 7 中，**NetworkManager** 提供的默认联网服务是一个动态网络控制和配置守护进程，它尝试在其可用时保持网络设备和连接处于活动状态。仍支持传统 **ifcfg** 类型配置文件。详情请查看 [第 1.8 节 “NetworkManager 及网络脚本”](#)。

表 1.1. 联网工具及应用程序概述

应用程序或工具	描述
NetworkManager	默认联网守护进程
nmtui	NetworkManager 的使用光标的简单文本用户界面 (TUI)
nmcli	允许用户及脚本与 NetworkManager 互动的命令行工具
control-center	GNOME Shell 提供的图形用户界面工具
nm-connection-editor	这是一个 GTK+ 3 应用程序，可用于尚未由 control-center 处理的某些任务的。

NetworkManager 可用于以下连接类型：以太网、VLAN、网桥、绑定、成组、Wi-Fi、移动宽带（比如移动网络 3G）及 IP-over-InfiniBand。在这些连接类型中，**NetworkManager** 可配置网络别名、IP 地址、静态路由器、DNS 信息及 VPN 连接以及很多具体连接参数。最后，**NetworkManager** 通过 D-bus 提供 API，D-Bus 允许应用程序查询并控制网络配置及状态。

1.4. 安装 NetworkManager

默认在 Red Hat Enterprise Linux 中安装 **NetworkManager**。必要时可作为 **root** 用户运行以下命令：

```
~]# yum install NetworkManager
```

有关用户授权及获取授权的详情，请查看 [《Red Hat Enterprise Linux 7 系统管理员指南》](#)。

1.4.1. NetworkManager 守护进程

默认情况下，是将使用 root 授权运行的 **NetworkManager** 守护进程配置为在引导时启动。可运行这个命令确定 **NetworkManager** 守护进程是否正在运行：

```
~]$ systemctl status NetworkManager
NetworkManager.service - Network Manager
   Loaded: loaded (/lib/systemd/system/NetworkManager.service; enabled)
   Active: active (running) since Fri, 08 Mar 2013 12:50:04 +0100; 3 days
   ago
```

如果 **NetworkManager** 服务未处于运行状态，则 **systemctl status** 命令会报告 **NetworkManager** 处于 **Active: inactive (dead)** 状态。请作为 root 用户运行下面的命令在当前会话中启动该服务：

```
~]# systemctl start NetworkManager
```

运行 **systemctl enable** 命令确定每次系统引导时都启动 **NetworkManager**：

```
~]# systemctl enable NetworkManager
```

有关启动、停止及管理服务的详情，请查看 [《Red Hat Enterprise Linux 7 系统管理员指南》](#)。

1.4.2. 与 NetworkManager 互动

用户不与 **NetworkManager** 系统服务直接互动，而是通过图形及命令行用户界面工具执行网络配置任务。Red Hat Enterprise Linux 7 中有以下工具可用：

1. **NetworkManager** 的简单基于光标的文本用户界面（TUI）**nmtui**。
2. 提供命令行工具 **nmcli**，允许用户及脚本与 **NetworkManager** 互动。注：**nmcli** 可用于缺少 GUI 的系统（比如服务器）以控制 **NetworkManager** 的各个方面。它与 GUI 工具处于同等地位。
3. GNOME Shell 还在其通知区域提供网络图标，代表 **NetworkManager** 报告的网络连接状态。该图标有多种状态，分别代表目前使用的连接状态。
4. GNOME Shell 提供的图形用户界面 **control-center** 还适用于桌面用户。它整合了 **Network** 设置工具。要启动该工具，请按 **Super** 键进入活动概述页面，输入 **control network**，然后按 **Enter** 键。**Super** 键以不同的形式在 gui 中出现，具体要看键盘及其他硬件配置，但通常是 Window 键或 Command 键，一般位于空格键的左侧。

- 图形用户界面工具 **nm-connection-editor** 可用于某些 **control-center** 尚未处理的任务。要启动该工具，请按 **Super** 键进入活动概述页面，输入 **network connections** 或 **nm-connection-editor**，并按 **Enter** 键。



图 1.1. 网络连接图标状态

1.5. 使用文本用户界面（nmtui）进行网络配置

NetworkManager 文本用户界面（TUI）工具 **nmtui** 可提供一个文本界面配置由 **NetworkManager** 控制的网络。该工具包含在 *NetworkManager-tui* 子软件包中。写入时，不会默认随 **NetworkManager** 安装该子软件包。要安装 *NetworkManager-tui*，请作为 **root** 运行以下命令：

```
~]# yum install NetworkManager-tui
```

如有必要，请根据 [第 1.4.1 节 “NetworkManager 守护进程”](#) 中的论述确定 **NetworkManager** 的运行方式。

要启动 **nmtui**，请按如下方式运行命令：

```
~]$ nmtui
```

此时会出现文本用户界面。要在该界面中导航，请使用箭头键，或按 **Tab** 在选项间前进，按 **press Shift+Tab** 后退。按 **Enter** 选择某个选项。**Space** 键切换选择库状态。

有以下命令可用：

```
» nmtui edit connection-name
```

如果未提供连接名称，则会出现选择菜单。如果提供连接名称，并正确验证，则会出现相关的 **编辑连接** 页面。

```
» nmtui connect connection-name
```

如果未提供连接名称，则会出现选择菜单。如果提供连接名称并正确验证，则会激活相关连接。如命令无效，则会输出用法信息。

编写时，**nmtui** 不支持所有连接类型。特别是无法编辑使用 WPA Enterprise 的 VPN、Wi-Fi 连接，或无法编辑使用 **802.1X** 的以太网连接。

1.6. 使用 NetworkManager 的 CLI（nmcli）进行网络配置

NetworkManager 命令行工具 **nmcli** 提供使用命令行配置由 **NetworkManager** 所控制联网的方法。默认会与 **NetworkManager** 一同安装。如有必要，可根据 [第 1.4.1 节 “NetworkManager 守护进程”](#) 中的说明验证 **NetworkManager** 的运行状态。

解释其他命令行方法及图形用户界面前，会尽可能为每个任务包含一个使用 **nmcli** 工具的示例。请在 [第 2.3 节 “使用 NetworkManager 命令行工具 nmcli”](#) 查看 **nmcli** 简介。

1.7. 使用命令行界面（CLI）进行网络配置

ip 程序的命令有时是 `iproute2` 中论述的 `upstream` 软件包名称，请参考 `man ip(8)` 页面。该软件包在 Red Hat Enterprise Linux 7 中的名称是 `iproute`。必要时，可按照以下方法检查其版本后确认安装的 **ip** 程序：

```
~]$ ip -V
ip utility, iproute2-ss130716
```

可使用 **ip** 命令添加和删除连接到接口的地址和路由，同时使用 **NetworkManager** 在 **nmcli**、**nmtui**、**control-center** 及 D-Bus API 中保留并识别它们。

注：**ip** 程序可代替 **ifconfig** 程序，因为 `net-tools` 软件包（提供 **ifconfig**）不支持 InfiniBand 地址。命令 **ip help** 可提供用法信息。可为 OBJECTS 提供具体帮助信息，例如：**ip link help** 和 **ip addr help**



注意

重启后，该命令行中给出的 **ip** 命令不会保留。如果需要保留，则请使用配置文件 **ifcfg**，或者在脚本中添加命令。

使用命令行及配置文件完成每个人物的示例均在 **nmtui** 和 **nmcli** 示例之后，但在 **NetworkManager** 图形用户界面（即 **control-center** 和 **nm-connection-editor**）使用说明之前。

1.8. NetworkManager 及网络脚本

在之前的 Red Hat Enterprise Linux 发行本中，默认使用 *网络脚本* 配置联网。术语 *网络脚本* 通常是指 `/etc/init.d/network` 及所有它调用的已安装脚本。用户提供的文件通常被视为配置文件，但也可以将其解读为对脚本的修改。

虽然 **NetworkManager** 提供默认联网服务，但 Red Hat 开发人员仍致力于确保脚本和 **NetworkManager** 之间可相互协作。习惯于使用脚本的管理员仍可继续使用脚本。我们的预期是两个系统可同时存在，并可很好地合作。大多数之前发行本中的用户 `shell` 脚本仍可正常工作。但 Red Hat 建议您在使用前对其进行测试。

运行网络脚本

只使用 **systemctl** 程序运行脚本则会清除所有现有环境变量，并确保一个干净的执行模式。该命令的格式如下：

```
systemctl start|stop|restart|status network
```

请勿直接调用 `/etc/init.d/servicename start|stop|restart|status` 运行任何服务。

注：在 Red Hat Enterprise Linux 7 中，首先启动 **NetworkManager**，此时 `/etc/init.d/network` 会使用 **NetworkManager** 检查，以避免破坏 **NetworkManager** 的连接。**NetworkManager** 主要在使用 `sysconfig` 配置文件的主要应用程序中使用，而 `/etc/init.d/network` 主要是作为备用程序在此要程序中使用。

`/etc/init.d/network` 脚本不是事件驱动，它可采用以下方式之一运行：

1. 手动（运行 `systemctl start|stop|restart network` 命令之一），
2. 如果启用网络服务，则会在引导和关机时运行（`systemctl enable network` 命令的结果）。

这是一个手动过程，不会与任何引导后发生的事件互动。用户还可以手动调用 **ifup** 和 **ifdown** 脚本。

自定义命令及网络脚本

只有在那些设备由 `/etc/init.d/network` 服务控制时方可执行脚本 `/sbin/ifup-local` 中的自定义命令 (`ifdown-pre-local` 和 `ifdown-local`)。如果修改 `initscripts` 本身 (比如 `/etc/sysconfig/network-scripts/ifup-eth`)，那么 `initscripts` 软件包更新会覆盖那些修改。因此建议避免直接修改 `initscripts`，而是使用 `/sbin/if*local` 脚本，以便在更新软件包后仍可保留您所做的更改。`Initscripts` 只检查是否有相关的 `/sbin/if*local`，并在该文件存在时运行该文件。`Initscripts` 不会在 `/sbin/if*local` 脚本中添加任何内容，`initscripts` RPM (或任意软件包) 也不会拥有或修改那些文件。

网络连接启动或断开时，可采用某些方法执行自定义任务，旧的网络脚本及 **NetworkManager** 均有此功能。启用 **NetworkManager** 后，`ifup` 和 `ifdown` 脚本会询问 **NetworkManager**，是否由 **NetworkManager** 管理在 `ifcfg` 文件的“`DEVICE=`”行中发现的接口。如果是 **NetworkManager** 管理该设备，且该设备未处于连接状态，则 `ifup` 会要求 **NetworkManager** 启动该连接。

- ✦ 如果该设备由 **NetworkManager** 管理，且它已经处于连接状态，则不需要任何操作。
- ✦ 如果该设备不是由 **NetworkManager** 管理，那么该脚本会使用旧的非 **NetworkManager** 机制 (即出现 **NetworkManager** 之前使用的方法) 启动该连接。

如果要调用 `ifdown`，且该设备由 **NetworkManager** 管理，那么 `ifdown` 会要求 **NetworkManager** 终止该连接。

该脚本会动态检查 **NetworkManager**，因此如果未运行 **NetworkManager**，则该脚本会故障转移至旧的、早于 **NetworkManager** 的基于脚本的机制。

1.9. 使用 `sysconfig` 文件进行网络配置

配置文件和脚本保存在 `/etc/sysconfig/` 目录中。大多数网络配置信息都保存在这里，VPN、移动宽带及 PPPoE 配置除外，这些配置保存在 `/etc/NetworkManager/` 子目录中。例如：接口的具体信息是保存在 `/etc/sysconfig/network-scripts/` 目录下的 `ifcfg` 文件中。

全局设置使用 `/etc/sysconfig/network` 文件。有关 VPN、移动宽带及 PPPoE 连接的信息保存在 `/etc/NetworkManager/system-connections/` 中。

在 Red Hat Enterprise Linux 7 中编辑 `ifcfg` 文件时，**NetworkManager** 不会自动意识到更改，需为其提供通知。如果使用以下工具之一更新 **NetworkManager** 配置文件，则 **NetworkManager** 会在使用该配置文件重新连接后方可实施那些更改。例如：如果使用编辑器更改配置文件，则必须让 **NetworkManager** 重新读取该配置文件。方法是作为 **root** 运行以下命令：

```
~]# nmcli connection reload
```

上述命令会读取所有连接配置文件。另外也可以运行下面的命令，只重新载入那些有变化的文件 `ifcfg-ifname`：

```
~]# nmcli con load /etc/sysconfig/network-scripts/ifcfg-ifname
```

该命令接受多个文件名。这些命令需要 **root** 授权。有关用户授权及获取授权的信息，请查看 [《Red Hat Enterprise Linux 7 系统管理员》](#) 及 `su(1)` 和 `sudo(8)` man page。

可使用类似 `nmcli` 的工具做出更改，这些工具不要求断开关联接口连接，然后再重新连接。该命令的运行格式如下：

```
nmcli dev disconnect interface-name
```

后接：


```
nmcli con up interface-name
```

NetworkManager 不会触发任何网络脚本，但在其运行时如果使用 **ifup** 命令，网络脚本会尝试启动 **NetworkManager**。有关网络脚本的说明，请查看 [第 1.8 节 “NetworkManager 及网络脚本”](#)。

ifup 脚本是一个通用脚本，可完成一些任务，并调用具体接口脚本，比如 **ifup-ethX**、**ifup-wireless**、**ifup-ppp** 等等。用户手动运行 **ifup eth0** 后：

1. **ifup** 会查找名为 **/etc/sysconfig/network-scripts/ifcfg-eth0** 的文件；
2. 如果存在 **ifcfg** 文件，**ifup** 会在那个文件中查找 **TYPE** 密钥，以确定要调用的脚本类型；
3. **ifup** 根据 **TYPE** 调用 **ifup-wireless** 或 **ifup-eth** 或 **ifup-XXX**；
4. 具体类型脚本执行具体类型设置；
5. 然后具体类型脚本让通用功能执行与 **IP** 相关的任务，比如 **DHCP** 或静态设置。

引导时，**/etc/init.d/network** 会读取所有 **ifcfg** 文件，并检查每个包含 **ONBOOT=yes** 的文件，确定是否已在 **ifcfg** 列出的设备中启动 **NetworkManager**。如果 **NetworkManager** 正在启动或已经启动那个设备，则不需要对那个文件进行任何操作，然后检查下一个包含 **ONBOOT=yes** 的文件。如果 **NetworkManager** 尚未启动那个设备，则 **initscripts** 会继续采用传统方式运行，并为那个 **ifcfg** 文件调用 **ifup**。

最终的结果是在系统启动后，会使用 **NetworkManager** 或 **initscripts** 启动所有包含 **ONBOOT=yes** 的 **ifcfg** 文件。这样可保证在 **NetworkManager** 无法处理某些传统的网络类型时，比如 **NetworkManager** 不处理的类型（ISDN 或模拟拨号调制解调器），以及 **NetworkManager** 尚不支持的新应用程序时，仍可使用 **initscripts** 正常启动它们。



注意

建议不要在保存目前使用的 **ifcfg** 文件的同一位置保存其备份文件。该脚本会运行 **ifcfg-***，扩展名 **.old**、**.orig**、**.rpmnew**、**.rpmorig** 和 **.rpmsave** 除外。最好是不要将备份文件保存在 **/etc/** 目录下。

1.10. 其他资料

以下信息资源为您提供有关 Red Hat Enterprise Linux 7 联网的附加资源。

1.10.1. 已安装文档

- » **man(1)** man page — 论述 man pages 及如何找到它们。
- » **NetworkManager(8)** man page — 论述网络管理守护进程。
- » **NetworkManager.conf(5)** man page — 论述 **NetworkManager** 配置文件。
- » **/usr/share/doc/initscripts-version/sysconfig.txt** — 论述配置文件及其指令。

第 2 章 配置 IP 联网

2.1. 静态和动态接口设置

什么时候使用静态寻址？什么时候使用动态寻址？这些都是主观决定，具体要看您的访问需要和具体要求。建立一个策略后，记录并一贯应用该策略比做出具体决定要重要。在传统公司 LAN 中，这种决定比较简单，因为通常服务器数目会比主机数少。部署和安装工具会方便为新主机提供静态配置，同时使用此类工具会改变您的工作流程和要求。以下两小节主要是为那些尚未经历此类决定的人员提供基本指导。有经验的系统管理员通常会有他们自己的一套规则和要求，这些规则和要求与在此讨论的内容有所不同。有关自动配置和管理的详情，请查看 [《Red Hat Enterprise Linux 7 系统管理员指南》](#) 中 **OpenLMI** 一节。[《Red Hat Enterprise Linux 7 安装指南》](#) 记录了 **kickstart** 的使用，还可使用该程序自动化网络设置分配。

2.1.1. 什么时候使用静态网络接口设置

在使用自动分配方法（比如 **DHCP**）时，要确保其网络可用性时，使用静态 **IP** 寻址。**DHCP**、**DNS** 和认证服务器是典型示例。带外（out-of-band）管理设备接口也应该使用静态设置配置，因为这些设备应该尽可能独立于其他网络架构工作。

对那些并不关键，但仍要求使用 **IP** 寻址的主机，请尽可能使用自动部署方法。例如：可将 **DHCP** 服务器配置为每次为同一主机提供 **IP** 主机。例如可使用这个方法设置公共打印机。

[第 2.1.3 节 “选择网络配置方法”](#) 中列出的所有配置工具都允许手动分配静态 **IP** 地址。**nmcli** 工具还适用于根据脚本分配网络配置。

2.1.2. 什么时候使用动态接口设置

启用并使用动态分配的 **IP** 地址及其他网络信息，无论是否有无法控制的原因不进行此操作。这样可从计划及编写手动设置中节省时间用于其他目的。*动态主机控制协议*（**DHCP**）是为主机动态分配网络配置的传统方法。有关此问题的详情请查看 [第 10.1 节 “为什么使用 DHCP”](#)。

将配置文件设定为自动获取地址，或者将接口配置文件的 **BOOTPROTO** 设定为 **dhcp** 后，**NetworkManager** 将默认调用 **DHCP** 客户端 **dhclient**。需要 **DHCP** 时，会为每个互联网协议启动 **dhclient**，即每个接口中的 **IPv4** 和 **IPv6**。若未运行 **NetworkManager**，或者未管理接口，旧的网络设备将根据需要调用 **dhclient** 实例。

2.1.3. 选择网络配置方法

- ✱ 要使用 **NetworkManager** 的文本用户界面工具 **nmtui** 配置接口，请执行 [第 2.2 节 “使用文本用户界面 nmtui”](#)。
- ✱ 要使用 **NetworkManager** 的命令行工具 **nmcli** 配置接口，请执行 [第 2.3 节 “使用 NetworkManager 命令行工具 nmcli”](#)。
- ✱ 要手动配置网络接口，请查看 [第 2.4 节 “使用命令行界面（CLI）”](#)。
- ✱ 要使用图形用户界面工具配置网络，请执行 [第 2.5 节 “在 GNOME 图形用户界面中使用 NetworkManager”](#)。

2.2. 使用文本用户界面 nmtui

可使用文本用户界面工具 **nmtui** 在终端窗口中配置接口。使用以下命令启动这个工具：

```
~]$ nmtui
```

此时会出现文本用户界面。无效命令会显示用法信息。

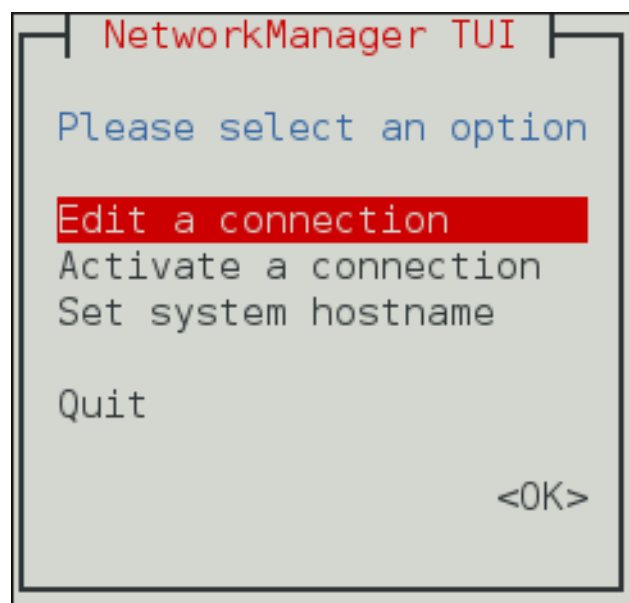


图 2.1. NetworkManager 文本用户界面启动菜单

使用箭头键或按 **Tab** 键向前选择选项，按 **Shift+Tab** 组合键返回。按 **Enter** 选择一个选项。按 **Space** 键选择复选框状态。

有关安装 **nmtui** 的详情请查看 [第 1.5 节“使用文本用户界面（nmtui）进行网络配置”](#)。

2.3. 使用 NetworkManager 命令行工具 nmcli

用户和脚本都可使用命令行工具 **nmcli** 控制 **NetworkManager**。该命令的基本格式为：

```
nmcli OPTIONS OBJECT { COMMAND | help }
```

其中 OBJECT 可为 **general**、**networking**、**radio**、**connection** 或 **device** 之一。最常用的选项为：**-t**，**--terse**（用于脚本）、**-p**，**--pretty** 选项（用于用户）及 **-h**，**--help** 选项。在 **nmcli** 中采用命令完成功能，无论何时您不确定可用的命令选项时，都可以按 **Tab** 查看。有关选项及命令的完整列表，请查看 **nmcli(1)** man page。

nmcli 工具有一些内置上下文相关的帮助信息。例如：运行以下两个命令，并注意不同之处：

```
~]$ nmcli help
Usage: nmcli [OPTIONS] OBJECT { COMMAND | help }

OPTIONS
  -t[erse]                terse output
  -p[retty]               pretty output
  -m[ode] tabular|multiline output mode
  -f[ields] <field1,field2,...>|all|common specify fields to output
  -e[scape] yes|no        escape columns separators in
values
  -n[ocheck]              don't check nmcli and
NetworkManager versions
  -a[sk]                  ask for missing parameters
  -w[ait] <seconds>       set timeout waiting for
```



```

finishing operations
-v[ersion]           show program version
-h[elp]             print this help

OBJECT
g[eneral]           NetworkManager's general status and operations
n[etworking]       overall networking control
r[adio]            NetworkManager radio switches
c[onnection]       NetworkManager's connections
d[evice]           devices managed by NetworkManager

```

```

~]$ nmcli general help
Usage: nmcli general { COMMAND | help }

COMMAND := { status | hostname | permissions | logging }

status

hostname [<hostname>]

permissions

logging [level <log level>] [domains <log domains>]

```

在上面的第二个示例中，这个帮助信息与对象 **general** 有关。

nmcli-examples(5) man page 有很多有帮助的示例，节选如下：

显示 **NetworkManager** 总体状态：

```
nmcli general status
```

要控制 **NetworkManager** 日志记录：

```
nmcli general logging
```

要显示所有链接：

```
nmcli connection show
```

要只显示当前活动链接，如下所示添加 **-a**, **--active**：

```
nmcli connection show --active
```

显示由 **NetworkManager** 识别到设备及其状态：

```
nmcli device status
```

可简化命令并省略一些选项。例如：可将命令

```
nmcli connection modify id 'MyCafe' 802-11-wireless.mtu 1350
```

简化为

```
nmcli con mod MyCafe 802-11-wireless.mtu 1350
```

可省略 **id** 选项，因为在这种情况下对于 **nmcli** 来说连接 ID（名称）是明确的。您熟悉这些命令后可做进一步简化。例如：可将

```
nmcli connection add type ethernet
```

改为

```
nmcli c a type eth
```



注意

如有疑问，请使用 tab 完成功能。

使用 nmcli 启动和停止接口

可使用 **nmcli** 工具启动和停止任意网络接口，其中包括主接口。例如：

```
nmcli con up id bond0
nmcli con up id port0
nmcli dev disconnect iface bond0
nmcli dev disconnect iface ens3
```



注意

建议使用 **nmcli dev disconnect iface *iface-name*** 命令，而不是 **nmcli con down id *id-string*** 命令，因为连接断开可将该接口放到“手动”模式，这样做用户让 **NetworkManager** 启动某个连接前，或发生外部事件（比如载波变化、休眠或睡眠）前，不会启动任何自动连接。

nmcli 互动连接编辑器

nmcli 工具具有一个互动连接编辑器。请运行以下命令使用该工具：

```
~]$ nmcli con edit
```

此时会提示您从显示的列表中选择有效连接类型。输入连接类型后，就会为您显示 **nmcli** 提示符。如果您熟悉连接类型，也可以在 **nmcli con edit** 命令中添加 **type** 选项，从而直接进入提示符。编辑现有连接配置的格式如下：

```
nmcli con edit [id | uuid | path] ID
```

要添加和编辑新连接配置，请采用以下格式：

```
nmcli con edit [type new-connection-type] [con-name new-connection-name]
```

在 **nmcli** 提示符后输入 **help** 查看可用命令列表。请使用 **describe** 命令获取设置及其属性描述，格式如下：

```
describe setting.property
```

例如：

```
nmcli> describe team.config
```

2.3.1. 了解 nmcli 选项

很多 **nmcli** 命令是可以顾名思义的，但有几个选项需要进一步了解：

type — 连接类型。

允许值为：**adsl**, **bond**, **bond-slave**, **bridge**, **bridge-slave**, **bluetooth**, **cdma**, **ethernet**, **gsm**, **infiniband**, **olpc-mesh**, **team**, **team-slave**, **vlan**, **wifi**, **wimax**。

每个连接了类型都有具体类型的命令选项。按 **Tab** 键查看该列表，或查看 **nmcli(1)** man page 中的 **TYPE_SPECIFIC_OPTIONS** 列表。**type** 选项可用于如下命令：**nmcli connection add** 和 **nmcli connection edit**。

con-name — 为连接配置分配的名称。

如果未指定连接名称，则会以如下格式生成名称：

```
type-ifname[-number]
```

连接名称是 *connection profile* 的名称，不应与代表某个设备的名称混淆（比如 **wlan0**、**ens3**、**em1** 等等）。虽然用户可为根据接口为链接命名，但这是两回事。一个设备可以有多个连接配置文件。这对移动设备，或者在不同设备间反复切换网线时很有帮助。与其编辑该配置，不如创建不同的配置文件，并根据需要将其应用到接口中。**id** 选项也是指连接配置文件名称。

id — 用户为连接配置文件分配的身份字符串。

可在 **nmcli connection** 命令中用来识别某个连接的 ID。输出结果中的 **NAME** 字段永远代表连接 ID（名称）。它指的是 **con-name** 给出的同一连接配置文件名称。

uuid — 系统为连接配置文件分配的独有身份字符串。

可在 **nmcli connection** 命令中用来识别某个连接的 UUID。

2.3.2. 使用 nmcli 连接到网络

请使用以下命令列出目前可用的网络连接：

```
~]$ nmcli con show
NAME                UUID                                TYPE
DEVICE
Auto Ethernet       9b7f2511-5432-40ae-b091-af2457dfd988  802-3-ethernet  --
ens3                 fb157a65-ad32-47ed-858c-102a48e064a2  802-3-ethernet  ens3
MyWiFi              91451385-4eb8-4080-8b82-720aab8328dd  802-11-wireless
wlan0
```

注：输出结果中的 NAME 字段永远代表连接 ID（名称）。它不是接口名称（尽管看起来很像）。在上述示例的第二个连接中，NAME 字段中的 **ens3** 代表用户为在 ens3 接口中应用的配置文件分配的连接 ID。在所示最后一个连接中，用户将连接 ID **MyWiFi** 分配给接口 wlan0。

添加以太网连接意味着生成一个配置文件，并将其分配给某个设备。生成新配置文件前，请检查可用设备，方法如下：

```
~]$ nmcli dev status
DEVICE  TYPE        STATE          CONNECTION
ens3    ethernet    disconnected    --
ens9    ethernet    disconnected    --
lo      loopback    unmanaged      --
```

添加动态以太网连接

要使用动态 **IP** 配置添加以太网配置文件，以便 **DHCP** 分配网络配置，可使用如下格式的命令：

```
nmcli connection add type ethernet con-name connection-name ifname
interface-name
```

例如：请使用以下命令创建名为 *my-office* 的动态连接配置文件：

```
~]$ nmcli con add type ethernet con-name my-office ifname ens3
Connection 'my-office' (fb157a65-ad32-47ed-858c-102a48e064a2) successfully
added.
```

NetworkManager 会将内部参数 **connection.autoconnect** 设定为 **yes**。**NetworkManager** 还会将设置保存到 **/etc/sysconfig/network-scripts/ifcfg-my-office** 文件中，在该文件中会将 **ONBOOT** 指令设定为 **yes**。

注：再次激活该接口前，**NetworkManager** 不会意识到对 **ifcfg** 文件的手动更改。有关使用配置文件的详情，请查看 [第 1.9 节“使用 sysconfig 文件进行网络配置”](#)。

请使用以下命令激活以太网连接：

```
~]$ nmcli con up my-office
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/5)
```

检查这些设备及连接的状态：

```
~]$ nmcli device status
DEVICE  TYPE        STATE          CONNECTION
ens3    ethernet    connected      my-office
ens9    ethernet    disconnected    --
lo      loopback    unmanaged      --
```

要更改主机发送到 **DHCP** 服务器的主机名，请按照以下方法修改 **dhcp-hostname** 属性：

```
~]$ nmcli con modify my-office my-office ipv4.dhcp-hostname host-name
ipv6.dhcp-hostname host-name
```

要更改主机发送到 **DHCP** 服务器的 **IPv4** 客户端 ID，请按照以下方法修改 **dhcp-client-id** 属性：

```
~]$ nmcli con modify my-office my-office ipv4.dhcp-client-id client-ID-string
```

没有用于 IPv6 的 `dhcp-client-id` 属性，`dhclient` 为 IPv6 生成识别符。详情请查看 `dhclient(8)` man page。

要忽略 DHCP 服务器发送到主机的 DNS 服务器，请按照以下操作修改 `ignore-auto-dns` 属性：

```
~]$ nmcli con modify my-office my-office ipv4.ignore-auto-dns yes
ipv6.ignore-auto-dns yes
```

有关属性及其设置的详情请查看 `nm-settings(5)` man page。

例 2.1. 使用互动编辑器配置动态以太网连接

运行以下命令使用互动编辑器配置动态以太网连接：

```
~]$ nmcli con edit type ethernet con-name ens3

===| nmcli interactive connection editor |===

Adding a new '802-3-ethernet' connection

Type 'help' or '?' for available commands.
Type 'describe [<setting>.<prop>]' for detailed property description.

You may edit the following settings: connection, 802-3-ethernet
(ethernet), 802-1x, ipv4, ipv6, dcb
nmcli> describe ipv4.method

=== [method] ===
[NM property description]
IPv4 configuration method. If 'auto' is specified then the appropriate
automatic method (DHCP, PPP, etc) is used for the interface and most other
properties can be left unset. If 'link-local' is specified, then a link-
local address in the 169.254/16 range will be assigned to the interface.
If 'manual' is specified, static IP addressing is used and at least one IP
address must be given in the 'addresses' property. If 'shared' is
specified (indicating that this connection will provide network access to
other computers) then the interface is assigned an address in the
10.42.x.1/24 range and a DHCP and forwarding DNS server are started, and
the interface is NAT-ed to the current default network connection.
'disabled' means IPv4 will not be used on this connection. This property
must be set.

nmcli> set ipv4.method auto
nmcli> save
Saving the connection with 'autoconnect=yes'. That might result in an
immediate activation of the connection.
Do you still want to save? [yes] yes
Connection 'ens3' (090b61f7-540f-4dd6-bf1f-a905831fc287) successfully
saved.
nmcli> quit
~]$
```

默认动作是持久保存连接配置文件。需要时可使用 **save temporary** 命令，在下次重启前只将该配置文件保存在内存中。

添加静态以太网连接

要添加使用静态 **IPv4** 配置的以太网连接，可使用以下命令：

```
nmcli connection add type ethernet con-name connection-name ifname
interface-name ip4 address gw4 address
```

可使用 **ip6** 和 **gw6** 选项添加 **IPv6** 地址和网关信息。

例如：使用以下命令创建只使用 **IPv4** 地址和网关的静态以太网连接：

```
~]$ nmcli con add type ethernet con-name test-lab ifname ens9 ip4
10.10.10.10/24 \
gw4 10.10.10.254
```

还可为该设备同时指定 **IPv6** 地址和网关，如下：

```
~]$ nmcli con add type ethernet con-name test-lab ifname ens9 ip4
10.10.10.10/24 \
gw4 10.10.10.254 ip6 abbe::cafe gw6 2001:db8::1
Connection 'test-lab' (05abfd5e-324e-4461-844e-8501ba704773) successfully
added.
```

NetworkManager 会将其内部参数 **ipv4.method** 设定为 **manual**，将 **connection.autoconnect** 设定为 **yes**。**NetworkManager** 还会将设置写入 **/etc/sysconfig/network-scripts/ifcfg-my-office** 文件，其中会将对应 **BOOTPROTO** 设定为 **none**，并将 **ONBOOT** 设定为 **yes**

注：再次激活该接口前，**NetworkManager** 不会意识到对 **ifcfg** 文件的手动更改。有关使用配置文件的详情，请查看 [第 1.9 节“使用 sysconfig 文件进行网络配置”](#)。

使用以下命令设定两个 **IPv4** DNS 服务器地址：

```
~]$ nmcli con mod test-lab ipv4.dns "8.8.8.8 8.8.4.4"
```

注：这样会替换之前设置的 **DNS** 服务器。要设置两个 **IPv6** DNS 服务器地址，请运行以下命令：

```
~]$ nmcli con mod test-lab ipv6.dns "2001:4860:4860::8888
2001:4860:4860::8844"
```

注：这样会替换之前设置的 **DNS** 服务器。也可以使用 **+** 作为前缀，在之前的任意设置中添加额外 **DNS** 服务器，如下：

```
~]$ nmcli con mod test-lab +ipv4.dns "8.8.8.8 8.8.4.4"
```

```
~]$ nmcli con mod test-lab +ipv6.dns "2001:4860:4860::8888
2001:4860:4860::8844"
```

请使用以下命令激活新的以太网连接：

```
~]$ nmcli con up test-lab ifname ens9
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/6)
```

检查这些设备及连接的状态：

```
~]$ nmcli device status
DEVICE  TYPE      STATE      CONNECTION
ens3    ethernet  connected  my-office
ens9    ethernet  connected  test-lab
lo      loopback  unmanaged  --
```

请使用以下命令查看新配置的连接详情：

```
~]$ nmcli -p con show test-lab
=====
===
                        Connection profile details (test-lab)
=====
===
connection.id:                test-lab
connection.uuid:              05abfd5e-324e-4461-844e-
8501ba704773
connection.interface-name:    ens9
connection.type:              802-3-ethernet
connection.autoconnect:       yes
connection.timestamp:         1410428968
connection.read-only:         no
connection.permissions:
connection.zone:              --
connection.master:            --
connection.slave-type:        --
connection.secondaries:
connection.gateway-ping-timeout: 0[output truncated]
```

使用 **-p**, **--pretty** 选项在输出结果中添加标题和分段。

例 2.2. 使用互动编辑器配置静态以太网连接

请运行以下命令，使用互动编辑器配置静态以太网连接：

```
~]$ nmcli con edit type ethernet con-name ens3

===| nmcli interactive connection editor |===

Adding a new '802-3-ethernet' connection

Type 'help' or '?' for available commands.
Type 'describe [>setting<.>prop<'] for detailed property description.

You may edit the following settings: connection, 802-3-ethernet
(ethernet), 802-1x, ipv4, ipv6, dcb
nmcli> set ipv4.addresses 192.168.122.88/24
Do you also want to set 'ipv4.method' to 'manual'? [yes]: yes
```

```
nmcli>
nmcli> save temporary
Saving the connection with 'autoconnect=yes'. That might result in an
immediate activation of the connection.
Do you still want to save? [yes] no
nmcli> save
Saving the connection with 'autoconnect=yes'. That might result in an
immediate activation of the connection.
Do you still want to save? [yes] yes
Connection 'ens3' (704a5666-8cbd-4d89-b5f9-fa65a3dbc916) successfully
saved.
nmcli> quit
~]$
```

默认动作是持久保存连接配置文件。需要时可使用 **save temporary** 命令，在下次重启前只将该配置文件保存在内存中。

将配置文件锁定至具体设备

要将配置文件锁定至某个具体接口设备，请在以上示例使用的命令中包含接口名称。例如：

```
nmcli connection add type ethernet con-name connection-name ifname
interface-name
```

要让某个配置文件在所有兼容以太网设备中都无法使用，请使用以下命令：

```
nmcli connection add type ethernet con-name connection-name ifname "*"
```

注：即使不想设置具体接口，也必须使用 **ifname** 参数。使用通配符 ***** 指定可在任意兼容设备中使用的配置文件。







要将配置文件锁定至某个具体 MAC 地址，请使用以下命令：

```
nmcli connection add type ethernet con-name "connection-name" ifname "*" mac
00:00:5E:00:53:00
```

添加 Wi-Fi 连接

请使用以下命令查看可用 Wi-Fi 访问点：

```
~]$ nmcli dev wifi list
```

SSID	MODE	CHAN	RATE	SIGNAL	BARS	SECURITY
FedoraTest	Infra	11	54 MB/s	98		WPA1
Red Hat Guest	Infra	6	54 MB/s	97		WPA2
Red Hat	Infra	6	54 MB/s	77		WPA2 802.1X
* Red Hat	Infra	40	54 MB/s	66		WPA2 802.1X
VoIP	Infra	1	54 MB/s	32		WEP
MyCafe	Infra	11	54 MB/s	39		WPA2

请使用以下命令生成使用静态 **IP** 配置，但允许自动 **DNS** 地址分配的 Wi-Fi 连接：

```
~]$ nmcli con add con-name MyCafe ifname wlan0 type wifi ssid MyCafe \
ip4 192.168.100.101/24 gw4 192.168.100.1
```


请使用以下命令设定 WPA2 密码，例如 “caffeine”：

```
~]$ nmcli con modify MyCafe wifi-sec.key-mgmt wpa-psk
~]$ nmcli con modify MyCafe wifi-sec.psk caffeine
```

有关密码安全的详情，请查看 [《Red Hat Enterprise Linux 7 安全指南》](#)。

请使用以下命令更改 Wi-Fi 状态：

```
~]$ nmcli radio wifi [on | off ]
```

更改具体属性

请使用以下命令检查具体属性，比如 `mtu`：

```
~]$ nmcli connection show id 'MyCafe' | grep mtu
802-11-wireless.mtu: auto
```

请使用以下命令更改设置的属性：

```
~]$ nmcli connection modify id 'MyCafe' 802-11-wireless.mtu 1350
```

请使用以下命令确认更改：

```
~]$ nmcli connection show id 'MyCafe' | grep mtu
802-11-wireless.mtu: 1350
```

注：NetworkManager 在设置中将 `802-3-ethernet` 和 `802-11-wireless` 视为参数，将 `mtu` 视为属性。有关属性及其设置的详情，请查看 `nm-settings(5)` man page。

2.3.3. 使用 nmcli 配置静态路由

要使用 `nmcli` 工具配置静态路由，则必须使用命令后或者交互式编辑器。

例 2.3. 使用 nmcli 配置静态路由

请使用命令行为现有以太网连接配置静态路由，输入如下命令：

```
~]# nmcli connection modify eth0 +ipv4.routes "192.168.122.0/24
10.10.10.1"
```

这样会将 `192.168.122.0/24` 子网的流量指向位于 `10.10.10.1` 的网关

例 2.4. 使用 nmcli 编辑器配置静态路由

请使用以下命令使用交互式编辑器配置静态路由：

```
~]$ nmcli con edit type ethernet con-name ens3

===| nmcli interactive connection editor |===
```

```
Adding a new '802-3-ethernet' connection
```

```
Type 'help' or '?' for available commands.
```

```
Type 'describe [>setting<.>prop<]' for detailed property description.
```

```
You may edit the following settings: connection, 802-3-ethernet
(ethernet), 802-1x, ipv4, ipv6, dcb
```

```
nmcli> set ipv4.routes 192.168.122.0/24 10.10.10.1
```

```
nmcli>
```

```
nmcli> save persistent
```

```
Saving the connection with 'autoconnect=yes'. That might result in an
immediate activation of the connection.
```

```
Do you still want to save? [yes] yes
```

```
Connection 'ens3' (704a5666-8cbd-4d89-b5f9-fa65a3dbc916) successfully
saved.
```

```
nmcli> quit
```

```
~]$
```

2.4. 使用命令行界面（CLI）

2.4.1. 使用 ifcfg 文件配置网络接口

接口配置文件控制各个独立网络设备的软件接口。系统引导后，它会使用这些文件决定激活哪些接口，以及如何配置它们。这些文件通常名为 **ifcfg-name**，其后缀指的是该配置文件控制的设备名称。通常 **ifcfg** 文件的后缀与配置文件本身的 **DEVICE** 指令给出的字符串相同。

静态网络设置

要让名为 eth0 的接口使用 **ifcfg** 文件配置使用静态网络设置的接口，请在 **/etc/sysconfig/network-scripts/** 目录中生成名为 **ifcfg-eth0** 的文件，如下所示：

```
DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
PREFIX=24
IPADDR=10.0.1.27
```

另外还可使用 **HWADDR** 指令指定硬件或 MAC 地址。注：这可能会影响设备命名步骤，如 [第 8 章 一致网络设备命名](#) 所述。不需要指定网络或广播地址，因为 **ipcalc** 会自动计算这些数值。

动态网络设置

要使用 **ifcfg** 文件为名为 em1 的接口配置使用动态网络设置的接口，请按照如下操作在 **/etc/sysconfig/network-scripts/** 目录中生成名为 **ifcfg-em1** 的文件：

```
DEVICE=em1
BOOTPROTO=dhcp
ONBOOT=yes
```

另外还可以使用 **HWADDR** 指令指定硬件或 MAC 地址。注：这可能会影响设备命名过程，如 [第 8 章 一致网络设备命名](#) 所述。

要配置一个向 **DHCP** 服务器发送不同的主机名的接口，请在 **ifcfg** 文件中添加以下行：

```
DHCP_HOSTNAME=hostname
```

要将接口配置为忽略由 **DHCP** 服务器发送的路由，请在 **ifcfg** 文件中添加以下行：

```
PEERDNS=no
```

这样可防止网络服务使用从 **DHCP** 服务器接收的 **DNS** 服务器更新 **/etc/resolv.conf**。

要配置一个接口以便使用具体 **DNS** 服务器，请如上所述设定 **PEERDNS=no**，并在 **ifcfg** 文件中添加以下行：

```
DNS1=ip-address
DNS2=ip-address
```

其中 *ip-address* 是 **DNS** 服务器的地址。这样就会让网络服务使用指定的 **DNS** 服务器更新 **/etc/resolv.conf**。

将配置文件设定为自动获取地址，或者将接口配置文件的 **BOOTPROTO** 设定为 **dhcp** 后，**NetworkManager** 将默认调用 **DHCP** 客户端 **dhclient**。需要 **DHCP** 时，会为每个互联网协议启动 **dhclient**，即每个接口中的 **IPv4** 和 **IPv6**。若未运行 **NetworkManager**，或者未管理接口，旧的网络设备将根据需要调用 **dhclient** 实例。

配置 DHCP 客户端

2.4.2. 使用 ip 命令配置网络接口

可使用 **ip** 程序为接口分配 **IP** 地址。这个命令的格式如下：

```
ip addr [ add | del ] address dev ifname
```

使用 ip 命令分配静态地址

请作为 **root** 使用以下命令为接口分配 **IP** 地址：

```
~]# ip address add 10.0.0.3/24 dev eth0
The address assignment of a specific device can be viewed as follows:
~]# ip addr show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
    link/ether f0:de:f1:7b:6e:5f brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.3/24 brd 10.0.0.255 scope global global eth0
        valid_lft 58682sec preferred_lft 58682sec
    inet6 fe80::f2de:f1ff:fe7b:6e5f/64 scope link
        valid_lft forever preferred_lft forever
```

更多示例及命令选项请查看 **ip-address(8)** manual page。

使用 ip 命令配置多个地址

因为 **ip** 程序支持为同一接口分配多个地址，所以不再需要使用别名接口方法在同一接口中绑定多个地址。可重复多次使用 **ip** 命令分配地址，这样就可以分配多个地址。例如：

```

~]# ip address add 192.168.2.223/24 dev eth1
~]# ip address add 192.168.4.223/24 dev eth1
~]# ip addr
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
    link/ether 52:54:00:fb:77:9e brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.223/24 scope global eth1
    inet 192.168.4.223/24 scope global eth1

```

有关 **ip** 程序的命令，请参考 **ip(8)** manual page。



注意

系统重启后会丢失命令行中给出的 **ip** 命令。

2.4.3. 静态路由及默认网关

静态路由是用于流量，不得用于或不应用于默认网关。路由通常是由网络中专门用于路由的设备提供（虽然所有设备都可被配置为执行路由）。因此，通常不需要在 Red Hat Enterprise Linux 服务器或客户端中配置静态路由。那些必须通过加密 VPN 通道，或者那些因为成本或安全原因应使用具体路由的流量除外。默认网关是用于那些目标不是本地网络，且未在路由表中指定首选路由的流量。默认网关通常是一个专用网络路由器。



注意

要扩展您的专业领域，[Red Hat 系统管理 I \(RH124\)](#) 培训课程可能会对您有所帮助。

使用命令行配置静态路由

如果需要静态路由，可使用 **ip route add** 命令在路由表中添加，使用 **ip route del** 命令删除。最常使用的 **ip route** 命令格式如下：

```
ip route [ add | del | change | append | replace ] destination-address
```

有关选项及格式的详情，请查看 **ip-route(8)** man page。

使用不附带任何选项的 **ip route** 命令显示 IP 路由表。例如：

```

~]$ ip route
default via 192.168.122.1 dev ens9 proto static metric 1024
192.168.122.0/24 dev ens9 proto kernel scope link src 192.168.122.107
192.168.122.0/24 dev eth0 proto kernel scope link src 192.168.122.126

```

要在主机地址中添加一个静态路由，即 **IP** 地址，请作为 **root** 运行以下命令：

```
ip route add 192.0.2.1 via 10.0.0.1 [dev ifname]
```

其中 **192.0.2.1** 是用点分隔的十进制符号中的 **IP** 地址，**10.0.0.1** 是下一个跃点，**ifname** 是进入下一个跃点的退出接口。

要在网络中添加一个静态路由，即代表 IP 地址范围的 IP 地址，请作为 **root** 运行以下命令：

```
ip route add 192.0.2.0/24 via 10.0.0.1 [dev ifname]
```

其中 192.0.2.1 是用点分隔的十进制符号中目标网络的 IP 地址，10.0.0.1 是网络前缀。网络前缀是在子网掩码中启用的位元。这个网络地址/网络前缀长度格式有时是指无类别域际路由选择（CIDR）表示法。

可在 `/etc/sysconfig/network-scripts/route-interface` 文件中为每个接口保存其静态路由配置。例如：接口 `eth0` 的静态路由可保存在 `/etc/sysconfig/network-scripts/route-eth0` 文件中。**route-interface** 文件有两种格式：**ip** 命令参数和网络/子网掩码指令，如下所述。

有关 **ip route** 命令的详情，请查看 **ip-route(8)** man page。

配置默认网关

确定默认网关时，首先是使用网络脚本解析 `/etc/sysconfig/network` 文件，然后是为处于“up”的接口解析 **ifcfg** 文件。**ifcfg** 文件是按照数字升序的顺序解析，使用最后读取的 GATEWAY 指令编写路由表中的默认路由。

因此，默认路由可使用 GATEWAY 指令代表，并可在全局或具体接口的配置文件中指定。但在 Red Hat Enterprise Linux 中，已经不再使用全局 `/etc/sysconfig/network` 文件，现在只能在每个接口的配置文件中指定网关。

在动态网络环境中，当使用 **NetworkManager** 管理主机时，网关信息一般是指具体接口，而且最好是由 DHCP 分配。在某些特殊情况下，如果要影响 **NetworkManager** 选择用来连接网关的退出接口，请在 **ifcfg** 文件中为那些不想连接默认网关的接口使用 **DEFROUTE=no** 命令。

2.4.4. 在 ifcfg 文件中配置静态路由

在命令提示符后使用 **ip** 设定的静态路由会在系统关机或重启后丢失。要配置静态路由以便在系统重启后仍可保留，则必须将其放在 `/etc/sysconfig/network-scripts/` 目录中。该文件名的格式应为 **route-ifname**。在该配置文件中使用的两类命令，即 **ip**，如 [第 2.4.4.1 节“使用 IP 命令参数格式的静态路由”](#) 所述；及 **网络/子网掩码** 格式，如 [第 2.4.4.2 节“网络/子网掩码指令格式”](#) 所述。

2.4.4.1. 使用 IP 命令参数格式的静态路由

如果需要根据接口设置的配置文件，例如 `/etc/sysconfig/network-scripts/route-eth0` 中第一行定义默认网关的路由。只有不是使用 **DHCP** 设置的网关需要此操作，且不会在 `/etc/sysconfig/network` 文件中进行全局设置：

```
default via 192.168.1.1 dev interface
```

其中 192.168.1.1 是默认网关的 IP 地址。*interface* 是连接到，或可连接网关的接口。可省略 **dev** 选项，它是自选项。注：这个设置可覆盖 `/etc/sysconfig/network` 文件中的设置。

如果需要路由到远程网络，可按以下方式指定静态路由。每行都解析为一个独立路由：

```
10.10.10.0/24 via 192.168.1.1 [dev interface]
```

其中 10.10.10.0/24 是网络地址及远程或目标网络的前缀长度。地址 192.168.1.1 是远程网络的第一个 IP 地址。首选下一个跃点地址，但也可以使用退出接口。“下一个跃点”的含义是链接的远端点，例如网关或路由器。可使用 **dev** 选项指定退出接口 *interface*，但不一定要这么做。根据需要添加所有静态路由。

以下是使用 **ip** 命令参数格式的 **route-interface** 文件示例。默认网关是 **192.168.0.1**，接口为 **eth0**，以及 **192.168.0.10** 中的租用专线或 WAN 连接。两个静态路由是用来连接 **10.10.10.0/24** 网络和 **172.16.1.10/32** 主机：

```
default via 192.168.0.1 dev eth0
10.10.10.0/24 via 192.168.0.10 dev eth0
172.16.1.10/32 via 192.168.0.10 dev eth0
```

在以上示例中，会将进入本地 **192.168.0.0/24** 网络的数据包指向附加到那个网络的接口。会将进入 **10.10.10.0/24** 网络和 **172.16.1.10/32** 主机的数据包指向 **192.168.0.10**。进入未知、远程、网络的数据包将使用默认网关，因此只应在默认路由不适用时为远程网络或主机配置静态路由。在这里远程是指没有直接连接到该系统的网络或主机。

指定退出接口为自选项，要强制让流量离开某个具体接口时有用。例如：在使用 VPN 时，可强制让流量通过远程网络使用 **tun0** 接口，即便该接口处于不同于目标网络的子网中。



重要

如果已由 **DHCP** 分配默认网关，且在配置文件中指定了使用同一跃点的同一网关，则会在启动时出错，或者在激活某个接口时出错，并显示以下出错信息：“RTNETLINK answers: File exists”。可忽略该信息。

2.4.4.2. 网络/子网掩码指令格式

还可以在 **route-interface** 文件中使用网络/子网掩码指令格式。以下是网络/子网掩码格式示例，并随后提供具体说明：

```
ADDRESS0=10.10.10.0
NETMASK0=255.255.255.0
GATEWAY0=192.168.1.1
```

- ✦ **ADDRESS0=10.10.10.0** 是要连接的远程网络或主机的网络地址。
- ✦ **NETMASK0=255.255.255.0** 是使用 **ADDRESS0=10.10.10.0** 定义的网络地址的子网掩码。
- ✦ **GATEWAY0=192.168.1.1** 是默认网关，或用来连接 **ADDRESS0=10.10.10.0** 的 IP 地址。

以下为使用网络/子网掩码指令格式的 **route-interface** 文件示例。默认网关为 **192.168.0.1**，但租用线路或 WAN 连接位于 **192.168.0.10**。这两个静态路由分别用于连接 **10.10.10.0/24** 和 **172.16.1.0/24** 网络：

```
ADDRESS0=10.10.10.0
NETMASK0=255.255.255.0
GATEWAY0=192.168.0.10
ADDRESS1=172.16.1.10
NETMASK1=255.255.255.0
GATEWAY1=192.168.0.10
```

后面的静态路由必须以数字顺序排列，且不能跳过任意数值。例如：**ADDRESS0**、**ADDRESS1**、**ADDRESS2** 等等。

2.4.5. 配置 VPN

Red Hat Enterprise Linux 7 中常见 VPN 的首选方法是使用 **Libreswan** 的 IPsec。使用命令行配置 IPsec VPN 的详情，请参考 [《Red Hat Enterprise Linux 7 安全指南》](#)。

2.5. 在 GNOME 图形用户界面中使用 NetworkManager

在 Red Hat Enterprise Linux 7 中，**NetworkManager** 本身没有图形用户界面（GUI）。GNOME Shell 提供的网络连接图标位于桌面右上角，同时新的 **control-center** GUI 提供 网络 设定配置工具。原有的 **nm-connection-editor** GUI 仍可用于某些任务。

2.5.1. 使用 GUI 连接到网络

可采用两种方法进入 **control-center** 的 网络 设定页面：

- ✧ 按 **网络** 键进入活动概述页面，输入 **control network**，如 [图 2.2 “在 GNOME 中选择网络工具”](#) 所示，并按 **Enter**。此时会出现 网络 设定工具。继续执行 [第 2.5.2 节 “配置网络及编辑现有连接”](#)。

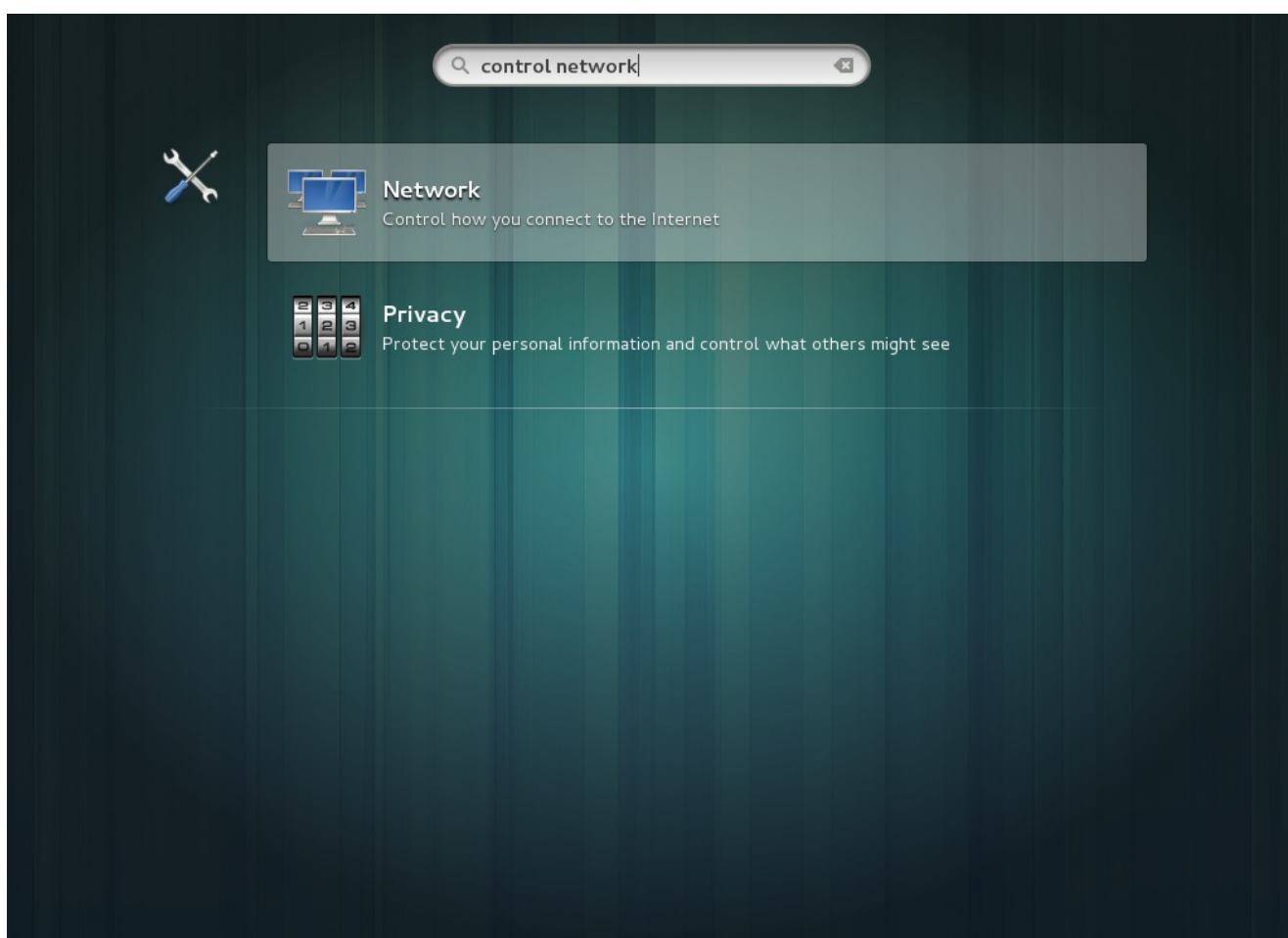


图 2.2. 在 GNOME 中选择网络工具

- ✧ 点击屏幕右上角的 GNOME Shell 网络连接按钮打开菜单。

点击 GNOME Shell 网络连接按钮后会为您显示：

- ✧ 目前已连接的网络分类列表（比如 **有线网络** 和 **Wi-Fi**）；
- ✧ **NetworkManager** 已探测出的所有**可用网络** 列表；
- ✧ 连接到任意已配置虚拟专用网络（VPN）的选项；及

✱ 选择 **网络设置** 菜单条目的选项。

如果已连接到某个网络，则会以符号 **ON** 按钮表示。点击按钮的任何位置均可更改该按钮的状态。

点击 **网络设置** 后会出现 **网络** 设置工具。继续执行 [第 2.5.2 节“配置网络及编辑现有连接”](#)。

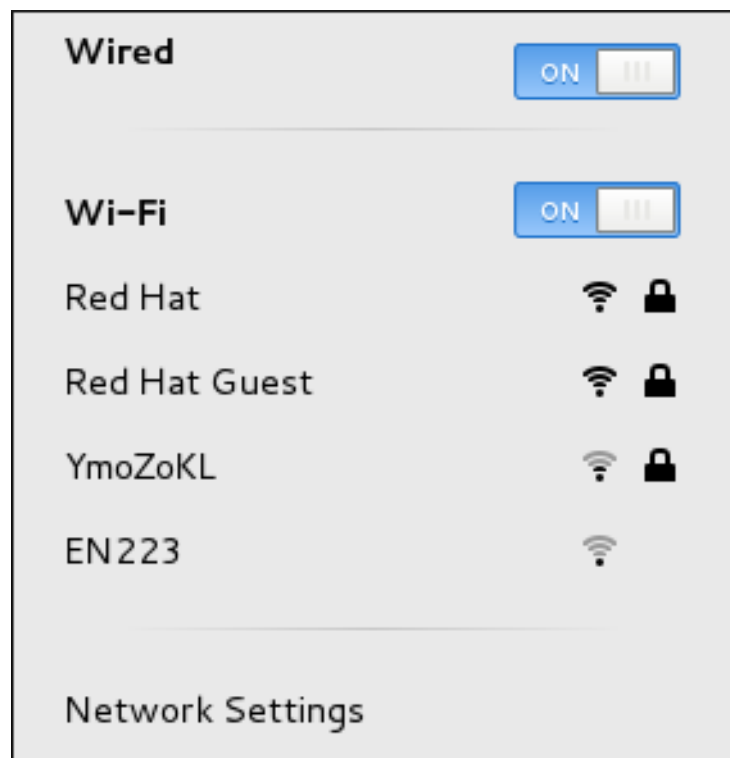


图 2.3. GNOME 网络菜单显示所有可用网络和已连接网络

2.5.2. 配置网络及编辑现有连接

网络 设置页面显示连接状态、其类型和接口、其 **IP** 地址及路由详情等等。



图 2.4. 使用网络设置页面配置网络

网络 设置页面的左侧有一个显示可用网络设备或接口的菜单。该菜单中包括软件接口，比如 VLAN、桥接、绑定或成组。右侧会显示所选网络设备或接口的 **连接配置文件**。配置文件是可应用于某个接口的命名设置集合。配置文件下面是加号按钮和减号按钮，可用来添加和删除新网络连接，右侧有一个齿轮图标，可用来编辑所选网络设备或 VPN 连接的详情。要添加新连接，请点击加号打开 **添加网络连接** 窗口，并继续执行 [第 2.5.2.1 节“配置新连接”](#)。

编辑现有连接

点击 **网络** 设置窗口中现有配置文件的齿轮图标，打开 **网络** 详情窗口，在那里可完成大多数网络配置任务，比如 **IP** 地址分配、**DNS** 和路由配置。



图 2.5. 使用网络连接详情窗口配置网络

2.5.2.1. 配置新连接

在 **网络** 设置窗口中点击菜单下面的加号，打开 **添加网络连接** 窗口。此时会显示可添加的连接类型列表。

然后配置：

- ✧ **VPN 连接**，点击 **VPN** 条目并继续执行 [第 2.5.7 节“建立 VPN 连接”](#)；
- ✧ **绑定连接**，点击 **绑定** 条目并继续执行 [第 4.6.1 节“建立绑定连接”](#)；
- ✧ **桥接连接**，点击 **桥接** 条目并继续执行 [第 6.4.1 节“建立桥接连接”](#)；
- ✧ **VLAN 连接**，点击 **VLAN** 条目并继续执行 [第 7.5.1 节“建立 VLAN 连接”](#)；或
- ✧ **成组连接**，点击 **成组** 条目并继续执行 [第 5.13 节“使用 GUI 创建网络成组”](#)。

2.5.3. 自动连接到网络

对于您要添加或配置的所有连接，都可以选择是否使用 **NetworkManager** 在网络可用时自动连接到该网络。

过程 2.1. 将 NetworkManager 配置为探测后自动连接到网络

1. 按 **Super** 键进入活动概述页面，输入 **control network**，然后按 **Enter** 键，此时会显示 **网络** 设置工具。

2. 在左侧按钮中选择网络接口。
3. 点击右侧菜单中连接配置文件中的齿轮图标。如果只有一个配置文件与所选接口关联，则会在右下角显示齿轮图标。此时会出现 **网络** 详情窗口。
4. 选择左侧的 **身份** 菜单，此时 **网络** 窗口会转变为身份视图。
5. 选择 **自动连接** 后，**NetworkManager** 就会在 **NetworkManager** 探测到网络可用时自动连接到网络。如果不想让 **NetworkManager** 自动连接，则请取消选择该复选框。如果清除该复选框，则必须在网络连接图标菜单中手动选择连接方可连接到该网络。

2.5.4. 系统范围及专用连接配置文件

NetworkManager 保存所有 **连接配置文件**。配置文件是一个可应用于某个接口的命名设置集合。**NetworkManager** 保存这些连接配置文件以便在系统范围内使用（**系统连接**），也可以作为 **用户连接** 配置文件使用。对这些连接配置文件的访问取决于 **NetworkManager** 中保存的权限。有关 **connection** 设置中 **permissions** 属性的详情请查看 **nm-settings(5)** man page。这些权限与 **ifcfg** 文件中的 **USERS** 指令对应。如果没有 **USERS** 指令，则该网络配置文件可用于所有用户。例如：**ifcfg** 文件中的如下命令可让该连接仅用于下列用户：

```
USERS="joe bob alice"
```

也可以使用图形用户界面工具设定。在 **nm-connection-editor** 的 **常规** 标签中有对应的 **所有用户均可访问这个网络** 复选框，同时在 **GNOME control-center** 网络设置身份窗口中也有 **使其可用于其他用户** 复选框与之对应。

NetworkManager 的默认策略允许所有用户创建和修改系统范围内的连接。引导时可用的配置文件不可能是专用的，因为用户登录前无法看到专用网络。例如：如果用户 **user** 创建一个连接配置文件 **user-em2**，选择了 **自动连接** 复选框，但没有选择 **使其可用于其他用户**，则该连接在引导时就无法使用。

要限制连接和联网，可单独或联合使用以下两个选项：

- ✱ 清除 **使其可用于其他用户** 复选框，这样就只有正在进行修改的用户可修改和使用该连接。
- ✱ 使用 **polkit** 固件为每位用户限制常规网络操作权限。

这两个选项合并可对安全性及联网控制进行微调。有关 **polkit** 的详情请查看 **polkit(8)** man page。

注：VPN 连接总是作为每位用户的专用连接创建，因为假设 VPN 连接比 Wi-Fi 或以太网连接更专用。

过程 2.2. 将连接改为用于具体用户而不是在系统范围内使用，反之亦然。

根据系统的策略，需要在系统中拥有 root 授权方可更改连接属性，以确定其是要用于具体用户，还是在系统范围内使用。

1. 按 **Super** 键进入活动概述页面，输入 **control network**，然后按 **Enter** 键，此时会显示 **网络** 设置工具。
2. 在左侧按钮中选择网络接口。
3. 点击右侧菜单中连接配置文件中的齿轮图标。如果只有一个配置文件与所选接口关联，则会在右下角显示齿轮图标。此时会出现 **网络** 详情窗口。
4. 选择左侧的 **身份** 菜单，此时 **网络** 窗口会转变为身份视图。
5. 选择 **使其可用于其他用户** 复选框可造成 **NetworkManager** 使该连接在系统范围内可用。

反之，如果清除 **使其可用于其他用户** 复选框，则该连接只可用于具体用户。

2.5.5. 配置有线（以太网）连接

要配置有线连接，请按 **Super** 键计入活动概述页面，然后输入 **control network**，并按 **Enter** 键。此时会显示 **网络** 设置工具。

在左侧菜单中选择 **有线** 网络接口（如果未突出显示该选项）。

系统会默认创建并配置单一有线连接配置文件，我们称之为**有线**。配置文件是可用于某个接口的设置集合命名。可根据需要为一个接口创建一个以上配置文件。无法删除默认配置文件，但可以更改其设置。点击齿轮图标即可编辑默认 **有线** 配置文件。点击 **添加配置文件** 按钮即可新建有线连接配置文件。右侧菜单中会显示与所选接口关联的连接配置文件。

点击 **添加配置文件** 按钮添加新连接后，**NetworkManager** 会为那个连接生成新配置文件，然后打开同一对话框编辑现有连接。这些对话框之间的不同在于现有连接配置文件有 **详情** 和 **重置** 菜单条目。实际上您一直在编辑连接配置文件；不同之处只是连接是之前就存在，还是在点击 **添加配置文件** 按钮后由 **NetworkManager** 生成。

2.5.5.1. 配置连接名称、自动连接行为及可用性设置

编辑对话框中的很多设置适用于所有连接类型，请查看 **身份** 视图（或者在使用 **nm-connection-editor** 时查看 **常规** 标签）：

- ✦ **名称** — 为网络连接输入描述性名称。会在 **网络** 窗口中使用这个名称代表这个连接。
- ✦ **MAC 地址** — 选择必须使用这个配置文件的接口 MAC 地址。
- ✦ **克隆的地址** — 必要时输入不同的 MAC 地址使用。
- ✦ **MTU** - 如有必要，请输入要使用的具体 *最大传输单位*（MTU）。MTU 值代表链接层可传输的最大数据包（单位：字节）。这个值默认为 **1500**，且一般不需要指定或更改。
- ✦ **防火墙区** — 如有必要，请选择要使用的不同防火墙区。有关防火墙区的详情，请查看 [《Red Hat Enterprise Linux 7 安全性指南》](#)。
- ✦ **自动连接** — 如果要让 **NetworkManager** 在该连接可用时自动连接到该连接，请选择这个复选框。详情请查看 [第 2.5.3 节“自动连接到网络”](#)。
- ✦ **使其可用于其他用户** — 要让创建可用于系统中其他用户的连接，请选中这个复选框。详情请查看 [第 2.5.4 节“系统范围及专用连接配置文件”](#)。
- ✦ **使用这个连接时自动连接到 VPN** — 如果要让 **NetworkManager** 在连接到这个连接配置文件时，自动连接到所选 VPN，请选中这个复选框。在下拉菜单中选择 VPN。

正在保存新的（或修改的）的连接并进一步完成配置

完成编辑有线连接后，点击 **应用** 按钮保存自定义配置。如果在编辑该配置文件时正在使用该文件，则需要重启连接方可让 **NetworkManager** 应用更改。如果该配置文件处于 OFF 状态，请在网络连接图标菜单中将其设定为 ON。有关使用新的或已更改连接的详情，请查看 [第 2.5.1 节“使用 GUI 连接到网络”](#)。

可在 **网络** 窗口中选择现有连接，并点击齿轮图标返回编辑对话框进行编辑。

然后配置：

- ✦ **基于端口的网络访问控制（PNAC）**，点击 **802.1X Security** 标签执行 [第 2.5.10.1 节“配置 802.1X 安全性”](#)；

- ✱ 该连接的 **IPv4** 设置，点击 **IPv4** 设置 标签执行 [第 2.5.10.4 节“配置 IPv4 设置”](#)；或者，
- ✱ 该连接的 **IPv4** 设置，点击 **IPv6** 设置标签，继续执行 [第 2.5.10.5 节“配置 IPv6 设置”](#)。

2.5.6. 配置 Wi-Fi 连接

本小结论述了如何使用 **NetworkManager** 为接入点配置 Wi-Fi（也称无线连接或 802.11a/b/g/n）连接。

有关配置移动宽带（比如 3G）连接的详情，请查看 [第 2.5.8 节“建立移动宽带连接”](#)。

快速连接到可用接入点

连接到可用接入点的最简单方法是点击网络连接图标激活该网络连接图标的菜单，找到 **Wi-Fi** 网络列表中接入点的 *服务集标识符*（SSID），并点击。挂锁符号表示该接入点要求认证。如果该接入点安全，则会出现对话框，提示您输入认证密钥或密码。

NetworkManager 尝试自动探测接入点使用的安全类型。如果有多个选项，**NetworkManager** 会猜测安全类型，并将其显示在 **Wi-Fi 安全** 下拉菜单中。在 WPA-PSK 安全性中（附带密码短语的 WPA）不需要选择。在 WPA Enterprise（802.1X）中，必须特别选择安全性，因为无法自动探测。如果不确定，请按顺序尝试连接每种类型。最后，在 **密码** 字段输入密钥或密码短语。某些密码类型，比如 40 字节长的 WEP 或 128 字节长的 WPA 密钥，除非满足要求的长度，否则会无效。连接 按钮在输入所选安全类型所需密钥长度前保持不活跃状态。有关无线安全性的详情，请查看 [第 2.5.10.2 节“配置 Wi-Fi 安全性”](#)。

如果 **NetworkManager** 成功连接到接入点，则网络连接图标会改为无线连接信号强度的图形指示符。

还可为这些自动生成的接入点连接编辑设置，就像这些连接是由您添加的一样。网络 窗口的 **Wi-Fi** 页面中有一个 **历史记录** 按钮。点击这个按钮可显示所有曾经尝试的连接列表。详情请查看 [第 2.5.6.2 节“编辑连接或创建全新连接”](#)。

2.5.6.1. 连接至隐藏 Wi-Fi 网络

所有接入点都有一个 *服务集标识符*（SSID）以供识别。但可将接入点设定为不广播其 SSID，就是说将其隐藏起来，因此不会出现在 **NetworkManager** 的 **可用** 网络列表中。只要您知道无线接入点的 SSID、认证方法及 secrets，即使其处于隐藏状态，也可以连接。

要连接到隐藏无线网络，请按 **Super** 键进入活动概述页面，输入 **control network**，然后按 **Enter**。此时会出现 **网络** 窗口。在菜单中选择 **Wi-Fi**，然后选择 **连接到隐藏网络**，此时会出现一个对话框。如果之前曾连接到隐藏网络，请使用 **连接** 下拉菜单选择，并点击 **连接** 按钮。如果之前没有进行过此类操作，则可选择 **连接** 下拉菜单中的 **新建** 选项，输入隐藏网络的 SSID，并选择其 **Wi-Fi 安全** 方法，输入正确的认证 secrets，并点击 **连接** 按钮。

有关无线安全设置的详情，请查看 [第 2.5.10.2 节“配置 Wi-Fi 安全性”](#)。

2.5.6.2. 编辑连接或创建全新连接

打开 **网络** 对话框中的 **Wi-Fi** 页面，并选择 Wi-Fi 连接名称右侧的齿轮图标，打开之前尝试或成功连接的 Wi-Fi 进行编辑。如果该网络目前不在可用范围内，请点击 **历史记录**，显示过去的连接。点击齿轮图标编辑出现的连接对话框。**详情** 窗口会显示连接详情。

要配置 SSID 处于可用范围内的新连接，首先请打开 **网络** 窗口，选择 **Wi-Fi** 菜单条目，并点击该连接名称（默认与 SSID 相同），尝试连接 Wi-Fi。如果该 SSID 不处于可用范围内，请查看 [第 2.5.6.1 节“连接至隐藏 Wi-Fi 网络”](#)。如果该 SSID 处于可用范围内，则请按照以下步骤操作：

1. 按 **Super** 键进入活动概述页面，输入 **control network**，然后按 **Enter** 键，此时会显示 **网络** 设置工具。

2. 在左侧菜单条目中选择 **Wi-Fi** 接口。
3. 在右侧菜单中选择要连接的 Wi-Fi 连接配置文件。挂锁符号表示需要密钥或密码。
4. 如有必要，请输入认证详情。

配置 SSID、自动连接行为及可用性设置

要编辑 Wi-Fi 的连接设置，请选择 **网络** 页面中的 **Wi-Fi**，然后选择 Wi-Fi 连接名称右侧的齿轮图标。选择 **身份**。此时会有以下设置可用：

SSID

接入点 (AP) 的 *服务集标识符* (SSID)

BSSID

处于 **架构** 模式时，要连接的具体无线接入点的 MAC 地址，也称 *硬件地址* 的 *基本服务集标识符* (BSSID)。默认情况下该字段为空白，且您可以使用 **SSID**，无需指定其 **BSSID** 就可以连接到无线接入点。如果指定 BSSID。则会强制系统只与具体接入点关联。

在临时网络中，**BSSID** 是在创建对等网络时由 **mac80211** 子系统随机生成。**NetworkManager** 不会显示该网络。

MAC 地址

选择 Wi-Fi 接口要使用的 MAC 地址，也称 *硬件地址*。

单一系统可以有一个或多个无线网络适配器与之连接。因此 **MAC 地址** 字段可让您将具体无线适配器与具体连接（或多个连接）关联。

克隆的地址

在真实硬件地址中使用克隆的 MAC 地址。除非要求，否则可保留空白。

以下设置适用于所有配置文件：

- ✧ **自动连接** — 如果要让 **NetworkManager** 在这个连接可用时与其自动相连，则请选择这个复选框。详情请查看 [第 2.5.3 节“自动连接到网络”](#)。
- ✧ **使其可用于其他用户** — 要让创建可用于系统中其他用户的连接，请选中这个复选框。详情请查看 [第 2.5.4 节“系统范围及专用连接配置文件”](#)。

正在保存新的（或修改的）的连接并进一步完成配置

完成无线连接编辑后，点击 **应用** 按钮保存您的配置。正确配置后，则可从网络连接图标菜单中选择修改的连接并与之连接。有关选择及连接网络的详情，请查看 [第 2.5.1 节“使用 GUI 连接到网络”](#)。

在 **网络** 窗口中选择现有连接，并点击齿轮图标显示连接详情，从而进一步配置现有连接。

然后配置：

- ✧ 无线网络的安全认证，请点击 **安全性**，并执行 [第 2.5.10.2 节“配置 Wi-Fi 安全性”](#)；
- ✧ 该连接的 **IPv4** 设置，请点击 **IPv4**，并执行 [第 2.5.10.4 节“配置 IPv4 设置”](#)；或者，
- ✧ 该连接的 **IPv6** 设置，请点击 **IPv6** 并执行 [第 2.5.10.5 节“配置 IPv6 设置”](#)。

2.5.7. 建立 VPN 连接

Libreswan 提供的 IPsec 是 Red Hat Enterprise Linux 7 用来创建 VPN 的首选方法。以下所述 GNOME 图形用户界面工具需要 *NetworkManager-libreswan-gnome* 软件包。必要时，请确保安装该软件包。方法是作为 **root** 运行以下命令：

```
~]# yum install NetworkManager-libreswan-gnome
```

有关在 Red Hat Enterprise Linux 7 中安装新软件包的详情，请查看 [《Red Hat Enterprise Linux 7 系统管理员指南》](#)。

建立虚拟专用网络（VPN）可启用局域网（LAN）之间的通讯，另外，也可启动远程局域网之间的通讯。可通过设置对中级网络（比如互联网）的通道访问达到此目的。通常采用认证和加密方法设置 VPN 通道。使用安全通道建立 VPN 连接后，VPN 路由器或网关会根据您传送的数据包执行以下动作：

1. 为路由及认证目的添加一个认证标头；
2. 加密数据包数据；并
3. 根据解密及处理步骤中的封装式安全措施负载（Encapsulating Security Payload, ESP）协议封装数据包中的数据。

接收 VPN 路由器会去除标头信息，解密该数据，并将其路由至目标系统（工作站或某个网络中的某个节点）。使用网络对网络连接时，本地网络中的接收节点会接收已解密并可进行处理的数据包。因此网络对网络 VPN 连接加密和解密过程对于客户端来说是透明的。

因为它们采用多层认证和加密，VPN 是连接多个远程节点将其作为统一 intranet 使用的安全、有效的方法。

过程 2.3. 添加新 VPN 连接

打开 **网络** 窗口，选择菜单下方的加号配置新 VPN 连接。

1. 按 **Super** 键进入活动概述页面，输入 **control network**，然后按 **Enter** 键，此时会显示 **网络** 设置工具。
2. 选择菜单下方的加号。此时会出现 **添加网络连接** 窗口。
3. 选择 **VPN** 菜单条目。该视图现在改为提供手动配置 VPN 的方法，或者导入 VPN 配置文件。

必须为要安装的 VPN 类型安装正确的 **NetworkManager** VPN 插件。请查看 [第 2.5.7 节“建立 VPN 连接”](#)。

4. 点击 **添加** 按钮打开 **选择 VPN 连接类型** 助手。
5. 从菜单中为要连接的网关选择 VPN 协议。菜单中可供选择的 VPN 协议与所安装 **NetworkManager** VPN 插件对应。详情请查看 [第 2.5.7 节“建立 VPN 连接”](#)。
6. **添加网络连接** 窗口改为显示为在上一步中所选 VPN 连接类型的自定义设置。

过程 2.4. 编辑现有 VPN 连接

可打开 **网络** 窗口，从列表中选择连接名称配置现有 VPN 连接。然后点击 **编辑** 按钮。

1. 按 **Super** 键进入活动概述页面，输入 **control network**，然后按 **Enter** 键，此时会显示 **网络** 设置工具。
2. 在左侧菜单中选择要编辑的 **VPN** 连接。
3. 点击 **配置** 按钮。

配置连接名称、自动连接行为及可用性设置

编辑 对话框中的五个设置适用于所有连接类型，请查看 **常规** 标签：

- ✦ **连接名称** — 为网络连接输入描述性名称。这个名称可用于在 **网络** 窗口中列出这个连接。
- ✦ **可用时自动连接到这个网络** — 如果需要 **NetworkManager** 在这个连接可用时自动连接，则请选择正规复选框。详情请查看 [第 2.5.3 节“自动连接到网络”](#)。
- ✦ **所有用户都可以连接到这个网络** — 如果要在系统中创建所有用户均可使用的连接，则请选择正规复选框。更改这个设置需要 root 权限。详情请查看 [第 2.5.4 节“系统范围及专用连接配置文件”](#)。
- ✦ **使用这个连接时自动连接到 VPN** — 如果要想让 **NetworkManager** 在该连接可用时自动连接到 VPN 连接，则请选择这个复选框。请从下拉菜单中选择 VPN。
- ✦ **防火墙区域** — 从下拉菜单中选择防火墙区域。有关防火墙区域到想起请查看 [《Red Hat Enterprise Linux 7 安全指南》](#)。

配置 VPN 标签

网关

远程 VPN 网关的名称或 IP 地址。

组名称

远程网关中配置的 VPN 组名称。

用户密码

需要时，请输入用来认证 VPN 的秘密。

组密码

需要时，请输入用来认证 VPN 的秘密。

用户名

需要时，请输入用来认证 VPN 的用户名。

阶段 1 算法

必要时，请输入用来认证的算法，并设置加密频道。

阶段 2 算法

必要时，请输入用于 IPsec 协商的算法。

域

必要时，请输入域名。

正在保存新的（或修改的）的连接并进一步完成配置

完成编辑新 VPN 连接后，点击 **保存** 按钮保存自定义配置。如果编辑该配置文件时正在使用该文件，则可重启连接电源以便 **NetworkManager** 应用所有更改。如果将配置文件设定为 OFF，将其设定为 ON 或者网络连接图标菜单中选择它。有关使用新的或更改的连接的详情，请查看 [第 2.5.1 节“使用 GUI 连接到网络”](#)。

在 **网络** 窗口对话框中选择现有连接进行进一步配置，并点击 **配置** 返回 **编辑** 对话框。

然后配置：

- ✱ 连接的 **IPv4** 设置，请点击 **IPv4 设置** 标签，并执行 [第 2.5.10.4 节“配置 IPv4 设置”](#)。

2.5.8. 建立移动宽带连接

可使用 **NetworkManager** 的移动宽带连接功能连接到以下 2G 和 3G 服务：

- ✱ 2G — *GPRS*（通用分组无线业务），*EDGE*（增强数据率的 GSM 演进），或者 *CDMA*（码分多址联接方式）。
- ✱ 3G — *UMTS*（通用移动通信系统），*HSPA*（高速分包存取），或者 *EVDO*（只演进数据）。

您的计算机必须有系统可发现并识别的移动宽带设备（调制解调器），以便创建连接。可将此类设备构建入您的计算机（比如很多笔记本电脑和上网本），或者可作为内置或外置硬件单独提供。其中包括 PC 卡、USB 调制解调器或硬件保护装置、移动或蜂窝电话作为调制解调器使用。

过程 2.5. 添加新移动宽带连接

打开 **网络连接** 工具，并选择 **移动宽带** 标签。

1. 按 **Super** 键进入活动概述页面，输入 **nm-connection-editor** 然后按 **Enter**。此时会出现 **网络连接** 工具。
2. 点击 **添加** 按钮。此时会打开 **选择连接类型** 菜单。
3. 选择 **移动宽带** 菜单条目。
4. 点击 **创建** 打开 **设置移动宽带连接** 助手。
5. 在 **为这个移动宽带设备创建连接** 菜单中选择要在这个连接中使用的 2G 或者 3G 设备。如果无法使用下拉菜单，则表示系统无法探测到移动宽带可用设备。在这种情况下，点击 **取消**，确定在计算机中连接并识别可用移动宽带设备，然后重试此步骤。点击 **继续** 按钮。
6. 选择服务供应商所在国家，并点击 **继续** 按钮。
7. 从列表中选择供应商或手动输入。点击 **继续** 按钮。
8. 从下拉菜单中选择支付计划，并确认接入点名称（APN）正确。点击 **继续** 按钮。
9. 检查并确认设置，然后点击 **应用** 按钮。
10. 请参考 [第 2.5.8.1 节“配置移动宽带标签”](#) 编辑具体移动宽带设置。

过程 2.6. 编辑现有移动宽带连接

按照这些步骤编辑现有移动宽带连接。

1. 按 **Super** 键进入活动概述页面，输入 **nm-connection-editor** 然后按 **Enter**。此时会出现 **网络连接** 工具。
2. 选择 **移动宽带** 标签。
3. 选择要编辑的连接并点击 **编辑** 按钮。
4. 配置连接名称、自动连接行为及可用性设置。

编辑 对话框中的五个设置适用于所有连接类型，请查看 **常规** 标签：

- ✧ **连接名称** — 为网络连接输入描述性名称。这个名称可用于在 **网络** 窗口中列出这个连接。
- ✧ **可用时自动连接到这个网络** — 如果需要 **NetworkManager** 在这个连接可用时自动连接，则请选择正规复选框。详情请查看 [第 2.5.3 节 “自动连接到网络”](#)。
- ✧ **所有用户都可以连接到这个网络** — 如果要在系统中创建所有用户均可使用的连接，则请选择正规复选框。更改这个设置需要 root 权限。详情请查看 [第 2.5.4 节 “系统范围及专用连接配置文件”](#)。
- ✧ **使用这个连接时自动连接到 VPN** — 如果要想让 **NetworkManager** 在该连接可用时自动连接到 VPN 连接，则请选择这个复选框。请从下拉菜单中选择 VPN。
- ✧ **防火墙区域** — 从下拉菜单中选择防火墙区域。有关防火墙区域到想起请查看 [《Red Hat Enterprise Linux 7 安全指南》](#)。

5. 请参考 [第 2.5.8.1 节 “配置移动宽带标签”](#) 编辑具体移动宽带设置。

正在保存新的（或修改的）的连接并进一步完成配置

完成编辑移动宽带连接后，点击 **应用** 按钮保存自定义配置。如果在编辑该配置文件时正在使用它，则需要重启该连接方可让 **NetworkManager** 应用所做更改。如果该配置文件处于 OFF 状态，请将其设定为 ON，或者网络连接的图标菜单中选择。有关使用新的或更改的连接的详情，请查看 [第 2.5.1 节 “使用 GUI 连接到网络”](#)。

可在 **网络连接** 窗口中选择现有连接，并点击 **编辑** 返回 **编辑对话框** 进行进一步配置。

然后配置：

- ✧ 该连接的 **点到点** 设置，请点击 **PPP 设置** 标签执行 [第 2.5.10.3 节 “配置 PPP（点对点）设置”](#)；
- ✧ 该连接的 **IPv4** 设置，点击 **IPv4 设置** 标签执行 [第 2.5.10.4 节 “配置 IPv4 设置”](#)；或者，
- ✧ 该连接的 **IPv4** 设置，点击 **IPv6** 设置标签，继续执行 [第 2.5.10.5 节 “配置 IPv6 设置”](#)。

2.5.8.1. 配置移动宽带标签

如果已使用助手添加新的移动宽带连接（步骤请参看 [过程 2.5 “添加新移动宽带连接”](#)），则可编辑 **移动宽带** 标签在家用网络不可用时禁用漫游，在使用该连接时分配一个网络 ID，或者让 **NetworkManager** 采用某些技术（比如 3G 或 2G）。

号码

使用基于 GSM 的移动宽带网络建立 PPP 连接的拨号号码。最初安装宽带设备时会自动填充这个字段。通常可保持此字段空白，并输入 **APN**。

用户名

输入用来认证网络的用户名。有些供应商不提供用户名，或在连接到该网络时不接受用户名。

密码

输入用来认证网络的密码。有些供应商不提供，或者不接受密码。

APN

输入用来与基于 GSM 网络建立连接的 **接入点名称（APN）**。为连接输入正确的 APN 很重要，因为它通常决定：

- ✧ 用户如何分配其网络用量；和/或

- ✎ 用户是否可以访问互联网、intranet 或子网。

网络 ID

输入 **网络 ID** 可让 **NetworkManager** 强制设备只在具体网络中注册。这样可保证在无法直接控制漫游时，该连接不会漫游。

类型

Any — 默认值 **Any** 可让调制解调器选择最快的网络。

3G (UMTS/HSPA) — 强制连接只使用 3G 技术。

2G (GPRS/EDGE) — 强制连接只使用 2G 技术。

首选 3G (UMTS/HSPA) — 首先尝试使用 3G 技术连接（比如 HSPA 或者 UMTS），只有连接失败后方返回使用 GPRS 或者 EDGE。

首选 2G (GPRS/EDGE) — 首先尝试使用 2G 技术，比如 GPRS 或者 EDGE，只有连接失败后方返回使用 HSPA 或者 UMTS。

家用网络不可用时允许漫游

如果要想让 **NetworkManager** 终止连接，而不是从家用网络转为漫游网网络，则请取消选择这个复选框，以避免可能的漫游费用。如果选择这个复选框，则 **NetworkManager** 从家用网络转而使用漫游服务以保持良好连接，反之亦然。

PIN

如果设备的 *SIM*（*用户身份模块*）会由 *PIN*（*个人身份号码*）锁定，输入 PIN 以便 **NetworkManager** 为该设备解锁。如果需要 PIN 方可使用该设备，则 **NetworkManager** 必须解锁 SIM。

CDMA 和 EVDO 的选项较少。它们没有 **APN**、**Network ID** 或者 **Type** 选项。

2.5.9. 启用 DSL 连接

本小节旨在用于那些主机中有 DSL 卡，而不是使用外置合并 DSL 调制解调器路由器的安装，通常用于私人用户或 SOHO 安装。

过程 2.7. 添加新 DSL 连接

可打开 **网络连接** 窗口配置新 DSL 连接，点击 **添加** 按钮并从新连接列表的 **硬件** 部分选择 **DSL**。

1. 按 **Super** 键进入活动概述页面，输入 **nm-connection-editor** 然后按 **Enter**。此时会出现 **网络连接** 工具。
2. 点击 **添加** 按钮。
3. 此时会出现 **选择连接类型** 列表。
4. 选择 **DSL** 并按 **创建** 按钮。
5. 此时会出现 **编辑 DSL 连接 1** 窗口。

过程 2.8. 编辑现有 DSL 连接

可打开 **网络连接** 窗口配置现有 DSL 连接，并从该列表中选择连接名称。然后点击 **编辑** 按钮。

1. 按 **Super** 键进入活动概述页面，输入 **nm-connection-editor** 然后按 **Enter**。此时会出现 **网络连接 工具**。
2. 选择要编辑的连接并点击 **编辑** 按钮。

配置连接名称、自动连接行为及可用性设置

编辑 对话框中的五个设置适用于所有连接类型，请查看 **常规** 标签：

- ✱ **连接名称** — 为网络连接输入描述性名称。这个名称可用于在 **网络** 窗口中列出这个连接。
- ✱ **可用时自动连接到这个网络** — 如果需要 **NetworkManager** 在这个连接可用时自动连接，则请选择正规复选框。详情请查看 [第 2.5.3 节 “自动连接到网络”](#)。
- ✱ **所有用户都可以连接到这个网络** — 如果要在系统中创建所有用户均可使用的连接，则请选择正规复选框。更改这个设置需要 root 权限。详情请查看 [第 2.5.4 节 “系统范围及专用连接配置文件”](#)。
- ✱ **使用这个连接时自动连接到 VPN** — 如果要想让 **NetworkManager** 在该连接可用时自动连接到 VPN 连接，则请选择这个复选框。请从下拉菜单中选择 VPN。
- ✱ **防火墙区域** — 从下拉菜单中选择防火墙区域。有关防火墙区域到想起请查看 [《Red Hat Enterprise Linux 7 安全指南》](#)。

配置 DSL 标签

用户名

输入用来与服务供应商认证的用户名。

服务

除非由服务供应商指导，否则保留其为空白。

密码

输入服务供应商提供的密码。

正在保存新的（或修改的）的连接并进一步完成配置

完成编辑 DSL 连接后，点击 **应用** 按钮保存自定义配置。如果编辑该配置文件时正在使用该文件，则需要重启该连接方可让 **NetworkManager** 应用所有更改。如果该配置文件处于 OFF 状态，则请将其设定为 ON，或从网络连接图标菜单中选择。有关使用新的或更改的连接的详情，请查看 [第 2.5.1 节 “使用 GUI 连接到网络”](#)。

可在 **网络连接** 窗口中选择现有连接，并点击 **编辑** 返回 **编辑对话框** 进行进一步配置。

然后配置：

- ✱ **MAC 地址和 MTU 设置**，点击 **有线** 标签并执行 [第 2.5.5.1 节 “配置连接名称、自动连接行为及可用性设置”](#)；
- ✱ 该连接的 **点到点** 设置，请点击 **PPP 设置** 标签执行 [第 2.5.10.3 节 “配置 PPP（点对点）设置”](#)；
- ✱ 连接的 **IPv4** 设置，请点击 **IPv4 设置** 标签，并执行 [第 2.5.10.4 节 “配置 IPv4 设置”](#)。

2.5.10. 配置连接设置

2.5.10.1. 配置 802.1X 安全性

802.1X 安全性是用于 *基于端口的访问控制协议* (PNAC) 的 IEEE 标准名称。也称其为 *WPA 企业级协议*。只要使用 802.1X 安全性即可控制物理机对 *本地网络* 的访问。所有要加入逻辑网络的客户端都必须通过该服务器 (比如路由器) 使用正确的 802.1X 认证方法认证。

802.1X 安全性大多是讨论安全的无线网络 (WLAN)，但也可用于防止可物理连接到网络 (比如 LAN) 的入侵者获得进入授权。过去是将 **DHCP** 服务器配置为不向非认证用户出租 **IP** 地址，但由于各种原因，这个方法既不实用，也不安全，因此不再推荐使用。而 802.1X 安全性是用来通过基于端口的认证，保证一个逻辑上安全的网络。

802.1X 为 WLAN 和 LAN 访问控制提供一个框架，并作为一个封包以便提供扩展认证协议 (EAP) 类型。EAP 类型是定义如何获得网络安全的协议。

可为有线或无线连接配置 802.1X 安全性。方法是打开 **网络** 窗口 (参看 [第 2.5.1 节 “使用 GUI 连接到网络”](#))，并按照以下可用步骤操作。按 **Super** 键进入活动概述页面，输入 **control network**，然后按 **Enter** 键。此时会出现 **网络** 设置工具。执行 [过程 2.9, “有线连接”](#) 或者 [过程 2.10, “无线连接”](#)：

过程 2.9. 有线连接

1. 请从左侧菜单中选择 **有线** 网络接口。
2. 可点击 **添加配置文件** 为要配置的 802.1X 安全性添加新的网络连接配置文件，也可以选择现有连接配置文件，并点击齿轮图标。
3. 然后选择 **安全性** 并将符号电源按钮设定为 **ON** 以便启用设置配置。
4. 执行 [第 2.5.10.1.1 节 “配置 TLS \(传输层安全性\) 设置”](#)

过程 2.10. 无线连接

1. 请从左侧菜单中选择 **无线** 网络接口。如果有必要，请将符号电源按钮设定为 **ON**，并检查硬件开关是否设定为 on。
2. 对于要配置的 802.1X 安全性，可以为新连接选择连接名称，也可以点击现有连接配置文件的齿轮图标。如果是新连接，请完成认证步骤以便完成连接，然后点击齿轮图标。
3. 选择 **安全性**。
4. 在下拉菜单中选择以下安全方法之一：**LEAP**、**Dynamic WEP (802.1X)** 或者 **WPA & WPA2 Enterprise**。
5. 请参考 [第 2.5.10.1.1 节 “配置 TLS \(传输层安全性\) 设置”](#) 中有关与 **安全性** 下拉菜单中所选 *扩展认证协议 (EAP)* 类型描述。

2.5.10.1.1. 配置 TLS (传输层安全性) 设置

使用传输层安全性，客户端及服务器可根据 TLS 协议相互认证。该服务器证明它拥有数码证书，客户端使用其客户端证书证明其身份，并交换密钥信息。认证完成后，则不再使用 TLS 通道，而是使用交换的密钥，通过 AES、TKIP 或 WEP 加密数据。

必须在所有要认证的客户端中分布证书说明 EAP-TLS 认证方法非常强大，但设置较为繁复。使用 TLS 安全性需要消耗公钥基础设施 (PKI) 来管理证书。使用 TLS 安全性的优点是不允许被破坏的密码访问 (W) LAN：入侵者必须也拥有认证客户端的私钥。

NetworkManager 不决定要支持的 TLS 版本。**NetworkManager** 收集由用户输入的参数，并将其转发给处理该进程的守护进程 **wpa_supplicant**。该进程会按顺序使用 OpenSSL 建立 TLS 通道。OpenSSL 本身会与 SSL/TLS 协议版本协商，并使用两端都支持的最高版本。

选择认证方法

从以下认证方法中选择一个：

- ✦ 为 *传输层安全性* 选择 **TLS**，并执行 [第 2.5.10.1.2 节“配置 TLS 设置”](#)；
- ✦ 为 *使用安全通道的灵活认证协议* 选择 **FAST**，并执行 [第 2.5.10.1.4 节“配置通道 TLS 设置”](#)；
- ✦ 为 *通道传输层安全性* 选择 **通道 TLS**，也称 TTLS 或者 EAP-TTLS，并执行 [第 2.5.10.1.4 节“配置通道 TLS 设置”](#)；
- ✦ 为 *保护的可扩展认证协议* 选择 **保护的 EAP (PEAP)**，并执行 [第 2.5.10.1.5 节“配置受保护的 EAP \(PEAP\) 设置”](#)。

2.5.10.1.2. 配置 TLS 设置

身份识别

提供这台服务器的身份识别。

用户证书

点击浏览，并选择内嵌 *可区别编码规则* (DER) 或者 *隐私增强邮件* (PEM) 的个人 X.509 证书文件。

CA 证书

点击浏览，并选择内嵌 *可区别编码规则* (DER) 或者 *隐私增强邮件* (PEM) 的 X.509 证书颁发机构证书文件。

私钥

点击浏览，并选择内嵌 *可区别编码规则* (DER)、*隐私增强邮件* (PEM) 或者 *个人信息交换语法标准* (PKCS #12) 的私钥文件。

私钥密码

在 **私钥** 字段输入私钥密码。选择 **显示密码** 即可在输入密码时看到输入内容。

2.5.10.1.3. 配置 FAST 设置

匿名身份

提供这台服务器的身份识别。

PAC 部署

选择该复选框启用该功能，并从 **匿名**、**认证的** 及 **二者均使用** 中选择。

PAC 文件

点击浏览，并选择 *受保护的访问凭据* (PAC) 文件。

内部认证

GTC — 通用令牌卡。

MSCHAPv2 — Microsoft 质询握手身份验证协议版本 2。

用户名

输入认证过程中要使用的用户名。

密码

输入认证过程中要使用的密码。

2.5.10.1.4. 配置通道 TLS 设置

匿名身份

这个值是用来解密身份。

CA 证书

点击浏览并选择证书颁发机构的证书。

内部认证

PAP — 密码认证协议。

MSCHAP — 质询握手身份认证协议。

MSCHAPv2 — Microsoft 质询握手身份验证协议版本 2。

CHAP — 质询握手身份认证协议。

用户名

输入认证过程中要使用的用户名。

密码

输入认证过程中要使用的密码。

2.5.10.1.5. 配置受保护的 EAP (PEAP) 设置

匿名身份

这个值是用来解密身份。

CA 证书

点击浏览并选择证书颁发机构的证书。

PEAP 版本

要使用的受保护 EAP 版本。自动为 0 或 1。

内部认证

MSCHAPv2 — Microsoft 质询握手身份验证协议版本 2。

MD5 — 消息摘要 5，这是一种加密哈希功能。

GTC — 通用令牌卡。

用户名

输入认证过程中要使用的用户名。

密码

输入认证过程中要使用的密码。

2.5.10.2. 配置 Wi-Fi 安全性

安全性

None — 不为 Wi-Fi 连接加密。

WEP 40/128-bit 密钥 — IEEE 802.11 标准中的有线等效保密 (WEP)，采用单一预共享密钥 (PSK)。

WEP 128-bit 密码短语 — 将使用密码短语的 MD5 哈希衍生出 WEP 密钥。

LEAP — Cisco 系统的轻型可延伸认证协议。

动态 WEP (802.1X) — WEP 密钥会动态更改。请与 [第 2.5.10.1.1 节“配置 TLS \(传输层安全性\) 设置”](#) 一同使用。

WPA & WPA2 个人 — IEEE 802.11i 标准中的 Wi-Fi 访问安全 (WPA)，WEP 的替代品。802.11i-2004 标准的 Wi-Fi 访问安全 II (WPA2)。个人模式使用预共享密钥 (WPA-PSK)。

WPA & WPA2 Enterprise — WPA 与 RADIUS 认证服务器一同提供 IEEE 802.1X 网络访问控制。与 [第 2.5.10.1.1 节“配置 TLS \(传输层安全性\) 设置”](#) 一同使用。

密码

输入认证过程中要使用的密码。

2.5.10.3. 配置 PPP (点对点) 设置

配置方法

使用点对点加密法 (MPPE)

Microsoft 点对点加密协议 ([RFC 3078](#))。

允许压缩 BSD 数据

PPP BSD 压缩协议 ([RFC 1977](#))。

允许 Deflate 数据压缩

PPP Deflate 协议 ([RFC 1979](#))。

使用 TCP 标头压缩

为低速串口链接压缩 TCP/IP 标头 ([RFC 1144](#))。

发送 PPP 回显数据包

用于环回测试的 LCP 回显请求和回显回复代码 ([RFC 1661](#))。

2.5.10.4. 配置 IPv4 设置

可使用 **IPv4 设置** 标签根据需要配置用来连接网络的方法、输入 IP 地址、路由器、及 DNS 信息。创建和修改以下连接类型之一时可使用 **IPv4 设置** 标签：有线、无线、移动宽带、VPN 或者 DSL。如果需要配置 IPv6 地址，请查看 [第 2.5.10.5 节“配置 IPv6 设置”](#)。如果需要配置静态路由，则请点击 **路由** 按钮，并执行 [第 2.5.10.6 节“配置路由”](#)。

如果使用 DHCP 从 DHCP 服务器中获取动态 IP 地址，则只需要将 **方法** 设定为 **自动 (DHCP)**。

设定方法

根据连接类型列出的可用 IPv4 方法

点击 **方法** 下拉菜单，根据要配置的连接类型，可从以下 **IPv4** 连接方法中选择一个。所有方法都是根据连接类型或类型列出，并关联至：

方法

自动 (DHCP) — 如果要连接的网络使用 **DHCP** 分配 **IP** 地址，则请选择这个选项。不需要填写 **DHCP 客户端 ID** 字段。

仅用于自动 (DHCP) 地址 — 如果要连接的网络使用 **DHCP** 服务器分配 **IP** 地址，但您希望使用 **DNS** 服务器手动分配地址，则请选择这个选项。

仅用于本地链接 — 如果要连接的网络没有 **DHCP** 服务器，且您不希望手动分配 **IP** 地址。将根据 [RFC 3927](#) 使用前缀 **169.254/16** 随机分配地址。

与其他计算机共享 — 如果要配置的接口是用来共享互联网或者 WAN 连接，则请选择这个选项。为该接口分配 **10.42.x.1/24** 范围中的地址，启动 **DHCP** 服务器和 **DNS** 服务器，同时会根据 *网络地址转换* (NAT) 将该接口连接到默认网络连接。

禁用 — 为该连接禁用 **IPv4**。

有线、无线及 DSL 连接方法

手动 — 如果要手动分配 **IP** 地址，则请选择这个选项。

移动宽带连接方法

自动 (PPP) — 如果要连接的网络为您自动分配 **IP** 地址和 **DNS** 服务器，则请选择这个选项。

只用于自动 (PPP) 地址 — 如果要连接的网络自动为您分配 **IP** 地址，但您希望手动指定 **DNS** 服务器，则请选择这个选项。

VPN 连接方法

自动 (VPN) — 如果要连接的网络自动为您分配 **IP** 地址和 **DNS** 服务器，则请选择这个选项。

只用于自动 (VPN) 地址 — 如果要连接的网络自动为您分配 **IP** 地址，但您希望手动指定 **DNS** 服务器，则请选择这个选项。

DSL 连接方法

自动 (PPPoE) — 如果要连接的网络自动为您分配 **IP** 地址和 **DNS** 服务器，则请选择这个选项。

只用于自动 (PPPoE) 地址 — 如果要连接的网络自动为您分配 **IP** 地址，但您希望手动指定 **DNS** 服务器，则请选择这个选项。

有关为网络连接配置静态路由的详情，请查看 [第 2.5.10.6 节“配置路由”](#)。

2.5.10.5. 配置 IPv6 设置

方法

忽略 — 如果要忽略该连接的 **IPv6** 设置，则请选择这个选项。

自动 — 选择这个选项并使用 *SLAAC* 根据硬件地址及 *路由器公告* (RA) 创建自动、无缝配置。

只用于自动地址 — 如果要连接的配置使用 *路由器公告* (RA) 创建自动、无缝配置，但您希望手动分配 **DNS** 服务器，则请选择这个选项。

只用于自动 DHCP — 如果使用 RA，但不需要直接来自 **DHCPv6** 的信息创建状态配置，则请选择这个选项。

手动 — 如果要手动分配 **IP** 地址，则请选择这个选项。

只用于本地链接 — 如果要连接的网络没有 **DHCP** 服务器，且您不希望手动分配 **IP** 地址，则请选择这个选项。将根据 [RFC 4862](#) 使用 **FE80::0** 分配随机地址。

地址

DNS 服务器 — 输入用逗号分开的 **DNS** 服务器。

搜索域 — 输入用逗号分开的域控制器。

有关为网络连接配置静态路由的详情，请查看 [第 2.5.10.6 节“配置路由”](#)。

2.5.10.6. 配置路由

会在主机路由表中自动填入直接连接网络的路由。这些路由是在检查处于“up”状态的网络接口时获得。本小节论述了在通过转换中间网络或连接（比如 **VPN** 通道或租赁线路）即可到达的网络或主机中输入静态路由的方法。要到达远程网络或主机，会从该系统中给出应发送流量的网关地址。

使用 **DHCP** 配置主机接口时，通常会分配指向 upstream 网络或互联网的网关地址。这个网关一般是指默认网关，因为这是在没有其他系统已知的更好的路由时（在路由表中显示）使用的网关。网络管理员通常使用该网络中的第一个或最后一个主机 **IP** 地址作为网关。例如：**192.168.10.1** 或者 **192.168.10.254**。请不要将其与代表网络自身的地址（在这个示例中是指 **192.168.10.0**），或者子网的广播地址（在这个示例中是指 **192.168.10.255**）混淆。

配置静态路由

要设定静态路由，请打开要配置连接的 **IPv4** 或者 **IPv6** 设置窗口。有关操作详情请查看 [第 2.5.1 节“使用 GUI 连接到网络”](#)。

路由

地址 — 输入远程网络、子网或主机的 **IP** 地址。

子网掩码 — 以上输入的 **IP** 地址子网掩码或前缀长度。

网关 — 连接到以上输入的远程网络、子网或主机的 **IP** 地址。

指标 — 网络成本，这个路由的首选值。较低的数值有较高优先级。

自动

当 Automatic 处于 **ON** 状态时，使用 **RA** 或 **DHCP**，但还可以添加额外的静态路由。如果它处于 **OFF** 状态时，则只使用定义的静态路由。

只在其网络的资源中使用这个连接

选择这个复选框可防止该连接成为默认路由。典型的示例是指当连接是一个连接到总部帐号的 **VPN** 通道或者租用线路，且不希望任何绑定互联网的流量通过这个连接。选择这个选项意味着只有特别为通过连接或在此手动输入路由指定的目标可通过这个连接路由。

2.6. 其他资料

以下信息资源提供有关本章内容的额外信息。

2.6.1. 已安装文档

- ✧ **ip(8)** man page — 描述 **ip** 程序的命令语法。
- ✧ **nmcli(1)** man page — 描述 **NetworkManager** 的命令行工具。
- ✧ **nmcli-examples(5)** man page — **nmcli** 命令示例。
- ✧ **nm-settings(5)** man page — 描述 **NetworkManager** 属性及其设置。

2.6.2. 在线文档

[《Red Hat Enterprise Linux 7 安全指南》](#)

描述基于 VPN 的 IPsec 及其配置。描述使用 DNSSEC 的认证 **DNS** 查询方法。

[RFC 1518](#) — 无类别域际路由选择 (CIDR)

描述 CIDR 地址分配及整合策略，包括可变长度子网划分。

[RFC 1918](#) — 论述专用网络地址分配

描述为专用网络保留的 **IPv4** 地址范围。

[RFC 3330](#) — IPv4 地址的特殊用法

描述互联网编号分配机构 (IANA) 分配的全球或指定 **IPv4** 地址块。

第 3 章 配置主机名

3.1. 了解主机名

hostname 有三种类型：static、pretty 和 transient

“static”主机名是可由用户选择的传统 **hostname**，并保存在 `/etc/hostname` 文件中。“transient”**hostname** 是由内核维护的动态主机名。它最初是默认的 static 主机名，其值默认为 “localhost”。可由 **DHCP** 或 **mDNS** 在运行时更改其默认值。“pretty” **hostname** 是为用户提供的任意格式 UTF8 主机名。



注意

主机名可以是任意格式的字符串，最长为 64 个字符。但 Red Hat 建议在 static 和 transient 名称与 **DNS** 采用的完全限定域名 (FQDN) 匹配，比如 `host.example.com`。还建议在 static 和 transient 名称中只包含 7 字节 ASCII 小写字母，无空格或点，并将其限制为使用 **DNS** 域名标签格式，尽管这不是一个严格要求。在以前的要求中不允许使用下划线，因此也不建议在此使用。

hostnamectl 工具会强制采用以下限制：static 和 transient 主机名只包含 **a-z**、**A-Z**、**0-9**、“-”、“_”和“.”，不能在开头或结尾处使用句点，不允许使用两个相连的句点。大小限制为 64 个字符。

3.1.1. 建议到命名方法

互联网名称与数字地址分配机构 (ICANN) 有时会在公共注册中添加之前未注册的顶级域（比如 `.yourcompany`）。因此 Red Hat 强烈建议您不要使用未委托给您的域名，即便是专用网络也不要使用，因为这样做可能会造成因为网络配置不同而对域名进行不同的解析，结果是导致网络资源不可用。使用非由您委托的域名还会造成 DNSSEC 部署和维护变得更为困难，因为域名冲突会要求手动配置以便启用 DNSSEC 验证。有关这个问题的详情请查看 [ICANN 有关域名冲突的常见问题](#)。

3.2. 使用文本用户界面 **nmtui** 配置主机名

可在终端窗口中使用文本用户界面工具 **nmtui** 配置主机名。运行下面的命令启动该工具：

```
~]$ nmtui
```

此时会出现文本用户界面。输入错误命令时会显示用法信息。

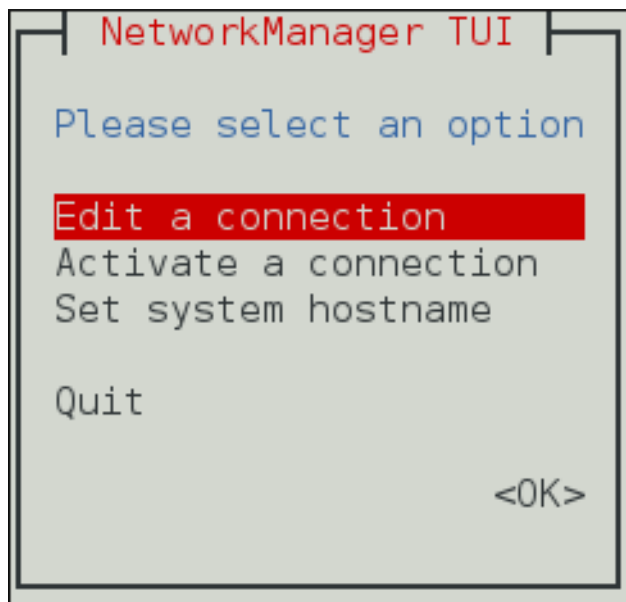


图 3.1. NetworkManager 文本用户界面启动菜单

导航时，使用箭头键或按 **Tab** 键在选项中前进，按 **Shift+Tab** 组合键后退。按 **Enter** 键选择一个选项。按 **Space** 键切换选择框状态。

有关安装 `nmtui` 的详情，请查看 [第 1.5 节 “使用文本用户界面（nmtui）进行网络配置”](#)。

可使用 **NetworkManager** 文本用户界面工具 `nmtui` 在 `/etc/hostname` 文件中查询和设置 static 主机。注：写入时，`hostnamectl` 不会意识到采用这个方法更改的主机名。

要强制 `hostnamectl` 注意这个 static 主机名更改，请作为 `root` 用户重启 `hostnamed`：

```
~]# systemctl restart systemd-hostnamed
```

3.3. 使用 `hostnamectl` 配置主机名

`hostnamectl` 工具是用来管理给定主机中使用的三种类型的主机名。

3.3.1. 查看所有主机名

请运行下面的命令查看所有当前主机名：

```
~]$ hostnamectl status
```

如果未指定任何选项，则默认使用 `status` 选项。

3.3.2. 设定所有主机名

请作为 `root` 用户运行下面的命令设定系统中的所有主机名：

```
~]# hostnamectl set-hostname name
```

这样会将 pretty、static 和 transient 主机名变得相似。Static 和 transient 主机名会简化为 pretty 主机名格式。使用 “-” 替换空格，并删除特殊字符。

3.3.3. 设定特定主机名

请作为 **root** 用户，使用附带相关选项的以下命令设定特定主机名：

```
~]# hostnamectl set-hostname name [option...]
```

其中 *option* 是 **--pretty**、**--static**、**--transient** 中的一个或多个选项。

如果 **--static** 或 **--transient** 选项与 **--pretty** 选项一同使用，则会将 static 和 transient 主机名简化为 pretty 主机名格式。使用 “-” 替换空格，并删除特殊字符。如果未使用 **--pretty** 选项，则不会发生简化。

设定 pretty 主机名时，如果该主机名中包含空格或单引号，请记住要使用正确的引号。例如：

```
~]# hostnamectl set-hostname "Stephen's notebook" --pretty
```

3.3.4. 清除特定主机名

要清除特定主机名，并将其还原为默认形式，请作为 **root** 用户使用附带相关选项的以下命令：

```
~]# hostnamectl set-hostname "" [option...]
```

其中 "" 是括起来的空白字符串，*option* 是 **--pretty**、**--static** 和 **--transient** 中的一个或多个选项。

3.3.5. 远程更改主机名

要在远程系统中运行 **hostnamectl** 命令，请使用 **-H**，**--host** 选项，如下所示：

```
~]# hostnamectl set-hostname -H [username]@hostname
```

其中 *hostname* 是要配置的远程主机。*username* 为自选项。**hostnamectl** 工具会使用 **SSH** 连接到远程系统。

3.4. 使用 nmcli 配置主机名

可使用 **NetworkManager** 工具 **nmcli** 查询和设定 **/etc/hostname** 文件中的主机名。注：写入时，**hostnamectl** 不会意识到采用这个方法更改的主机名。

请运行下面的命令查询 static 主机名：

```
~]$ nmcli general hostname
```

请作为 **root** 用户运行下面的命令将 static 主机名设定为 *my-server*：

```
~]# nmcli general hostname my-server
```

要强制 **hostnamectl** 注意这个 static 主机名更改，请作为 **root** 用户重启 **hostnamed**：

```
~]# systemctl restart systemd-hostnamed
```

3.5. 其他资料

以下信息资源提供有关 **hostnamectl** 的附加信息。

3.5.1. 已安装文档

- ✧ **hostnamectl(1)** man page — 描述 **hostnamectl**，包括命令及命令选项。
- ✧ **hostname(1)** man page — 包括 **hostname** 和 **domainname** 命令的说明。
- ✧ **hostname(5)** man page — 包含主机名文件、其内容和使用的说明。
- ✧ **hostname(7)** man page — 包含主机名解析说明。
- ✧ **machine-info(5)** man page — 描述本地机信息文件及其包含的环境变量。
- ✧ **machine-id(5)** man page — 描述本地机 ID 配置文件。
- ✧ **systemd-hostnamed.service(8)** man page — 描述 **hostnamectl** 使用的 **systemd-hostnamed** 系统服务。

第 4 章 配置网络绑定

Red Hat Enterprise Linux 7 可让管理员将多个网络接口绑定在一起作为单一、绑定的频道。频道绑定可让两个或多个接口作为一个接口动作，同时增加带宽，并提供冗余。



警告

不支持对不使用网络交换机的直接线缆连接进行绑定操作。如果没有网络交换机，在此论述的这个故障转移机制就无法工作。详情请查看 Red Hat 知识库文章 [《为什么在使用交叉线缆的直接连接中不支持绑定？》](#)



注意

active-backup、balance-tlb 和 balance-alb 模式不需要交换机的任何特殊配置。其他绑定模式需要配置交换机以便整合链接。例如：Cisco 交换机需要在模式 0、2 和 3 中使用 EtherChannel，但在模式 4 中需要 LACP 和 EtherChannel。有关交换机附带文档，请查看 <https://www.kernel.org/doc/Documentation/networking/bonding.txt>。

4.1. 了解主接口及从属接口的默认行为

使用 **NetworkManager** 守护进程控制绑定的从属接口时，特别是在查找出现问题时，请记住以下几点：

1. 启动主接口不会自动启动从属接口。
2. 启动从属接口总是启动主接口。
3. 停止主接口也可以停止从属接口。
4. 没有从属接口的主接口可启动静态 **IP** 连接。
5. 没有从属接口的主接口会在启动 **DHCP** 连接时等待从属接口。
6. 有 **DHCP** 连接的主接口会在添加有载波的从属接口时等待从属接口完成。
7. 有 **DHCP** 连接的主接口会在添加没有载波的从属接口时等待从属接口完成。

4.2. 使用文本用户界面 nmtui 配置绑定

可在终端窗口中使用文本用户界面工具 **nmtui** 配置绑定。请运行以下命令启动该工具：

```
~]$ nmtui
```

此时会出现文本用户界面，无效命令会显示用法信息。

可使用箭头键或按 **Tab** 在选项间前进，并按 **Shift+Tab** 后退。按 **Enter** 选择该选项。按 **Space** 键选择复选框状态。

1. 在开始菜单中选择 **编辑连接**。选择 **添加** 时会打开 **编辑连接** 页面。

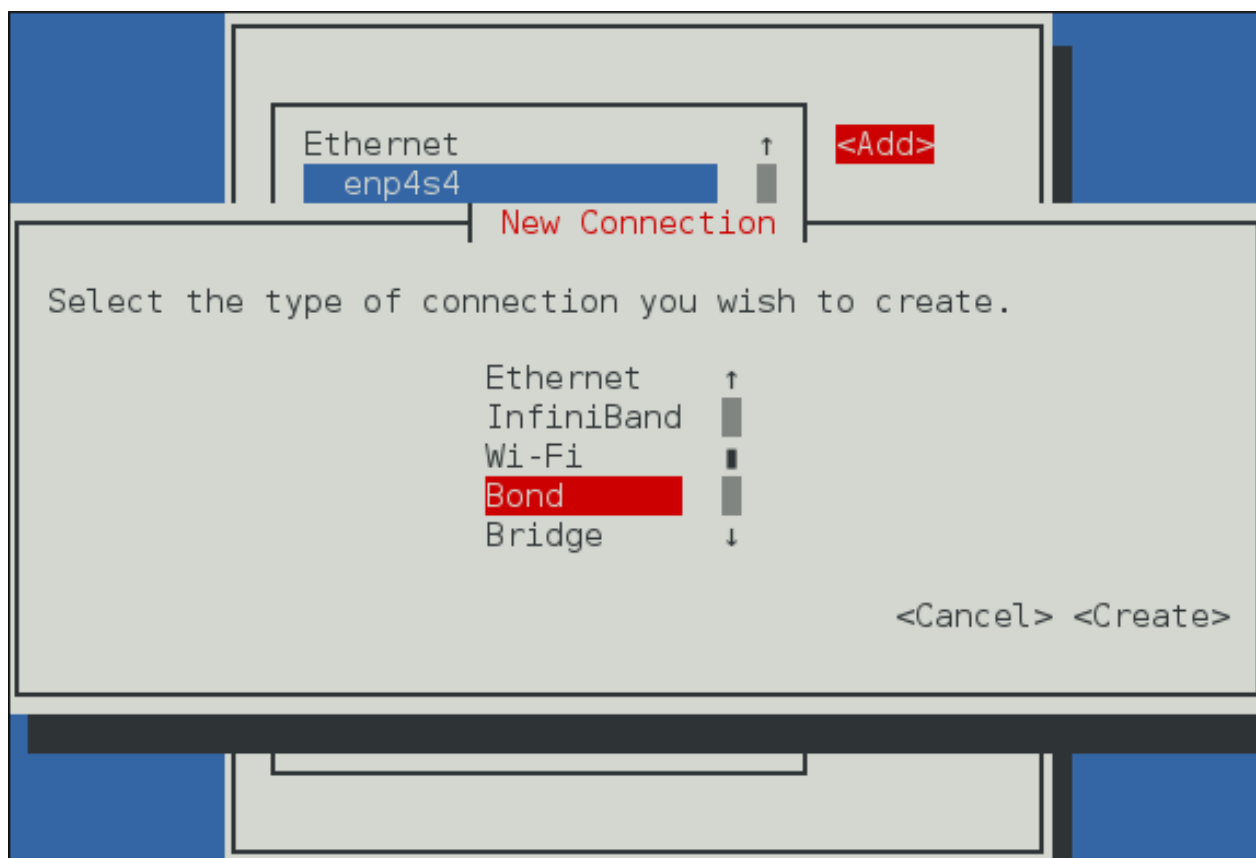


图 4.1. NetworkManager 文本用户界面的添加绑定连接菜单

2. 选择 **绑定**，然后选择 **创建** 打开绑定的 **编辑连接** 页面。

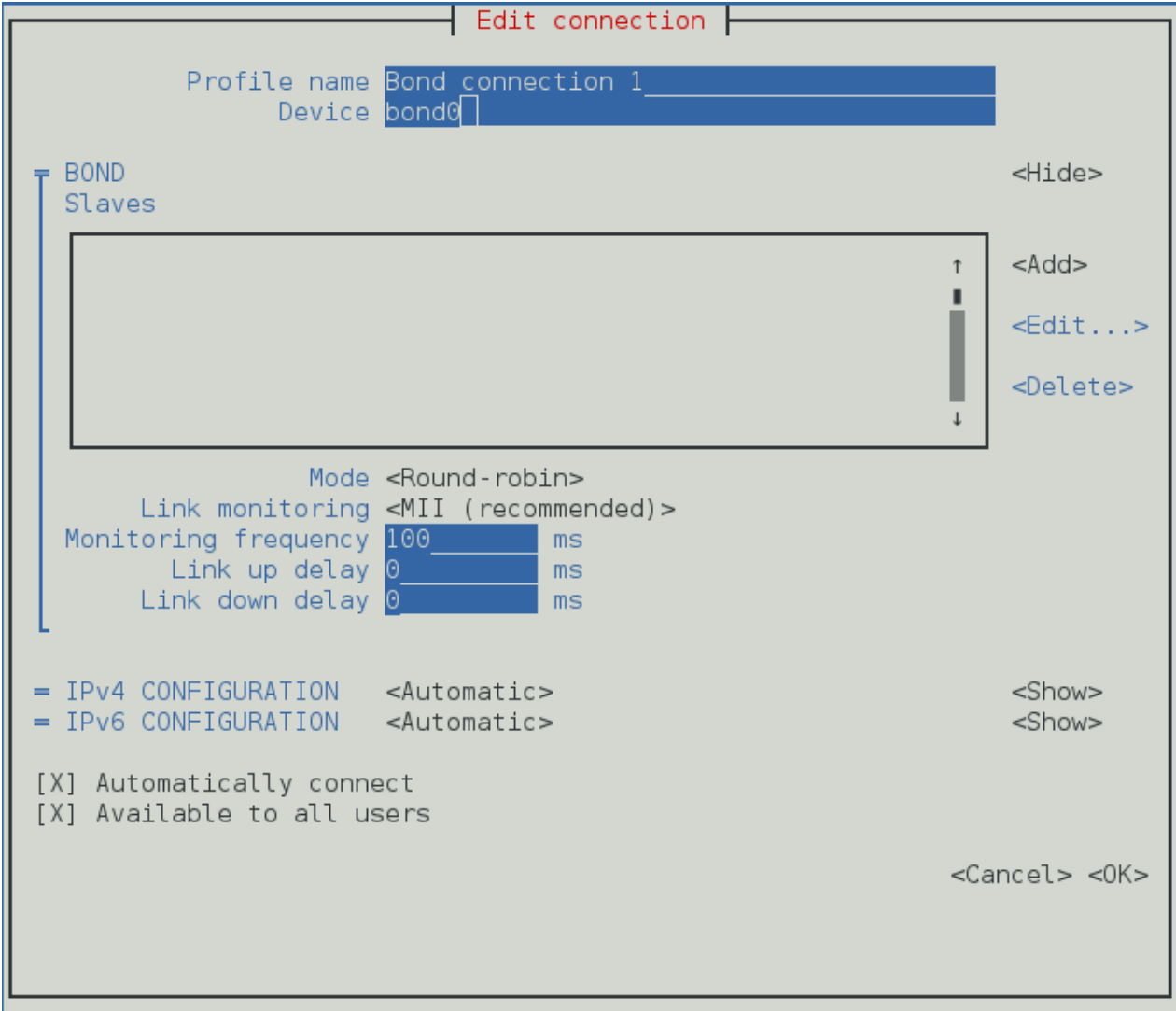


图 4.2. NetworkManager 文本用户界面的配置绑定连接菜单

- 3. 此时需要在绑定中添加从属接口。要添加从属接口，请选择 **添加** 打开 **新建连接** 页面。选择连接类型后，请点击 **创建** 按钮。

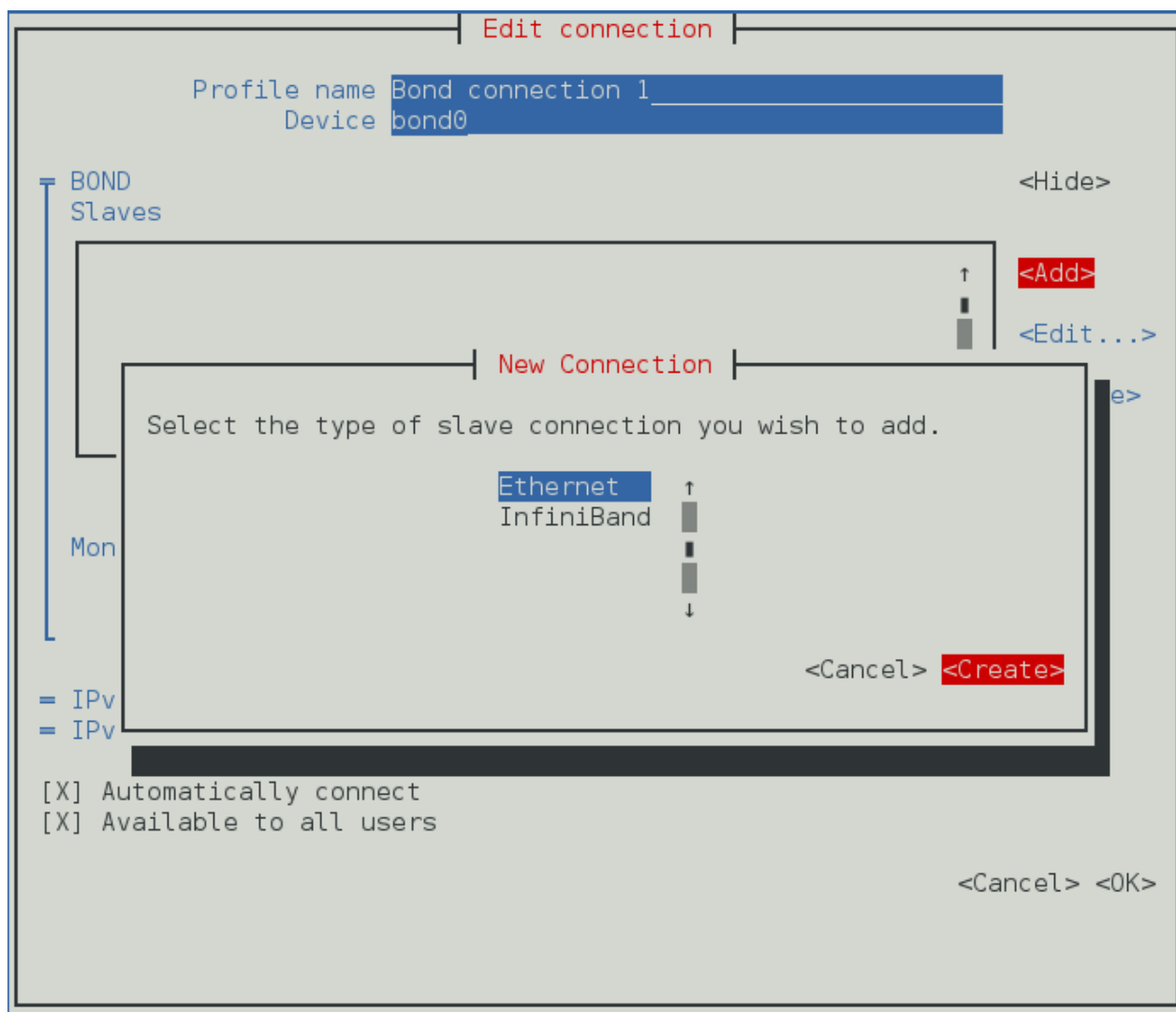


图 4.3. NetworkManager 文本用户界面的配置绑定从属连接菜单

4. 显示从属连接 **编辑连接** 页面。在 **Device** 字段输入所需从属设备名称或 MAC 地址。如有必要，选择 **以太网** 标签右侧的 **显示**，输入作为绑定的 MAC 地址的克隆 MAC 地址。选择 **确定** 按钮保存辅设备。



注意

如果没有为该设备指定 MAC 地址，则会在重新载入 **编辑连接** 窗口时自动填写 **Device** 部分，但只能在成功找到该设备时方可有此效果。

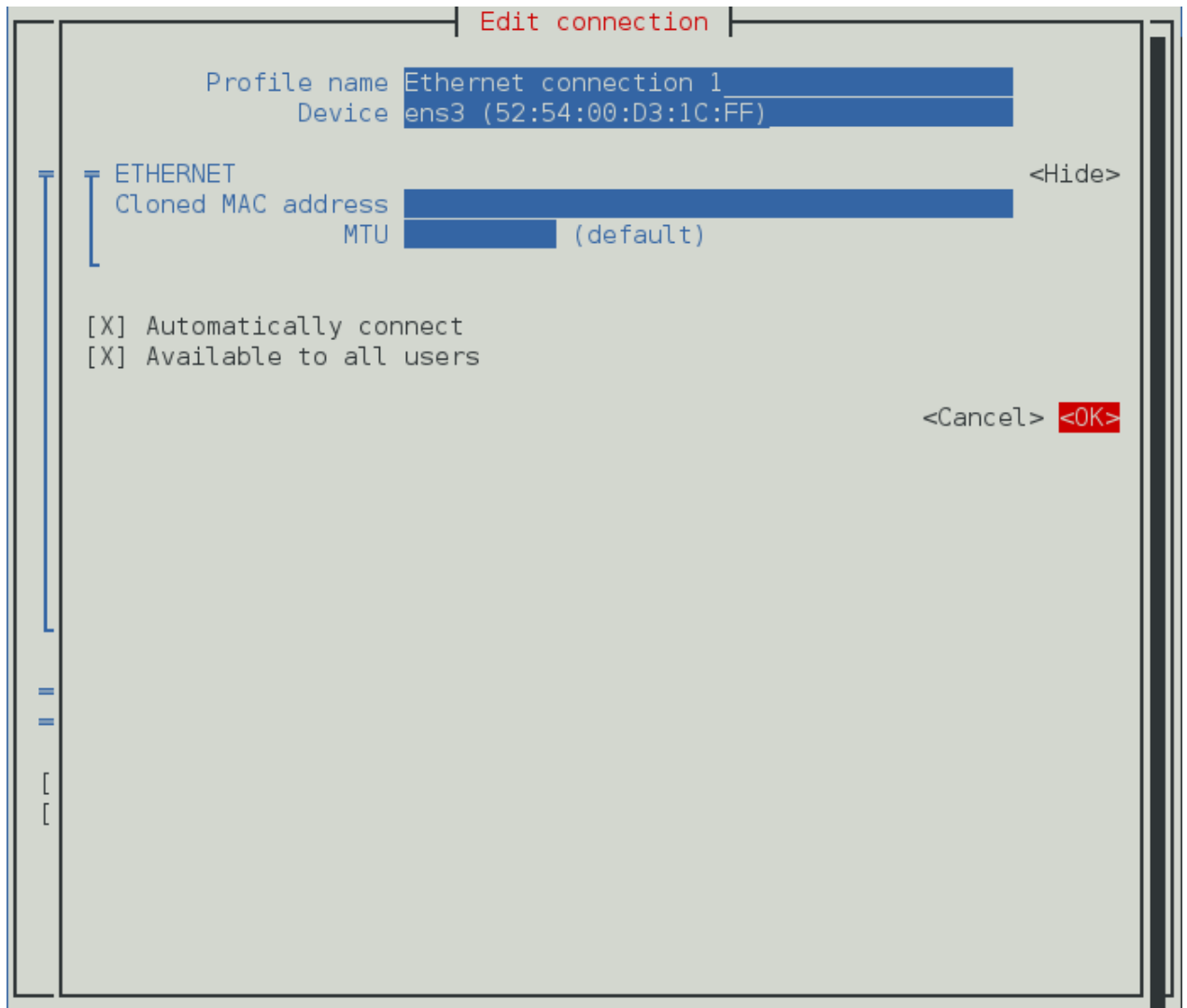


图 4.4. NetworkManager 文本用户界面的配置绑定从属连接菜单

5. **从属连接** 部分显示绑定从属连接名称。重复上述步骤添加其他从属连接。
6. 点击 **确定** 按钮前请检查并确定设置。

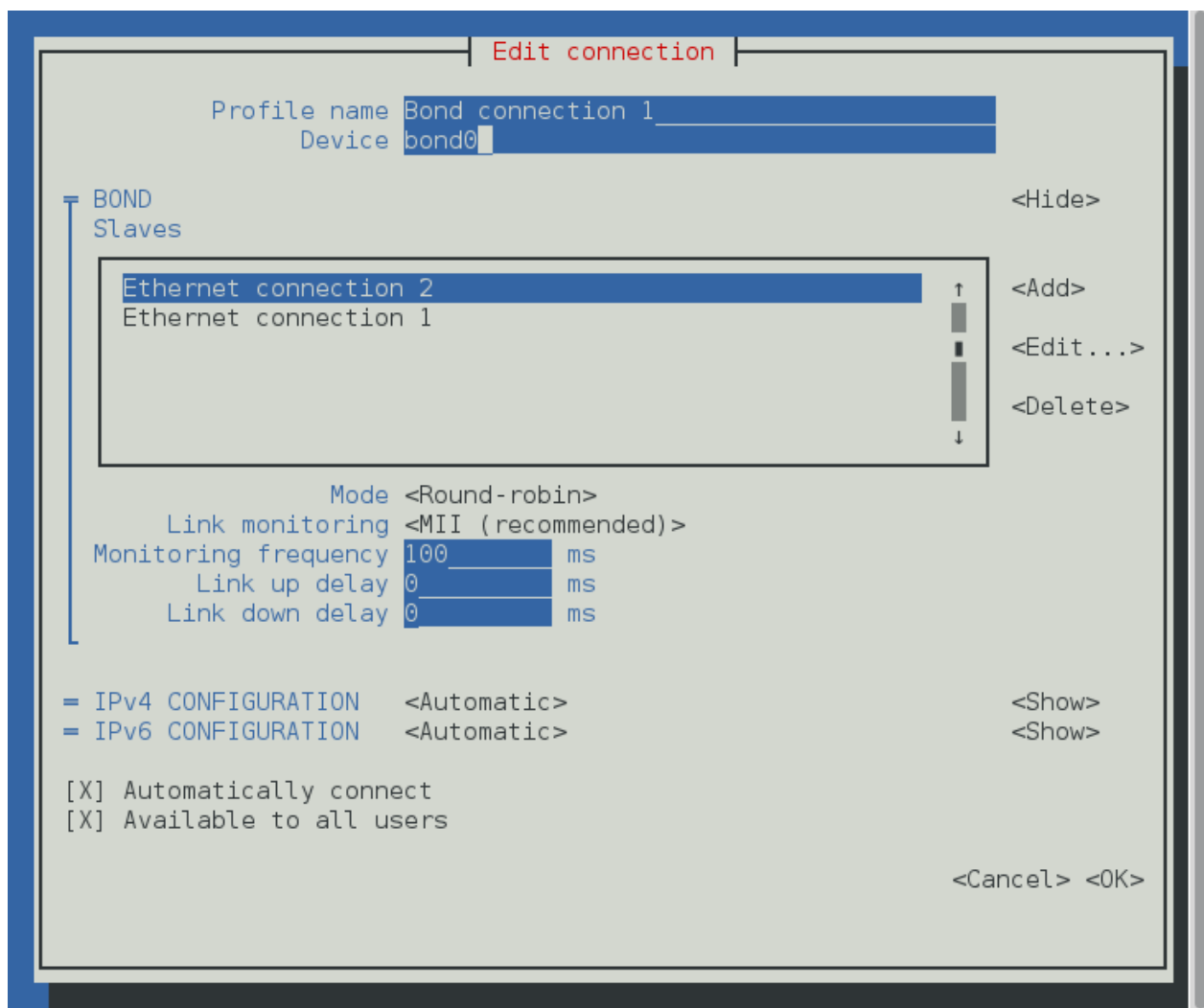


图 4.5. NetworkManager 文本用户界面中完成的绑定

有关绑定术语的定义请查看 [第 4.6.1.1 节 “配置绑定标签”](#)。

有关安装 `nmtui` 的信息请查看 [第 1.5 节 “使用文本用户界面（nmtui）进行网络配置”](#)。

4.3. 使用 NetworkManager 命令行工具 nmcli

请运行以下命令创建名为 `mybond0` 的绑定：

```
~]$ nmcli con add type bond con-name mybond0 ifname mybond0 mode active-backup
Connection 'mybond0' (9301ff97-abbc-4432-aad1-246d7faea7fb) successfully added.
```

请运行以下格式的命令添加从属接口：

```
~]$ nmcli con add type bond-slave ifname ens7 master mybond0
```

要添加其他从属接口，请重复上一个命令，在命令中使用新的接口，例如：

```
~]$ nmcli con add type bond-slave ifname ens3 master mybond0
Connection 'bond-slave-ens3-1' (50c59350-1531-45f4-ba04-33431c16e386) successfully added.
```

注：因为没有为从属接口提供 **con-name**，则该名称是接口名称加上类型构成。到目前为止，**nmcli** 只支持以太网从属接口。

要启动绑定，则必须首先启动从属接口，如下：

```
~]$ nmcli con up bond-slave-ens7
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/14)
```

```
~]$ nmcli con up bond-slave-ens3
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/15)
```

现在可使用以下方法启动绑定：

```
~]$ nmcli con up bond-mybond0
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/16)
```

有关 **nmcli** 的说明，请参考 [第 2.3 节 “使用 NetworkManager 命令行工具 nmcli”](#)。

4.4. 使用命令行接口 (CLI)

绑定是由 **bonding** 内核模块和名为 *频道绑定接口* 的特殊网络接口生成。

4.4.1. 检查是否已安装 Bonding 内核模块

在 Red Hat Enterprise Linux 7 中默认载入 **bonding** 模块。可作为 **root** 运行以下命令载入该模块：

```
~]# modprobe --first-time bonding
```

系统重启后则不会保留这个激活。有关持久载入该模块的详情请查看 [《Red Hat Enterprise Linux 7 系统管理员指南》](#)。注：使用 **BONDING_OPTS** 指令给出正确的配置文件，则会根据需要载入绑定模式，因此不需要分别载入。

请使用以下命令显示该模块的信息：

```
~]$ modinfo bonding
```

更多命令选项请查看 **modprobe(8)** man page。

4.4.2. 创建频道绑定接口

要创建频道绑定接口，请在 **/etc/sysconfig/network-scripts/** 目录中创建名为 **ifcfg-bondN** 的文件，使用接口号码替换 **N**，比如 **0**。

可根据要绑定接口类型的配置文件编写该文件的内容，比如以太网接口。最主要的区别是 **DEVICE** 指令是 **bondN**（使用接口号码替换 **N**）和 **TYPE=Bond**。此外还设置 **BONDING_MASTER=yes**。

例 4.1. ifcfg-bond0 接口配置文件示例

频道绑定接口示例。

```
DEVICE=bond0
NAME=bond0
TYPE=Bond
BONDING_MASTER=yes
IPADDR=192.168.1.1
PREFIX=24
ONBOOT=yes
BOOTPROTO=none
BONDING_OPTS="bonding parameters separated by spaces"
```

NAME 指令在 **NetworkManager** 命名连接配置文件时非常有用。ONBOOT 决定如何在引导时启动该配置文件（也就是说，如何自动连接某个设备）。



重要

必须在 **ifcfg-bondN** 接口文件的 **BONDING_OPTS="bonding parameters"** 指令中，使用以空格分开的列表指定 bonding 内核模块。请不要在 **/etc/modprobe.d/bonding.conf** 文件或弃用的 **/etc/modprobe.conf** 文件中为绑定设备指定选项。

max_bonds 参数不是具体接口的参数，且不应在使用 **BONDING_OPTS** 指令的 **ifcfg-bondN** 文件中设定，因为这个指令会让网络脚本根据需要创建绑定接口。

有关配置 bonding 模块及查看绑定参数的操作，请查看 [第 4.5 节“使用频道绑定”](#)。

4.4.3. 创建从属接口

频道绑定接口是“主”接口，同时要绑定的接口是“从属”接口。创建频道绑定接口后，必须在从属接口的配置文件中添加 **MASTER** 和 **SLAVE** 指令，以便配置要绑定在一起的接口。每个从属接口的配置文件都几乎一样。

例 4.2. 从属接口配置文件示例

例如：将两个以太网接口 **eth0** 和 **eth1** 以频道方式绑定，它们可类似如下：

```
DEVICE=ethN
NAME=bond0-slave
TYPE=Ethernet
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
```

在这个示例中，使用该接口的数字值替换 **N**。注：如果某个接口有一个以上配置文件，或配置文件中包含 **ONBOOT=yes**，则可能会产生彼此竞争，同时激活普通的 **TYPE=Ethernet** 配置文件，而不是绑定从属接口。

4.4.4. 激活频道绑定

要激活绑定，则要启动所有从属接口。请作为 **root** 运行以下命令：

```
~]# ifup ifcfg-eth0
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/7)
```

```
~]# ifup ifcfg-eth1
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/8)
```

注：如果为已经处于“up”的接口编辑接口文件，请首先将其设定为 down，如下：

```
ifdown ethN
```

。完成后，启动所有从属接口以便启动绑定（不将其设定为“down”）。

要让 **NetworkManager** 了解所做更改，请在每次进行更改后，作为 **root** 运行一个命令：

```
~]# nmcli con load /etc/sysconfig/network-scripts/ifcfg-device
```

另外，也可以重新载入所有接口：

```
~]# nmcli con reload
```

NetworkManager 的默认行为是不会意识到所进行的更改，并继续使用旧的配置数据。这是由 **NetworkManager.conf** 文件中的 **monitor-connection-files** 选项决定。有关详情请查看 **NetworkManager.conf(5)** manual page。

请运行以下命令查看绑定接口的状态：

```
~]# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
DEFAULT
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master bond0 state UP mode DEFAULT qlen 1000
    link/ether 52:54:00:e9:ce:d2 brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master bond0 state UP mode DEFAULT qlen 1000
    link/ether 52:54:00:38:a6:4c brd ff:ff:ff:ff:ff:ff
4: bond0: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP mode DEFAULT
    link/ether 52:54:00:38:a6:4c brd ff:ff:ff:ff:ff:ff
```

4.4.5. 创建多个绑定

在 Red Hat Enterprise Linux 7 中，会为每个绑定创建一个频道绑定接口，其中包括 **BONDING_OPTS** 指令。使用这个配置方法可让多个绑定设备使用不同的配置。请按照以下操作创建多个频道绑定接口：

- ✦ 创建多个 **ifcfg-bondN** 文件，这些文件中包含 **BONDING_OPTS** 指令。这个指令可让网络脚本根据需要创建绑定接口。
- ✦ 创建或编辑要绑定的现有接口配置文件，添加 **SLAVE** 指令。

✱ 使用 **MASTER** 指令工具在频道绑定接口中分配要绑定的接口，即从属接口。

例 4.3. 多 ifcfg-bondN 接口配置文件示例

以下是频道绑定接口配置文件示例：

```
DEVICE=bondN
NAME=bondN
TYPE=Bond
BONDING_MASTER=yes
IPADDR=192.168.1.1
PREFIX=24
ONBOOT=yes
BOOTPROTO=none
BONDING_OPTS="bonding parameters separated by spaces"
```

在这个示例中，使用绑定接口的号码替换 *N*。例如：要创建两个接口，则需要使用正确的 **IP** 地址创建两个配置文件 **ifcfg-bond0** 和 **ifcfg-bond1**。

使用 [例 4.2 “从属接口配置文件示例”](#) 创建要绑定的接口，并根据需要使用 **MASTER=bondN** 指令将其分配到绑定接口。例如：在上述示例中，如果需要在每个绑定中有两个接口，则两个绑定则要创建四个接口配置文件，并使用 **MASTER=bond0** 分配前两个配置文件，使用 **MASTER=bond1** 分配后两个配置文件。

4.5. 使用频道绑定

为增强性能，可调整可用模块选项确定最佳组合。特别要注意 **miimon** 或者 **arp_interval** 和 **arp_ip_target** 参数。有关可用选项列表及如何迅速决定要绑定的最佳接口详情，请查看 [第 4.5.1 节 “Bonding 模块指令”](#)。

4.5.1. Bonding 模块指令

最好是在将绑定接口添加到绑定接口配置文件（例如：**ifcfg-bond0**）的 **BONDING_OPTS="bonding parameters"** 指令前，测试最适合的频道 bonding 模块参数。修改 **sysfs** 文件系统中的文件即可在不载入（及重新载入）bonding 模块的情况下配置绑定接口参数。

sysfs 是一个虚拟文件系统，将内核对象视为目录、文件及符号链接。可使用 **sysfs** 查询内核对象信息，也可以通过使用常规文件系统命令处理那些对象。**sysfs** 虚拟文件系统是挂载到 **/sys/** 目录下。所有绑定接口都可以通过与 **/sys/class/net/** 目录的文件互动和操作动态进行配置。

为确定绑定接口的最佳参数，可创建一个绑定接口文件，比如 **ifcfg-bond0**，方法如 [第 4.4.2 节 “创建频道绑定接口”](#) 所述。在每个绑定到 **bond0** 的接口的配置文件中插入 **SLAVE=yes** 和 **MASTER=bond0** 指令。完成后，即可测试这些参数。

首先，请作为 **root** 运行 **ifup bondN** 启动您创建的绑定：

```
~]# ifup bond0
```

如果已正确创建 **ifcfg-bond0** 绑定接口文件，则可以作为 **root** 运行 **ip link show** 命令，查看该命令输出结果中列出的 **bond0**：

```
~]# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
```

```

DEFAULT
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master bond0 state UP mode DEFAULT qlen 1000
    link/ether 52:54:00:e9:ce:d2 brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master bond0 state UP mode DEFAULT qlen 1000
    link/ether 52:54:00:38:a6:4c brd ff:ff:ff:ff:ff:ff
4: bond0: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP mode DEFAULT
    link/ether 52:54:00:38:a6:4c brd ff:ff:ff:ff:ff:ff

```

要查看所有现有绑定（包括未启动的绑定），请运行：

```

~]$ cat /sys/class/net/bonding_masters
bond0

```

可处理 `/sys/class/net/bondN/bonding/` 目录中的各个文件配置每个绑定。首先，您要配置的绑定必须处于 down 状态：

```

~]# ifdown bond0

```

例如：要以 1 秒为间隔启用 bond0 中的 MII 监控，请作为 **root** 运行：

```

~]# echo 1000 > /sys/class/net/bond0/bonding/miimon

```

要为 **balance-alb** 模式配置 bond0，请运行：

```

~]# echo 6 > /sys/class/net/bond0/bonding/mode

```

.....或者使用该模式名称：

```

~]# echo balance-alb > /sys/class/net/bond0/bonding/mode

```

配置有问题绑定的选项后，可运行 **ifup bondN** 启动并测试该连接。如果决定要更改选项，则需要先断开该接口，使用 **sysfs** 修改参数，然后启动接口，并再次测试。

确定绑定连接的最佳参数后，为要配置的绑定接口在 `/etc/sysconfig/network-scripts/ifcfg-bondN` 文件的 **BONDING_OPTS=** 指令中以空格分开的方式添加那些参数。无论何时连接该接口时（例如：若设置 **ONBOOT=yes** 指令，则在系统引导序列中），**BONDING_OPTS** 中指定的绑定选项即可生效。

以下列表提供了很多常见的频道绑定参数名称及其功能说明。有关详情，请查看 **modinfo bonding** 输出中每个 **parm** 的概述，或查看 <https://www.kernel.org/doc/Documentation/networking/bonding.txt> 了解更详细的内容。

绑定接口参数

ad_select=value

指定要使用的 802.3ad 聚合选择逻辑，可能值为：

- ✱ **stable** 或者 **0** — 默认设置。由最大聚合带宽选择的活跃聚合器。只有全部从属聚合器失效，或活跃聚合器没有从属聚合器时，才会重新选择活跃聚合器。

✧ **bandwidth** 或者 **1** — 活动聚合器由最大聚合带宽选择。出现在以下情况下会重新选择：

- 在绑定中添加或删除从属；
- 任意从属链接发生变化；
- 任意与 802.3ad 有关的状态变化；
- 绑定的管理状态改为 up。

✧ **count** 或者 **2** — 活动聚合器由号码最大从属连接选择。重新选择的条件与上述 **bandwidth** 一致。

bandwidth 和 **count** 选择策略允许活动聚合器部分失灵时进行 802.3ad 聚合故障转移。这样可保证聚合器高度可用，即 **bandwidth** 或从属连接数一直保持活动。

arp_interval=time_in_milliseconds

以毫秒为单位指定 ARP 监控的频繁度。



重要

关键是要指定 **arp_interval** 或 **arp_ip_target** 参数，或者指定 **miimon** 参数。否则会在链接失败时使网络性能降级。

如果在 **mode=0** 或者 **mode=2**（两种负载平衡模式）中使用这个设置，则必须配置网络交换机，以便使用网卡平均发送数据包。有关如何完成此操作的详情，请查看 <https://www.kernel.org/doc/Documentation/networking/bonding.txt>。

默认将这个数值设定为 **0**，即禁用该功能。

arp_ip_target=ip_address[, ip_address_2,...ip_address_16]

启用 **arp_interval** 参数后，指定 ARP 请求的目标 IP 地址。在使用逗号分开的列表中最多可指定 16 个 IP 地址。

arp_validate=value

验证 ARP 探测的源/分配，默认为 **none**。其他值为 **active**、**backup** 和 **all**。

downdelay=time_in_milliseconds

以毫秒为单位指定从链接失败到禁用该链接前要等待的时间。该值必须是 **miimon** 参数中的多个数值。默认将其设定为 **0**，即禁用该功能。

fail_over_mac=value

指定 active-backup 模式是否应该将所有从属连接设定为使用同一 MAC 地址作为 enslavement（传统行为），或在启用时根据所选策略执行绑定 MAC 地址的特殊处理。可能值为：

- ✧ **none** 或 **0** -- 默认设置。这个设置禁用 **fail_over_mac**，并造成在 enslavement 时间内将所有 active-backup 绑定的从属连接绑定到同一 MAC 地址。
- ✧ **active** 或者 **1** — “active” **fail_over_mac** 策略表示绑定的 MAC 地址应永远是目前活动从属连接的 MAC 地址。从属连接的 MAC 地址不会更改，但在故障转移过程中会更改绑定的 MAC 地址。

这个策略对永远无法更改其 MAC 地址的设备，或拒绝使用其自主源 MAC 地址传入多播的设备（影响 ARP 监控）很有帮助。这个策略的缺点是该网络中的每个设备必须通过免费 ARP 更新，这与切换 snoop 传入流量以便更新其 ARP 表的常规方法相反。如果免费 ARP 链接丢失，则可能破坏通讯。

使用这个策略同时采用 MII 监控时，在可真正传输并接受数据前就声明链接处于 up 状态的设备很可能会丢失免费 ARP，并可能需要设置正确的呼叫建立延迟（updelay）。

- **follow** 或者 **2** — “follow” **fail_over_mac** 策略可保证正常选择绑定的 MAC 地址（通常是绑定的第一从属链接的 MAC 地址）。但第二从属连接及之后的从属连接不适用这个 MAC 地址，虽然他们是备份角色；故障转移时从属连接是使用绑定的 MAC 地址编程（之前活动的从属连接接收新激活的从属 MAC 地址）。

这个策略对使用同一 MAC 地址编程时变得混乱或发生性能损失的多端口设备有帮助。

lacp_rate=value

指定链接伙伴应使用 802.3ad 模式传输 LACPDU 的速率：

- **slow** 或者 **0** — 默认设置。这是让链接伙伴每 30 秒传输一次 LACPDU。
- **fast** 或者 **1** — 默认设置。这是让链接伙伴每 1 秒传输一次 LACPDU。

miimon=time_in_milliseconds

以毫秒为单位指定 MII 链接监控的频率。这在需要高可用性时有用，因为 MII 是用来验证网卡是否激活。要验证某个支持 MII 工具的具体网卡的驱动程序，请作为 root 运行以下命令：

```
~]# ethtool interface_name | grep "Link detected:"
```

在这个命令中使用设备接口（比如 **eth0**），而不是绑定接口替换 *interface_name*。如果支持 MII，则该命令会返回：

```
Link detected: yes
```

如果为高可用性使用绑定的接口，则每个网卡的模块都必须支持 MII。将该值设定为 **0**（默认设置）即关闭此功能。配置这个设定时，最好从 **100** 开始。



重要

关键是要指定 **arp_interval** 或 **arp_ip_target** 参数，或者指定 **miimon** 参数。否则会在链接失败时使网络性能降级。

mode=value

允许您指定绑定的策略。*value* 可为以下之一：

- **balance-rr** 或者 **0** — 为容错及负载平衡设定轮询机制。从第一个可用的绑定从属接口开始按顺序接收和发送传输数据。
- **active-backup** 或者 **1** — 为容错设定 active-backup 策略。通过第一个可用的绑定从属接口接收和发送传输文件。只有在活动的绑定从属接口失败时才使用其他绑定从属接口。

- **balance-xor** 或者 **2** — 只根据所选哈希策略传输数据。默认为使用源的 XOR 和目标 MAC 地址与从属接口数的余数相乘生成哈希。在这个模式中，指向具体对等接口的模式流量总是使用同一接口发送。因为目标是由 MAC 地址决定，因此这个方法最适合相同链接或本地网络的对等接口流量。如果流量必须通过单一路由器，那么这个流量平衡模式将是次选模式。
- **broadcast** 或者 **3** — 为容错设定广播策略。可在所有从属接口中传输所有数据。
- **802.3ad** 或者 **4** — 设定 IEEE 802.3ad 动态链接聚合策略。创建一个共享同一速度和双工设置的聚合组。在所有活跃聚合器中传输和接受数据。需要兼容 802.3ad 的交换机。
- **balance-tlb** 或者 **5** — 为容错及负载均衡设定传输负载均衡（TLB）策略。传出流量会根据每个从属接口的当前负载分布。传入流量由当前从属接口接收。如果接收数据从属接口失败，另一个从属接口会接管失败从属接口的 MAC 地址。这个模式只适用于内核绑定模式了解的本地地址，因此无法在桥接后的虚拟机中使用。
- **balance-alb** 或者 **6** — 为容错及负载均衡设定自适应负载均衡（ALB）策略，包括用于 IPv4 流量的传输及接收负载均衡。使用 ARP 协商获得接收负载均衡。这个模式只适用于内核 binding 模块了解的本地地址，因此无法在桥接后的虚拟机中使用。

primary=interface_name

指定主设备的接口名称，比如 **eth0**。主设备是要使用的第一个绑定接口，且在其失败前不会放弃。当绑定接口的一个网卡较快并可处理较大负载时，这个设置特别有帮助。

只有在绑定接口处于 **active-backup** 模式时这个设置才有用。详情请查看 <https://www.kernel.org/doc/Documentation/networking/bonding.txt>。

primary_reselect=value

为主从属接口指定重新选择策略。这会影响在活动从属接口失败或恢复主从属接口时，将主从属接口选为活动从属接口的方式。这个参数是用来防止主从属接口和其他从属接口之间的反转。可能值为：

- **always** 或者 **0**（默认） — 无论何时只要有备份时，主从属接口成为活动从属接口。
- **better** 或者 **1** — 如果主从属接口的速度及双工由于当前活动从属接口的速度及双工，有备份时，主从属接口成为活动从属接口。
- **failure** 或者 **2** — 只有当前活动从属接口失败且主从属接口处于 up 状态时，主从属接口方成为活动从属接口。

在两种情况下会忽略 **primary_reselect** 设置：

- 如果没有从属接口处于活动状态，第一个要恢复的从属接口则成为活动从属接口。
- 成为从属接口后，主从属接口总是活动从属接口。

使用 **sysfs** 更改 **primary_reselect** 策略后，会立即根据新策略选择最佳活动从属接口。这可能或可能不会造成活动从属接口变化，要看具体情况。

resend_igmp=range

指定故障转移事件后要进行的 IGMP 成员报告数。故障转移后会立即提交一个报告，之后会每隔 200 毫秒发送数据包。

有效值范围为 **0** 到 **255**，默认值为 **1**。数值 **0** 可防止发出的 IGMP 成员报告响应故障转移事件。

这个选项在绑定模式 **balance-rr** (mode 0)、**active-backup** (mode 1)、**balance-tlb** (mode 5) 和 **balance-alb** (mode 6) 中有帮助，这样可让故障转移将 IGMP 流量从一个从属接口转移到另一个从属接口。因此必须刷新 IGMP 报告以便让交换机将传入 IGMP 流量通过新选择的从属接口转发。

updelay=time_in_milliseconds

以毫秒为单位指定启用某个链接前要等待的时间。该数值必须是在 **miimon** 参数值指定值的倍数。默认设定为 **0**，即禁用该参数。

use_carrier=number

指定 **miimon** 是否应该使用 MII/ETHTOOL `ioctl`s 或者 `netif_carrier_ok()` 来决定该链接状态。`netif_carrier_ok()` 功能以来设备驱动程序维持其 `netif_carrier_on/off` 状态，大多数设备驱动程序支持此功能。

MII/ETHTOOL `ioctl`s 工具利用内核中的弃用调用序列。但这还是可以配置的，以防您的设备驱动程序不支持 `netif_carrier_on/off`。

有效数值是：

- ✱ **1** — 默认设置。启用 `netif_carrier_ok()` 功能。
- ✱ **0** — 启用 MII/ETHTOOL `ioctl`s 功能。



注意

如果绑定接口坚持链接应处于 up 状态（即使它不应处于该状态），原因可能是因为您的网络设备驱动程序不支持 `netif_carrier_on/off`。

xmit_hash_policy=value

选择 **balance-xor** 和 **802.3ad** 模式中用来选择从属接口的传输哈希策略。可能值为：

- ✱ **0** 或者 **layer2** — 默认设置。这个参数使用硬件 MAC 地址的 XOR 生成哈希。使用的公式为：

$$(source_MAC_address \text{ XOR } destination_MAC) \text{ MODULO } slave_count$$

这个算法会将所有流量放到同一从属接口的特定对等网络中，且该网络兼容 802.3ad。

- ✱ **1** 或者 **layer3+4** — 使用上层协议信息（可用时）生成该哈希。这样可让流量进入特定对等网络并跨多个从属接口，虽然单一连接无法跨多个从属接口。

未碎片化 TCP 及 UDP 数据包使用的公式为：

$$((source_port \text{ XOR } dest_port) \text{ XOR } ((source_IP \text{ XOR } dest_IP) \text{ AND } 0xffff)) \text{ MODULO } slave_count$$

在碎片化 TCP 或 UDP 以及所有其他 IP 协议流量中会省略源及目标端口信息。非 IP 流量的公式与 **layer2** 传输哈希策略相同。

这个策略旨在模拟某些交换机的行为，特别是附带 PFC2 以及一些 Foundry 和 IBM 产品的 Cisco 交换机。

这个策略使用的算法不兼容 802.3ad。

- ✱ **2 或者 layer2+3** — 联合使用 layer2 和 layer3 协议信息生成该哈希。

使用硬件 MAC 地址及 **IP** 地址生成该哈希。这个公式为：

```
((source_IP XOR dest_IP) AND 0xffff) XOR
 ( source_MAC XOR destination_MAC ))
MODULO slave_count
```

这个算法将所有流量放到同一从属接口的特定对等网络中。非 **IP** 流量的公式与 layer2 传输哈希策略相同。

这个策略旨在提供比单独使用 layer2 更平衡的流量分配，特别是在需要 layer3 网关设备的环境，以便连接到大多数目标。

这个算法兼容 802.3ad。

4.6. 使用 GUI 创建绑定连接

您可以使用 GNOME **control-center** 工具程序让 **NetworkManager** 为两个或多个有限或 InfiniBand 连接创建绑定。不一定要首先创建需要绑定的连接，可将其作为配置该绑定过程的一部分。您必须有该接口的 MAC 地址方可完成配置过程。

4.6.1. 建立绑定连接

过程 4.1. 添加新绑定连接

按照以下步骤创建新绑定连接。

1. 按 **Super** 键进入活动概述页面，输入 **control network** 然后按 **Enter**。此时会出现 **Network** 设置工具，详情请查看 [第 2.5 节“在 GNOME 图形用户界面中使用 NetworkManager”](#)。
2. 点击加号打开选择列表。选择 **绑定**。此时会出现 **编辑绑定连接 1** 窗口。

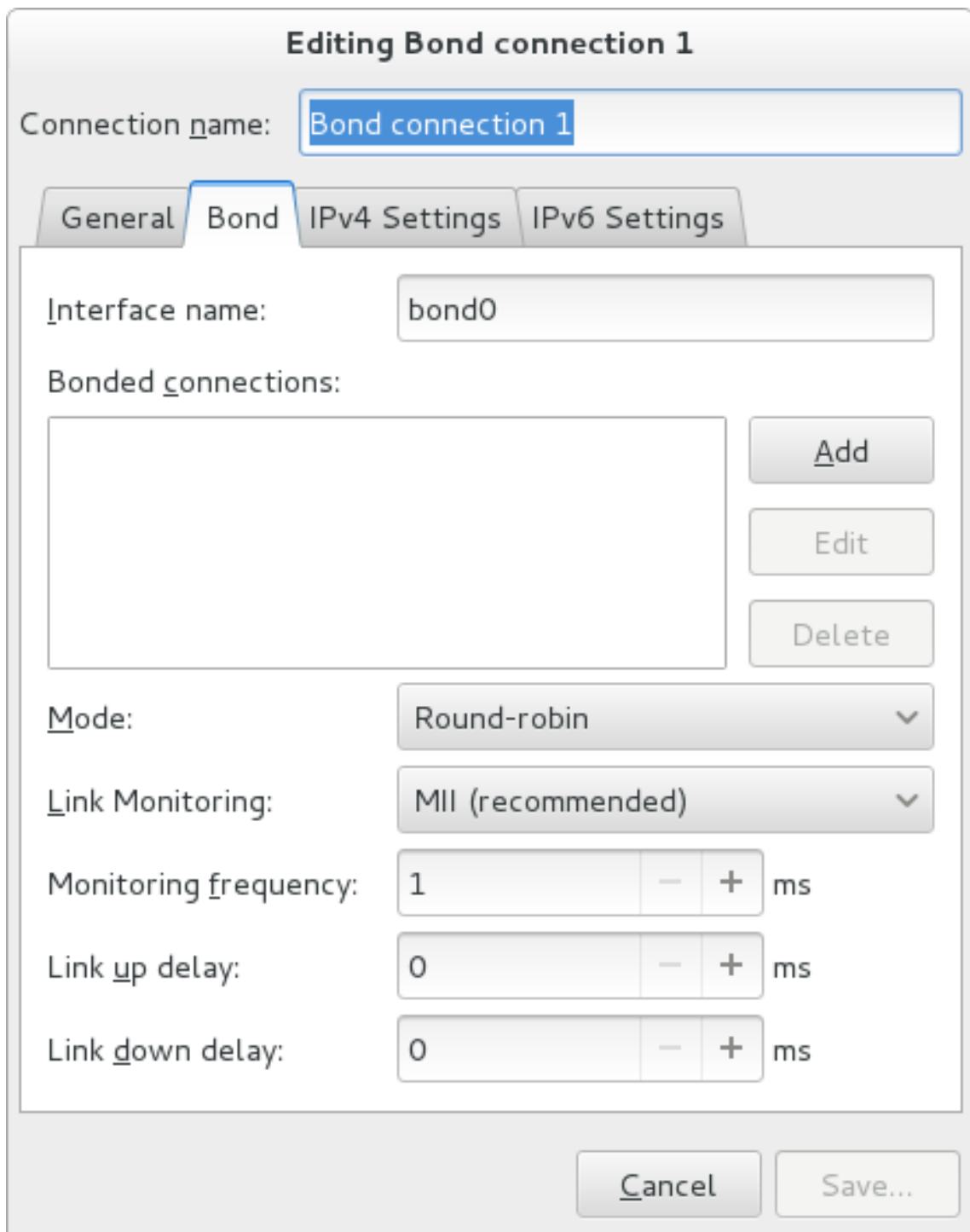


图 4.6. NetworkManager 图形用户界面的添加绑定菜单

3. 在 **绑定** 标签中点击 **添加** 并选择在绑定连接中要使用的接口类型。点击 **创建** 按钮。注：只有在创建第一个从属接口时才会出现选从属类型的对话框；之后会自动在所有从属接口中使用同一类型。
4. 此时会出现 **编辑 bond0 slave 1** 窗口。使用 **设备 MAC 地址** 下拉菜单选择要绑定接口的 MAC 地址。第一个从属 MAC 地址会作为绑定接口的 MAC 地址使用。必要时请输入作为绑定 MAC 地址使用的克隆 MAC 地址。请点击 **保存** 按钮。

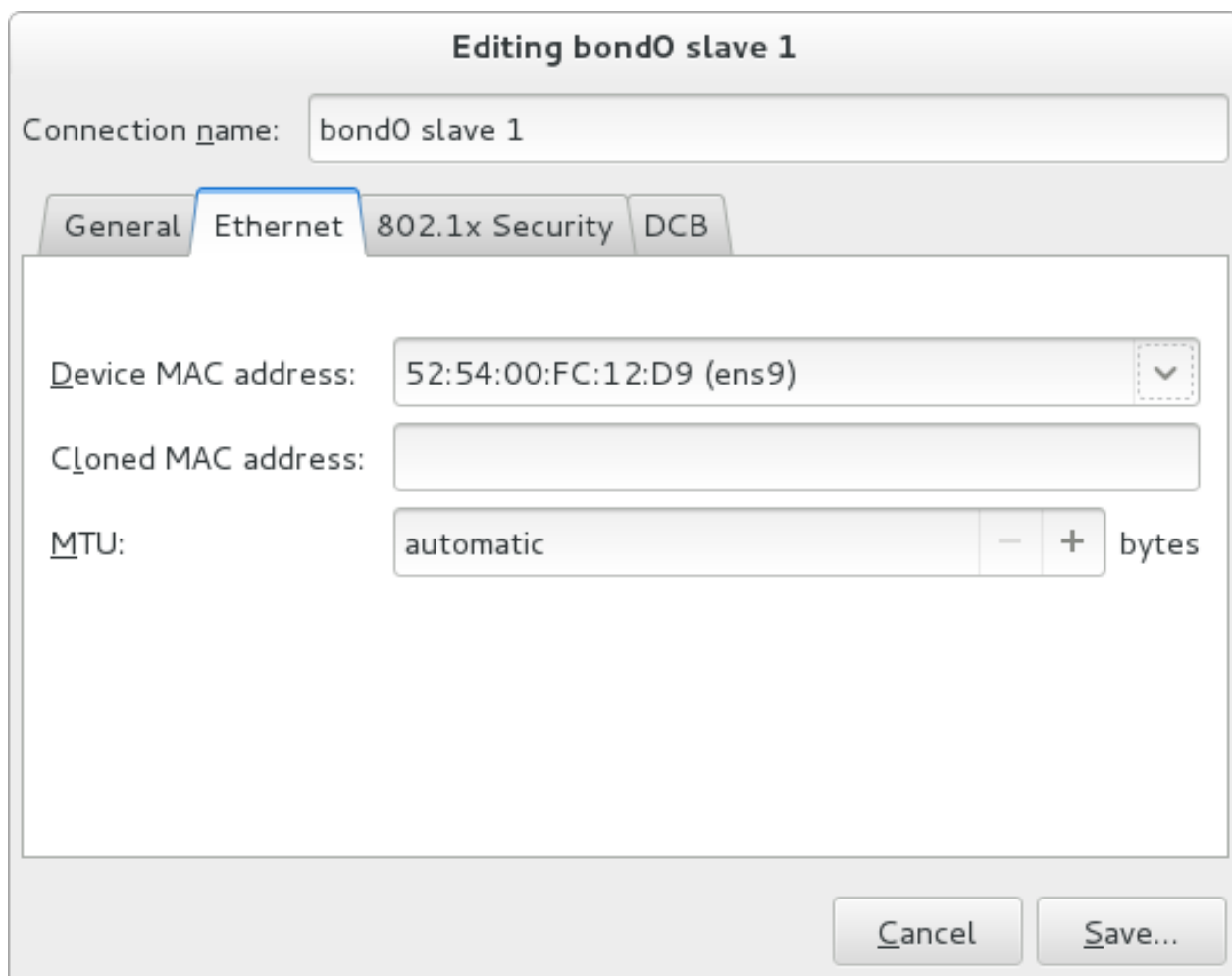


图 4.7. NetworkManager 文本用户界面的添加绑定连接菜单

5. 在 **绑定的连接** 窗口中会出现绑定的从属。点击 **添加** 按钮添加其他从属连接。
6. 检查并确定设置，然后点击 **保存** 按钮。
7. 请参考 [第 4.6.1.1 节 “配置绑定标签”](#) 编辑具体绑定设置。

过程 4.2. 编辑现有绑定连接

按照这些步骤编辑现有绑定连接。

1. 按 **Super** 键进入活动概述，输入 **control network**，然后按 **Enter** 键。此时会出现 **Network** 设置工具。
2. 选择要编辑的连接，并点击 **选项** 按钮。
3. 选择 **常规** 标签。
4. 配置连接名称、自动连接行为及可用性设置。

编辑 对话框中的五项设置适用于所有连接类型，请查看 **常规** 标签：

- ✱ **连接名称** — 为网络连接输入描述性名称。这个名称可用于在 **网络** 窗口中列出这个连接。
- ✱ **网络可用时自动连接到该网络** — 如果要想 **NetworkManager** 在这个连接可用时自动与之连接，则请选择这个复选框。详情请查看 [第 2.5.3 节 “自动连接到网络”](#)。
- ✱ **所有用户都可以连接到这个网络** — 要创建可用于系统中所有用户的连接，请选择这个复选框。详

情请查看 [第 2.5.4 节 “系统范围及专用连接配置文件”](#)。

- ✎ **使用此连接时自动连接到 VPN** — 如果要让 **NetworkManager** 在可用时自动连接到 VPN 连接，请选择正规复选框。请在下拉菜单中选择 VPN。
- ✎ **防火墙区** — 请在下拉菜单中选择防火墙区。有关防火墙区的详情，请查看 [《Red Hat Enterprise Linux 7 安全指南》](#)。

5. 请参考 [第 4.6.1.1 节 “配置绑定标签”](#) 编辑具体绑定设置。

保存新的（或修改的）连接并做进一步配置

完成编辑绑定连接后，请点击 **保存** 按钮保存自定义配置。如果编辑配置文件时该文件正在使用，则需要重启连接方可让 **NetworkManager** 以能够用所有更改。如果该配置文件处于 OFF 状态，则请将其设定为 ON，或者在网络连接图标菜单中选择它。有关使用新的或更改连接的详情，请查看 [第 2.5.1 节 “使用 GUI 连接到网络”](#)。

若要对现有连接做进一步的配置，请在 **网络** 窗口中选中该连接，并点击 **选项 返回 编辑** 对话框。

然后配置：

- ✎ 该连接的 **IPv4** 设置，点击 **IPv4 设置** 标签执行 [第 2.5.10.4 节 “配置 IPv4 设置”](#)；或者，
- ✎ 若要为该连接进行 **IPv6** 设置，请点击 **IPv6 设置** 标签并执行 [第 2.5.10.5 节 “配置 IPv6 设置”](#)

保存后，就会在网络设置工具中显示该绑定及其所有从属连接。

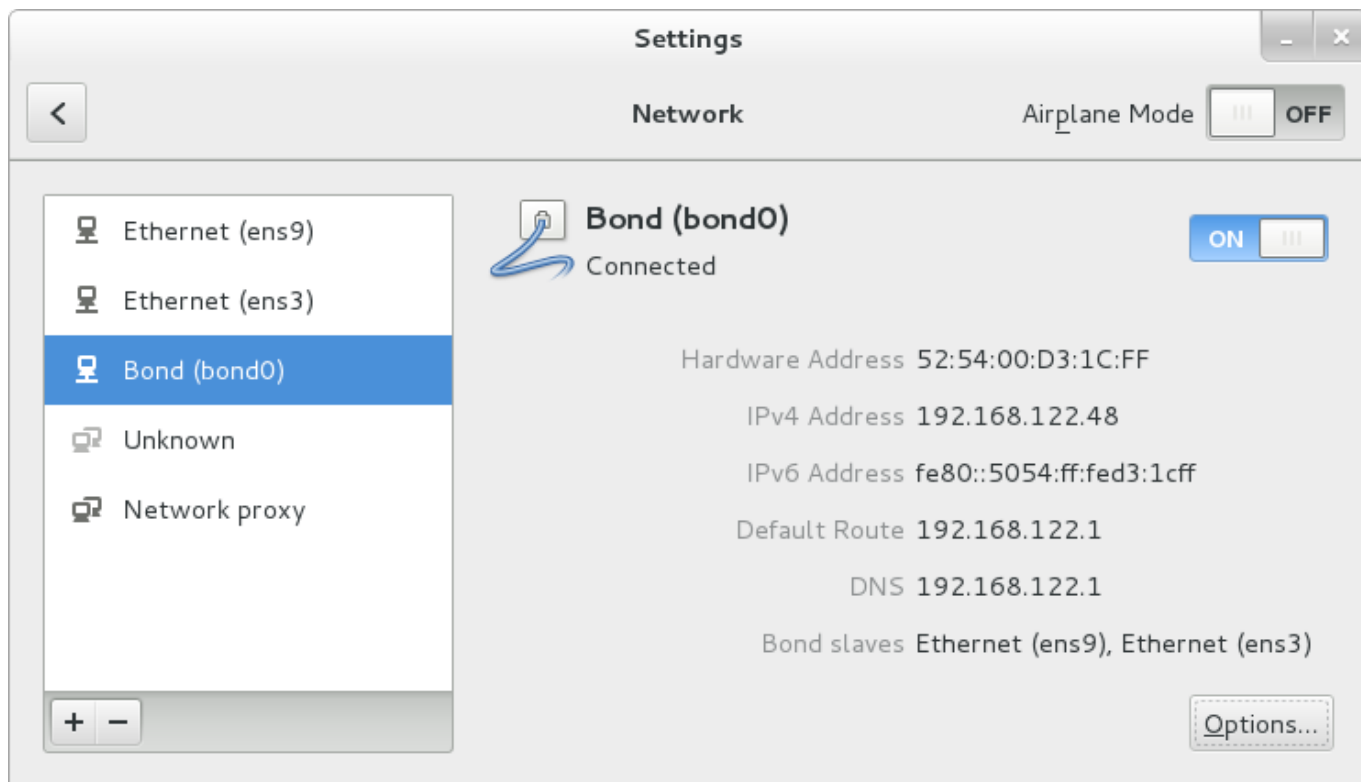


图 4.8. 附带绑定的 NetworkManager 图形用户界面

4.6.1.1. 配置绑定标签

如果已添加新绑定连接（步骤请参考 [过程 4.1, “添加新绑定连接”](#)），则可以编辑 **绑定** 标签，设定要和使用的负载共享模式及链接监控类型，以便探测从属连接失败。

模式

用来在构成绑定的从属连接间共享流量的模式。默认为 **轮询**。可使用下拉菜单列表工具选择其他负载共享模式，比如 **802.3ad**。

链接监控

监控从属输送网络流量能力的方法。

可在 **模式** 下拉菜单中选择以下负载共享模式：

轮询

为容错及负载平衡设定轮询策略。传输是从第一个可用绑定从属接口开始按顺序接收和发送。如果没有附加交换机配置，这个模式可能无法在桥接后的虚拟机中使用。

Active backup

为容错设定 active-backup 策略。传输时通过第一个可用绑定从属接口接收和发送。只有在活动绑定从属接口失败时才会使用另一个绑定从属接口。注：这是 InfiniBand 设备绑定的唯一可用模式。

XOR

设定 XOR（排他）策略。根据所选哈希策略传输。默认为将源及目标 MAC 地址与从属接口号模数生成哈希。在这个模式中，通向具体对等接口的流量永远使用同一接口发送。因为目标是由 MAC 地址是决定，所以这个方法是同一链接或本地网络对等加快的最佳方法。如果流量是通过单一路由器，则这个流量平衡模式是次优的。

广播

为容错设定广播策略。所有传输都是通过从属接口传输。若未另外配置交换机，则这个模式可能无法在使用虚拟机的桥接中使用。

802.3ad

设定 IEEE **802.3ad** 动态链接聚合成策略。创建共享同样速度及双工设置的聚合组。在活动聚合器的所有从属接口中传输和接收所有从属。需要兼容 **802.3ad** 的网络交换机。

适配器传输负载平衡

为容错及负载平衡设定适配器传输负载平衡（TLB）策略。传出流量是根据每个从属接口的当前负载分配。如果接收从属接口失败，另一个从属接口会接管失败从属接口的 MAC 地址。这个模式只适用于内核 binding 模块了解的本地地址，因此无法在使用虚拟机的桥接中使用。

自适应负载平衡

为容错及负载平衡设置自适应负载平衡（ALB）策略，包括 **IPv4** 流量的传输及接收负载平衡，通过 **ARP** 协商获得接收负载平衡。这个模式只适用于内核绑定模式了解的本地地址，因此无法在桥接后的虚拟机中使用。

可在 **链接监控** 下拉菜单列表中选择链接监控类型。最好是可以测试最适合您绑定接口的频道 bonding 模块参数。

MII（介质无关接口）

监控该接口载波状态。可通过查询该驱动程序，即直接查询 MII 注册表，或者使用 **ethtool** 查询该设备完成。这里有三个选项可用：

监控频率

查询该驱动程序或 MII 注册表的时间间隔（单位：毫秒）。

链接启动延迟

以毫秒为单位设定尝试使用已报告为 up 状态的链接。如果在将该连接报告为“up”状态后，紧接着丢失一些免费 ARP 请求，则可以使用这个延迟。这种情况可能会在交换机初始化过程中发生。

链接关闭延迟

以毫秒为单位设定在将之前活动连接报告为“down”后，等待多长时间再更换为另一个链接。如果连接的交换机需要相当长的时间改为备用模式时可使用这个延迟。

ARP

地址解析协议（ARP）是用来探测一个或多个对等连接，以便决定链接层连接的工作情况。这与提供传输时间及其最后接收时间的设备驱动程序不同。

有两个选项可用：

监控频率

发送 ARP 请求的时间间隔，单位为毫秒。

ARP 目标

以逗号分开，向其发送 ARP 请求的 IP 地址列表。

4.7. 其他资料

以下信息资源提供有关网络绑定的附加信息。

4.7.1. 已安装文档

- ✧ **nmcli(1)** man page — 描述 **NetworkManager** 的命令行工具。
- ✧ **nmcli-examples(5)** man page — 提供 **nmcli** 命令示例。
- ✧ **nm-settings(5)** man page — 描述 **NetworkManager** 连接的设置及参数。

4.7.2. 在线文档

[*《Red Hat Enterprise Linux 7 系统管理员指南》*](#)

解释内核模块功能的用法

https://access.redhat.com/site/node/28421/Configuring_VLAN_devices_over_a_bonded_interface

有关使用绑定接口配置 VLAN 设备的 Red Hat 知识库文章中。

第 5 章 配置网络成组 (Network Teaming)

5.1. 了解网络成组

联合或合并网络连接，以提供具有较高吞吐量的本地连接或冗余的方式可称为“频道绑定”、“以太网绑定”、“端口聚合”、“频道成组”、“NIC 成组”、“链接合并”等等。这个最初在 Linux 内核中应用的概念泛指“绑定”。现使用网络成组 (Network Teaming) 代表这个概念的最新应用。这不会影响现有的绑定驱动程序，网络成组会作为备选方法提供，且不会替换 Red Hat Enterprise Linux 7 中的绑定。

网络成组或成组旨在通过提供小内核驱动程序，以便使用不同的方法应用这个概念，实现数据包流的快速处理，并让各种用户空间应用程序在用户空间执行各种任务。该驱动程序有一个 *应用程序编程接口* (API)，即“成组 Netlink API”，可使用该接口进行 Netlink 通讯。用户空间程序库使用这个 API 与该驱动程序通讯。库指的是“lib”，可用来进行成组 Netlink 通讯及 RT Netlink 信息在用户空间的换行。同时还提供应用程序守护进程 **teamd** 使用 Libteam 库。**teamd** 的实例可控制成组驱动程序中的实例。该守护进程通过使用附加代码（即“运行程序”）采用负载均衡及 active-backup 逻辑（比如轮询）。通过使用这个方式分离代码，可方便网络成组对负载均衡及冗余要求的扩展及延伸解决方案。例如：使用 **teamd** 编写自定义运行程序应用新的逻辑可相对简单，即使 **teamd** 为自选程序，用户仍可编写其自己的应用程序以便使用 **libteam**。

teamdctl 提供一个用来控制使用 D-bus 运行 **teamd** 实例的工具。它可为 **teamd** D-Bus API 提供 D-Bus 换行程序。默认情况下，**teamd** 使用 Unix 域套接字 (Unix Domain Socket) 进行侦听和通讯，但仍监控 D-Bus。这样做是保证能够在没有 D-Bus 或者尚未载入 D-Bus 的环境中使用 **teamd**。例如：引导 **teamd** 链接时不一定载入 D-Bus。可在运行时使用 **teamdctl** 工具读取配置、连接监控程序状态、端口状态检查及变更、添加和删除端口以及将端口状态在 active 和 backup 状态间切换。

成组 Netlink API 通信使用 Netlink 信息与用户空间应用程序通讯。用户空间库 **libteam** 不会直接与这个 API 互动，但会使用 **libnl** 或 **teamnl** 与驱动程序 API 互动。

总之，不会直接配置或控制内核中运行的成组驱动程序实例。所有配置均采用用户空间应用程序完成，比如 **teamd** 程序。然后该程序会根据需要指向内核驱动程序。

5.2. 了解主接口及从属接口的默认行为

使用 **NetworkManager** 守护进程控制成组的端口接口时，特别是发现错误时，请记住以下要点：

1. 启动主接口不会自动启动端口接口。
2. 启动端口接口总是会启动主接口。
3. 停止主接口总是会停止端口接口。
4. 没有端口的主机可启动静态 **IP** 连接。
5. 没有端口的主机在启动 **DHCP** 连接时会等待端口。
6. 添加附带载波的端口后，使用 **DHCP** 连接的主机会等待端口完成连接。
7. 添加不附带载波的端口后，使用 **DHCP** 连接的主机会让端口继续等待。



警告

不支持对使用线缆但没有网络交换机的连接进行成组操作。如果没有网络交换机，在此论述的这个故障转移机制就无法工作。详情请查看 Red Hat 知识库文章 [《为什么在使用交叉线缆的直接连接中不支持绑定？》](#)

5.3. 网络成组和绑定对比

表 5.1. 绑定及成组功能对比

功能	绑定	成组
多播 Tx 策略	是	是
轮询 Tx 策略	是	是
active-backup Tx 策略	是	是
LACP (802.3ad) 支持	是 (仅适用于 passive)	是
基于哈希字符的 Tx 策略	是	是
用户可设定哈希功能	否	是
Tx 负载均衡支持 (TLB)	是	是
LACP 哈希端口选择	是	是
用于 LACP 支持的负载均衡	否	是
Ethtool 链接监控	是	是
ARP 链接监控	是	是
NS/NA (IPv6) 链接监控	否	是
端口启动/断开延迟	是	是
端口优先权及粘性 (“主要”选项加强)	否	是
根据端口链接进行独立监控的设置	否	是
多链接监控设置	有限监控	是
无锁 Tx/Rx 路径	无 (rwlock)	有 (RCU)
VLAN 支持	是	是
用户空间运行时控制	有限监控	全面控制
用户空间逻辑	否	是
延展性	困难	容易
模块设计	否	是
性能开销	低	非常低
D-Bus 接口	否	是
多设备堆叠	是	是
使用 LLDP 进行零配置	否	(计划中)
NetworkManager 支持	是	是

5.4. 了解网络成组守护进程及“运行程序”

成组守护进程 **teamd** 使用 **libteam** 控制成组驱动器中的一个实例。这个成组驱动器实例添加硬件设备驱动程序实例以构成网络链接“成组”。这个成组驱动器为内核的其他部分提供网络接口，比如 team0。文档中会为由成组驱动程序实例创建的接口按顺序命名，比如 team0、team1 等等。这样便于理解，但可使用其他名称。**teamd** 采用在所有成组方法中通用的逻辑；不同负载分享及备份方法中特有的功能，比如轮询，则是由不同的代码单位，也称“运行程序”实施。因为类似“模块”和“模式”等词语有与内核相关的特别含义，所以选择“运行程序”代表这些代码单元。用户可在 JSON 形式的配置文件中指定运行程序，然后就会在创建实例时将其代码编译到 **teamd** 实例中。运行程序不是插件，因为运行程序的代码是在创建该程序时编译到 **teamd** 中。以后可能会有将代码作为 **teamd** 插件创建的需求。

可在编写时使用以下运行程序。

- » broadcast (可将数据传送到所有端口)
- » round-robin (可按顺序将数据传送到所有端口)

- ✱ **active-backup**（使用一个端口或链接时其他则处于备用状态）
- ✱ **loadbalance**（使用主动 Tx 负载均衡及基于 BPF 的 Tx 端口选择程序）
- ✱ **lacp**（采用 802.3ad 链接合并控制协议）

此外还可使用以下链接监视程序：

- ✱ **ethtool**（Libteam lib 使用 **ethtool** 监视链接状态变化）。若没有在配置中指定其他链接监控程序，则默认使用该程序。
- ✱ **arp_ping**（使用 **arp_ping** 程序监控使用 ARP 数据包的远端硬件地址状态。）
- ✱ **nsna_ping**（使用 **IPv6** 邻居发现协议中的的邻居播发和邻居请求给你监控邻居的接口状态。）

代码中没有对使用具体运行程序防止使用特定链接监视程序的限制，但使用 **lacp** 运行程序时，只推荐使用 **ethtool** 链接监视程序。

5.5. 安装网络成组守护进程

默认不会安装网络成组守护进程 **teamd**。要安装 **teamd**，请作为 **root** 运行以下命令：

```
~]# yum install teamd
```

5.6. 将绑定转换为成组

可使用 **bond2team** 工具将现有绑定配置文件转换为成组配置文件。它可将 **ifcfg** 格式的绑定配置文件转换为 **ifcfg** 或 JSON 格式的成组配置文件。注：重命名后可能会破坏与原来的接口名称关联的防火墙规则、别名接口及其他信息，因为这个工具只更改 **ifcfg** 文件，其他什么都不会做。

请运行以下命令查看命令格式示例：

```
~]$ bond2team --examples
```

会在以 **/tmp/bond2team.XXXXXXX/** 开头的目录中创建新文件，其中 **XXXXXX** 是随机字符串。创建新配置文件后，请将旧的绑定文件移动到备份文件夹中，然后将新文件移动到 **/etc/sysconfig/network-scripts/** 目录下。

例 5.1. 将绑定转换为成组

请作为 **root** 运行以下命令将当前的 **bond0** 配置转换为成组 **ifcfg**：

```
~]# /usr/bin/bond2team --master bond0
```

注：这样会保留名称 **ifcfg**。要使用新名称保存该配置，请使用 **--rename** 选项，如下：

```
~]# /usr/bin/bond2team --master bond0 --rename team0
```

添加 **--json** 选项输出 JSON 格式文件，而不是 **ifcfg** 文件。有关 JSON 格式示例请查看 **teamd.conf(5)** man page。

例 5.2. 将绑定转换为成组并指定文件路径

要将 **bond0** 配置转换为成组 **ifcfg**，并手动指定 **ifcfg** 文件路径，请作为 **root** 运行以下命令：

```
~]# /usr/bin/bond2team --master bond0 --configdir /path/to/ifcfg-file
```

添加 **--json** 选项输出 JSON 格式文件，而不是 **ifcfg** 文件。

例 5.3. 使用 Bond2team 生成成组配置

可使用 **bond2team** 工具并附加一组绑定参数列表创建成组配置。例如：

```
~]# /usr/bin/bond2team --bonding_opts "mode=1 miimon=500"
```

还可以如下方式在命令行中提供端口：

```
~]# /usr/bin/bond2team --bonding_opts "mode=1 miimon=500 primary=eth1 \
primary_reselect=0" --port eth1 --port eth2 --port eth3 --port eth4
```

详情请查看 **bond2team(1)** man page。有关绑定参数的解释，请查看 [第 4.5 节 “使用频道绑定”](#)。

5.7. 选择作为网络成组端口使用的接口

请运行以下命令查看可用接口：

```
~]$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP > mtu 65536 qdisc noqueue state UNKNOWN mode
DEFAULT
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP > mtu 1500 qdisc pfifo_fast state
UP mode DEFAULT qlen 1000
    link/ether 52:54:00:6a:02:8a brd ff:ff:ff:ff:ff:ff
3: em2: <BROADCAST,MULTICAST,UP,LOWER_UP > mtu 1500 qdisc pfifo_fast state
UP mode DEFAULT qlen 1000
    link/ether 52:54:00:9b:6d:2a brd ff:ff:ff:ff:ff:ff
```

在可用接口中确定适合添加到您的网络成组中的接口，然后执行 [第 5.8 节 “选择网络成组配置方法”](#)。

**注意**

成组开发人员倾向于使用术语 “port”，而不是 “slave”，但 **NetworkManager** 使用 “team-slave” 代表组成一个成组的接口。

5.8. 选择网络成组配置方法

要使用 **NetworkManager** 的文本用户界面工具 **nmtui** 配置网络成组，请按照 [第 5.9 节 “使用文本用户界面 nmtui 配置网络成组”](#) 操作。

要使用命令行工具 **nmcli** 创建网络成组，请按照 [第 5.10.1 节“使用 nmcli 配置网络成组”](#) 操作。

要使用成组守护进程 **teamd** 创建网络成组，请按照 [第 5.10.2 节“使用 teamd 创建网络成组”](#) 操作。

要使用配置文件创建网络成组，请按照 [第 5.10.3 节“使用 ifcfg 文件创建网络成组”](#) 操作。

要使用图形用户界面配置网络成组，请按照 [第 5.13 节“使用 GUI 创建网络成组”](#) 操作。

5.9. 使用文本用户界面 nmtui 配置网络成组

可使用文本用户界面工具 **nmtui** 在终端窗口中配置成组。运行以下命令启动这个工具：

```
~]$ nmtui
```

此时会出现文本用户界面。输入无效的命令行时会显示用法信息。

请使用箭头键导航，或按 **Tab** 在选项间前进，按 **Shift+Tab** 后退。按 **Enter** 选择某个选项。按 **Space** 键更改复选框状态。

1. 在启动菜单中，请选择 **编辑连接**。选择 **添加**，此时会打开 **新建连接** 页面。

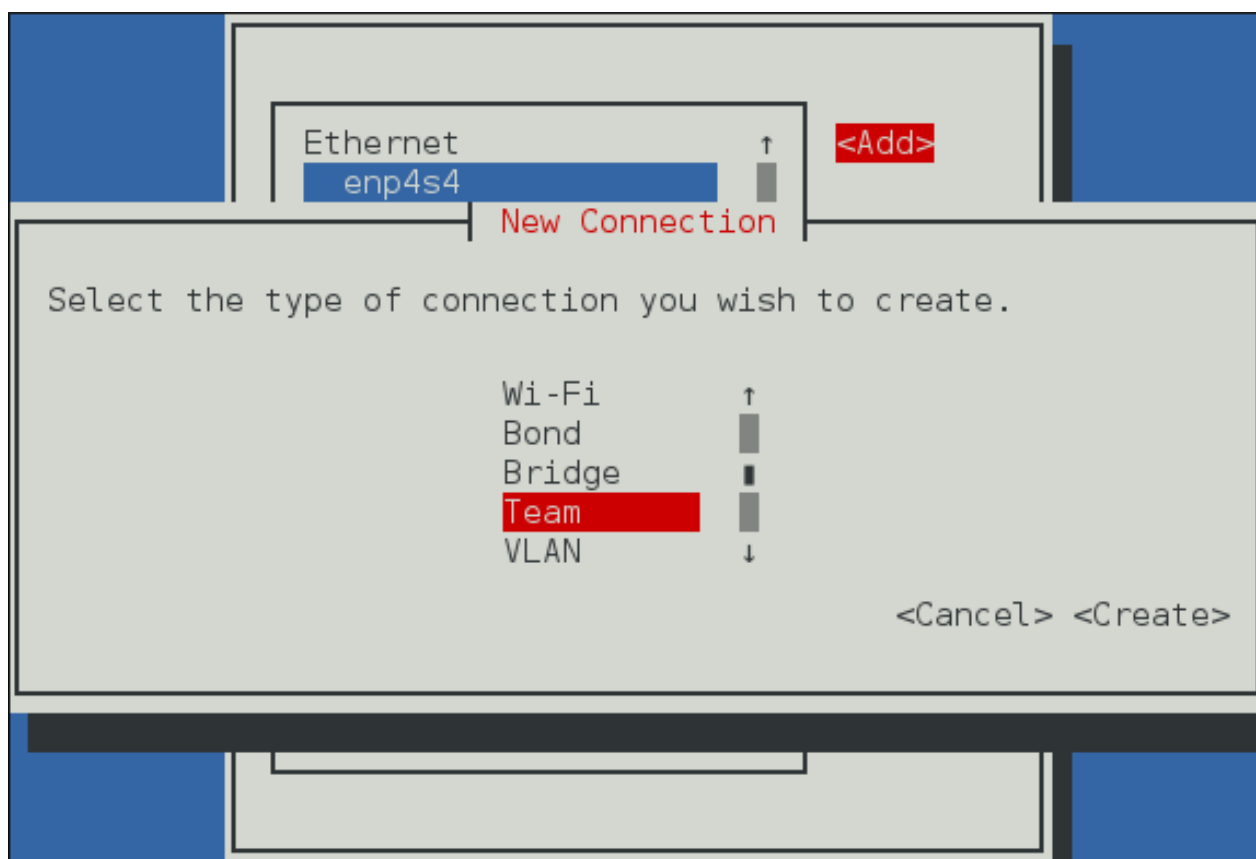


图 5.1. 添加成组连接的 NetworkManager 文本用户界面菜单

2. 选择 **成组** 打开 **编辑连接** 页面。

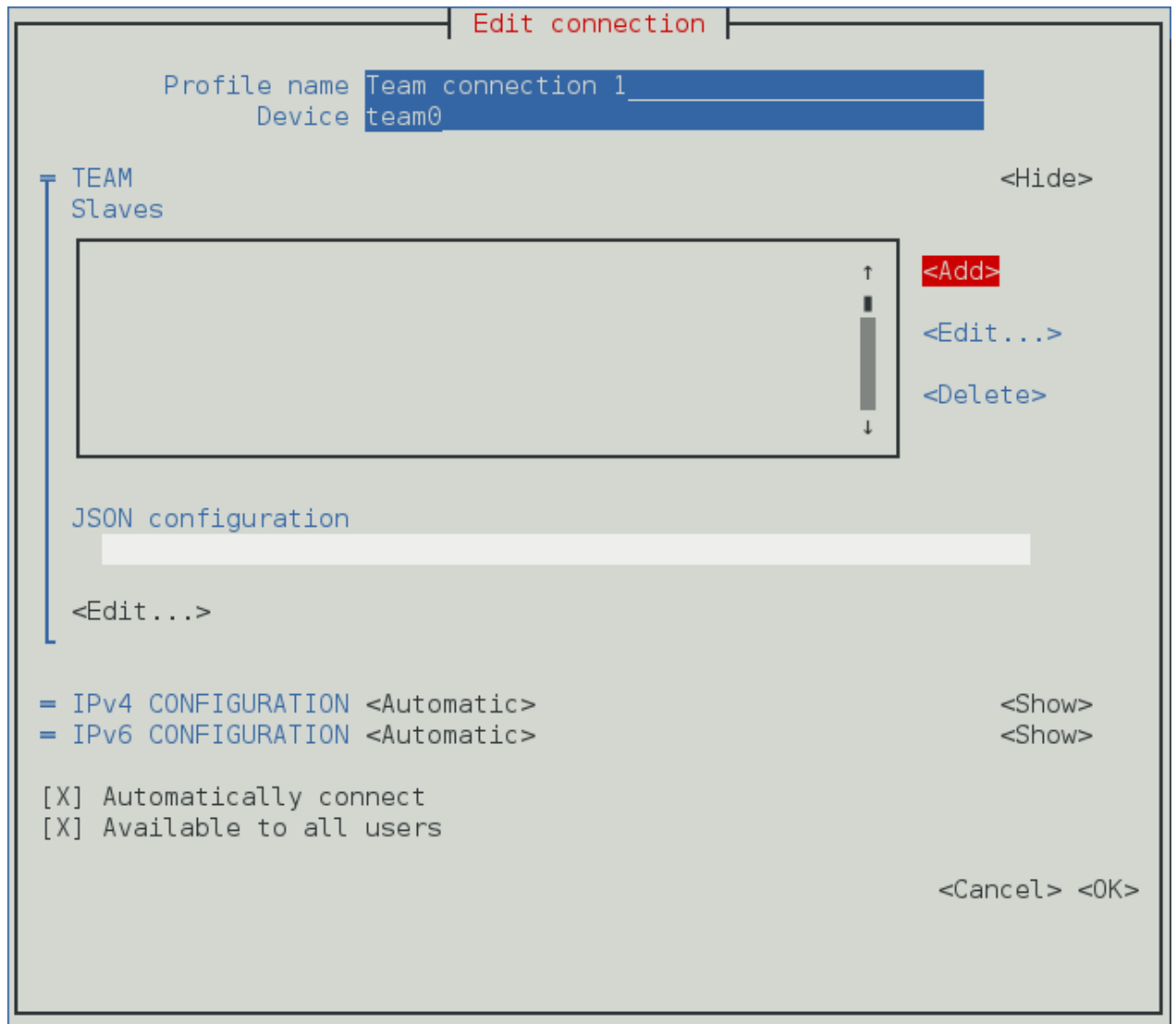


图 5.2. NetworkManager 文本用户界面法配置成组连接菜单

3. 要做成组中添加端口接口，请选择 **添加** 打开 **新建连接** 页面。选择连接类型后，点击 **创建** 按钮显示 **编辑连接** 页面。

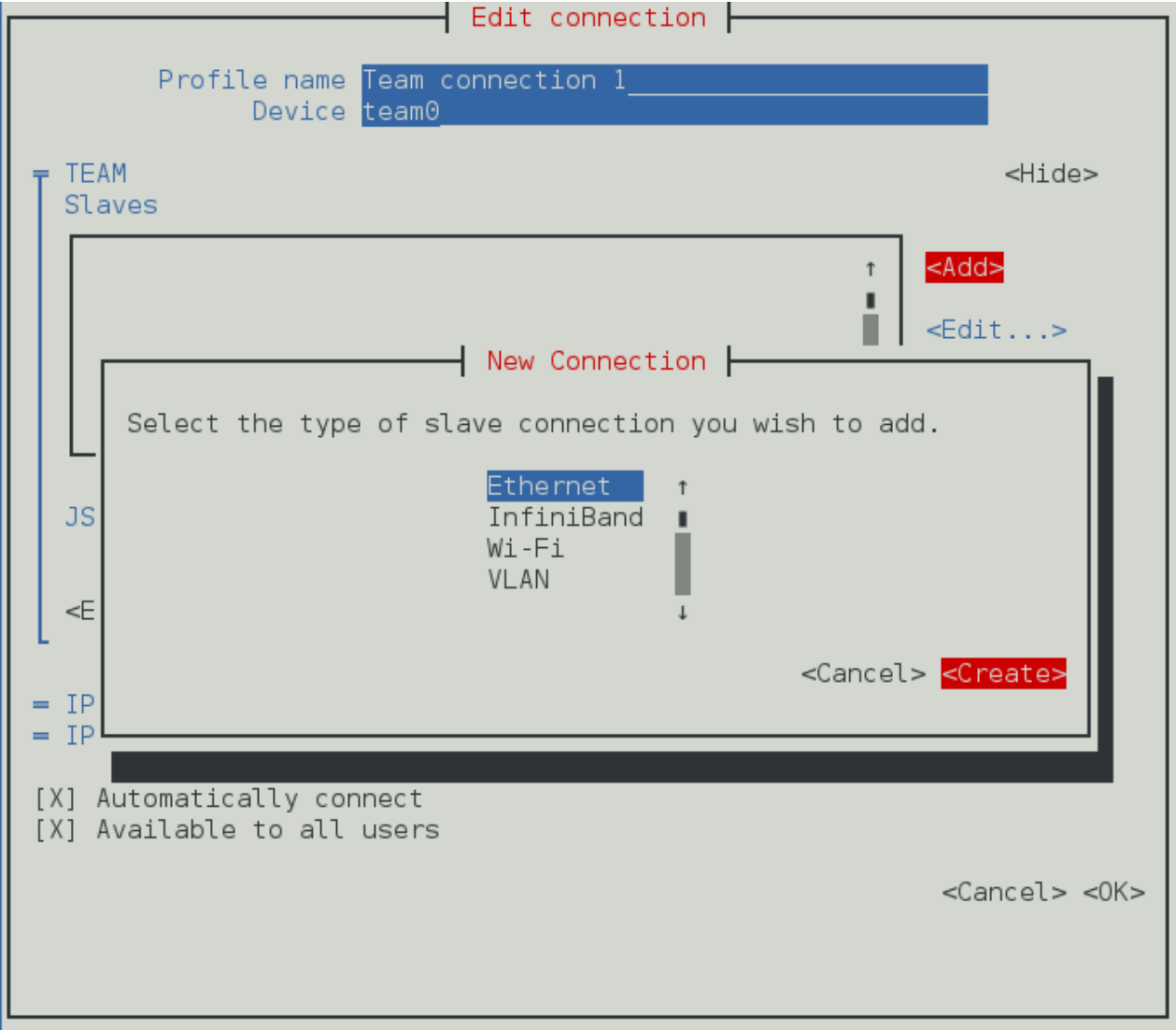



图 5.3. NetworkManager 文本用户界面中的配置新建成组端口接口连接菜单

- 4. 输入所需从属设备名称或 设备 部分的 MAC 地址。如有必要，选择 以太网 标签旁的 显示，输入成组 MAC 地址使用的克隆 MAC 地址。选择 确定 按钮。



注意

如果指定该设备时没有指定 MAC 地址，则会在重新载入 编辑连接 时自动填充 设备 部分，但首先要成功找到该设备。

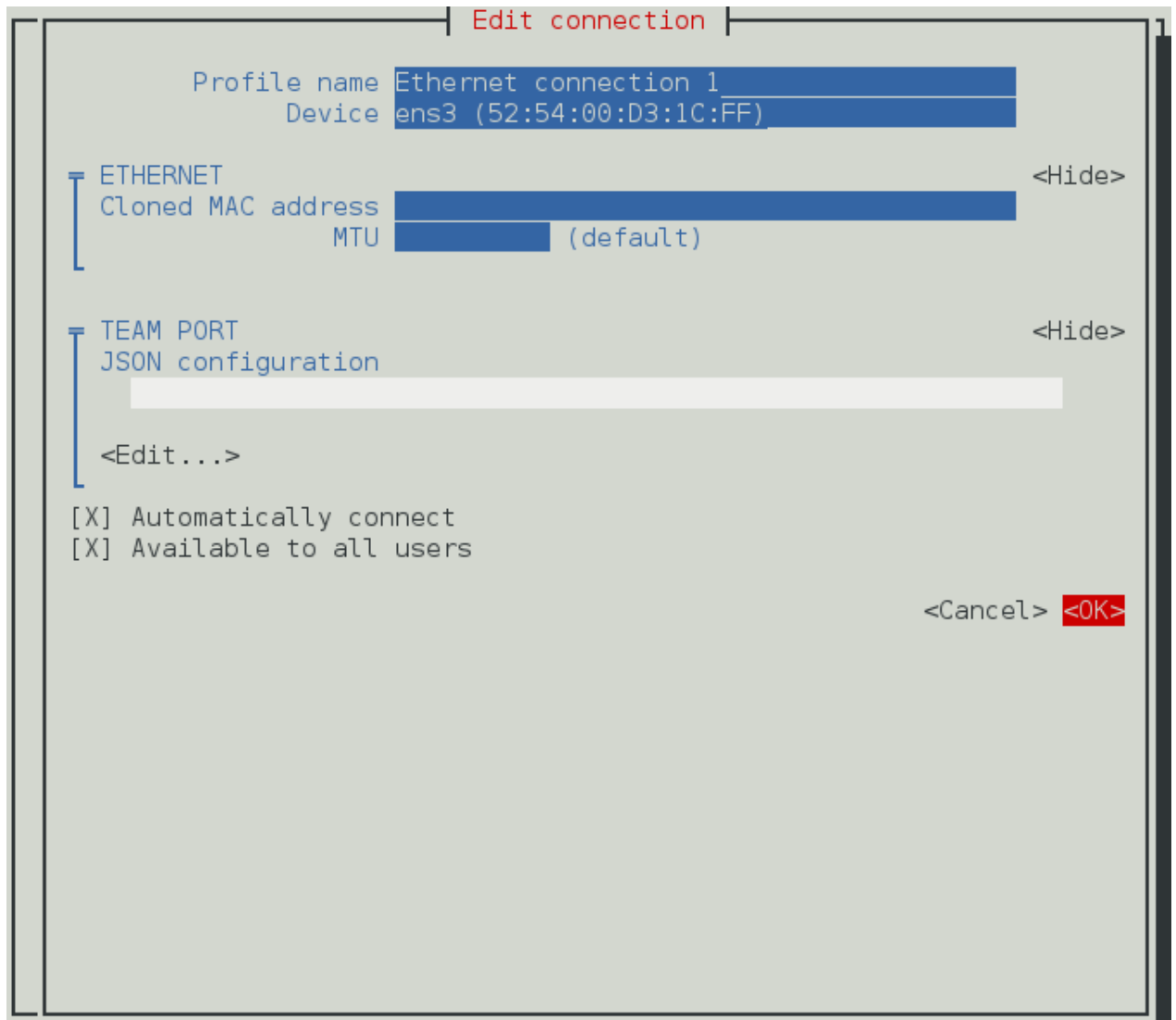


图 5.4. NetworkManager 文本用户界面中的配置成组端口接口连接菜单

5. 从属接口 部分会出现成组从属连接名称。重复上述步骤添加更多的从属连接。
6. 如果要应用自定义端口设置，请选择 **JSON 配置** 部分中的 **编辑** 按钮。这样会启动 **vim** 控制台方便您应用更改。在 **vim** 完成更改编写后，确定 **JSON 配置** 部分中显示的 JSON 字符串与满足要求。
7. 选择 **确定** 按钮前，请检查并确认设置。

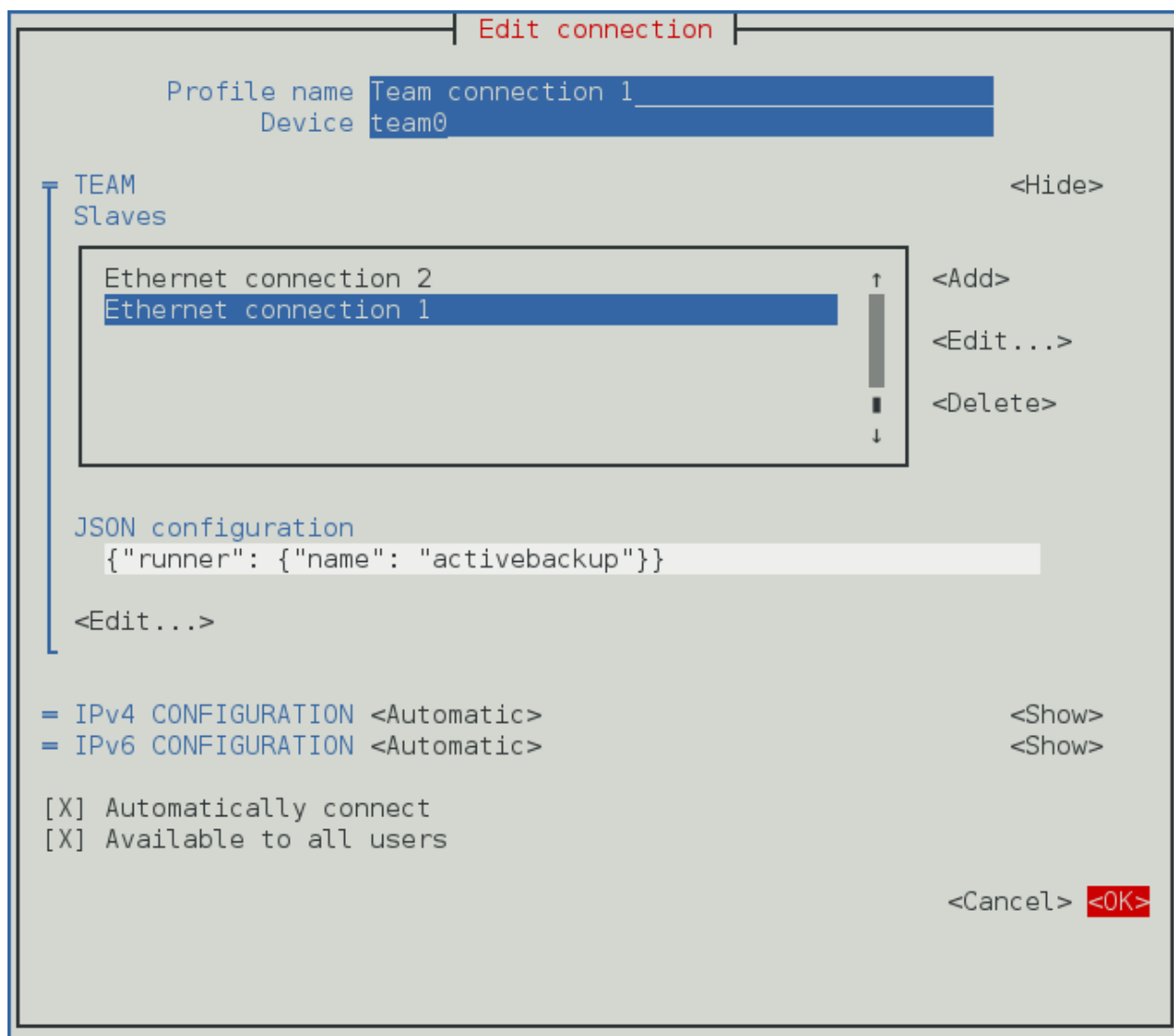


图 5.5. NetworkManager 文本用户界面法配置成组连接菜单

有关 JSON 字符串示例请查看 [第 5.12 节“配置 teamd 运行程序”](#)。注：只能在使用 `nmtui` 进行的成组或端口配置中使用示例字符串中的相关部分。请勿将“Device”指定为 JSON 字符串的一部分。例如：成组 JSON 配置字段中只能使用“device”之后，“port”之前的 JSON 字符串。所有与端口有关的 JSON 字符串都必须添加到 port 配置字段。

有关安装 `nmtui` 的详情请查看 [第 1.5 节“使用文本用户界面（nmtui）进行网络配置”](#)。

5.10. 使用命令行配置网络成组

5.10.1. 使用 nmcli 配置网络成组

运行以下命令查看系统中的可用设备：

```
~]$ nmcli connection show
```

NAME	UUID	TYPE	DEVICE
eth1	0e8185a1-f0fd-4802-99fb-bedbb31c689b	802-3-ethernet	--
eth0	dfe1f57b-419d-4d1c-aaf5-245deab82487	802-3-ethernet	--

请运行以下命令使用名称 `team-ServerA` 创建新的成组接口：

```
~]$ nmcli connection add type team ifname team-ServerA
Connection 'team-ServerA' (b954c62f-5fdd-4339-97b0-40efac734c50)
successfully added.
```

NetworkManager 会将其内部参数 **connection.autoconnect** 设定为 **yes**，这样就不会将给出 **ipv4.method** 的 IP 地址设定为 **auto**。**NetworkManager** 还会将配置写入 **/etc/sysconfig/network-scripts/ifcfg-team-ServerA**，其中会将对应的 **ONBOOT** 设定为 **yes**，并将 **BOOTPROTO** 设定为 **dhcp**。

注：再次启用该接口前，**NetworkManager** 不会意识到对 **ifcfg** 文件的手动更改。有关使用配置文件的详情，请查看 [第 1.9 节“使用 sysconfig 文件进行网络配置”](#)。

请运行以下命令查看其他分配的值：

```
~]$ nmcli con show team-ServerA
connection.id:                team-ServerA
connection.uuid:              b954c62f-5fdd-4339-97b0-40efac734c50
connection.interface-name:    ServerA
connection.type:              team
connection.autoconnect:      yes...
ipv4.method:                  auto[ 输出结果截选]
```

因为没有指定任何 JSON 配置文件，所以采用默认值。有关成组 JSON 参数及其默认值的详情，请查看 **teamd.conf(5)** man page。注：该名称衍生自接口名称，在接口名称的前面添加类型。另外还可使用 **con-name** 选项指定一个名称，如下：

```
~]$ nmcli connection add type team con-name Team0 ifname ServerB
Connection 'Team0' (5f7160a1-09f6-4204-8ff0-6d96a91218a7) successfully
added.
```

运行以下命令查看刚刚配置的成组接口：

```
~]$ nmcli con show
NAME                UUID                                TYPE
DEVICE
team-ServerA        b954c62f-5fdd-4339-97b0-40efac734c50 team
ServerA
eth1                0e8185a1-f0fd-4802-99fb-bedbb31c689b 802-3-ethernet -
-
eth0                dfe1f57b-419d-4d1c-aaf5-245deab82487 802-3-ethernet -
-
Team0               5f7160a1-09f6-4204-8ff0-6d96a91218a7 team
ServerB
```

运行以下格式的命令更改为成组分配的名称：

```
nmcli con mod old-team-name connection.id new-team-name
```

要为现有成组载入成组配置文件，请使用以下格式的命令：

```
nmcli connection modify team-name team.config JSON-config
```

可将成组配置文件指定为 JSON 字符串，或提供包含该配置的文件名。该文件名可包括路径。两种情况都会在

team.config 属性中保存 JSON 字符串。如果是 JSON 字符串，请使用单引号将字符串括起来，并将整个字符串粘贴到命令行中。

运行以下格式的命令检查 **team.config** 属性：

```
nmcli con show team-name | grep team.config
```

运行以下命令在 **Team0** 中添加名为 *Team0-port1* 的接口 *eth0*：

```
~]$ nmcli con add type team-slave con-name Team0-port1 ifname eth0 master Team0
Connection 'Team0-port1' (ccd87704-c866-459e-8fe7-01b06cf1cffc) successfully added.
```

同样，可使用以下命令添加另一个名为 *Team0-port2* 的接口 *eth1*：

```
~]$ nmcli con add type team-slave con-name Team0-port2 ifname eth1 master Team0
Connection 'Team0-port2' (a89ccff8-8202-411e-8ca6-2953b7db52dd) successfully added.
```

编写时，**nmcli** 只支持以太网接口。

要启用成组，必须首先激活这些端口，如下：

```
~]$ nmcli connection up Team0-port1
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/2)
```

```
~]$ nmcli connection up Team0-port2
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/3)
```

可通过激活这些端口验证是否已激活成组接口，如下：

```
~]$ ip link
3: Team0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT
    link/ether 52:54:00:76:6f:f0 brd ff:ff:ff:ff:ff:f
```

另外还可使用命令启用该接口组，如下：

```
~]$ nmcli connection up Team0
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/4)
```

有关 **nmcli** 简介请查看 [第 2.3 节“使用 NetworkManager 命令行工具 nmcli”](#)。

5.10.2. 使用 **teamd** 创建网络成组

**注意**

使用 `teamd` 创建的配置不会持久，因此应按照 [第 5.10.1 节 “使用 nmcli 配置网络成组”](#) 或 [第 5.10.3 节 “使用 ifcfg 文件创建网络成组”](#) 中规定的步骤创建成组接口。

要创建网络成组，作为成组端口或链接接口的虚拟接口需要一个 JSON 格式的配置文件。快捷的方法是复制示例配置文件，然后使用有 **root** 授权的编辑器进行编辑。请输入以下命令列出可用示例配置文件：

```
~]$ ls /usr/share/doc/teamd-*/example_configs/
activebackup_arp_ping_1.conf  activebackup_multi_lw_1.conf
loadbalance_2.conf
activebackup_arp_ping_2.conf  activebackup_nсна_ping_1.conf
loadbalance_3.conf
activebackup_ethtool_1.conf  broadcast.conf                random.conf
activebackup_ethtool_2.conf  lacp_1.conf
roundrobin_2.conf
activebackup_ethtool_3.conf  loadbalance_1.conf            roundrobin.conf
```

请使用以下命令查看包含的文件之一，比如 **activebackup_ethtool_1.conf**：

```
~]$ cat /usr/share/doc/teamd-*/example_configs/activebackup_ethtool_1.conf
{
  "device": "team0",
  "runner": {"name": "activebackup"},
  "link_watch": {"name": "ethtool"},
  "ports": {
    "eth1": {
      "prio": -10,
      "sticky": true
    },
    "eth2": {
      "prio": 100
    }
  }
}
```

创建工作配置目录保存 **teamd** 配置文件。例如：作为常规用户输入以下格式的命令：

```
~]$ mkdir ~/teamd_working_configs
```

将选择的文件复制到工作目录中，并根据需要进行编辑。例如：可使用以下命令格式：

```
~]$ cp /usr/share/doc/teamd-*/example_configs/activebackup_ethtool_1.conf \
~/teamd_working_configs/activebackup_ethtool_1.conf
```

要编辑该文件以适应您的环境（例如更改作为网络成组端口的接口），请打开要编辑的文件，如下：

```
~]$ vi ~/teamd_working_configs/activebackup_ethtool_1.conf
```

进行必要的更改并保存文件。有关使用 **vi** 编辑器或使用首选编辑器的详情，请查看 **vi(1)** man page。

注：在该成组内作为端口使用的接口不得处于 `active` 状态，就是说将其添加到成组设备时，它们必须处于 `“down”` 状态。运行以下命令检查其状态：

```
~]$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
DEFAULT
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
mode DEFAULT qlen 1000
    link/ether 52:54:00:d5:f7:d4 brd ff:ff:ff:ff:ff:ff
3: em2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
mode DEFAULT qlen 1000
    link/ether 52:54:00:d8:04:70 brd ff:ff:ff:ff:ff:ff
```

在这个示例中可以看到要使用的两个接口都处于 `“UP”` 状态。

请作为 **root** 用户，使用以下命令格式禁用接口：

```
~]# ip link set down em1
```

根据需要在每个接口中执行这个操作。

要创建基于配置文件的成组接口，请作为 **root** 用户进入可使用的配置目录（在这个示例中为 `teamd_working_configs`）：

```
~]# cd /home/user/teamd_working_configs
```

然后运行以下格式的命令：

```
~]# teamd -g -f activebackup_ethtool_1.conf -d
Using team device "team0".
Using PID file "/var/run/teamd/team0.pid"
Using config file
"/home/user/teamd_working_configs/activebackup_ethtool_1.conf"
```

`-g` 选项用于显示 debug 信息，`-f` 选项用于指定要载入的配置文件，`-d` 选项可让该进程中启动后作为守护进程运行。有关其他选项的详情请查看 **teamd(8)** man page。

请作为 **root** 用户使用以下命令检查成组接口状态：

```
~]# teamdctl team0 state
setup:
  runner: activebackup
ports:
  em1
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
  em2
    link watches:
      link summary: up
      instance[link_watch_0]:
```

```

        name: ethtool
        link: up
runner:
  active port: em1

```

请作为 **root** 用户，使用以下命令在网络成组接口 team0 中应用一个地址：

```
~]# ip addr add 192.168.23.2/24 dev team0
```

请运行以下命令检查成组接口的 IP 地址：

```

~]$ ip addr show team0
4: team0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether 16:38:57:60:20:6f brd ff:ff:ff:ff:ff:ff
    inet 192.168.23.2/24 scope global team0
        valid_lft forever preferred_lft forever
    inet6 2620:52:0:221d:1438:57ff:fe60:206f/64 scope global dynamic
        valid_lft 2591880sec preferred_lft 604680sec
    inet6 fe80::1438:57ff:fe60:206f/64 scope link
        valid_lft forever preferred_lft forever

```

请作为 **root** 用户，使用以下命令激活该成组接口，或使其处于 “up” 状态：

```
~]# ip link set dev team0 up
```

请作为 **root** 用户，使用以下命令暂时取消激活成组接口，或使其处于 “down” 状态：

```
~]# ip link set dev team0 down
```

请作为 **root** 用户，使用以下命令格式终止或杀死成组守护进程的实例：

```
~]# teamd -t team0 -k
```

使用 **-k** 选项指定要杀死的与设备 team0 关联的守护进程实例。其他选项请查看 **teamd(8)** man page。

请运行以下命令获得 **teamd** 命令行选项帮助信息：

```
~]$ teamd -h
```

此外，请查看 **teamd(8)** man page。

5.10.3. 使用 ifcfg 文件创建网络成组

要 **ifcfg** 文件创建网络成组，在 **/etc/sysconfig/network-scripts/** 目录中创建一个文件，如下：

```

DEVICE=team0
DEVICETYPE=Team
ONBOOT=yes
BOOTPROTO=none
IPADDR=192.168.11.1
PREFIX=24
TEAM_CONFIG='{"runner": {"name": "activebackup"}, "link_watch": {"name":
"ethtool"}}'

```

这样可为该成组创建接口，换句话说这就是主接口。

要创建属于 team0 成员的端口，请在 `/etc/sysconfig/network-scripts/` 目录中创建一个或多个文件，如下：

```
DEVICE=eth1
HWADDR=D4:85:64:01:46:9E
DEVICETYPE=TeamPort
ONBOOT=yes
TEAM_MASTER=team0
TEAM_PORT_CONFIG='{"prio": 100}'
```

根据要求添加与上述接口类似的附加端口接口，修改 `DEVICE` 和 `HWADDR` 字段使其与端口（网络设备）匹配。如果未根据 `prio` 指定端口优先权，则默认为 `0`。该数值可为负数和正数，范围在 `-32,767` 到 `+32,767` 之间。

使用 `HWADDR` 指令指定硬件或 MAC 地址会影响设备命名过程。原因请查看 [第 8 章 一致网络设备命名](#)。

请作为 `root` 运行以下命令启用网络成组：

```
~]# ifup team0
```

运行以下命令查看网络成组：

```
~]$ ip link show
```

5.10.4. 使用 `iputils` 在网络成组中添加端口

请作为 `root`，使用 `ip` 工具程序运行以下命令，在网络成组 `team0` 中添加端口 `em1`：

```
~]# ip link set dev em1 down
~]# ip link set dev em1 master team0
```

根据需要添加附加端口。成组驱动程序会自动启用这些端口。

5.10.5. 使用 `teamnl` 列出成组连接的端口

请作为 `root` 使用 `teamnl` 工具程序，运行以下命令查看或列出网络成组中的端口：

```
~]# teamnl team0 ports
em2: up 100 full duplex
em1: up 100 full duplex
```

5.10.6. 使用 `teamnl` 配置成组连接选项

请作为 `root` 使用 `teamnl` 工具程序，运行以下命令查看或列出当前可用选项：

```
~]# teamnl team0 options
```

请作为 `root` 运行以下命令将网络成组配置为使用 `active-backup` 模式：

```
~]# teamnl team0 setoption mode activebackup
```

5.10.7. 使用 iputils 在网络成组中添加地址

请作为 **root** 使用 **ip** 工具程序，运行以下命令在成组接口 **team0** 中添加地址：

```
~]# ip addr add 192.168.252.2/24 dev team0
```

5.10.8. 使用 iputils 在网络成组中激活接口

请作为 **root**，使用 **ip** 工具程序，运行以下命令激活接口，或在网络成组中“启用”接口：

```
~]# ip link set team0 up
```

5.10.9. 使用 teamnl 查看成组连接的 activeport 选项

请作为 **root**，使用 **teamnl** 程序，运行以下命令查看或列出网络成组中的 **activeport** 选项：

```
~]# teamnl team0 getoption activeport
0
```

5.10.10. 使用 teamnl 设置成组连接的 activeport 选项

请作为 **root**，使用 **teamnl** 工具程序，运行以下命令在网络成组中设置 **activeport** 选项：

```
~]# teamnl team0 setoption activeport 5
```

请作为 **root**，使用以下命令检查成组端口选项变更：

```
~]# teamnl team0 getoption activeport
5
```

5.11. 使用 teamnl 控制 teamd

请使用控制工具 **teamdctl** 查询运行的 **teamd** 实例，以获取统计或配置信息。

请作为 **root** 使用以下命令查看成组 **team0** 的当前状态：

```
~]# teamdctl team0 state view
```

要了解更详细的信息，请运行：

```
~]# teamdctl team0 state view -v
```

运行以下命令获取 **team0** 的完整 JSON 格式状态转储（对机器处理有帮助）：

```
~]# teamdctl team0 state dump
```

运行以下命令获取 **team0** 的 JSON 格式配置转储：

```
~]# teamdctl team0 config dump
```

运行以下命令查看作为成组 team0 一部分的端口 em1 配置：

```
~]# teamdctl team0 port config dump em1
```

5.11.1. 在网络成组中添加端口

请作为 **root** 运行以下命令在网络成组 team0 中添加端口 em1：

```
~]# teamdctl team0 port add em1
```

5.11.2. 从网络成组中删除端口

请作为 **root** 运行以下命令，从网络成组 team0 中删除接口 em1：

```
~]# teamdctl team0 port remove em1
```

5.11.3. 在网络成组中为端口应用配置

要在网络成组 team0 为端口 em1 应用 JSON 格式配置，请作为 **root** 运行以下格式的命令：

```
~]# teamdctl team0 port config update em1 JSON-config-string
```

其中 *JSON-config-string* 是 JSON 格式文本中的字符串文本配置。这样就可使用提供的 JSON 格式字符串更新端口配置。用来配置端口的有效 JSON 字符串示例类似如下：

```
{
  "prio": -10,
  "sticky": true
}
```

使用单引号括起 JSON 配置字符串，并忽略换行符号。

注：会覆盖旧的配置，同时会将省略的选项重置为默认值。有关成组守护进程控制工具命令示例，请查看 **teamdctl(8)** man page。

5.11.4. 查看网络成组中端口的配置

要复制网络成组 team0 中端口 em1 的配置，请作为 **root** 运行以下命令：

```
~]# teamdctl team0 port config dump em1
```

这会将该端口的 JSON 格式配置转储为标准输出。

5.12. 配置 teamd 运行程序

运行程序是创建实例时编译到成组连接守护进程中的代码单元。有关 **teamd** 运行程序的简介，请查看 [第 5.4 节“了解网络成组守护进程及“运行程序”](#)。

5.12.1. 配置 broadcast 运行程序

要配置 broadcast 运行程序，请作为 **root** 使用编辑器，在成组 JSON 格式配置文件中添加以下内容：

```
{
  "device": "team0",
  "runner": {"name": "broadcast"},
  "ports": {"em1": {}, "em2": {}}
}
```

详情请查看 **teamd.conf(5)** man page。

5.12.2. 配置 random 运行程序

Random 运行程序与 roundrobin 运行程序行为类似。

要配置 random 运行程序，请作为 **root** 使用编辑器，在成组 JSON 格式配置文件中添加以下内容：

```
{
  "device": "team0",
  "runner": {"name": "random"},
  "ports": {"em1": {}, "em2": {}}
}
```

详情请查看 **teamd.conf(5)** man page。

5.12.3. 配置 roundrobin 运行程序

要配置 roundrobin 运行程序，请作为 **root** 使用编辑器在成组 JSON 格式配置文件中添加以下内容：

```
{
  "device": "team0",
  "runner": {"name": "roundrobin"},
  "ports": {"em1": {}, "em2": {}}
}
```

Roundrobin 的最基本配置。

详情请查看 **teamd.conf(5)** man page。

5.12.4. 配置 active-backup 运行程序

active-backup 运行程序可使用所有链接监视程序确定成组中链接的状态。以下任意示例都可添加到成组 JSON 格式配置文件中：

```
{
  "device": "team0",
  "runner": {
    "name": "activebackup"
  },
  "link_watch": {
    "name": "ethtool"
  },
}
```

```

    "ports": {
      "em1": {
        "prio": -10,
        "sticky": true
      },
      "em2": {
        "prio": 100
      }
    }
  }
}

```

这个示例配置使用 active-backup 运行程序 **ethtool** 作为链接监视程序。端口 em2 有较高优先权。这个粘性标签可保证 em1 处于 active 状态，只要链接处于连接状态，它就仍将保持 active 状态。

```

{
  "device": "team0",
  "runner": {
    "name": "activebackup"
  },
  "link_watch": {
    "name": "ethtool"
  },
  "ports": {
    "em1": {
      "prio": -10,
      "sticky": true,
      "queue_id": 4
    },
    "em2": {
      "prio": 100
    }
  }
}

```

这个示例配置添加了队列 ID 4。它使用 active-backup 运行程序，将 **ethtool** 作为链接监视程序。端口 em2 有较高的优先权。但粘性标签可保证 em1 处于 active 状态。只要链接处于连接状态，它就会处于 active 状态。

要使用 **ethtool** 配置 active-backup 运行程序，并应用延迟，请作为 **root** 使用编辑器在成组 JSON 格式配置文件中添加以下内容：

```

{
  "device": "team0",
  "runner": {
    "name": "activebackup"
  },
  "link_watch": {
    "name": "ethtool",
    "delay_up": 2500,
    "delay_down": 1000
  },
  "ports": {
    "em1": {
      "prio": -10,
      "sticky": true
    },
    "em2": {

```



```

        "prio": 100
    }
}
}

```

这个示例配置使用 `active-backup` 运行程序，将 `ethtool` 作为链接监视程序使用。端口 `em2` 有较高优先权。但粘性标签保证如果 `em1` 处于 `active` 状态，则只要链接处于链接状态，它就会保持 `active` 状态。链接更改不会立即计入该运行程序，但会应用延迟。

详情请查看 `teamd.conf(5)` man page。

5.12.5. 配置 loadbalance 运行程序

这个运行程序可用于两种负载平衡，即主动和被动负载平衡。在主动模式中，通过使用最新流量统计不断进行流量再平衡，以便尽可能平均分配流量。在静态模式中，流量会在可用链接之间随机分配。鉴于较低的处理开销，这样有速度优势。在高流量应用程序中通常首选此方式，因为流量通常由多个要在可用链接间随机分配的流组成，使用这种方式时，无需 `teamd` 介入即可完成负载分布。

要为被动传送 (Tx) 负载平衡配置 `loadbalance` 运行程序，请作为 **root** 在成组 JSON 格式配置文件中添加以下内容：

```

{
  "device": "team0",
  "runner": {
    "name": "loadbalance",
    "tx_hash": ["eth", "ipv4", "ipv6"]
  },
  "ports": {"em1": {}, "em2": {}}
}

```

配置基于哈希的被动传送 (Tx) 负载平衡。

要为主动传送 (Tx) 负载平衡配置 `loadbalance` 运行程序，请作为 **root** 在成组 JSON 格式配置文件中添加以下内容：

```

{
  "device": "team0",
  "runner": {
    "name": "loadbalance",
    "tx_hash": ["eth", "ipv4", "ipv6"],
    "tx_balancer": {
      "name": "basic"
    }
  },
  "ports": {"em1": {}, "em2": {}}
}

```

使用基本负载平衡程序配置主动传送 (Tx) 负载平衡。

详情请查看 `teamd.conf(5)` man page。

5.12.6. 配置 LACP (802.3ad) 运行程序

要使用 `ethtool` 作为链接监视器配置 LACP 运行程序，请作为 **root** 用户使用编辑器中成组 JSON 格式配置文件中添加以下内容：

```
{
  "device": "team0",
  "runner": {
    "name": "lacp",
    "active": true,
    "fast_rate": true,
    "tx_hash": ["eth", "ipv4", "ipv6"]
  },
  "link_watch": {"name": "ethtool"},
  "ports": {"em1": {}, "em2": {}}
}
```

配置可使用 [链接聚合控制协议](#)（LACP）对应连接。LCAP 运行程序应使用 **ethtool** 监控链接状态。使用 **ethtool** 以外的链接监控方法没有任何意义，因为使用 **arp_ping** 时，该链接可能永远无法激活。原因是必须首先建立该链接，也只有在建立链接后，数据包（包括 ARP）方可通过。使用 **ethtool** 可防止此类情况，因为它对每一层中的链接单独监控。

使用这个运行程序可能会以 **loadbalance** 运行程序类似的方式进行主动负载平衡。要启用主动传送（Tx）负载平衡，请添加以下内容：

```
"tx_balancer": {
  "name": "basic"
}
```

详情请查看 **teamd.conf(5)** man page。

5.12.7. 配置链接状态监控

以下为可用链接状态监控方法。要采用其中之一，请以 **root** 授权，使用编辑器在成组 JSON 格式配置文件中添加 JSON 格式字符串。

5.12.7.1. 在 Ethtool 中配置链接状态监控

要在激活链接之后到通知运行程序之前到时间段内添加或编辑现有延迟（单位：毫秒），可按照如下方法操作：

```
"link_watch": {
  "name": "ethtool",
  "delay_up": 2500
}
```

要在断开链接后到通知运行程序前添加或编辑现有延迟（单位：毫秒），可按照如下方法操作：

```
"link_watch": {
  "name": "ethtool",
  "delay_down": 1000
}
```

5.12.7.2. 为链接状态监控配置 ARP Ping

成组守护进程 **teamd** 会向链接远端的地址发送 ARP REQUEST，以确定该链接是否处于连接状态。这个方法与 **arping** 程序的功能相同，但不会使用该工具程序。

准备一个包含 JSON 格式新配置的文件，类似如下：

```
{
  "device": "team0",
  "runner": {"name": "activebackup"},
  "link_watch": {
    "name": "arp_ping",
    "interval": 100,
    "missed_max": 30,
    "source_host": "192.168.23.2",
    "target_host": "192.168.23.1"
  },
  "ports": {
    "em1": {
      "prio": -10,
      "sticky": true
    },
    "em2": {
      "prio": 100
    }
  }
}
```

这个配置使用 **arp_ping** 作为链接监视器。**missed_max** 选项是最多可允许的丢失回复数（比如 ARP 回复）。可与 **interval** 选项一同使用，以决定将链接报告为断开前的总时间长度。

要为成组端口 em2 从包含 JSON 配置的文件中载入新配置，请作为 **root** 运行以下命令：

```
~]# port config update em2 JSON-config-file
```

注：会覆盖旧的配置，同时会将省略的选项重置为默认值。有关成组守护进程控制工具命令示例，请查看 **teamdctl(8)** man page。

5.12.7.3. 为链接状态监控配置 IPv6 NA/NS

```
{
  "device": "team0",
  "runner": {"name": "activebackup"},
  "link_watch": {
    "name": "nsna_ping",
    "interval": 200,
    "missed_max": 15,
    "target_host": "fe80::210:18ff:feaa:bbcc"
  },
  "ports": {
    "em1": {
      "prio": -10,
      "sticky": true
    },
    "em2": {
      "prio": 100
    }
  }
}
```

要配置发送 NS/NA 数据包的间隔，请添加或编辑以下内容：

```
"link_watch": {
  "name": "nsna_ping",
  "interval": 200
}
```

该数值为正数（单位：毫秒）。可与 **missed_max** 选项一同使用，以决定将链接状态报告为断开前的总时间长度。

要配置将链接状态报告为断开前最多可丢失的 NS/NA 数据包数，请添加或编辑以下内容：

```
"link_watch": {
  "name": "nsna_ping",
  "missed_max": 15
}
```

最多可丢失的 NS/NA 回复数。如果超过这个数值，则会将该链接状态报告为断开。**missed_max** 选项是允许丢失回复（比如 ARP 回复）的最大数值。它可与 **interval** 选项一同使用，以决定报告链接断开前的总时间长度。

要配置解析为 **IPv6** 地址（即 NS/NA 数据包的目标地址）的主机名，请添加或编辑以下内容：

```
"link_watch": {
  "name": "nsna_ping",
  "target_host": "MyStorage"
}
```

“target_host”选项包含要转换为 **IPv6** 地址的主机名，会将该地址作为 NS/NA 数据包的目标地址使用。可在主机名中使用 **IPv6** 地址。

详情请查看 **teamd.conf(5)** man page。

5.12.8. 配置端口选择覆盖

该物理端口传输的帧一般由成组驱动程序内核部分选择，而不是由用户或系统管理员决定。输出端口由所选成组模式（**teamd** 运行程序）策略选择。但在实现较为复杂的策略时，偶尔将某些类型的传出流量指向某些物理接口有帮助。默认情况下，成组驱动程序可识别多队列，并在启动该驱动程序时创建 16 个队列。如果需要更多或较少的队列，可在创建成组驱动程序实例时，使用 Netlink 属性 **tx_queues** 更改这个数值。

可使用端口配置选项 **queue_id** 配置端口的队列 ID，如下：

```
{
  "queue_id": 3
}
```

这些队列 ID 可与 **tc** 工具程序一同使用，以便配置多队列原则和过滤器，倾向于将某些流量传送到某些端口设备中。流入：如果使用上述配置，并要强制所有绑定在 **192.168.1.100** 的流量使用成组中的 **eth1** 作为输出设备，则请作为 **root** 用户运行以下格式的命令：

```
~]# tc qdisc add dev team0 handle 1 root multiq
~]# tc filter add dev team0 protocol ip parent 1: prio 1 u32 match ip dst \
  192.168.1.100 action skbedit queue_mapping 3
```

这个机制覆盖运行程序选择逻辑，以便将流量绑定到具体端口，该机制适用于所有运行程序。

5.12.9. 配置基于 BPF 的 Tx 端口选择程序

loadbalance 和 LACP 运行程序使用数据包哈希为网络流量排序。这个哈希计算机制是基于伯克利数据包过滤器 (BPF) 代码。BPF 代码是用来生成哈希，而不是决定传出数据包的策略。哈希长度为 8 字节，有 256 种变体。就是说很多不同的套接字缓冲 (SKB) 可有相同的哈希，并因此将流量传递给同一链接。使用短哈希是将流量分入不同 stream 的快速方法，以便在多个链接间取得平衡负载。在静止模式中，只使用哈希决定应该向那个端口发送流量。在活动模式中，运行程序将继续创新将哈希分配给不同的端口，以达到最佳平衡状态。

可在数据包 Tx 哈希计算中使用以下碎片类型或字符串：

- ✦ **eth** — 使用源和目标 MAC 地址。
- ✦ **vlan** — 使用 VLAN ID。
- ✦ **ipv4** — 使用源和目标 IPv4 地址。
- ✦ **ipv6** — 使用源和目标 IPv6 地址。
- ✦ **ip** — 使用源和目标 IPv4 和 IPv6 地址。
- ✦ **l3** — 使用源和目标 IPv4 和 IPv6 地址。
- ✦ **tcp** — 使用源和目标 TCP 端口。
- ✦ **udp** — 使用源和目标 UDP 端口。
- ✦ **sctp** — 使用源和目标 SCTP 端口。
- ✦ **l4** — 使用源和目标 TCP、UDP 和 SCTP 端口。

可通过在负载平衡运行程序中添加如下格式的一行使用这些字符串：

```
"tx_hash": ["eth", "ipv4", "ipv6"]
```

。示例请参考 [第 5.12.5 节“配置 loadbalance 运行程序”](#)。

5.13. 使用 GUI 创建网络成组

5.13.1. 建立成组连接

可使用 GNOME control-center 程序让 NetworkManager 使用两个或更多有线或 InfiniBand 链接创建成组。不需要首先创建要成组的链接。可在配置成组的过程中配置这些连接。必须有可用的 MAC 地址方可完成配置过程。

过程 5.1. 添加新成组连接

按照以下步骤操作添加新的成组连接。

1. 按 **Super** 键进入活动概述，输入 **control network**，并按 **Enter** 键。此时会显示 **Network** 设置工具。[第 2.5 节“在 GNOME 图形用户界面中使用 NetworkManager”](#) 中有关于这个步骤的详细论述。
2. 点击加号打开选择列表。选择 **成组**。此时会出现 **编辑成组连接 1** 窗口。

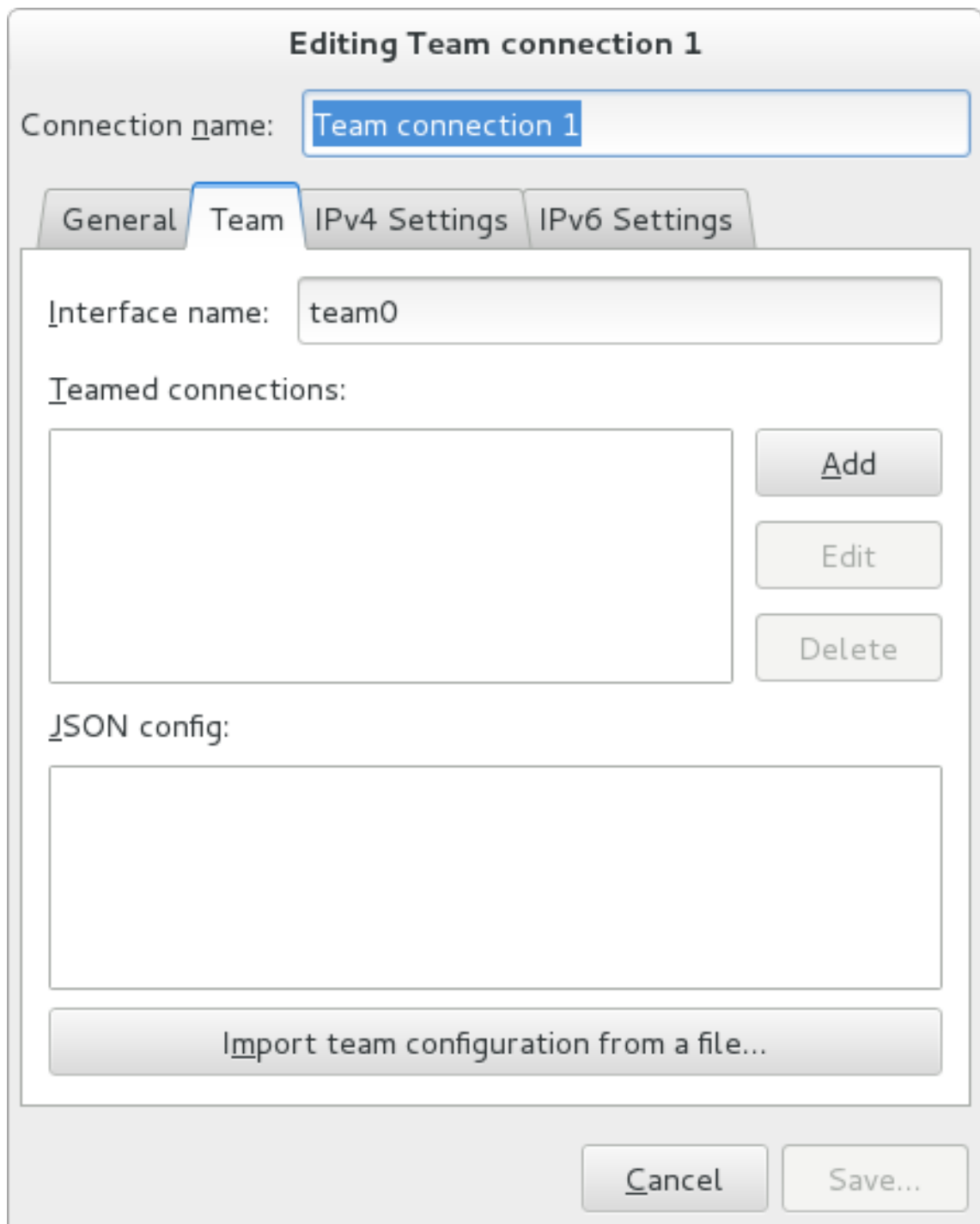


图 5.6. NetworkManager 图形用户界面的添加菜单

3. 在 **成组** 标签中点击 **添加**按钮，并选择要在成组连接中使用的接口类型。点击 **创建** 按钮。注：只在创建第一个端口时会显示选择端口类型对话框，之后将会自动在所有端口中使用同样的类型。
4. 此时会出现 **编辑 team0 端口 1**窗口。填写要在成组中添加的第一个接口地址。

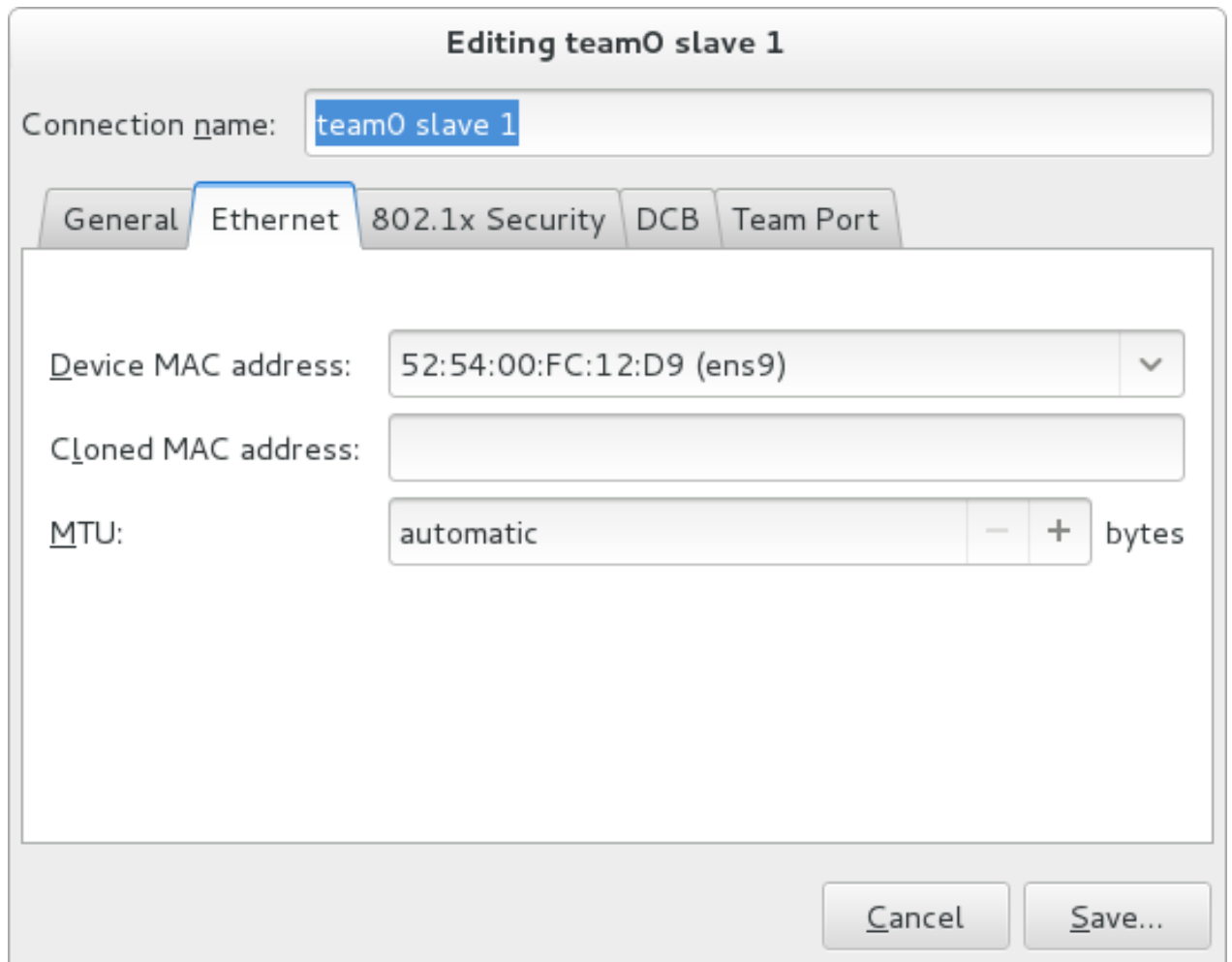


图 5.7. NetworkManager 图形用户界面的添加从属连接菜单

5. 如果要应用自定义端口设置，请点击 **成组端口** 标签，并输入 JSON 配置，或使用文件导入该配置。
6. 点击 **保存** 按钮。
7. 此时会在 **成组连接** 窗口中出现成组的端口。点击 **添加** 按钮添加未来的端口连接。
8. 检查并确认设置，然后点击 **保存** 按钮。
9. 参考下面的 [第 5.13.1.1 节 “配置成组标签”](#) 部分，编辑具体成组设置。

过程 5.2. 编辑现有成组链接

按照以下步骤操作编辑现有成组连接。

1. 按 **Super** 键进入活动概述页面，输入 **control network**，然后按 **Enter** 键。此时会出现 **Network** 设置工具。
2. 选择要编辑的连接，并点击 **选项** 按钮。
3. 选择 **常规** 标签。
4. 配置连接名称、自动连接行为及可用性设置。

编辑 对话框中的五个设置适用于所有连接类型，请参看 **常规** 标签：

- ✦ **连接名称** — 为网络配置选择一个描述性名称。可使用这个名称在 **网络** 窗口中列出这个连接。

- ✧ **可用时将其自动连接到这个网络** — 如果要让 **NetworkManager** 在其可用时自动连接这个连接，选择这个复选框。详情请查看 [第 2.5.3 节 “自动连接到网络”](#)。
- ✧ **所有用户都可以连接到这个网络** — 创建可用于系统中所有用户的连接时，选中这个复选框。更改这个设置需要 root 授权。详情请查看 [第 2.5.4 节 “系统范围及专用连接配置文件”](#)。
- ✧ **使用这个连接自动连接到 VPN** — 如果要让 **NetworkManager** 在 VPN 连接可用时自动连接，请选择这个复选框。从下拉菜单中选择 VPN。
- ✧ **防火墙区域** — 从下拉菜单中选择防火墙区域。有关防火墙区域的详情，请查看 [《Red Hat Enterprise Linux 7 安全指南》](#)。

5. 参考下面的 [第 5.13.1.1 节 “配置成组标签”](#) 部分，编辑具体成组设置。

保存新的（或修改的）连接并进行进一步配置

完成编辑成组连接后，点击 **保存** 按钮保存自定义配置。如果编辑时正在使用该配置文件，重启电源以便 **NetworkManager** 应用所做更改。如果该配置文件处于 OFF 状态，请将其设定为 ON，或者网络连接图标菜单中选择它。有关使用新建或更改连接的详情，请查看 [第 2.5.1 节 “使用 GUI 连接到网络”](#)。

可在 **网络Network** 窗口中选择现有连接对其进行编辑，然后点击 **选项 返回 编辑** 对话框。

然后配置：

- ✧ 该连接的 **IPv4** 设置，点击 **IPv4 Settings** 标签，继续 [第 2.5.10.4 节 “配置 IPv4 设置”](#)；或者，
- ✧ 该链接的 **IPv6** 设置，点击 **IPv6 Settings** 标签，继续 [第 2.5.10.5 节 “配置 IPv6 设置”](#)。

保存后，该成组会出现在 Network 设置工具中。

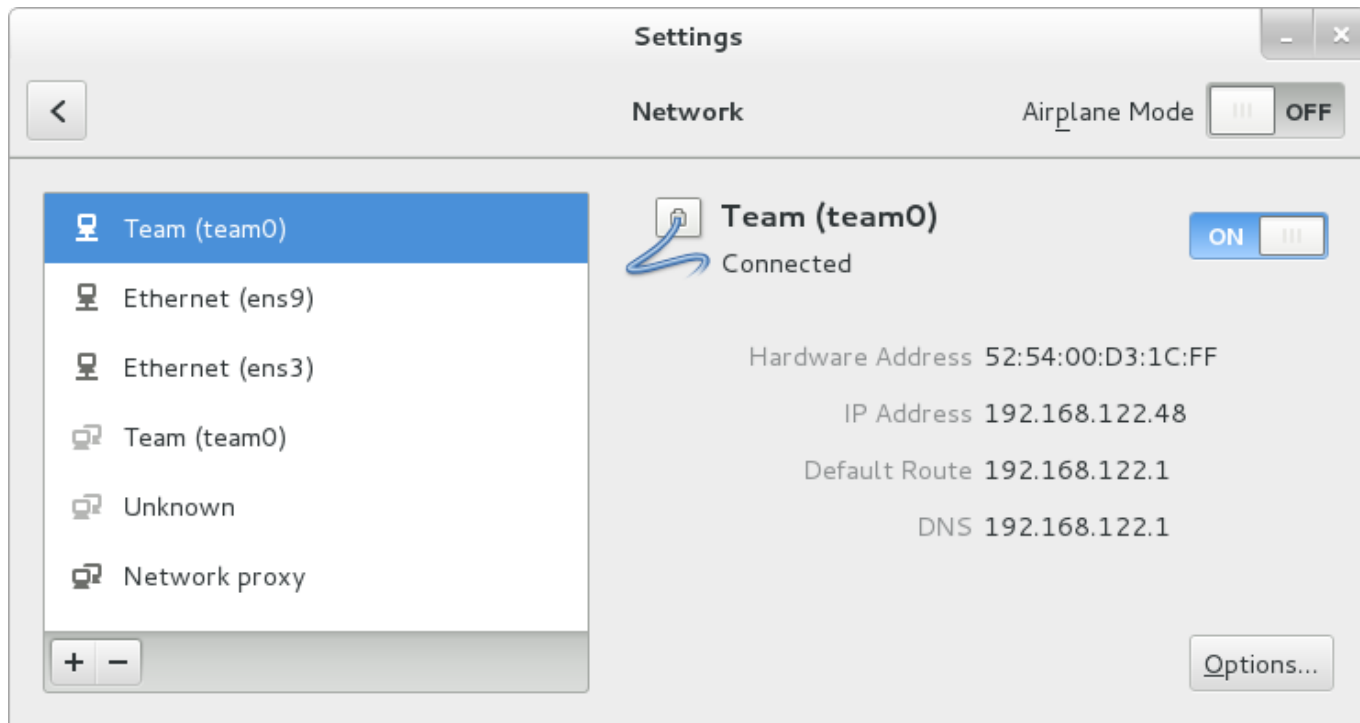


图 5.8. 附带成组的 NetworkManager 图形用户界面

5.13.1.1. 配置成组标签

如果已添加新的成组连接，则可以在文本框中输入自定义 JSON 配置字符串，或导入配置文件。点击 **保存** 在成组界面中应用 JSON 配置。

有关 JSON 字符串的详情，请查看 [第 5.12 节“配置 teamd 运行程序”](#)。

有关如何添加新成组的说明，请查看 [过程 5.1, “添加新成组连接”](#)。

5.14. 其他资料

以下信息资源提供有关网络成组的附加信息。

5.14.1. 已安装文档

- ✧ **teamd(8)** man page — 描述 **teamd** 服务。
- ✧ **teamdctl(8)** man page — 描述 **teamd** 控制工具。
- ✧ **teamd.conf(5)** man page — 描述 **teamd** 配置文件。
- ✧ **teamnl(8)** man page — 描述 **teamd** Netlink 库。
- ✧ **bond2team(1)** man page — 描述将绑定选项转换为成组的工具。

5.14.2. 在线文档

http://www.w3schools.com/json/json_syntax.asp

JSON 句法说明。

第 6 章 配置网络桥接

网络桥接是一个链接层设备，可在基于 MAC 地址的网络间转发流量。它根据通过侦听网络流量建立的 MAC 地址表了解每个网络连接的主机，并做出转发决定。在 Linux 主机中可使用软件桥接模拟硬件桥接，例如与一个或多个虚拟网卡共享 NIC 的虚拟化应用程序。

注：无法在以 *临时* 或 *架构* 模式运行的 Wi-Fi 网络建立桥接。这是 IEEE 802.11 标准造成的，该标准指定在 Wi-Fi 中使用 3-地址帧，以便有效使用放映时间（airtime）。

6.1. 使用文本用户界面 nmtui 配置桥接

可使用文本用户界面工具 **nmtui** 在终端窗口中配置桥接。运行以下命令启动该工具：

```
~]$ nmtui
```

此时会出现文本用户界面。使用无效命令会显示用法信息。

使用箭头键导航或按 **Tab** 在选项中前进，按 **Shift+Tab** 后退。按 **Enter** 选择一个选项。按 **Space** 键更改选择框状态。

1. 在起始菜单中选择 **编辑连接**。选择 **添加**，打开 **新建连接** 页面。

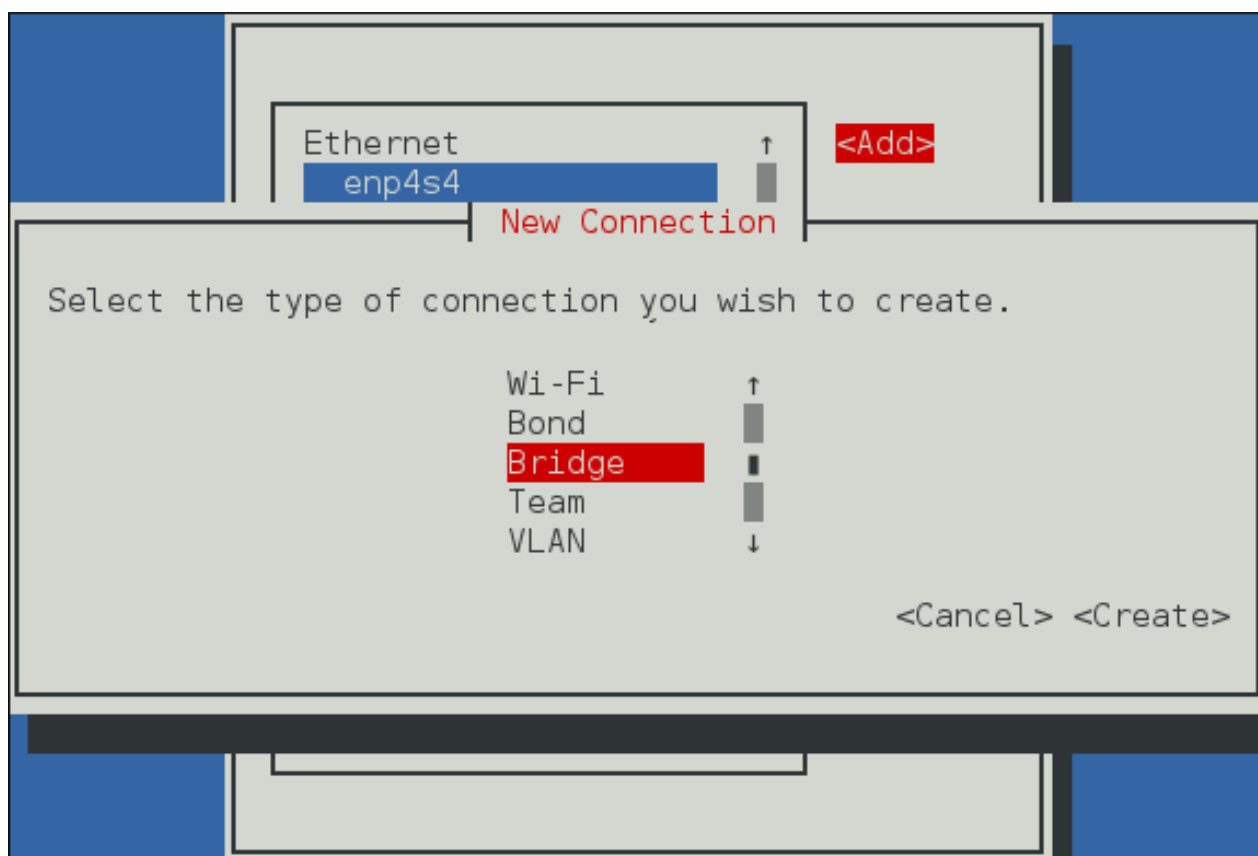


图 6.1. 添加桥接链接的 NetworkManager 文本用户界面菜单

2. 选择 **桥接**，打开 **编辑连接** 页面。
3. 要在桥接中添加从属接口，请选择 **添加** 打开 **新建连接** 页面。选择连接类型后，选择 **创建** 按钮显示 **编辑连接** 页面。

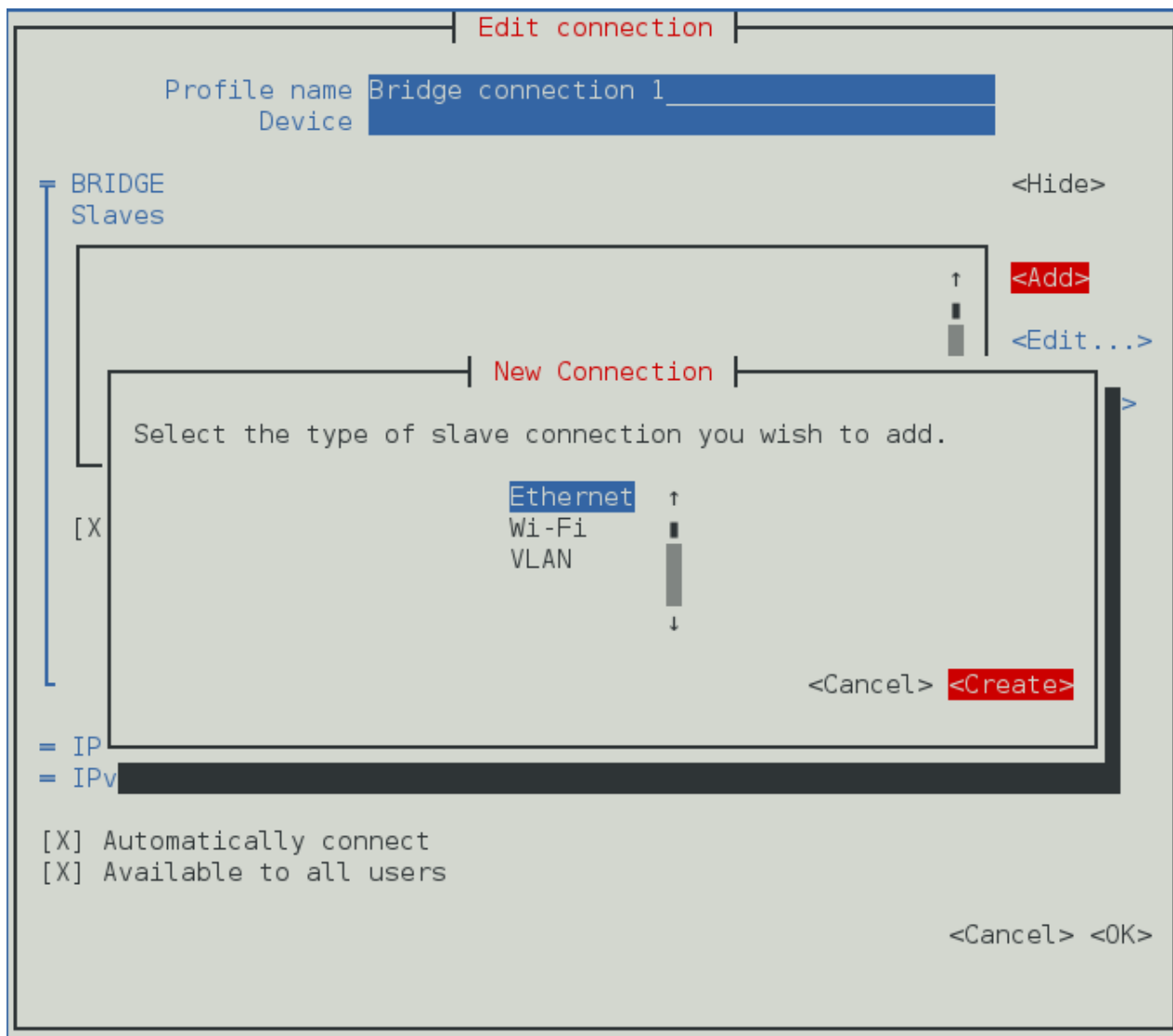


图 6.2. NetworkManager 文本用户界面的添加新桥接从属链接菜单

4. 在 设备 字段输入所需从属设备名称或地址。如果需要，选择 以太网 标签右侧的 显示 输入作为桥接 MAC 地址的克隆 MAC 地址。选择 确定 按钮。



注意

如果指定设备时未指定 MAC 地址，则会在重新载入 编辑连接 后自动填入 设备 字段，前提是它可以成功找到该设备。

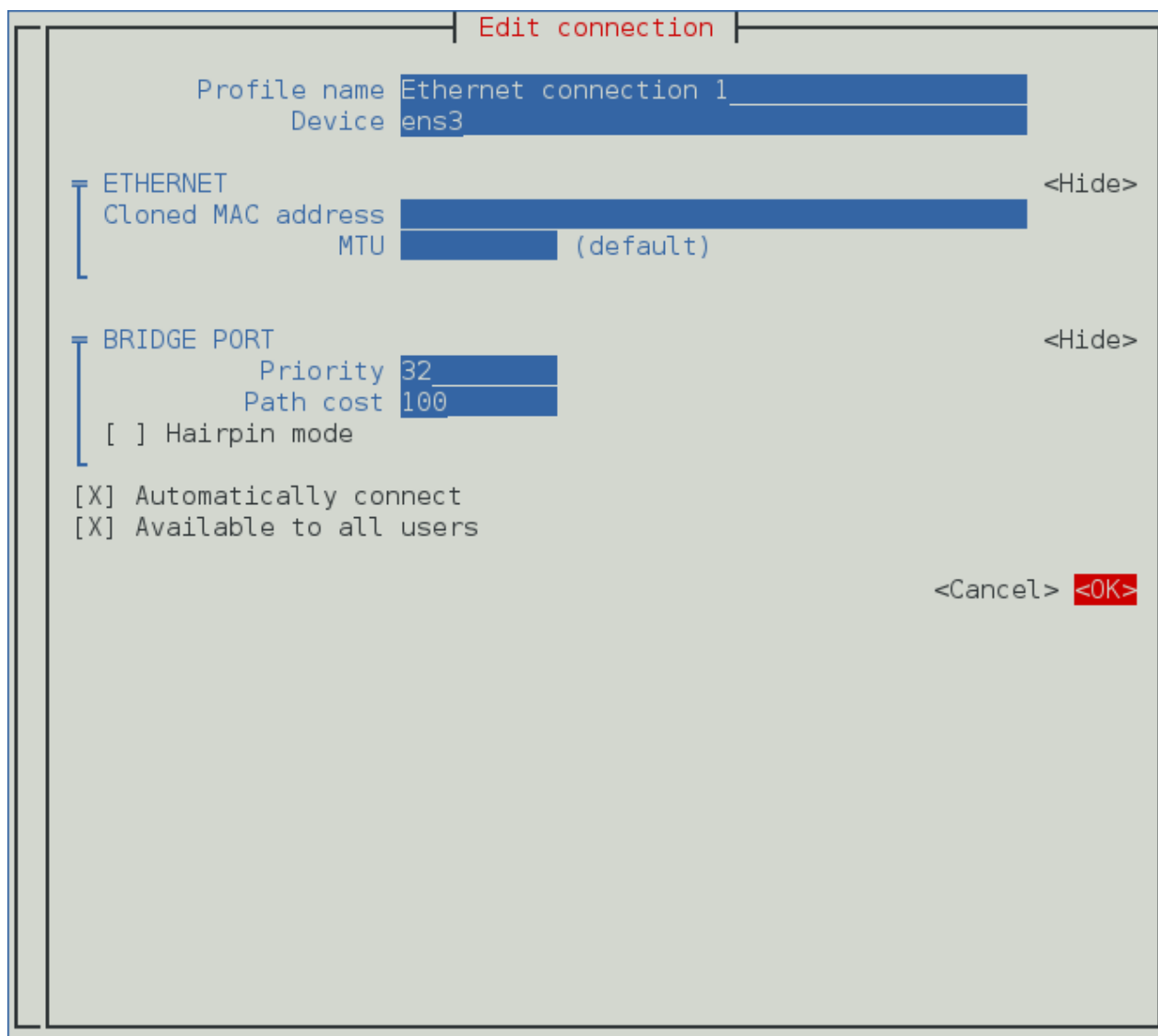


图 6.3. 配置桥接从属连接的 NetworkManager 文本用户界面菜单

5. 从属 部分会显示桥接从属连接名称。重复上述步骤添加更多从属连接。
6. 选择 **确定** 按钮前检查并确认设置。

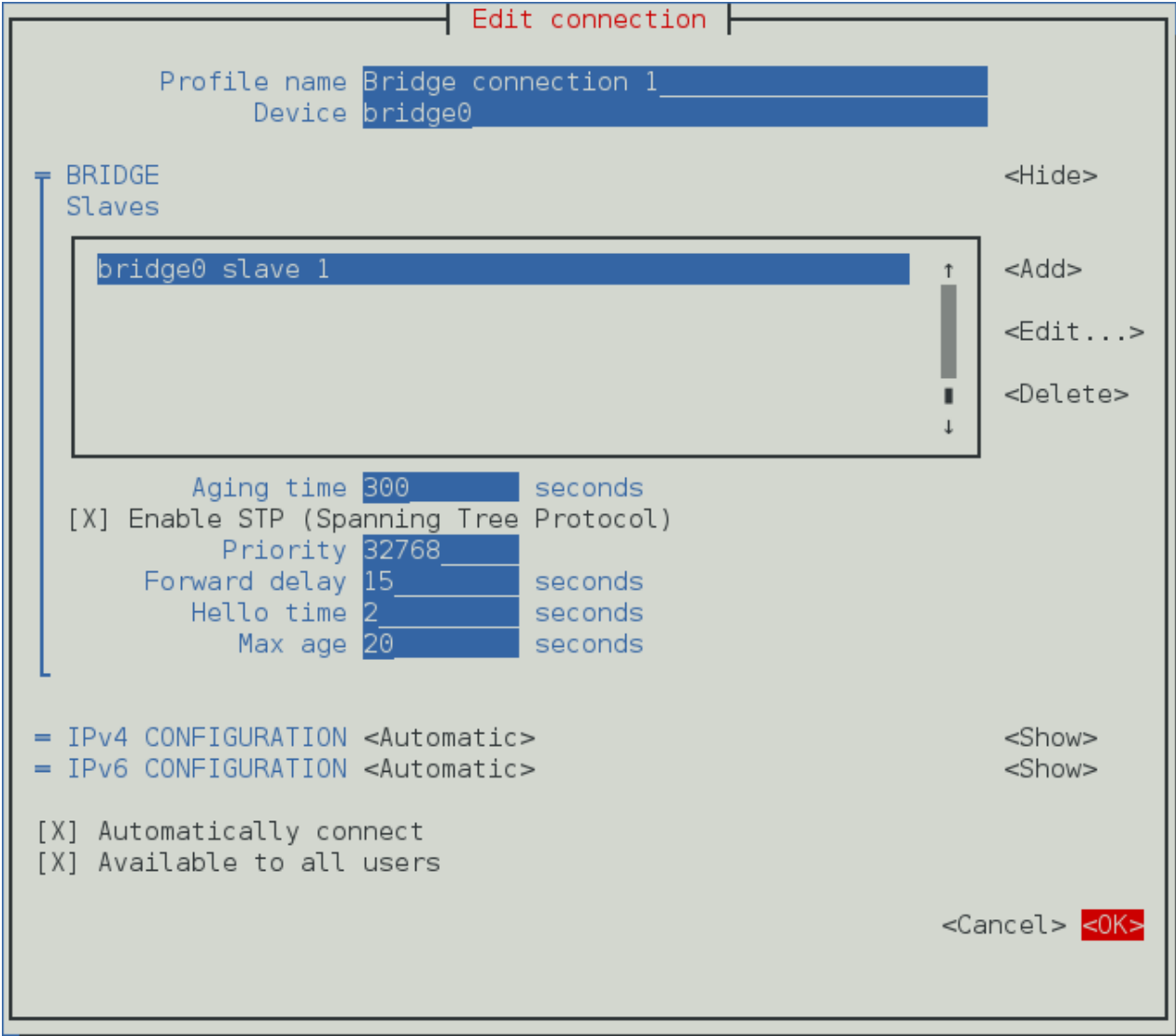


图 6.4. 配置桥接菜单的 NetworkManager 文本用户界面

有关桥接术语请参考 [第 6.4.1.1 节 “配置桥接标签”](#)。
有关 nmtui 的信息请查看 [第 1.5 节 “使用文本用户界面 \(nmtui\) 进行网络配置”](#)。

6.2. 使用 NetworkManager 命令行工具 nmcli

请作为 **root** 运行以下命令创建名为 bridge-br0 的桥接：

```
~]# nmcli con add type bridge ifname br0
Connection 'bridge-br0' (6ad5bba6-98a0-4f20-839d-c997ba7668ad) successfully
added.
```

如果未指定任何接口名称，则默认使用 bridge、bridge-1、bridge-2，以此类推。

运行以下命令查看连接：

```
~]$ nmcli con show
```

NAME	UUID	TYPE	DEVICE
bridge-br0	79cf6a3e-0310-4a78-b759-bda1cc3eef8d	bridge	br0
eth0	4d5c449a-a6c5-451c-8206-3c9a4ec88bca	802-3-ethernet	eth0

默认启用 *跨树协议* (STP)。使用 IEEE 802.1D-1998 标准中的数值。要为这个桥接禁用 **STP**，请作为 **root** 运行以下命令：

```
~]# nmcli con modify bridge-br0 bridge.stp no
```

要为这个桥接重新启用 **802.1D STP**，请作为 **root** 用户运行以下命令：

```
~]# nmcli con modify bridge-br0 bridge.stp yes
```

802.1D STP 的默认桥接优先级为 **32768**。数字越小越倾向于作为 root 桥接使用。例如：使用优先级为 **28672** 的桥接要先于优先级为 **32768** 的桥接（默认）作为 root 桥接使用。要创建使用非默认值的桥接，请运行以下命令：

```
~]$ nmcli con add type bridge ifname br5 stp yes priority 28672
Connection 'bridge-br5' (86b83ad3-b466-4795-aeb6-4a66eb1856c7) successfully
added.
```

允许值范围为 **0** 到 **65535**。

要将现有桥接的桥接优先级改为非默认值，请运行以下格式的命令：

```
~]$ nmcli connection modify bridge-br5 bridge.priority 36864
```

允许值范围为 **0** 到 **65535**。

运行以下命令查看桥接设置：

```
~]$ nmcli -f bridge con show bridge-br0
```

802.1D STP 的更多选项请查看 **nmcli(1)**。

运行以下命令在桥接 bridge-br0 中添加或支配接口，例如 eth1：

```
~]$ nmcli con add type bridge-slave ifname eth1 master bridge-br0
Connection 'bridge-slave-eth1' (70ffae80-7428-4d9c-8cbd-2e35de72476e)
successfully added.
```

写入时，**nmcli** 只支持以太网从属接口。

运行以下命令，使用互动模式更改数值：

```
~]$ nmcli connection edit bridge-br0
```

此时会为您显示 **nmcli** 提示符。

```
nmcli> set bridge.priority 4096
nmcli> save
Connection 'bridge-br0' (79cf6a3e-0310-4a78-b759-bda1cc3eef8d) successfully
saved.
nmcli> quit
```

有关 **nmcli** 的介绍请查看 [第 2.3 节 “使用 NetworkManager 命令行工具 nmcli”](#)。

6.3. 使用命令行界面 (CLI)

6.3.1. 检查是否安装 Bridging 内核模块

在 Red Hat Enterprise Linux 7 中会默认载入 bridging 模块。如有必要，可作为 **root** 运行以下命令确定已载入该模块：

```
~]# modprobe --first-time bridge
modprobe: ERROR: could not insert 'bridge': Module already in kernel
```

运行以下命令显示有关该模块的信息：

```
~]$ modinfo bridge
```

更多命令选项请查看 **modprobe(8)** man page。

6.3.2. 创建网络桥接

要创建网络桥接，请在 `/etc/sysconfig/network-scripts/` 目录中创建名为 `ifcfg-brN` 的文件，使用该接口号替换 *N*，比如 **0**。

该文件的内容和与之建立桥接的接口类型相似，比如以太网接口。本示例的不同之处在于：

- ✦ 为 **DEVICE** 指令分配一个接口名称作为参数，格式为 **brN**，其中使用接口号替换 *N*。
- ✦ 为 **TYPE** 指令分配参数 **Bridge**。这个指令决定设备类型及参数，区分大小写。
- ✦ 为桥接接口配置文件分配 **IP** 地址，其物理接口配置文件必须只含有 MAC 地址（如下）。
- ✦ 在桥接中添加额外指令 **DELAY=0**，防止桥接在监控流量、了解主机位置及构建用来决定主机过滤的 MAC 地址表时等待。如果不可能有任何路由循环，则不需要默认的 15 秒延迟。

例 6.1. ifcfg-br0 接口配置文件示例

以下桥接接口配置文件示例使用静态 **IP** 地址：

```
DEVICE=br0
TYPE=Bridge
IPADDR=192.168.1.1
PREFIX=24
BOOTPROTO=none
ONBOOT=yes
DELAY=0
```

要完成桥接，需创建另一个接口，或修改现有接口，并将其指向桥接接口。

例 6.2. ifcfg-ethX 接口配置文件示例

以下是指向桥接接口的以太网接口配置文件示例。在 `/etc/sysconfig/network-scripts/ifcfg-ethX` 中配置您的物理接口，其中 *X* 是与具体接口对应的独有数字，如下：


```

DEVICE=ethX
TYPE=Ethernet
HWADDR=AA:BB:CC:DD:EE:FF
BOOTPROTO=none
ONBOOT=yes
BRIDGE=br0

```

使用 **NAME** 指令自选指定名称。如果未指定名称，则 **NetworkManager** 插件 **ifcfg-rh** 会为该连接配置文件生成格式为“类型接口”的名称。在这个示例中意味着桥接名为 **Bridge br0**。另外，如果在 **ifcfg-br0** 文件中添加 **NAME=bridge-br0**，则该连接的配置文件名称应为 **bridge-br0**。



注意

在 **DEVICE** 指令中，可使用大多数接口名称，因为它不决定设备类型。不一定需要 **TYPE=Ethernet**。如果未设置 **TYPE**，则需将该设备视为以太网设备（除非其名称与不同的接口配置文件完全匹配）。

指令区分大小写。

使用 **HWADDR** 指令指定硬件或 MAC 地址会影响设备命名过程，如 [第 8 章 一致网络设备命名](#) 所述。



警告

如果在远程主机中配置桥接，同时通过要配置的物理网卡连接到那个主机，请在执行前考虑可能丢失连接的情况。重启该服务可能会丢失连接，同时如果出现任何错误，可能会很难重获连接。建议使用控制台或带外访问。

要启动新或最新配置的接口，请作为 **root**，采用以下格式运行以下命令：

```
ifup device
```

这个命令将探测 **NetworkManager** 是否正在运行，并调用 **nmcli con load UUID**，然后调用 **nmcli con up UUID**。

另外，可作为 **root** 运行以下命令重启所有接口：

```
~]# systemctl restart network
```

这个命令将停止网络服务，启动该网络服务，然后为所有使用 **ONBOOT=yes** 的 **ifcfg** 调用 **ifup**。



注意

NetworkManager 默认不会意识到 **ifcfg** 文件更改，并在该接口下次启动前继续使用旧的配置数据。这是由 **NetworkManager.conf** 文件中的 **monitor-connection-files** 选项设定。详情请查看 **NetworkManager.conf(5)** manual page。

6.3.3. 附带绑定的网络桥接

在此给出由两个或更多绑定的以太网接口组成的网络桥接示例，因为这是虚拟化环境中的常见应用程序。如果不熟悉绑定接口的配置文件，请参考 [第 4.4.2 节“创建频道绑定接口”](#)。

创建或编辑两个或更多绑定的以太网接口配置文件，如下：

```
DEVICE=ethX
TYPE=Ethernet
SLAVE=yes
MASTER=bond0
BOOTPROTO=none
HWADDR=AA:BB:CC:DD:EE:FF
```



注意

最常用的接口名称为 **ethX**，但通常可以使用任何名称。

创建或编辑接口配置文件 `/etc/sysconfig/network-scripts/ifcfg-bond0`，如下：

```
DEVICE=bond0
ONBOOT=yes
BONDING_OPTS='mode=1 miimon=100'
BRIDGE=brbond0
```

有关配置 bonding 模块及查看绑定参数的进一步说明及建议，请查看 [第 4.5 节“使用频道绑定”](#)。

创建或编辑接口配置文件 `/etc/sysconfig/network-scripts/ifcfg-brbond0`，如下：

```
DEVICE=brbond0
ONBOOT=yes
TYPE=Bridge
IPADDR=192.168.1.1
PREFIX=24
```

我们现在有两个或更多包含 **MASTER=bond0** 指令的接口配置文件。这些接入点指向名为 `/etc/sysconfig/network-scripts/ifcfg-bond0` 的配置文件，该文件包含 **DEVICE=bond0** 指令。这个 `ifcfg-bond0` 会按顺序指向 `/etc/sysconfig/network-scripts/ifcfg-brbond0` 配置文件，该文件包含 **IP** 地址，并作为该主机内部的虚拟网络的接口。

要启动新或最新配置的接口，请作为 **root**，采用以下格式运行以下命令：

```
ifup device
```

这个命令将探测 **NetworkManager** 是否正在运行，并调用 `nmcli con load UUID`，然后调用 `nmcli con up UUID`。

另外，可作为 **root** 运行以下命令重启所有接口：

```
~]# systemctl restart network
```

这个命令将停止网络服务，启动该网络服务，然后为所有使用 **ONBOOT=yes** 的 ifcfg 调用 **ifup**。



注意

NetworkManager 默认不会意识到 `ifcfg` 文件更改，并在该接口下次启动前继续使用旧的配置数据。这是由 `NetworkManager.conf` 文件中的 `monitor-connection-files` 选项设定。详情请查看 `NetworkManager.conf(5)` manual page。

6.4. 使用 GUI 配置网络桥接

启动桥接接口时，**NetworkManager** 会在开始配置任何独立网络 IP 前（比如 **DHCP** 或者 **IPv6** 自动配置），至少等待有一个端口进入 “forwarding” 状态。连接任何从属接口或端口，或开始转发数据包前，允许静态 IP 寻址。

6.4.1. 建立桥接连接

过程 6.1. 添加新桥接连接

按照以下步骤创建新桥接连接。

1. 要使用图形 **Network** 设置工具，请按 **Super** 键进入活动概述，输入 `control network`，然后按 **Enter**。此时会出现 **Network** 设置工具。这个步骤在 [第 2.5 节 “在 GNOME 图形用户界面中使用 NetworkManager”](#) 中有具体论述。
2. 选择菜单下方的加号。此时会出现 **添加网络连接** 窗口。
3. 选择 **桥接** 菜单条目。此时会出现 **编辑桥接连接 1** 窗口。

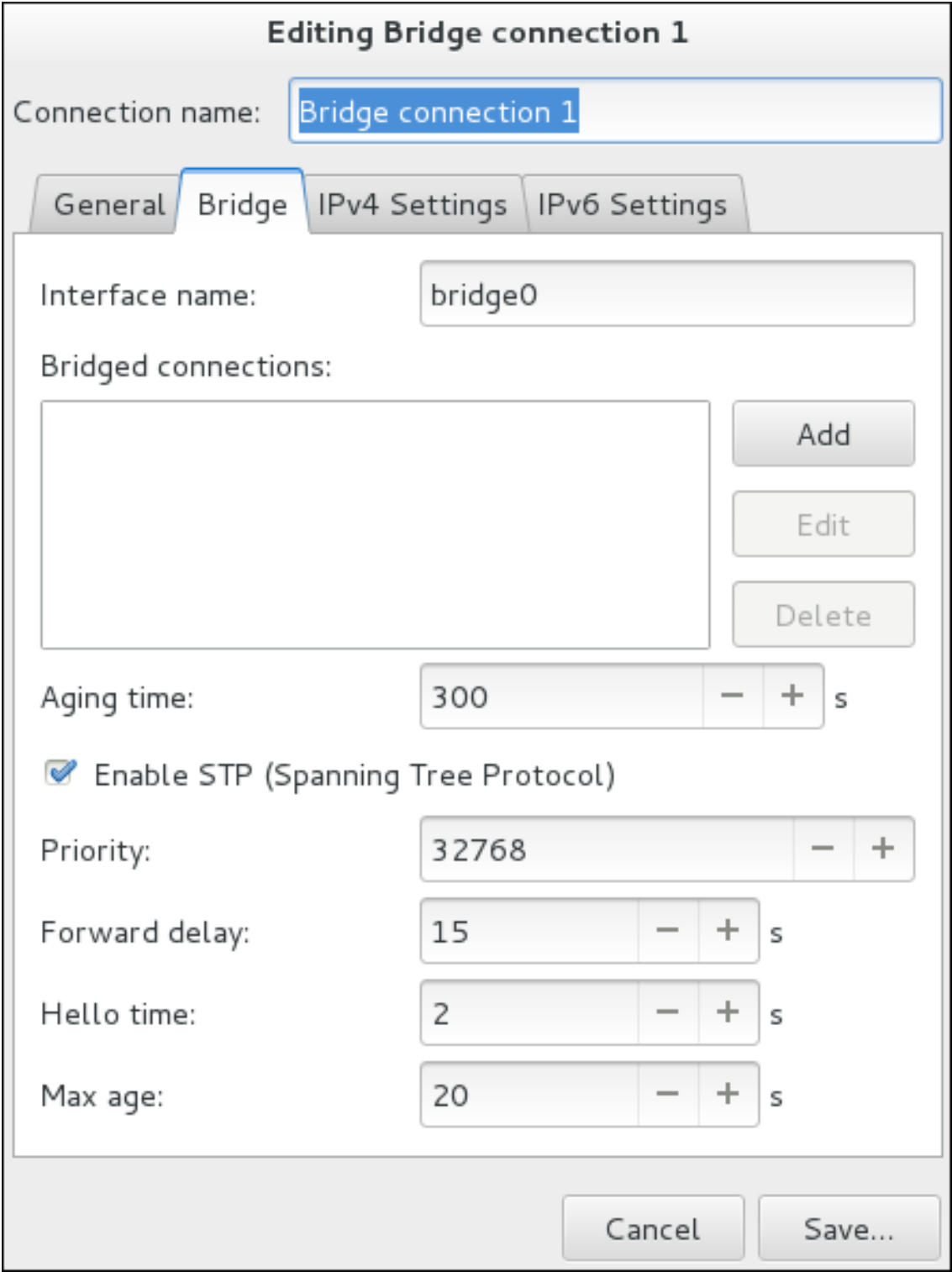


图 6.5. 编辑桥接连接 1

4. 添加从属设备的详情请参考 [过程 6.3, “在桥接中添加从属接口”](#) 如下：

过程 6.2. 编辑现有桥接连接

要编辑现有桥接连接，请打开 **网络** 窗口并从列表中选择该连接。然后单击 **编辑** 按钮。

1. 按 **Super** 键进入活动概述，输入 **control network**，然后按 **Enter** 键。此时会出现 **Network** 设置工具。
2. 在左侧菜单中选择要编辑的 **桥接** 连接。

3. 点击 选项 按钮。

配置连接名称、自动连接行为及可用性设置

编辑 对话框中的五项设置适用于所有连接类型，请查看 **常规** 标签：

- ✱ **连接名称** — 为网络连接输入描述性名称。这个名称可用于在 **网络** 窗口中列出这个连接。
- ✱ **网络可用时自动连接到该网络** — 如果要让 **NetworkManager** 在这个连接可用时自动与之连接，则请选择这个复选框。详情请查看 [第 2.5.3 节 “自动连接到网络”](#)。
- ✱ **所有用户都可以连接到这个网络** — 要创建可用于系统中所有用户的连接，请选择这个复选框。详情请查看 [第 2.5.4 节 “系统范围及专用连接配置文件”](#)。
- ✱ **使用这个连接时自动连接到 VPN** — 如果要让 **NetworkManager** 在 VPN 连接可用时自动与之连接，则请选择这个复选框。请从下拉菜单中选择该 VPN。
- ✱ **Firewall Zone** — 请从下拉菜单中选择防火墙区域。有关防火墙区域的详情请查看 [《Red Hat Enterprise Linux 7 安全指南》](#)。

6.4.1.1. 配置桥接标签

接口名称

连接到桥接的接口名称。

桥接的连接

一个或多个从属接口

老化时间

以秒为单位的时间、MAC 地址将保存在 MAC 地址转发数据库中。

启用 STP（跨树协议）

如有必要，请选择该复选框以便启用 STP。

优先级

桥接优先级；会将优先级最低的桥接作为 root 桥接使用。

转发延迟

进入转发（Forwarding）状态前侦听（Listening）和了解（Learning）状态所消耗时间，单位：秒。

问好时间

使用桥接协议数据单元（BPDU）发送配置信息的间隔，单位：秒。

Max age

保存 BPDU 中配置信息的最长时间，单位：秒。这个是数值应大于 Hello Time + 1，小于 Forwarding Delay -1。

过程 6.3. 在桥接中添加从属接口

1. 要在桥接中添加端口，请在 **编辑桥接连接 1** 窗口中选择 **桥接** 标签，并按照 [过程 6.2, “编辑现有桥接连接”](#) 的步骤打开这个窗口。
2. 点击 **添加** 按钮。此时会出现 **选择连接类型** 菜单。
3. 从该列表中选择要创建的连接的类型。点击 **创建**。此时会出现用来选择连接类型的窗口。

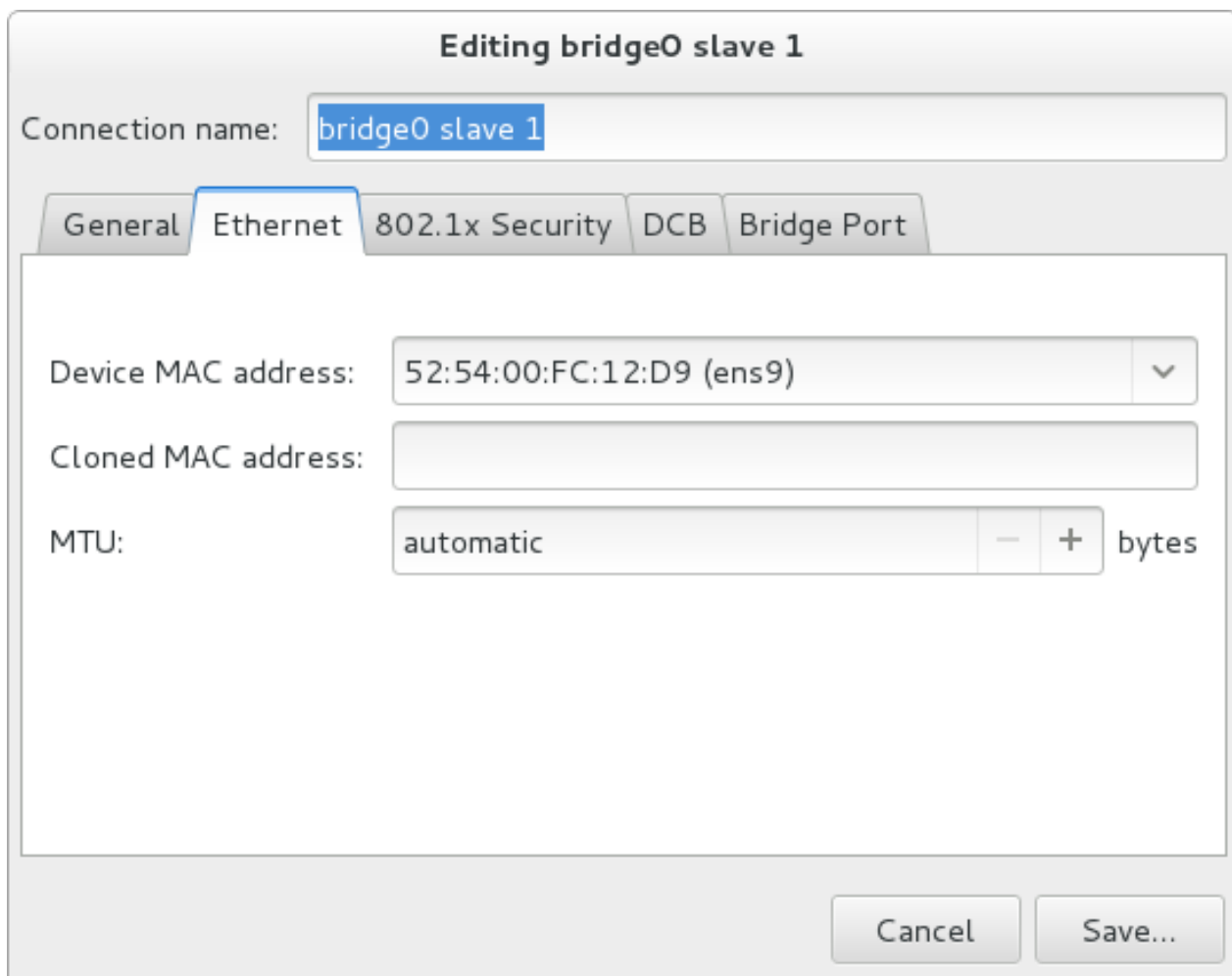


图 6.6. NetworkManager 图形用户界面添加桥接连接

4. 选择 **桥接端口** 标签。根据需要配置 **优先级** 和 **路径成本**。注：桥接的 STP 优先级受 Linux 内核限制。但该标准的允许值为 **0** 到 **255**，Linux 的允许值仅为 **0** 到 **63**。在此默认值为 **32**。

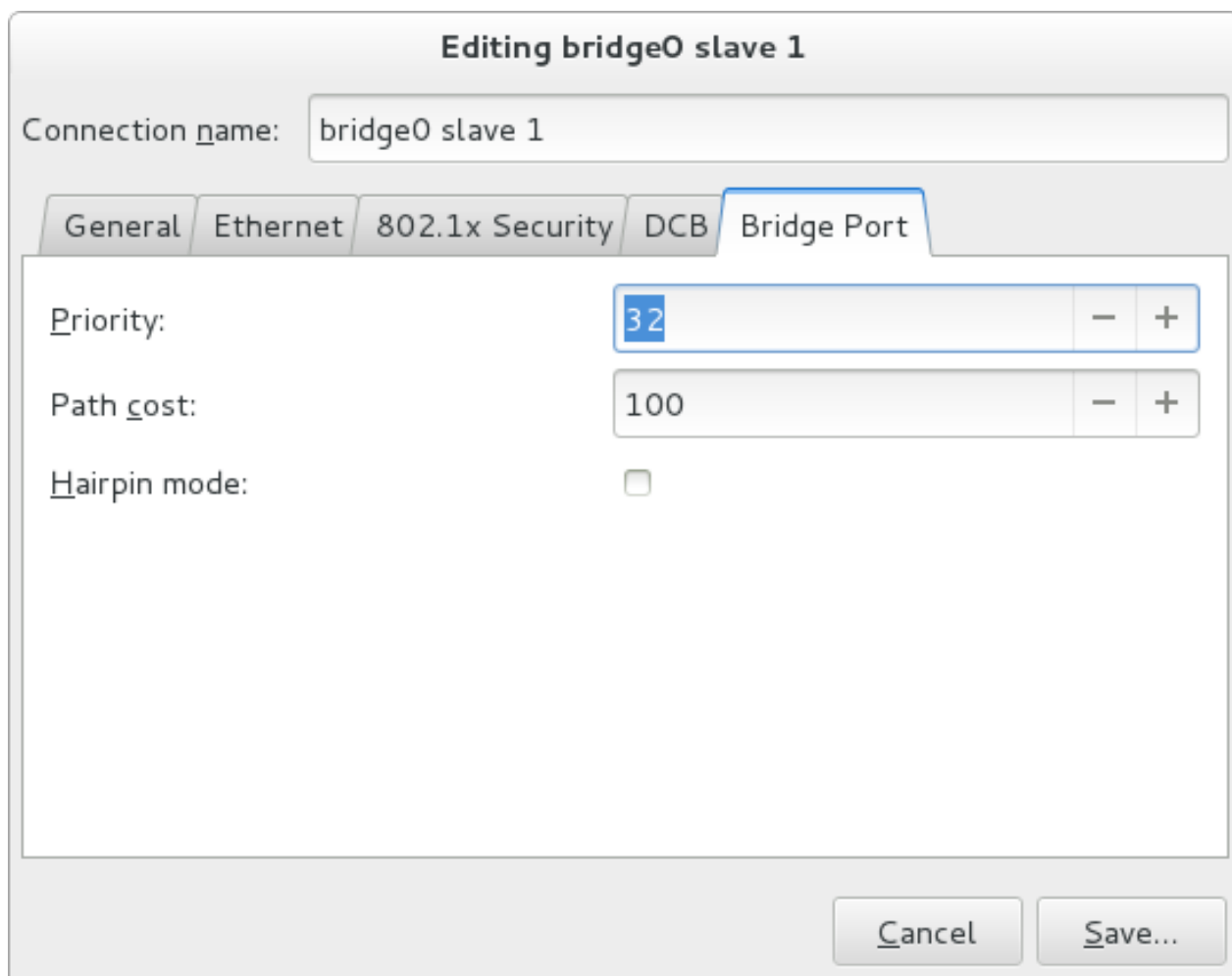


图 6.7. NetworkManager 图形用户界面桥接端口标签

5. 需要时请选择 **Hairpin 模式** 复选框为外部处理启用帧转发功能。该模式也称**虚拟以太网端口汇聚器 (VEPA)** 模式。

然后配置：

- ✧ 一个以太网从属接口，点击 **以太网** 标签并执行 [第 2.5.5.1 节“配置连接名称、自动连接行为及可用性设置”](#)，或者；
- ✧ 一个绑定从属接口，点击 **绑定** 标签并执行 [第 4.6.1.1 节“配置绑定标签”](#)，或者；
- ✧ 一个成组从属接口，点击 **成组** 标签并执行 [第 5.13.1.1 节“配置成组标签”](#)，或者；
- ✧ 一个 VLAN 从属接口，点击 **VLAN** 标签并执行 [第 7.5.1.1 节“配置 VLAN 标签”](#)，或者；

保存新的（或修改的）连接并做进一步配置

完成编辑新桥接连接后，请点击 **保存** 按钮保存您的自定义配置。如果在编辑过程中正在使用该配置文件，则需要重启该连接方可让 **NetworkManager** 应用这些更改。如果该配置文件处于 OFF 状态，请将其设定为 ON，或者在网络连接图标菜单中选中该连接。有关使用新的或更改的连接的信息，请查看 [第 2.5.1 节“使用 GUI 连接到网络”](#)。

若要对现有连接做进一步的配置，请在 **网络** 窗口中选中该连接，并点击 **选项 返回 编辑** 对话框。

然后配置：

- ✧ 若要为该连接进行 **IPv4** 设置，请点击 **IPv4 设置** 标签并执行 [第 2.5.10.4 节“配置 IPv4 设置”](#)，或者；

✧ 若要为该连接进行 **IPv6** 设置，请点击 **IPv6 设置** 标签并执行 [第 2.5.10.5 节“配置 IPv6 设置”](#)

保存后，就会在网络设置工具中显示该桥接及其从属连接。

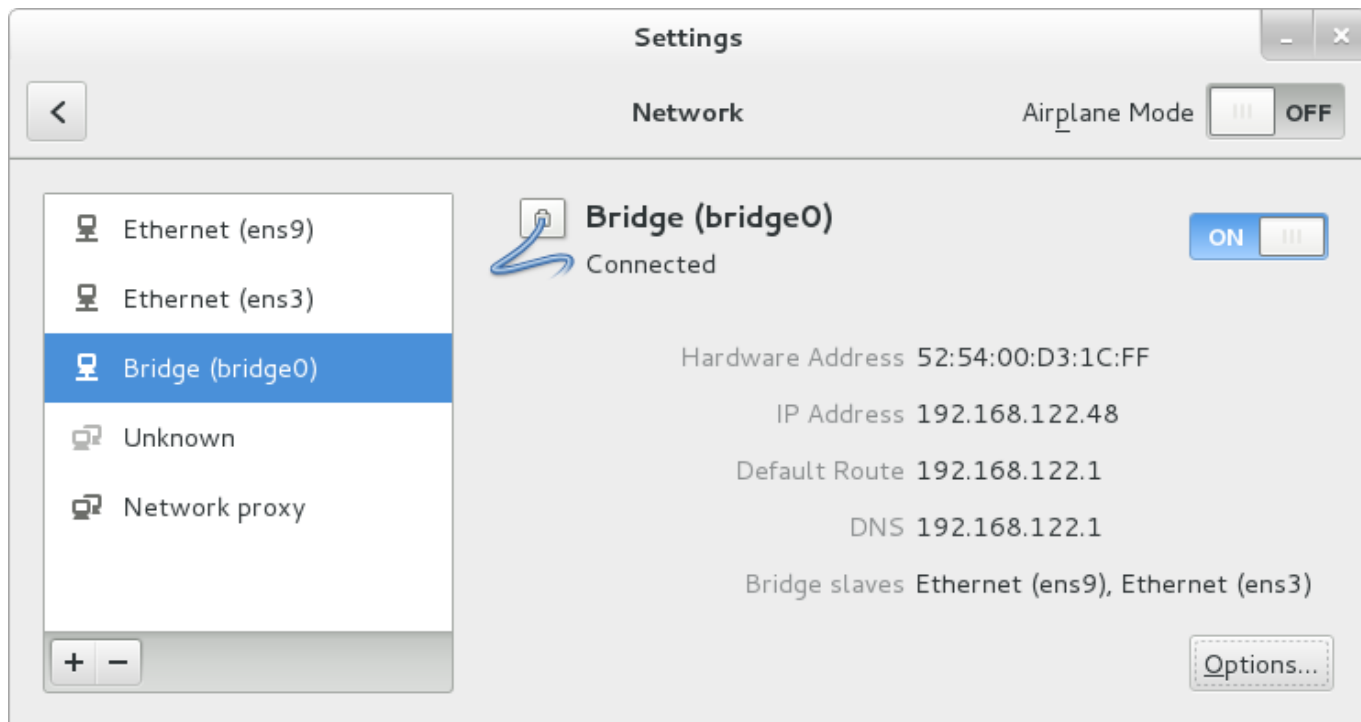


图 6.8. 附带桥接的 NetworkManager 图形用户界面

6.5. 其他资源

以下资源可为您提供有关网络桥接的附加资源。

6.5.1. 已安装文档

- ✧ **nmcli(1)** man page — 论述 **NetworkManager** 的命令行工具。
- ✧ **nmcli-examples(5)** man page — 提供 **nmcli** 命令示例。
- ✧ **nm-settings(5)** man page — 论述 **NetworkManager** 连接的设置及参数。

第 7 章 配置 802.1Q VLAN 标记

要创建 VLAN，需在另一个接口（即**上级接口**）中创建一个接口。VLAN 接口会为数据包添加 VLAN ID 标记，以便其可通过该接口，并为返回的数据包取消标签。可采用与其他接口相似的方式配置 VLAN 接口。上级接口不能是以太网接口。可在网桥（bridge）、绑定（bond）及成组接口（team interface）中创建 802.1Q VLAN 标记接口，但需要注意以下几个方面：

- ✦ 在使用绑定的 VLAN 中，关键是该绑定有从属接口，且在启动 VLAN 接口前，所有从属接口都处于“up”状态。写入时，无法在没有从属接口的绑定中添加 VLAN 接口
- ✦ 无法在使用 **fail_over_mac=follow** 选项的绑定接口中配置 VLAN 从属接口，因为 VLAN 虚拟设备无法更改其 MAC 地址使其与上级接口的新 MAC 地址匹配。在这种情况下，现在仍使用不正确的源 MAC 地址发送流量。
- ✦ 通过网络交换机发送使用 VLAN 标记的数据包时需要配置该交换机。有关交换机的信息请参考其文档。例如：在 Cisco 交换机中，必须将端口分配给一个 VLAN，或者将其配置为中继端口，以便从多个 VLAN 接收标记的数据包。中继端口还可以处理未标记的数据包，并将其视为**本机 VLAN**，但这样做有安全隐患，因此可能已被禁用，或者默认不启用该功能，具体要看交换机的生产厂家。
- ✦ 有些老的网卡、环回接口、Wimax 卡和一些 InfiniBand 设备可能有**VLAN 问题**，就是说不支持 VLAN。这通常是因为这些设备无法处理与标记数据包有关的 VLAN 标头和较大的 MTU。

7.1. 选择 VLAN 接口配置方法

- ✦ 要使用 NetworkManager 的文本用户界面工具 nmtui 配置 VLAN 接口，请执行 [第 7.2 节“使用文本用户界面 nmtui 配置 802.1Q VLAN 标记”](#)。
- ✦ 要使用 NetworkManager 的命令行工具 nmcli 配置 VLAN 接口，请执行 [第 7.3 节“使用命令行工具 nmcli 配置 802.1Q VLAN 标记”](#)。
- ✦ 要手动配置网络接口，请查看 [第 7.4 节“使用命令行配置 802.1Q VLAN 标记”](#)。
- ✦ 要使用图形用户界面工具配置网络，请执行 [第 7.5 节“使用 GUI 配置 802.1Q VLAN 标记”](#)。

7.2. 使用文本用户界面 nmtui 配置 802.1Q VLAN 标记

可在终端窗口中，使用文本界面工具 nmtui 配置 802.1Q VLAN。运行以下命令启动该工具：

```
~]$ nmtui
```

此时会出现文本用户界面。输入任何无效命令都会显示用法信息。

请使用箭头键或按 **Tab** 在选项间前进，按 **Shift+Tab** 后退。按 **Enter** 选择一个选项。按 **Space** 键更改复选框状态。

在开始菜单中选择 **编辑连接**。选择 **添加**，此时会打开 **新建连接** 页面。

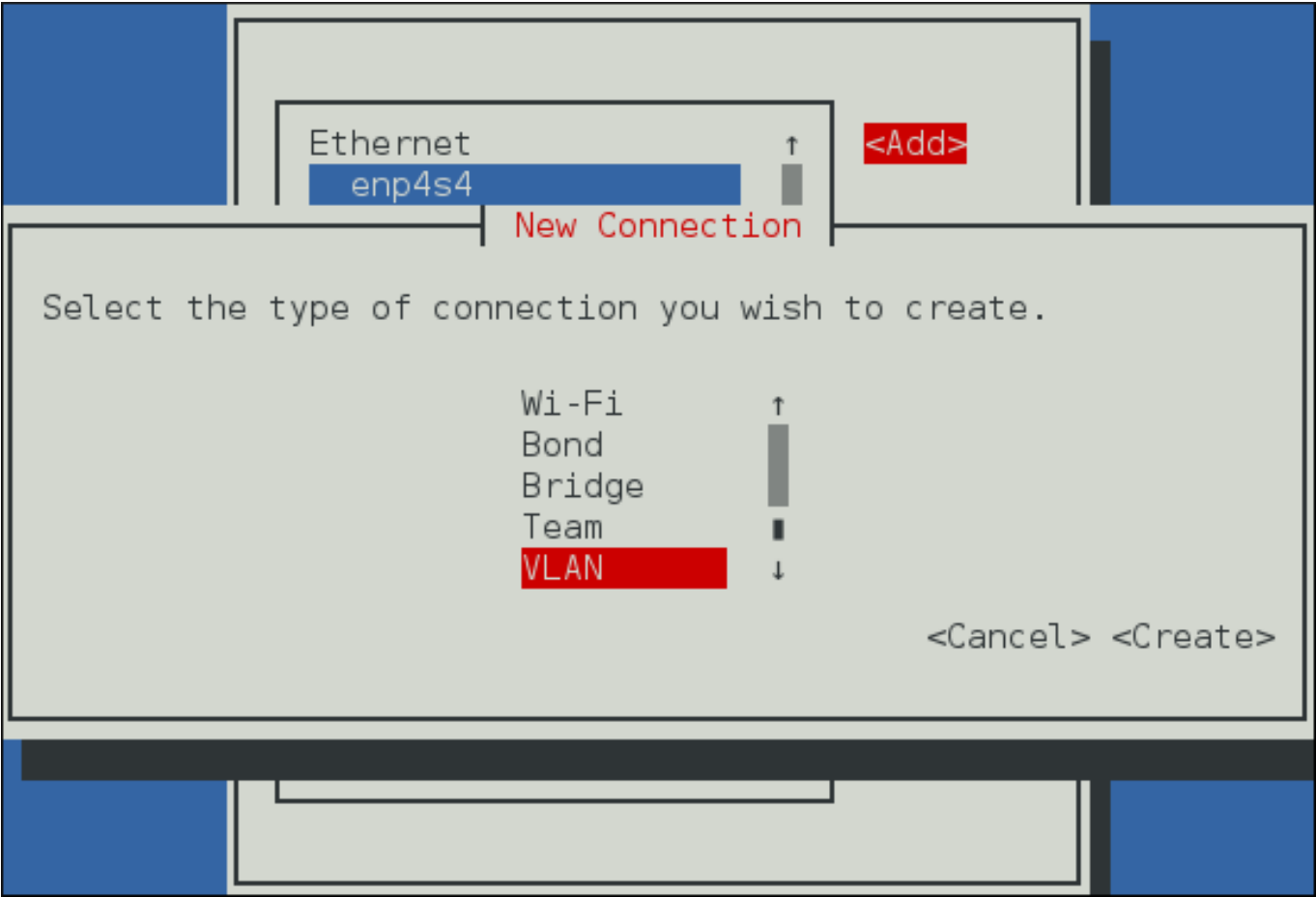


图 7.1. NetworkManager 文本用户界面中的添加 VLAN 连接菜单

选择 **VLAN**，此时会打开 **编辑连接** 页面。按照页面提示完成配置。

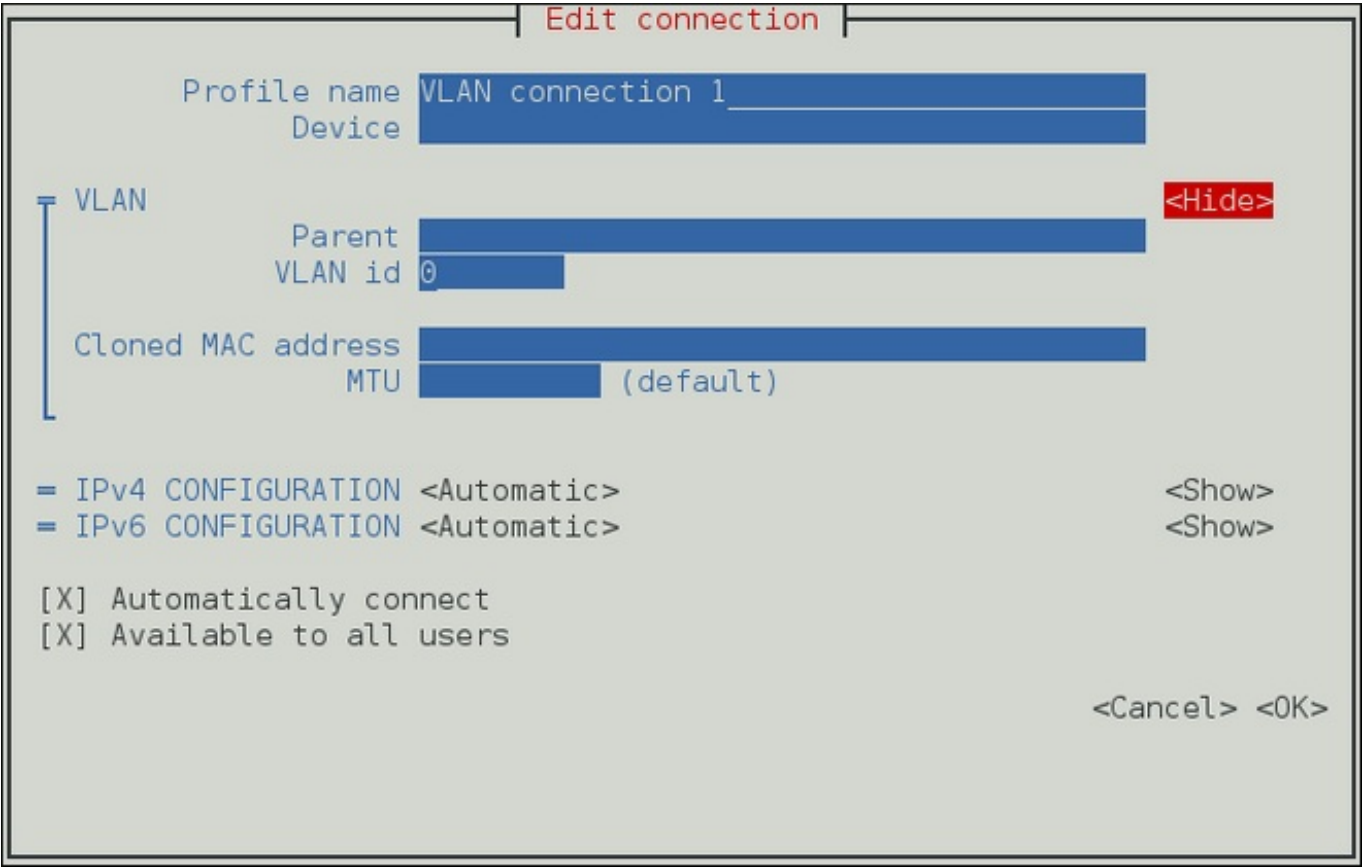


图 7.2. NetworkManager 文本用户界面中的配置 VLAN 连接菜单

有关 VLAN 术语定义详情，请查看 [第 7.5.1.1 节“配置 VLAN 标签”](#)。

有关安装 `nmtui` 的详情，请查看 [第 1.5 节“使用文本用户界面（nmtui）进行网络配置”](#)。

7.3. 使用命令行工具 `nmcli` 配置 802.1Q VLAN 标记

请运行以下命令查看系统中的可用接口：

```
~]$ nmcli con show
NAME                UUID                                TYPE                DEVICE
System eth1         9c92fad9-6ecb-3e6c-eb4d-8a47c6f50c04  802-3-ethernet      eth1
System eth0         5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03  802-3-ethernet      eth0
```

注：输出结果中的 `NAME` 字段总是表示连接 ID，而不是接口名称，尽管它们很相似。可在 `nmcli connection` 命令中使用该 ID 以识别连接。在其他应用程序中可使用 `DEVICE` 名称，比如 `firewalld`。

请运行以下命令，使用 VLAN 接口 `VLAN10` 及 ID `10` 在以太网接口中创建 802.1Q VLAN 接口：

```
~]$ nmcli con add type vlan ifname VLAN10 dev eth0 id 10
Connection 'vlan-VLAN10' (37750b4a-8ef5-40e6-be9b-4fb21a4b6d17) successfully
added.
```

注：没有为 VLAN 接口提供 `con-name`，该名称由接口名称及类型构成。另外，也可以使用 `con-name` 选项指定名称，如下：

```
~]$ nmcli con add type vlan con-name VLAN12 dev eth0 id 12
Connection 'VLAN12' (b796c16a-9f5f-441c-835c-f594d40e6533) successfully
added.
```

为 VLAN 接口分配地址

可使用 `nmcli` 命令分配静态和动态接口地址及其他接口。

例如：创建使用静态 `IPv4` 地址和网关的命令，如下：

```
~]$ nmcli con add type vlan con-name VLAN20 dev eth0 id 20 ip4 10.10.10.10/24 \
gw4 10.10.10.254
```

请运行以下命令创建使用动态分配地址的 VLAN 接口：

```
~]$ nmcli con add type vlan con-name VLAN30 dev eth0 id 30
```

有关使用 `nmcli` 命令配置接口的示例，请查看 [第 2.3.2 节“使用 nmcli 连接到网络”](#)。

请使用以下命令检查创建的 VLAN 接口：

```
~]$ nmcli con show
NAME                UUID                                TYPE                DEVICE
VLAN12              4129a37d-4feb-4be5-ac17-14a193821755  vlan                eth0.12
```

System eth1	9c92fad9-6ecb-3e6c-eb4d-8a47c6f50c04	802-3-ethernet	eth1
System eth0	5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03	802-3-ethernet	eth0
vlan-VLAN10	1be91581-11c2-461a-b40d-893d42fed4f4	vlan	VLAN10

请使用以下命令查看新配置连接的详情：

```
~]$ nmcli -p con show VLAN12
=====
===
                                Connection profile details (VLAN12)
=====
===
connection.id:                  VLAN12
connection.uuid:                4129a37d-4feb-4be5-ac17-
14a193821755
connection.interface-name:      --
connection.type:                vlan
connection.autoconnect:         yes...
-----
802-3-ethernet.port:           --
802-3-ethernet.speed:          0
802-3-ethernet.duplex:         --
802-3-ethernet.auto-negotiate: yes
802-3-ethernet.mac-address:    --
802-3-ethernet.cloned-mac-address: --
802-3-ethernet.mac-address-blacklist:
802-3-ethernet.mtu:            auto...
vlan.interface-name:           --
vlan.parent:                   eth0
vlan.id:                       12
vlan.flags:                    0 (NONE)
vlan.ingress-priority-map:
vlan.egress-priority-map:
-----
=====
===
                                Activate connection details (4129a37d-4feb-4be5-ac17-14a193821755)
=====
===
GENERAL.NAME:                  VLAN12
GENERAL.UUID:                  4129a37d-4feb-4be5-ac17-
14a193821755
GENERAL.DEVICES:               eth0.12
GENERAL.STATE:                 activating[output truncated]
```

VLAN 命令的其他选项，请查看 **nmcli(1)** man pag 的 VLAN 部分。在 man page 中，创建该 VLAN 的设备是作为上级设备使用。在上述示例中是使用其接口名称 **eth0** 指定该设备，也可以使用连接 UUID 或者 MAC 地址指定。

请运行以下命令，使用与以太网接口 **eth1** 映射的入口优先级、名称 **VLAN1** 及 ID **13** 创建 802.1Q VLAN 连接配置文件：

```
~]$ nmcli con add type vlan con-name VLAN1 dev eth2 id 13 ingress "2:3,3:5"
```

请运行以下命令查看与创建上述 VLAN 有关的所有参数：

```
~]$ nmcli connection show vlan-VLAN10
```

请运行以下命令更改 MTU：

```
~]$ nmcli connection modify vlan-VLAN10 802.mtu 1496
```

MTU 设置决定网络层数据包的最大大小。链路层帧能够承受的最大有效负载会反过来限制网络层 MTU。标准以太网帧为 1500 字节 MTU。设置 VLAN 时不需要更改 MTU，因为链接层标头会增大 4 字节以适应 802.1Q 标记。

写入时，**connection.interface-name** 和 **vlan.interface-name** 必须一致（如果设置这两个选项）。因此必须使用 **nmcli** 的互动模式同时更改它们。请运行以下命令更改 VLAN 名称：

```
~]$ nmcli con edit vlan-VLAN10
nmcli> set vlan.interface-name superVLAN
nmcli> set connection.interface-name superVLAN
nmcli> save
nmcli> quit
```

可使用 **nmcli** 程序设置和清除 **ioct1** 标签，这样会更改 802.1Q 代码的作用方式。**NetworkManager** 支持以下 VLAN 标签：

- ✱ 0x01 - 将输出数据包标头重新排序
- ✱ 0x02 - 使用 GVRP 协议
- ✱ 0x04 - 将接口与其上级接口松散绑定

VLAN 的状态与上级接口或主接口（即创建 VLAN 的接口或设备）状态同步。如果将上级接口设定为“down”管理状态，则会将所有关联的 VLAN 设定为 down 状态，并在路由表中刷新所有路由。标签 **0x04** 启用 **松散连接** 模式，在该模式中，可将运行状态从上级接口传递给关联的 VLAN，但不会更改 VLAN 状态。

请运行以下命令设置 VLAN 标签：

```
~]$ nmcli connection modify vlan-VLAN10 vlan.flags 1
```

有关 **nmcli** 的介绍请参考 [第 2.3 节“使用 NetworkManager 命令行工具 nmcli”](#)。

7.4. 使用命令行配置 802.1Q VLAN 标记

在 Red Hat Enterprise Linux 7 中，默认载入 **8021q** 模块。如有必要，可作为 **root** 运行以下命令确定已载入该模块：

```
~]# modprobe --first-time 8021q
modprobe: ERROR: could not insert '8021q': Module already in kernel
```

请运行以下命令显示该模块信息：

```
~]$ modinfo 8021q
```

更多命令选项请查看 **modprobe(8)** man page。

7.4.1. 使用 ifcfg 文件设置 802.1Q VLAN 标记

1. 在 `/etc/sysconfig/network-scripts/ifcfg-ethX` 中配置上级接口，其中 `X` 是与具体接口对应的唯一号码，如下：

```
DEVICE=ethX
TYPE=Ethernet
BOOTPROTO=none
ONBOOT=yes
```

2. 在 `/etc/sysconfig/network-scripts/` 目录中配置 VLAN 接口。配置文件名应为上级接口加上 `.` 字符再加上 VLAN ID 号码。例如：如果 VLAN ID 为 192，上级接口为 `eth0`，那么配置文件名应为 **`ifcfg-eth0.192`**：

```
DEVICE=ethX.192
BOOTPROTO=none
ONBOOT=yes
IPADDR=192.168.1.1
PREFIX=24
NETWORK=192.168.1.0
VLAN=yes
```

如果需要在同一接口 `eth0` 中配置第二个 VLAN，比如 VLAN ID 193，请添加名为 **`eth0.193`** 的新文件，文件中包含 VLAN 配置详情。

3. 重启联网服务以便更改生效。请作为 **root** 运行以下命令：

```
~]# systemctl restart network
```

7.4.2. 使用 ip 命令配置 802.1Q VLAN 标记

要在以太网接口 `eth0` 中创建名为 `VLAN8`、ID 为 **8** 的 802.1Q VLAN 接口，请作为 **root** 运行以下命令：

```
~]# ip link add link eth0 name eth0.8 type vlan id 8
```

请运行以下命令查看 VLAN：

```
~]$ ip -d link show eth0.8
4: eth0.8@eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP mode DEFAULT
    link/ether 52:54:00:ce:5f:6c brd ff:ff:ff:ff:ff:ff promiscuity 0
    vlan protocol 802.1Q id 8 <REORDER_HDR>
```

注：如果使用 **0x** 开头，则 **ip** 程序会将 VLAN ID 解析为十六进制数值，如果使用 **0** 开头，则将其解析为八进制数值。，如果要为 VLAN ID 分配十进制数值 **22**，则一定不能在开头添加任何 0。

请作为 **root** 运行以下命令移除 VLAN：

```
~]# ip link delete eth0.8
```



注意

系统关闭或重启后，会丢失使用 **ip** 命令在命令提示符后创建的 VLAN 接口。要将接口配置为在系统重启后仍保留，请使用 **ifcfg** 文件。详情请查看 [第 7.4.1 节“使用 ifcfg 文件设置 802.1Q VLAN 标记”](#)。

7.5. 使用 GUI 配置 802.1Q VLAN 标记

7.5.1. 建立 VLAN 连接

可将 GNOME **control-center** 程序设定为让 **NetworkManager** 使用现有接口做为上级接口创建 VLAN。写入时只能在以太网设备中创建 VLAN。注：如果将上级接口设定为自动连接，则只能自动创建 VLAN 设备。

过程 7.1. 添加新 VLAN 连接

可打开 **网络** 窗口，点击加号符号，并从列表中选择 **VLAN** 以便添加 VLAN 连接。

1. 按 **Super** 键进入活动概述页面，输入 **control network**，并按 **Enter**。此时会出现 **网络** 设置工具。
2. 点击加号符号打开选择列表。选择 **VLAN**。此时会出现 **编辑 VLAN 连接 1** 窗口。
3. 在 **VLAN** 标签中，从下拉菜单中选择 VLAN 连接要使用的上级接口。
4. 输入 VLAN ID
5. 输入 VLAN 接口名称。这是要创建的 VLAN 接口名称。例如：**eth0.1** 或者 **vlan2**。（通常这可以使上级接口名称+“.”和 VLAN ID，或者“**vlan**”+VLAN ID。）
6. 检查并确认设置，然后点击 **保存** 按钮。
7. 要编辑 VLAN 具体设置，请查看 [第 7.5.1.1 节“配置 VLAN 标签”](#)。

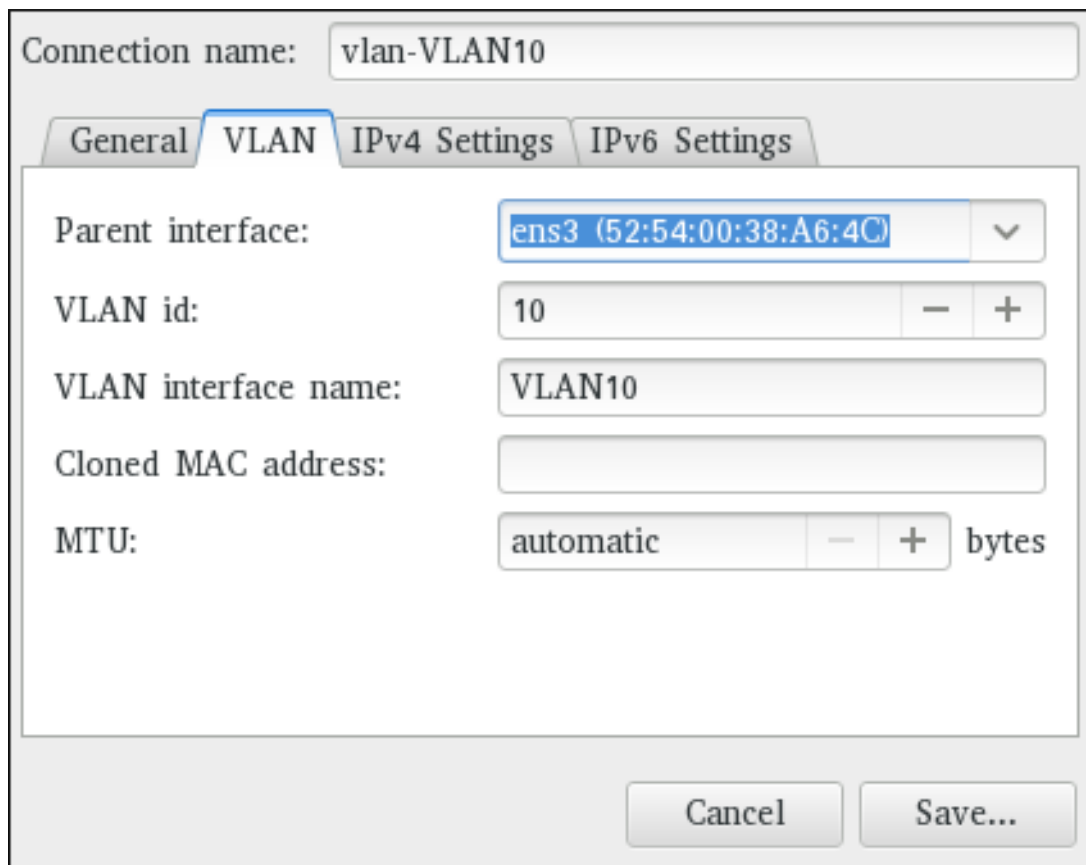


图 7.3. 添加新 VLAN 连接

过程 7.2. 编辑现有 VLAN 连接

按照以下步骤编辑现有 VLAN 连接。

1. 按 **Super** 键进入活动概述页面，输入 **control network**，并按 **Enter**。此时会出现 **网络** 设置工具。
2. 选择要编辑的连接，并点击 **选项** 按钮。
3. 选择 **常规** 标签。
4. 配置连接名称、自动连接行为及可用性设置。

编辑 对话框中的这些设置适用于所有连接类型：

- ✧ **连接名称** — 为您的网络连接输入一个描述性名称。可使用这个名称在 **网络** 窗口的 **VLAN** 部分列出这个连接。
- ✧ **可用时自动连接到这个网络** — 如果要想让 **NetworkManager** 在连接可用时自动连接到该网络，则请选择这个复选框。详情请参考 [第 2.5.3 节“自动连接到网络”](#)。
- ✧ **可用于所有用户** — 如果要想让该连接可用于所有用户，则请选择这个复选框。更改这个设置需要 root 授权。详情请参考 [第 2.5.4 节“系统范围及专用连接配置文件”](#)。

5. 要编辑 VLAN 具体设置，请查看 [第 7.5.1.1 节“配置 VLAN 标签”](#)。

保存新的（或更改的）连接并进行进一步配置

完成编辑 VLAN 连接后，请点击 **保存** 按钮保存自定义配置。如果编辑配置文件时该文件正在使用，则需要重启连接方可让 **NetworkManager** 以能够用所有更改。如果该配置文件处于 OFF 状态，则请将其设定为 ON，或者在网络连接图标菜单中选择它。有关使用新的或更改连接的详情，请查看 [第 2.5.1 节“使用 GUI 连接到网络”](#)。

要进一步配置现有连接，请在 **网络** 窗口，并点击 **选项 返回 编辑** 对话框。

然后，要配置：

- ✱ 该连接的 IPv4 设置，请点击 **IPv4 设置** 标签，执行 [第 2.5.10.4 节“配置 IPv4 设置”](#)。

7.5.1.1. 配置 VLAN 标签

如果已添加新 VLAN 连接（步骤请参考 [过程 7.1, “添加新 VLAN 连接”](#)），则可以编辑 **VLAN** 标签，设定上级接口和 VLAN ID。

上级接口

可在下拉菜单中选择之前配置的接口。

VLAN ID

用来标记 VLAN 网络流量的标识号

VLAN 接口名称

要创建的 VLAN 接口名称。例如：**eth0.1** 或者 **vlan2**。

克隆的 MAC 地址

自选设定一个可替换 MAC 地址，用来识别 VLAN 接口。可使用它为发送到这个 VLAN 的数据包更改源 MAC 地址。

MTU

自选设定在 VLAN 连接在发送数据包使用的最大传输单元（MTU）。

7.6. 其他资料

以下资源提供有关网络成组的附加信息。

7.6.1. 已安装文档

- ✱ **ip-link(8)** man page — 描述 **ip** 程序的网络设备配置命令。
- ✱ **nmcli(1)** man page — 描述 **NetworkManager** 的命令行工具。
- ✱ **nmcli-examples(5)** man page — 提供 **nmcli** 命令示例。
- ✱ **nm-settings(5)** man page — 描述 **NetworkManager** 连接的设置及参数。

第 8 章 一致网络设备命名

Red Hat Enterprise Linux 7 提供在网络接口中使用一致且可预期的网络设备命名方法。这些功能会更改系统中的网络接口名称，以便定位和区分这些接口。

通常 Linux 中的网络接口枚举如下 **eth[0123...]**，但这些名称不一定与底盘实际标签对应。使用多个网络适配器的现代服务器平台会有不确定和不直观的接口命名。这会影响到主板内嵌的网络适配器（集成网卡主板（*Lan-on-Motherboard*），或 LOM）及外接（单个或多个）适配器。

在 Red Hat Enterprise Linux 7 中，**udev** 支持大量不同的命名方案。默认是根据固件、拓扑及位置信息分配固定名称。这样做的优点是命名可完全自动进行，并可预期，即使添加或删除硬件后也会保留其名称（不会出现重复枚举的情况），同时可顺利更换损坏的硬件。不足之处是，相比传统的名称，比如 **eth0** 或 **wlan0**，这些名称有时会比较难理解。例如：**enp5s0**。

8.1. 命名方案层级结构

默认情况下，**systemd** 会使用以下策略，采用支持的命名方案为接口命名：

- ✦ **方案 1**：如果固件或 BIOS 信息适用且可用，则使用整合了为板载设备提供索引号的固件或 BIOS 的名称（例如：**eno1**），否则请使用方案 2。
- ✦ **方案 2**：如果固件或 BIOS 信息适用且可用，则使用整合了为 PCI 快速热插拔插槽提供索引号的固件或 BIOS 名称（例如：**ens1**），否则请使用方案 3。
- ✦ **方案 3**：如果硬件连接器物理位置信息可用，则使用整合了该信息的名称（例如：**enp2s0**），否则请使用方案 5。
- ✦ **方案 4**：默认不使用整合接口 MAC 地址的名称（例如：**enx78e7d1ea46da**），但用户可选择使用此方案。
- ✦ **方案 5**：传统的不可预测的内核命名方案，在其他方法均失败后使用（例如：**eth0**）。

这个策略（如上所述）是默认策略。如果该系统已启用 **biosdevname**，则会使用该方案。注：启用 **biosdevname** 需要添加 **biosdevname=1** 作为命令行参数（Dell 系统除外），此时只要安装 **biosdevname**，就会默认使用该方案。如果用户已添加 **udev** 规则，该规则会更高内核设备名称，则会优先使用这些规则。

8.2. 了解设备重命名过程

设备命名过程如下：

1. **/usr/lib/udev/rules.d/60-net.rules** 文件中的规则会让 **udev** 帮助工具 **/lib/udev/rename_device** 查看所有 **/etc/sysconfig/network-scripts/ifcfg-suffix** 文件。如果发现包含 **HWADDR** 条目的 **ifcfg** 文件与某个接口的 MAC 地址匹配，它会将该接口重命名为 **ifcfg** 文件中由 **DEVICE** 指令给出的名称。
2. **/usr/lib/udev/rules.d/71-biosdevname.rules** 中的规则让 **biosdevname** 根据其命名策略重命名该接口，即在上一步中没有重命名该接口、已安装 **biosdevname**、且在 **boot** 命令中将 **biosdevname=0** 作为内核命令给出。
3. **/lib/udev/rules.d/75-net-description.rules** 中的规则让 **udev** 通过检查网络接口设备，填写内部 **udev** 设备属性值 **ID_NET_NAME_ONBOARD**、**ID_NET_NAME_SLOT**、**ID_NET_NAME_PATH**。注：有些设备属性可能处于未定义状态。

4. `/usr/lib/udev/rules.d/80-net-name-slot.rules` 中的规则让 `udev` 重命名该接口，优先顺序如下：ID_NET_NAME_ONBOARD、ID_NET_NAME_SLOT、ID_NET_NAME_PATH。并提供如下信息：没有在步骤 1 或 2 中重命名该接口，同时未给出内核参数 `net.ifnames=0`。如果一个参数未设定，则会按列表的顺序设定下一个。如果没有设定任何参数，则不会重命名该接口。

第 3 步和第 4 步采用命名规则 1、2、3，可自选方案 4，如 [第 8.1 节“命名方案层级结构”](#) 所述。第 2 步在 [第 8.6 节“使用 biosdname 保持网络设备命名一致”](#) 中有详细论述。

8.3. 了解可预期网络接口设备名称

根据接口类型以两个字母开头：

- 1. `en` 代表以太网，
- 2. `wl` 代表无线局域网（WLAN），
- 3. `ww` 代表无线广域网（WWAN）。

名称有以下类型：

表 8.1. 设备名称类型

格式	描述
<code>o<index></code>	板载设备索引号
<code>s<slot>[f<function>][d<dev_id>]</code>	热插拔插槽索引号
<code>x<MAC></code>	MAC 地址
<code>p<bus>s<slot>[f<function>][d<dev_id>]</code>	PCI 地理位置
<code>p<bus>s<slot>[f<function>][u<port>][.][c<config>][i<interface>]</code>	USB 端口链

- 所有多功能 PCI 设备都在其设备名称中包含 `[f<function>]` 号，其中包括 `function 0` 设备。
- 在 USB 设备中会组成集线器端口号完整链。如果该名称超过 15 个字符上限，则无法导出该名称。
- 已取消 `USB configuration descriptors == 1` 和 `USB interface descriptors == 0`（如果只有一个 USB 配置或接口存在，则默认值为 `configuration == 1` 及 `interface == 0`）。

8.4. 在 System z 中用于 Linux 系统的网络设备命名规则

使用总线 ID 为 System z 实例中的 Linux 系统网络接口生成可预期设备名称。该总线 ID 可识别 s390 频道子系统中的设备。总线 ID 可识别 Linux 实例范围内的设备。对于 CCW 设备而言，该总线 ID 是该设备以 `0.n` 开头的设备号，其中 `n` 是子频道组 ID。例如：`0.1.0ab1`。

以太网设备类型的网络接口命名方式如下：

```
enccw0.0.1234
```

设备类型 SLIP 的 CTC 网络设备命名方式如下：

```
slccw0.0.1234
```

使用 `znetconf -c` 命令或 `lscss -a` 命令显示可用网络设备及其总线 ID。

表 8.2. System z 中 Linux 系统的设备名称类型

格式	描述
enccwbus-ID	以太网设备类型
slccwbus-ID	设备类型 SLIP 的 CTC 网络设备

8.5. VLAN 接口命名方案

通常，VLAN 接口名称格式为：*interface-name.VLAN-ID*。**VLAN-ID** 范围为 **0** 到 **4096**，最多为四位数，接口名称总计不超过 15 个字符。接口名称的长度限制由内核标头规定，且是一个通用限制，会影响所有应用程序。

在 Red Hat Enterprise Linux 7 中支持四种 VLAN 接口名称的命名规则：

VLAN + VLAN ID

单词 **vlan** 加上 VLAN ID。例如：vlan0005

VLAN + VLAN ID，不填充

单词 **vlan** 加上 VLAN ID，不会在前面添加额外的两个零。例如：vlan5

设备名称 + VLAN ID

上级接口名称加上 VLAN ID。例如：eth0.0005

设备名称 + VLAN ID，不填充

上级接口名称加上 VLAN ID，不中前面添加额外的两个零。例如：eth0.05

8.6. 使用 biosdevname 保持网络设备命名一致

通过 **biosdevname udev** 帮助程序实施此功能，可将所有内嵌网络接口名称、PCI 卡网络接口名称、以及现有 **eth[0123...]** 的虚拟功能网络接口名称改为新的命名规范，如 [表 8.3 “biosdevname 命名惯例”](#) 所示。注：除非使用 Dell 系统，或特别明确说明启用 **biosdevname**（如 [第 8.6.2 节 “启用和禁用该功能”](#) 所述），否则会优先使用 **systemd** 命名惯例。

表 8.3. biosdevname 命名惯例

设备	旧名称	新名称
内嵌网络接口（LOM）	eth[0123...]	em[1234...] [a]
PCI 卡网络接口	eth[0123...]	p<slot>p<ethernet port> [b]
虚拟功能	eth[0123...]	p<slot>p<ethernet port>_<virtual interface> [c]
[a] 新枚举从 1 开始。		
[b] For example: p3p4		
[c] For example: p3p4_1		

8.6.1. 系统要求

biosdevname 程序使用来自系统 BIOS 的信息，特别是 SMBIOS 中包含。type 9（系统插槽）和 type 41（板载设备扩展信息）字段。如果系统的 BIOS 没有 SMBIOS 版本 2.6 或更高版本和这个数据，则不会使用新的命名规则。大多数老硬件不支持这个功能，因为缺少有正确 SMBIOS 版本的 BIOS 和字段信息。有关 BIOS 和 MSBIOS 版本信息，请联络您的硬件销售商。

必须安装 `biosdevname` 软件包方可或使用这个功能。要安装这个软件包，请作为 **root** 用户运行以下命令：

```
~]# yum install biosdevname
```

8.6.2. 启用和禁用该功能

要禁用这个功能，请在安装过程中及安装后，在 `boot` 命令行中使用以下选项：

```
biosdevname=0
```

要启用这个功能，请在安装过程中及安装后，在 `boot` 命令行中使用以下选项：

```
biosdevname=1
```

除非系统达到最低要求，否则会忽略这个选项，同时系统会使用 **systemd** 命名方案，如本章开始部分所述。

如果指定 **biosdevname** 安装选项，那么它就必须在该系统的声明周期内作为其引导选项使用。

8.7. 管理员备注

很多系统自定义文件都包含网络接口名称，因此在系统中使用新的命名惯例时需要更新这些文件。如果使用新的命名惯例，则还需要在自定义 **iptables** 规则、脚本变更 **irqbalance** 及其他类似配置文件中更新网络接口名称。同时，为安装启用这个更改还要求修改现有 **kickstart** 文件（该文件通过 **ksdevice** 使用设备名称）。需要将这些 **kickstart** 文件更新为使用网络设备的 MAC 地址或网络设备的新名称。



注意

接口名称的长度限制由内核标头规定，且是一个通用限制，会影响所有应用程序。

8.8. 控制网络设备名称选择

可以如下方式控制设备命名：

根据网络接口设备识别

在 **ifcfg** 文件中使用 **HWADDR** 指令设定 MAC 地址，这样就可由 **udev** 识别。会从 **DEVICE** 指令提供的字符串中提取该名称，根据惯例，该名称应使用与 **ifcfg** 相同的后缀。例如：**ifcfg-eth0**。

通过打开或关闭 biosdevname

可使用由 **biosdevname** 提供的名称（如果 **biosdevname** 可确定）。

通过打开或关闭 systemd-udev 的命名方案

可使用由 **systemd-udev** 提供的名称（如果 **systemd-udev** 可确定）。

8.9. 禁用一致网络设备命名

请选择以下方法之一禁用一致网络设备命名：

✎ 通过屏蔽默认策略中的 **udev** 规则文件 禁止分配固定名称 以便重新使用不可预期的内核名称 可为

通过屏蔽来自内核的 **udev** 规则文件，禁止为接口分配名称，以便系统使用与可预期的网络接口。通过 **/dev/null** 生成一个符号链接完成“屏蔽”。请作为 **root** 用户运行以下命令：

```
~]# ln -s /dev/null /etc/udev/rules.d/80-net-name-slot.rules
```

- ✧ 创建自己的手动命名方案。例如：将接口命名为“internet0”、“dmz0”或“lan0”。要创建自己的 **udev** 规则文件，并为那些设备设置 **NAME** 属性。确定在使用默认策略文件前使用该文件。例如：将其命名为 **/etc/udev/rules.d/70-my-net-names.rules**。
- ✧ 修改策略文件，使其选择不同的命名方案后。例如：默认根据接口的 **MAC** 地址命名所有接口。作为 **root** 复制默认策略文件，如下：

```
~]# cp /usr/lib/udev/rules.d/80-net-name-slot.rules /etc/udev/rules.d/80-net-name-slot.rules
```

在 **/etc/udev/rules.d/** 目录中编辑文件，并根据需要修改。

- ✧ 在 GRUB 2 菜单的内核命令行中添加以下指令：

```
net.ifnames=0
```

更新所有 GRUB 2 内核菜单条目，作为 **root** 用户输入以下命令：

```
~]# grubby --update-kernel=ALL --args=net.ifnames=0
```

有关使用 GRUB 2 的详情请查看 [《Red Hat Enterprise Linux 7 系统管理员指南》](#)。

8.10. 网络设备命名故障排除

如果可能，会根据 [第 8.2 节“了解设备重命名过程”](#) 所述过程为每个接口分配一个可预期的接口名称。要查看 **udev** 可能使用的名称列表，请作为 **root** 运行以下命令：

```
~]# udevadm info /sys/class/net/iface | grep ID_NET_NAME
```

其中 **iface** 是用以下命令中列出的接口之一：

```
~]$ ls /sys/class/net/
```

udev 会根据 [第 8.2 节“了解设备重命名过程”](#) 所述规则应用可能的名称之一，并总结如下：

- ✧ **/usr/lib/udev/rules.d/60-net.rules** - 来自 **initscripts**，
- ✧ **/usr/lib/udev/rules.d/71-biosdevname.rules** - 来自 **biosdevname**，
- ✧ **/usr/lib/udev/rules.d/80-net-name-slot.rules** - 来自 **systemd**

从以上规则文件列表中可以看出，如果使用 **initscripts** 或 **biosdevname** 完成接口命名，它总是优先于 **udev** 原始命名。但如果没有使用 **initscripts** 命名，同时禁用了 **biosdevname**，那么要更改接口名称，就需要将 **/usr/** 中的 **80-net-name-slot.rules** 复制到 **/etc/**，并相应编辑该文件。换句话说，根据具体顺序注释出或安排要使用的方案。

例 8.1. 有些接口使用来自内核名称空间的名称（eth[0,1,2...]），同时可使用 **udev** 重命名其他名称。

混合方案大多是因为内核无法向 **udev** 提供某些硬件的可用信息，因此无法确定为 **udev** 提供的名称或信息是否合适，比如非唯一设备 ID。后者更常见，通常可使用 `ifcfg` 文件中的秘密方案，或编辑 `80-net-name-slot.rules` 更改使用的 **udev** 方案解决这个问题。

例 8.2. 在 `/var/log/messages` 或 `systemd` 日志中可以看到为每个设备重新命名两次。

使用 `ifcfg` 内置命名方案，但未重新创建 `initrd` 映像的系统通常会遇到这个问题。最初，在引导初期且仍处于 `initrd` 中时分配该接口名称（使用 `biosdevname` 或 `udev`）。当切换到真实 `rootfs` 后，会第二次重新命名，并由 `udev` 衍生的 `/usr/lib/udev/rename_device` 二进制文件决定新接口名称，因为使用的是 `60-net.rules`。可忽略此类信息。

例 8.3. 在附带 `ethX` 名称的 `ifcfg` 文件中使用命名方案不可行

不建议使用来自内核名称空间的接口名称。要获得可预期且稳定的接口名称，请使用 "eth" 以外的其他前缀。

8.11. 其他资料

以下资源提供有关网络成组的附加信息。

8.11.1. 已安装文档

- ✦ **udev(7)** man page — 描述 Linux 动态设备管理守护进程 `udevd`。
- ✦ **systemd(1)** man page — 描述 `systemd` 系统和服务管理器。
- ✦ **biosdevname(1)** man page — 描述获取 BIOS 给定设备名称的程序。

8.11.2. 在线文档

- ✦ IBM 知识中心出版物 SC34-2710-00 [《Red Hat Enterprise Linux 7 中的设备驱动程序、功能及命令行》](#) 中包含用于 IBM z 设备及附件的“可预期网络设备名称”信息。

部分 II. InfiniBand 和 RDMA 联网

这部分论述了如何设置 RDMA、InfiniBand 及通过 InfiniBand 网络连接的 IP。

第 9 章 配置 InfiniBand 和 RDMA 网络

9.1. 了解 InfiniBand 和 RDMA 技术

InfiniBand 是指两种完全不同的东西。一个是 InfiniBand 网络的物理链接层协议，另一个是高级编程 API，名为 InfiniBand Verbs API。InfiniBand Verbs API 是一种 *远程直接内存访问*（RDMA）技术的实施。

RDMA 通讯与一般 IP 通讯不同，原因是他们会绕过通讯过程中的内核干扰，并极大减少一般处理网络通讯所需的 CPU 消耗。在典型的 IP 数据传输中，机器 A 中的应用程序 X 会向机器 B 中的应用程序 Y 发送同样的数据。作为传输的一部分，机器 B 的内核必须首先接收数据，解码数据包标头，确定该数据属于应用程序 Y，然后唤醒应用程序 Y，等待应用程序 Y 在内核中执行读取 syscall，然后必须手动将该数据从内核的自主内部内存空间复制到应用程序 Y 提供的缓存中。这个过程意味着大多数网络流量都必须在系统的主内存总线间至少复制两次（一次是主机适配器使用 DMA 将该数据放到内核提供的内存缓存中，另一次是内核将该数据移动到应用程序的内存缓存中），同时也意味着计算机必须执行大量上下文切换，以便在内核上下文和应用程序 Y 上下文之间进行切换。这些操作都会在网络通讯处于极高频率时造成极高的系统 CPU 负载。

RDMA 协议可让机器中的主机适配器了解网络何时会有数据包，那个应用程序应该接收该数据包，以及该数据应进入该应用程序内存空间的那个部分。这样就不会向内核发送该数据包并进行处理，然后再复制到用户的应用程序内存，而是将该数据包直接放到应用程序的缓存中，没有任何进一步的干预。这样就可以极大减少高速网络通讯的负载。但大多数 IP 联网应用程序依赖的标准伯克利套接字 API 无法实现此功能，因此必须提供其自己的 API，即 InfiniBand Verbs API，同时必须在应用程序可直接使用 RDMA 技术前将其移植到这个 API。

Red Hat Enterprise Linux 7 supports both the InfiniBand hardware and the InfiniBand Verbs API. In addition, there are two additional supported technologies that allow the InfiniBand Verbs API to be utilized on non-InfiniBand hardware. They are iWARP (Internet Wide Area RDMA Protocol) and RoCE/IBoE (RDMA over Converged Ethernet, which was later renamed to InfiniBand over Ethernet). Both of these technologies have a normal IP network link layer as their underlying technology, and so the majority of their configuration is actually covered in the [第 2 章 配置 IP 联网](#) chapter of this document. For the most part, once their IP networking features are properly configured, their RDMA features are all automatic and will show up as long as the proper drivers for the hardware are installed. The kernel drivers are always included with each kernel Red Hat provides, however the user-space drivers must be installed manually if the InfiniBand package group was not selected at machine install time.

这些是需要的用户空间软件包：

iWARP

Chelsio hardware — `libcxgb3` 或者 `libcxgb4`，具体要看硬件版本。

RoCE/IBoE

Mellanox hardware — `libmlx4` 或者 `libmlx5`，具体要看硬件版本。另外，要求用户编辑 `/etc/rdma/mlx4.conf` 或者 `/etc/rdma/mlx5.conf`，以便为 RoCE/IBoE 使用设定正确的端口类型。要求用户编辑 `/etc/modprobe.d/mlx4.conf` 或者 `/etc/modprobe.d/mlx5.conf` 文件，以便在以太网中为无损服务配置数据包优先响应（在一些交换机中称之为“no-drop”），以此切换连接到该网络的网卡。

With these driver packages installed (in addition to the normal RDMA packages typically installed for any InfiniBand installation), a user should be able to utilize most of the normal RDMA applications to test and see RDMA protocol communication taking place on their adapters. However, not all of the programs included in Red Hat Enterprise Linux 7 will properly support iWARP or RoCE/IBoE devices. This is because the connection establishment protocol on iWARP in particular is different than it is on real InfiniBand link-layer connections. If the program in question uses the `librdmacm` connection management library, it will handle the differences between iWARP and InfiniBand silently and the program should work. If the application tries to do its own connection management, then it must specifically support iWARP or else it will not work.

9.2. 与 InfiniBand 及 RDMA 相关的软件包

因为 RDMA 应用程序与基于伯克利套接字的应用程序有很大不同，而在一般 IP 联网中，大多数在 IP 网络中使用的应用程序无法直接在 RDMA 网络中使用。Red Hat Enterprise Linux 7 为 RDMA 网络管理、测试及调试、高级软件开发 API 及性能分析提供大量不同的软件包。

要使用这些网络，需要安装这些软件包的一部分或全部（这个列表并不全面，但包括与 RDMA 有关的最重要软件包）。

必须安装的软件包：

- ✦ ***rdma*** — 负责 RDMA 栈的内核初始化。
- ✦ ***libibverbs*** — 提供 InfiniBand Verbs API。
- ✦ ***opensm*** — 子网管理器（只需要在一台机器中安装，且只能在没有激活子网管理器的构造中安装）。
- ✦ **user space driver for installed hardware** — 以下软件包之一：*infinipath-psm*、*libcxgb3*、*libcxgb4*、*libehca*、*libipathverbs*、*libmthca*、*libmlx4*、*libmlx5*、*libnes* 及 *libocrdma*。
注：*libehca* 只用于 IBM Power Systems 服务器。

推荐的软件包：

- ✦ ***librdmacm*、*librdmacm-utils* 和 *ibacm*** — 可以识别 InfiniBand、iWARP 和 RoCE 之间不同的连接管理库，也可以正确打开跨这些硬件类型的连接，运行确认该网络操作的一些简单测试程序，并可将该库整合到缓存守护进程，以便在大型集群中更快地进行主机解析。
- ✦ ***libibverbs-utils*** — 基于简单 Verbs 的程序查询安装的硬件，并确认使用该结构的通讯。
- ✦ ***infiniband-diags* 或 *ibutils*** — 为 InfiniBand 结构管理提供大量有用的调试工具。这些工具只为 iWARP 或 RoCE 提供有限功能，因为大多数工具可在 InfiniBand 链接层工作，但无法在 Verbs API 层使用。
- ✦ ***perftest* 和 *qperf*** — 用于各种 RDMA 通讯类型的性能测试应用程序。

自选软件包：

这些软件包位于自选频道中。从自选频道安装这些软件包前，请查看 [覆盖范围详情](#)。有关订阅自选频道的信息，请查看 Red Hat 知识库解决方案，[如何访问自选及辅助频道](#)。

- ✦ ***dapl*、*dapl-devel* 及 *dapl-utils*** — 为 RDMA 提供不同于 Verbs API 的 API。这些软件包中均包含运行时组件及开发组件。
- ✦ ***openmpi*、*mvapich2* 及 *mvapich2-psm*** — 可使用 RDMA 通讯的 MPI 栈。写入这些栈的用户空间应用程序不一定会知道发生的 RDMA 通讯。

9.3. 配置基础 RDMA 子系统

9.3.1. RDMA 软件包安装

The *rdma* package is not part of the default install package set. If the InfiniBand package group was not selected during install, the *rdma* package (as well as a number of others as listed in the previous section) can be installed after the initial installation is complete. If it was not installed at machine installation time and instead was installed manually later, then it is necessary to rebuild the **initramfs** images using **dracut** in order for it to function fully as intended. Issue the following commands as **root**:

```
~]# yum install rdma
dracut -f
```

Startup of the **rdma** service is automatic. When RDMA capable hardware, whether InfiniBand or iWARP or RoCE/IBoE is detected, **udev** instructs **systemd** to start the **rdma** service. Users need not enable the **rdma** service, but they can if they want to force it on all the time. To do that, issue the following command:

```
~]# systemctl enable rdma
```

9.3.2. rdma.conf file 文件配置

The **rdma** service reads **/etc/rdma/rdma.conf** to find out which kernel-level and user-level RDMA protocols the administrator wants to be loaded by default. Users should edit this file to turn various drivers on or off.

可启用和禁用的各个驱动程序为：

- ✧ **IPoIB** — 这是 **IP** 网络模拟层，以便 **IP** 应用程序在 InfiniBand 网络 中运行。
- ✧ **SRP** — This is the SCSI Request Protocol. It allows a machine to mount a remote drive or drive array that is exported via the **SRP** protocol on the machine as though it were a local hard disk.
- ✧ **SRPT** — This is the target mode, or server mode, of the **SRP** protocol. This loads the kernel support necessary for exporting a drive or drive array for other machines to mount as though it were local on their machine. Further configuration of the target mode support is required before any devices will actually be exported. See the documentation in the *targetd* and *targetcli* packages for further information.
- ✧ **ISER** — 这个是用 Linux 内核常规 iSCSI 层的底层驱动程序，可通过 InfiniBand 网络为 iSCSI 设备提供传输。
- ✧ **RDS** — This is the Reliable Datagram Service in the Linux kernel. It is not enabled in Red Hat Enterprise Linux 7 kernels and so cannot be loaded.

9.3.3. 70-persistent-ipoib.rules 用法

rdma 软件包提供 **/etc/udev.d/rules.d/70-persistent-ipoib.rules** 文件。这个 **udev** 规则文件可用来修改 IPoIB 设备的默认名称（比如 **ib0** 和 **ib1**），以便提供更有描述性的名称。用户必须编辑此文件以便确定如何命名其文件。首先，请找到要重命名文件的 GUID 地址：

```
~]$ ip link show ib0
8: ib0: >BROADCAST,MULTICAST,UP,LOWER_UP< mtu 65520 qdisc pfifo_fast state
UP mode DEFAULT qlen 256
    link/infiniband
    80:00:02:00:fe:80:00:00:00:00:00:00:f4:52:14:03:00:7b:cb:a1 brd
    00:ff:ff:ff:ff:12:40:1b:ff:ff:00:00:00:00:00:00:ff:ff:ff:ff
```

link/infiniband 后紧接着的是 IPoIB 接口的 20 字节硬件地址。重新命名只需要以上使用黑体标注的最后 8 位地址。用户可采用适合他们的任何命名方案。例如：如果将 **mlx4** 设备连接到 **ib0** 子网结构中，则可使用 *device_fabric* 命名规则将其命名为 **mlx4_ib0**。唯一要避免的是使用标准名称，比如 **ib0** 或者 **ib1**，因为这些名称会与内核自动分配的名称冲突。下一步即可在该规则文件中添加条目。将现有示例复制到该规则文件中，使用要重新命名设备的凸显 8 位字节替换 **ATTR{address}** 条目中的 8 个字节，并在 **NAME** 字段输入新名称。

9.3.4. 为用户解除 memlock 限制

RDMA 通讯需要所要连接计算机中的物理内存（就是说不允许该内核在计算机启动运行的可用内存短缺时将该内存交换到要页面文件）。锁定内存一般是非常特殊的操作。要让 **root** 以外的用户运行大 RDMA 程序，则需要增大非 **root** 用户在系统中可锁定的内存。方法是在 `/etc/security/limits.d/` 目录中添加有以下内容的文件：

```
~]$ more /etc/security/limits.d/rdma.conf
# configuration for rdma tuning
*      soft    memlock      unlimited
*      hard    memlock      unlimited
# rdma tuning end
```

9.3.5. 为以太网操作配置 Mellanox 卡

Mellanox 中的某些硬件既可在 InfiniBand 模式中运行，也可在以太网模式中运行。这些网卡通常默认是 InfiniBand。用户可将这些网卡设定为以太网模式。目前只支持在 ConnectX 产品线硬件（使用 **mlx4** 驱动程序）中设定模式。要设定此模式。用户应按照 `/etc/rdma/mlx4.conf` 中的说明为其给定的硬件找到正确的 PCI 设备 ID。然后使用该设备 ID 及要求使用的端口类型中该文件中生成一行内容。然后应重新构建其 **initramfs**，以便确定将更新的端口设置复制到 **initramfs**。

端口类型设定完毕后，如果一个或两个端口均设定为 Ethernet，那么用户会在其日志中看到这样的信息：**mlx4_core 0000:05:00.0: Requested port type for port 1 is not supported on this HCA**。这很正常，不会影响操作。负责设定端口类型的脚本不可能知道该驱动程序何时会完成内部从端口 2 到所需类型，而且从该脚本发出请求切换端口 2 到此切换完成前，尝试将端口 1 设定为不同的类型的请求都会被拒绝。该脚本会不断重试直到该命令成功，或者直到其超过超时值，后者表示该端口切换一直没有完成。

9.4. 配置子网管理器

9.4.1. 确定必要性

Most InfiniBand switches come with an embedded subnet manager. However, if a more up to date subnet manager is required than the one in the switch firmware, or if more complete control than the switch manager allows is required, Red Hat Enterprise Linux 7 includes the **opensm** subnet manager. All InfiniBand networks **must** have a subnet manager running for the network to function. This is true even when doing a simple network of two machines with no switch and the cards are plugged in back to back, a subnet manager is required for the link on the cards to come up. It is possible to have more than one, in which case one will act as master, and any other subnet managers will act as slaves that will take over should the master subnet manager fail.

9.4.2. 配置 opensm 主配置文件

The **opensm** program keeps its master configuration file in `/etc/rdma/opensm.conf`. Users may edit this file at any time and edits will be kept on upgrade. There is extensive documentation of the options in the file itself. However, for the two most common edits needed, setting the GUID to bind to and the PRIORITY to run with, it is highly recommended that the **opensm.conf** file is not edited but instead edit `/etc/sysconfig/opensm`. If there are no edits to the base `/etc/rdma/opensm.conf` file, it will get upgraded whenever the **opensm** package is upgraded. As new options are added to this file regularly, this makes it easier to keep the current configuration up to date. If the **opensm.conf** file has been changed, then on upgrade, it might be necessary to merge new options into the edited file.

9.4.3. 配置 opensm 启动选项

`/etc/sysconfig/opensm` 文件中的选项控制子网管理器的实际启动方式，以及启动的子网管理器副本数量。例如：在双端口 InfiniBand 卡中，会将每个端口与独立的物理网络相连，就是说需要在每个端口中都有一个运行的子网管理器副本。`opensm` 子网管理器只会在管理应用程序实例的一个子网，且必须为每个需要管理的子网启动一次。另外，如果有一个以上的 `opensm` 服务器，则需要为每台服务器设置优先顺序，以决定哪些是从属服务器，哪些是主服务器。

`/etc/sysconfig/opensm` 文件是用来设定子网管理器优先顺序并控制子网管理器所绑定 GUID 的简单工具。`/etc/sysconfig/opensm` 文件本身包括其选项的详尽说明。用户只需要阅读该文件并按照其说明操作即可启用 `opensm` 的故障转移及多结构操作功能。

9.4.4. 创建 P_Key 定义

默认情况下，`opensm.conf` 会寻找文件 `/etc/rdma/partitions.conf` 以获取要在其中创建结构的分区列表。所有结构必须包含 `0x7fff` 子网，且所有交换机及所有主机都必须属于那个结构。可在该结构外另行创建其他分区，且所有主机及交换机不一定是那些附加分区的成员。这样就可以让管理员在 InfiniBand 结构中创建类似以太网 VLAN 的子网。如果使用给定速率（比如 40Gbps）定义某个分区，而在该网络中有一台主机无法达到 40Gbps，则那台主机即使有授权也无法加入该分区，因为它无法满足速度要求，因此建议将分区速率设定为所有有权限加入该分区主机中的最慢速度主机。如果某些主机子网需要较快的速度，则可使用较高的速度创建不同分区。

以下分区文件会得到默认 `0x7fff` 分区（速度降低 10 Gbps）以及速度为 40 Gbps 的分区 `0x0002`：

```
~]$ more /etc/rdma/partitions.conf
# For reference:
# IPv4 IANA reserved multicast addresses:
#   http://www.iana.org/assignments/multicast-addresses/multicast-
#   addresses.txt
# IPv6 IANA reserved multicast addresses:
#   http://www.iana.org/assignments/ipv6-multicast-addresses/ipv6-multicast-
#   addresses.xml
#
# mtu =
#   1 = 256
#   2 = 512
#   3 = 1024
#   4 = 2048
#   5 = 4096
#
# rate =
#   2 = 2.5 GBit/s
#   3 = 10 GBit/s
#   4 = 30 GBit/s
#   5 = 5 GBit/s
#   6 = 20 GBit/s
#   7 = 40 GBit/s
#   8 = 60 GBit/s
#   9 = 80 GBit/s
#   10 = 120 GBit/s

Default=0x7fff, rate=3, mtu=4, scope=2, defmember=full:
    ALL, ALL_SWITCHES=full;
Default=0x7fff, ipoib, rate=3, mtu=4, scope=2:
    mgid=ff12:401b::ffff:ffff          # IPv4 Broadcast address
    mgid=ff12:401b::1                  # IPv4 All Hosts group
    mgid=ff12:401b::2                  # IPv4 All Routers group
    mgid=ff12:401b::16                 # IPv4 IGMP group
```



```

        mgid=ff12:401b::fb                # IPv4 mDNS group
        mgid=ff12:401b::fc                # IPv4 Multicast Link Local Name
Resolution group
        mgid=ff12:401b::101               # IPv4 NTP group
        mgid=ff12:401b::202               # IPv4 Sun RPC
        mgid=ff12:601b::1                 # IPv6 All Hosts group
        mgid=ff12:601b::2                 # IPv6 All Routers group
        mgid=ff12:601b::16                # IPv6 MLDv2-capable Routers group
        mgid=ff12:601b::fb                # IPv6 mDNS group
        mgid=ff12:601b::101               # IPv6 NTP group
        mgid=ff12:601b::202               # IPv6 Sun RPC group
        mgid=ff12:601b::1:3               # IPv6 Multicast Link Local Name
Resolution group
        ALL=full, ALL_SWITCHES=full;

ib0_2=0x0002, rate=7, mtu=4, scope=2, defmember=full:
        ALL, ALL_SWITCHES=full;
ib0_2=0x0002, ipoib, rate=7, mtu=4, scope=2:
        mgid=ff12:401b::ffff:ffff        # IPv4 Broadcast address
        mgid=ff12:401b::1                 # IPv4 All Hosts group
        mgid=ff12:401b::2                 # IPv4 All Routers group
        mgid=ff12:401b::16                # IPv4 IGMP group
        mgid=ff12:401b::fb                # IPv4 mDNS group
        mgid=ff12:401b::fc                # IPv4 Multicast Link Local Name
Resolution group
        mgid=ff12:401b::101               # IPv4 NTP group
        mgid=ff12:401b::202               # IPv4 Sun RPC
        mgid=ff12:601b::1                 # IPv6 All Hosts group
        mgid=ff12:601b::2                 # IPv6 All Routers group
        mgid=ff12:601b::16                # IPv6 MLDv2-capable Routers group
        mgid=ff12:601b::fb                # IPv6 mDNS group
        mgid=ff12:601b::101               # IPv6 NTP group
        mgid=ff12:601b::202               # IPv6 Sun RPC group
        mgid=ff12:601b::1:3               # IPv6 Multicast Link Local Name
Resolution group
        ALL=full, ALL_SWITCHES=full;

```

9.4.5. 启用 opensm

opensm 服务安装时默认不启动，用户需要自行启动该服务。请作为 **root** 运行以下命令：

```
~]# systemctl enable opensm
```

9.5. 测试早期 InfiniBand RDMA 操作



注意

本小节只适用于 InfiniBand 设备。因为 iWARP 和 RoCE/IBoE 设备是基于 **IP** 的设备，用户应在配置 IPoIB 且设备有 **IP** 地址后，按照执行 RDMA 测试操作一节的内容进行测试。

启用 **rdma** 服务和 **opensm** 服务（若需要），并为具体硬件安装适当的用户空间库后，就可以进行用户空间 **rdma** 操作。**libibverbs-utils** 软件包中的简单测试程序可帮助您确定 RDMA 操作是否正常。**ibv_devices** 程序

显示该系统中目前所有设备，而 **ibv_devinfo** 命令会给出每个设备的具体信息。例如：

```

~]$ ibv_devices
device                node GUID
-----
mlx4_0                0002c903003178f0
mlx4_1                f4521403007bcb0
~]$ ibv_devinfo -d mlx4_1
hca_id: mlx4_1
  transport:          InfiniBand (0)
  fw_ver:             2.30.8000
  node_guid:          f452:1403:007b:cb0
  sys_image_guid:     f452:1403:007b:cb03
  vendor_id:          0x02c9
  vendor_part_id:     4099
  hw_ver:             0x0
  board_id:           MT_1090120019
  phys_port_cnt:      2
    port: 1
      state:           PORT_ACTIVE (4)
      max_mtu:         4096 (5)
      active_mtu:      2048 (4)
      sm_lid:          2
      port_lid:        2
      port_lmc:        0x01
      link_layer:      InfiniBand
    port: 2
      state:           PORT_ACTIVE (4)
      max_mtu:         4096 (5)
      active_mtu:      4096 (5)
      sm_lid:          0
      port_lid:        0
      port_lmc:        0x00
      link_layer:      Ethernet
~]$ ibstat mlx4_1
CA 'mlx4_1'
  CA type: MT4099
  Number of ports: 2
  Firmware version: 2.30.8000
  Hardware version: 0
  Node GUID: 0xf4521403007bcb0
  System image GUID: 0xf4521403007bcb03
  Port 1:
    State: Active
    Physical state: LinkUp
    Rate: 56
    Base lid: 2
    LMC: 1
    SM lid: 2
    Capability mask: 0x0251486a
    Port GUID: 0xf4521403007bcb01
    Link layer: InfiniBand
  Port 2:
    State: Active
    Physical state: LinkUp

```

```

Rate: 40
Base lid: 0
LMC: 0
SM lid: 0
Capability mask: 0x04010000
Port GUID: 0xf65214fffe7bcba2
Link layer: Ethernet

```

ibv_devinfo 和 **ibstat** 命令输出信息稍有不同（比如端口 MTU 信息是在 **ibv_devinfo** 而不是 **ibstat** 输出中显示，而端口 PUID 信息是在 **ibstat** 而不是 **ibv_devinfo** 输出中显示。同时有些信息的命名方式也不同，例如：**ibstat** 输出中的基础本地标识符（LID）与 **ibv_devinfo** 输出中的 **port_lid** 是相同的信息。

可使用简单的 ping 程序，比如 *infiniband-diags* 软件包中的 **ibping** 测试 RDMA 连接性。**ibping** 程序采用客户端/服务器模式。必须首先在一台机器中启动 **ibping** 服务器，然后再另一台机器中将 **ibping** 作为客户端运行，并让它与 **ibping** 服务器相连。因为我们要测试基础 RDMA 功能，因此需要用于 RDMA 的地址解析方法，而不是使用 IP 地址指定服务器。

在服务器机器中，用户可使用 **ibv_devinfo** 和 **ibstat** 命令输出 **port_lid**（或基础 lid）以及所要测试端口的端口 GUID（假设是上述接口的端口 1，则 **port_lid**/基础 LID 是 2，而端口 GUID 是 **0xf4521403007bcba1**）。然后使用所需选项启动 **ibping** 捆绑至要测试的网卡和端口，同时还要指定 **ibping** 以服务器模式运行。使用 **-?** 或者 **--help** 选项即可查看 **ibping** 的所有可用选项，但在这个实例中可使用 **-S** 或 **--Server** 选项，同时在绑定到具体网卡和端口时可使用 **-C** 或者 **--Ca** 以及 **-P** 或者 **--Port** 选项。注：这个实例中的端口不会指示端口号，但会在使用多端口网卡时指示物理端口号。要测试所使用 RDMA 结构的连接性，比如多端口网卡的第二端口，则需要让 **ibping** 捆绑至网卡的端口 2。使用单一端口网卡时不需要这个选项。例如：

```
~]$ ibping -S -C mlx4_1 -P 1
```

然后切换至客户端机器并运行 **ibping**。记录 **ibping** 程序所绑定端口的端口 GUID 或者 **ibping** 程序所绑定服务器端口的本地标识符（LID）。另外，还需要记录客户端机器中与服务器为所捆绑网卡和端口连接网络相同的网卡和端口。例如：如果服务器中第一网卡的第二端口所捆绑的网络是辅 RDMA 结构，那么就需要在客户端中指定一个也连接到第二结构的网卡和端口。完成配置后，请作为客户端运行 **ibping** 程序，使用在服务器中找到的端口 LID 或者 GUID 作为地址连接到服务器。例如：

```

~]$ ibping -c 10000 -f -C mlx4_0 -P 1 -L 2
--- rdma-host.example.com.(none) (Lid 2) ibping statistics ---
10000 packets transmitted, 10000 received, 0% packet loss, time 816 ms
rtt min/avg/max = 0.032/0.081/0.446 ms

```

或

```

~]$ ibping -c 10000 -f -C mlx4_0 -P 1 -G 0xf4521403007bcba1 \
--- rdma-host.example.com.(none) (Lid 2) ibping statistics ---
10000 packets transmitted, 10000 received, 0% packet loss, time 769 ms
rtt min/avg/max = 0.027/0.076/0.278 ms

```

这个结果会验证端到端 RDMA 通讯是否在用户空间应用程序中正常工作。

可能会看到以下出错信息：

```

~]$ ibv_devinfo
libibverbs: Warning: no userspace device-specific driver found for
/sys/class/infiniband_verbs/uverbs0
No IB devices found

```


这个出错信息表示未安装所需用户空间库。管理员需要安装一个在 [第 9.2 节“与 InfiniBand 及 RDMA 相关的软件包”](#) 小节中列出的用户空间库（根据其硬件要求）。这种情况极少发生，可能是因为用户为驱动程序或 `libibverbs` 安装了错误的架构类型。例如：如果 `libibverbs` 使用架构 `x86_64`，且安装了 `libmlx4` 而不是 `i686` 类型，则会得到出错信息。



注意

很多简单应用程序喜欢使用主机名或地址而不是 LID 打开服务器与客户端之间的通讯。在那些应用程序中需要在尝试测试端到端 RDMA 通讯前设置 IPoIB。`ibping` 应用程序通常不属于此列，它可以接受简单 LID 作为寻址方式，同时也使其成为解决 IPoIB 寻址测试可能出现问题的最简单测试方法，因此让我们能够从更独立的角度确定简单 RDMA 通讯是否可行。

9.6. 配置 IPoIB

9.6.1. 了解 IPoIB 角色

如 [第 1.2 节“IP 网络 vs 非 IP 网络”](#) 所述，大多数网络都是 **IP** 网络。IPoIB 的角色是在 InfiniBand RDMA 网络顶层提供 **IP** 网络模拟层。这样就可以让现有应用程序无需修改即可在 InfiniBand 网络中运行。但那些应用程序的性能相比原本使用 RDMA 通讯编写的应用程序要低。因此大多数 InfiniBand 网络中有一些应用程序集合必须使用其全部网络性能，而对另一些应用程序来说，如果不需要修改为使用 RDMA 通讯，则性能降级是可以接受的。IPoIB 可让那些不那么主要的应用程序在网络中以其原有形式运行。

因为 iWARP 和 RoCE/IBoE 网络实际上是在其 **IP** 链接层顶层附带 RDMA 层的 **IP** 网络，他们不需要 IPoIB。因此，内核会拒绝在 iWARP 或 RoCE/IBoE 设备中创建任何 IPoIB 设备。

9.6.2. 了解 IPoIB 通讯方式

可将 IPoIB 设备配置为以数据报或连接模式运行。不同之处在于 IPoIB 层尝试在通讯另一端机器中打开的队列对类型。在数据报模式中会打开不可靠、断开连接的队列对；而在连接模式中会打开可靠且连接的队列对。

使用数据报模式时，不可靠、断开的队列对不允许大于 InfiniBand 链接层 MTU 的数据包通过。IPoIB 层会在要传输的 **IP** 数据包顶部添加一个 4 字节的 IPoIB 标头。因此，IPoIB MTU 必须比 InfiniBand 链接层小 4 字节。因为在 InfiniBand 链接层 MTU 一般为 2048 字节，则数据报模式中的 IPoIB 设备则通常为 2044 字节。

使用连接模式时，可靠且连接的队列对类型允许大于 InfiniBand 链接层 MTU 的信息通过，同时在两端均采用主机适配器处理数据包片段并将其组合。这样使用连接模式时不会对通过 InfiniBand 适配器发送的 IPoIB 信息有大小限制，但 **IP** 数据包只有 16 个字节字段，因此最大字节数只能为 **65535**。最大允许的 MTU 实际较小，因为我们必须确定不同 TCP/IP 标头也适合这个大小限制。因此，连接模式中获得 IPoIB MTU 最大为 **65520**，这样可保证为所有需要的 **TCP** 标头提供足够的空间。

连接模式选项通常有较高性能，但也会消耗更多的内核内存。因为大多数系统考虑的是性能而不是用内存消耗，因此连接模式是最常用的用户模式。

但如果将系统配置为连接模式，它必须可以使用数据报模式发送多播流量（InfiniBand 交换机及结构无法使用连接模式传递多播流量），且可以在与未配置连接模式的主机通讯时返回数据报模式。管理员应意识到如果他们要运行发送多播数据的程序，且那些程序要以接口中的最大 MTU 发送多播数据，则需要将该接口配置为数据报操作，或者使用将多播程序配置为将其发送数据包大小限制在数据报允许的数据包范围内的其他方法。

9.6.3. 了解 IPoIB 硬件地址

IPoIB 设备有 20 字节的硬件地址。已弃用程序 `ifconfig` 无法读取全部 20 字节，因此不应再用来尝试并查找 IPoIB 设备的正确硬件地址。`iproute` 软件包中的 `ip` 程序可正常工作。

IPoIB 硬件地址的前四个字节是标签及队列对号码。接下来的八个字节是子网前缀。首次创建 IPoIB 后，会使用默认的子网前缀 `0xfe:80:00:00:00:00:00:00`。该设备在建立与子网管理器之间的联系前都会使用这个默认子网前缀（`0xfe80000000000000`）。建立联系后，它会重新设置子网前缀使其与子网管理器为其配置的前缀匹配。最后八位字节是 IPoIB 设备所连接 IPoIB 端口的 GUID 地址。因为前四个字节及随后的八位字节会随时更改，因此在为 IPoIB 接口指定硬件地址时不会使用这些字节进行匹配。[第 9.3.3 节“70-persistent-ipoib.rules 用法”](#)一节中论述了如何导出该地址，即在 `udev` 规则文件的 `ATTR{address}` 字段中不使用前 12 位字节以便设备映射会可靠。配置 IPoIB 接口时，该配置文件的 `HWADDR` 字段可包含所有 20 位字节，但只使用最后 8 位映射并查找配置文件指定的硬件。但如果在设备配置文件中的 `TYPE=InfiniBand` 条目拼写有误，且 `ifup-ib` 不是用来实际启动 IPoIB 接口的脚本，则系统会给出出错信息，说明该系统无法找到该配置指定的硬件。在 IPoIB 接口中，配置文件的 `TYPE=` 字段只能是 `InfiniBand` 或者 `infiniband`（该条目区分大小写，但该脚本不区分）。

9.6.4. 了解 InfiniBand P_Key 子网

使用不同 `P_Key` 子网可将 InfiniBand 结构采用逻辑分段的方式分为虚拟子网。这与在以太网接口中使用 VLAN 极其相似。所有交换机及主机均必须是默认 `P_Key` 子网中的成员，但管理员可以创建附加子网，并将该子网成员限制为该结构中主机或交换机的子网。主机可使用 `P_Key` 子网前必须使用子网管理器定义该子网。有关使用 `opensm` 子网管理器定义 `P_Key` 子网的详情，请查看 [第 9.4.4 节“创建 P_Key 定义”](#)一节。在 IPoIB 接口中，创建 `P_Key` 子网后，可为 `P_Key` 子网创建特殊的 IPoIB 配置文件。与在以太网设备中使用 VLAN 接口一样，每个 IPoIB 接口的行为都好像其处于与其他 IPoIB 接口完全不共的结构，他们共享同一链接，但使用不同的 `P_Key` 值。

对 IPoIB `P_Key` 的名称有特殊要求。所有 IPoIB `P_Key` 必须在 `0x0000` 到 `0x7fff` 的范围内，在高端字节中，`0x8000` 表示 `P_Key` 中的成员关系为完全成员，而不是部分成员。Linux 内核的 IPoIB 驱动程序只支持 `P_Key` 子网中的完全成员，因此在 Linux 无法连接的子网中总是要设置 `P_Key` 号的高端字节。就是说如果使用 Linux 系统的计算机加入 `P_Key 0x0002`，加入后，它的实际 `P_Key` 号码将会变为 `0x8002`，表示它是 `P_Key 0x0002` 的完全成员。因此，如 [第 9.4.4 节“创建 P_Key 定义”](#)一节所述在 `opensm partitions.conf` 文件中创建 `P_Key` 定义时，要求制定不包含 `0x8000` 的 `P_Key` 值，但在定义 Linux 客户端中的 `P_Key` IPoIB 接口时，则需要在 `P_Key` 指中添加 `0x8000` 值。

9.7. 使用文本用户界面 nmtui 配置 InfiniBand

可在终端窗口中使用文本用户界面工具 `nmtui` 配置 InfiniBand。运行以下命令接口启动该工具：

```
~]$ nmtui
```

此时会出现文本用户界面。输入无效命令会显示用法信息。

请使用箭头按键导航或按 `Tab` 前进，按 `Shift+Tab` 组合件后退。按 `Enter` 选择某个选项。按 `Space` 键选择复选框状态。

在开始菜单中选择 **编辑连接**。选择 **添加**，此时会打开 **新建连接** 页面。

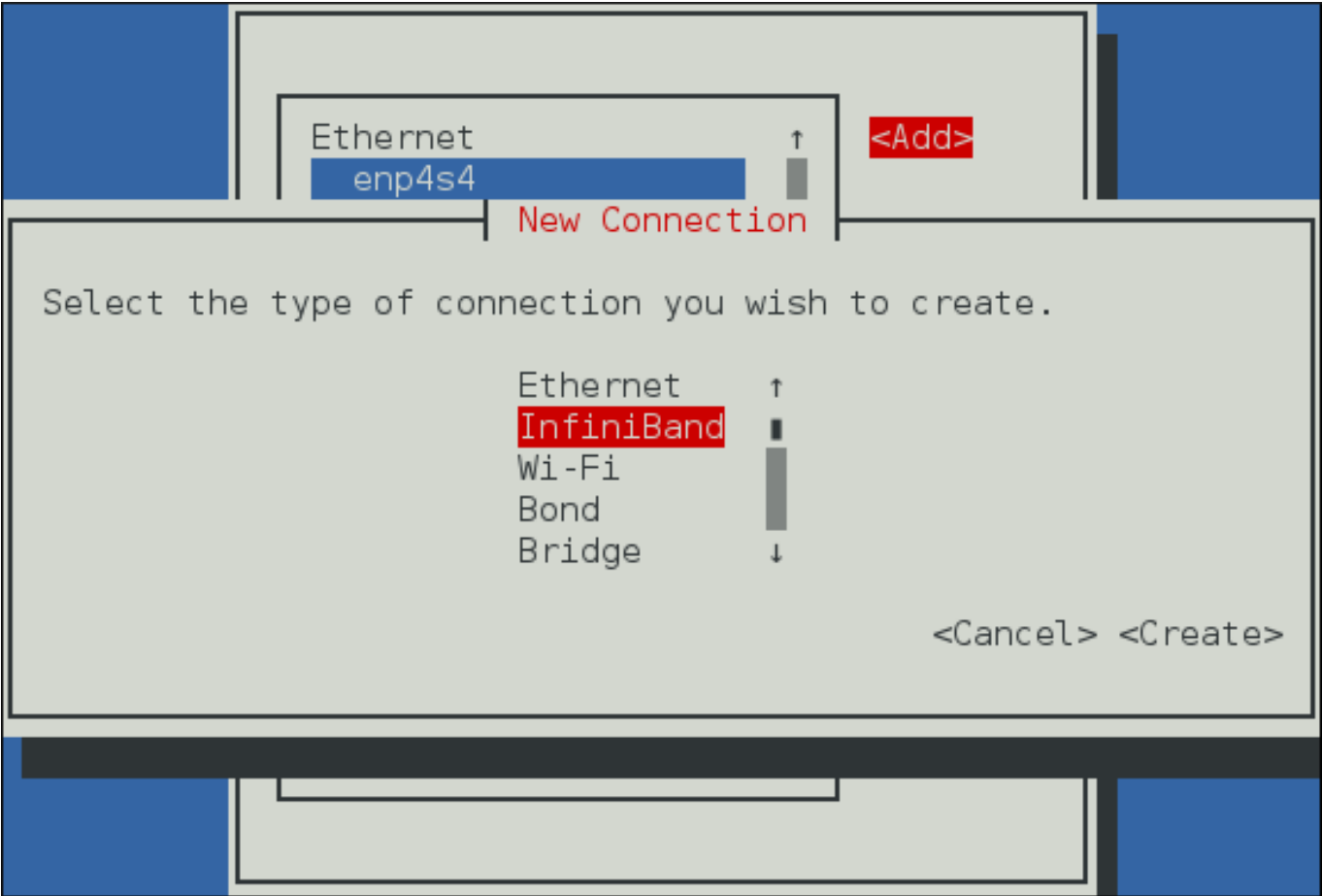


图 9.1. NetworkManager 文本用户界面添加 InfiniBand 连接菜单

选择 **InfiniBand**，此时会打开 编辑连接 页面。根据页面提示完成配置。

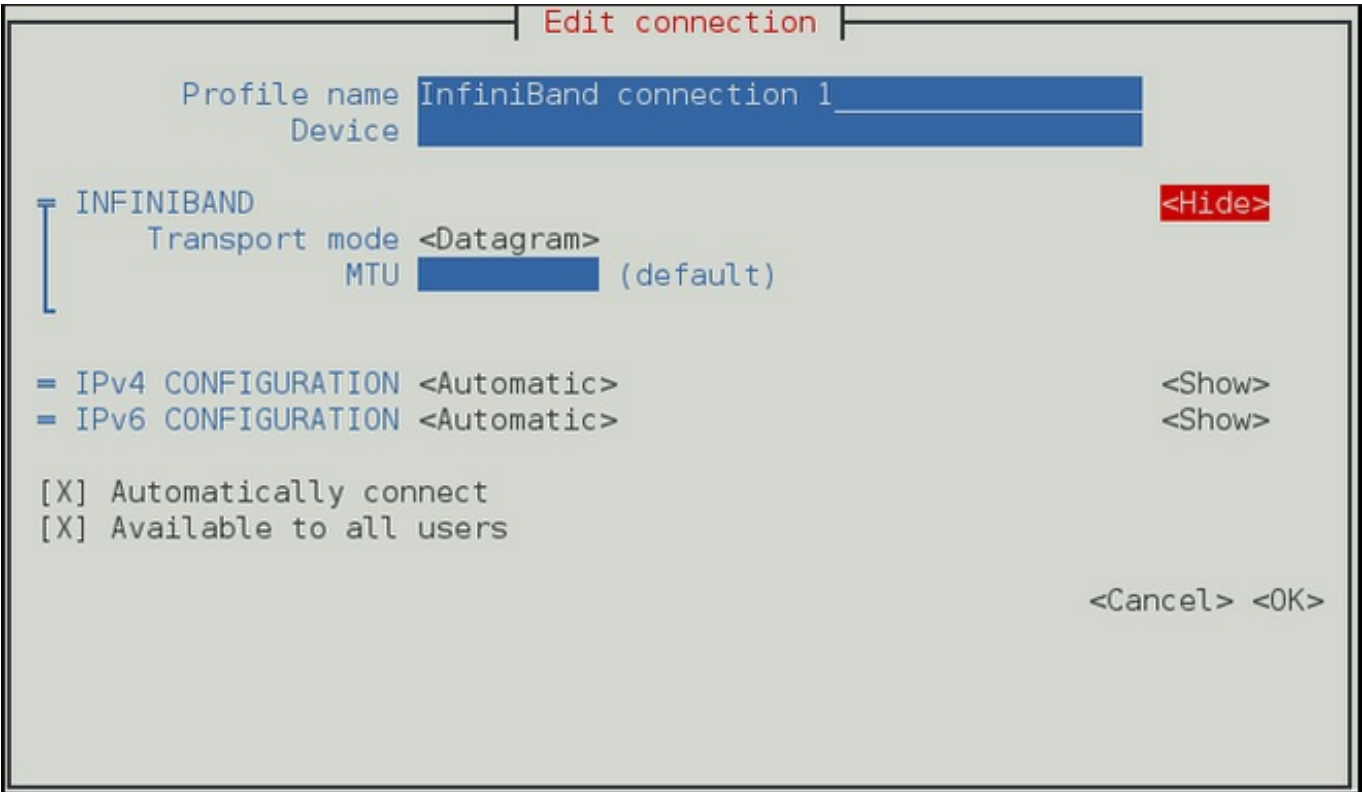


图 9.2. NetworkManager 文本用户界面配置 InfiniBand 连接菜单

有关 InfiniBand 术语定义请查看 [第 9.11.1 节“配置 InfiniBand 标签”](#)。

有关安装 `nmtui` 的详情请查看 [第 1.5 节“使用文本用户界面 \(nmtui\) 进行网络配置”](#)。

9.8. 使用命令行工具 `nmcli` 配置 IPoIB

首先确定是否需要重新命名默认的 IPoIB 设备，如果是，则请按照 [第 9.3.3 节“70-persistent-ipoib.rules 用法”](#) 一节中的说明，使用 `udev` 重命名规则重新命名这些设备。用户可强制重命名 IPoIB 设备而无需重启，方法是删除 `ib_ipoib` 内核模块，然后按以下方法重新载入：

```
~]$ rmmod ib_ipoib
~]$ modprobe ib_ipoib
```

按要求重新命名设备后，请使用 `nmcli` 工具创建 IPoIB 接口，如下：

```
~]$ nmcli con add type infiniband con-name mlx4_ib0 ifname mlx4_ib0
transport-mode connected mtu 65520
Connection 'mlx4_ib0' (8029a0d7-8b05-49ff-a826-2a6d722025cc) successfully
added.
~]$ nmcli con edit mlx4_ib0

===| nmcli interactive connection editor |===

Editing existing 'infiniband' connection: 'mlx4_ib0'

Type 'help' or '?' for available commands.
Type 'describe [>setting<.>prop<|>' for detailed property description.

You may edit the following settings: connection, infiniband, ipv4, ipv6
nmcli> set infiniband.mac-address
80:00:02:00:fe:80:00:00:00:00:00:00:f4:52:14:03:00:7b:cb:a3
nmcli> save
Connection 'mlx4_ib3' (8029a0d7-8b05-49ff-a826-2a6d722025cc) successfully
updated.
nmcli> quit
```

此时已创建名为 `mlx4_ib0` 的 IPoIB 接口，并将其设定为连接模式，采用最大连接模式 MTU，在 **IPv4** 或 **IPv6** 中使用 **DHCP**。如果为集群流量使用 IPoIB 接口，而为集群以外通讯使用以太网接口，则很可能需要按要求禁用 IPoIB 接口中的默认路由及所有默认名称服务器。

```
~]$ nmcli con edit mlx4_ib0

===| nmcli interactive connection editor |===

Editing existing 'infiniband' connection: 'mlx4_ib0'

Type 'help' or '?' for available commands.
Type 'describe [>setting<.>prop<|>' for detailed property description.

You may edit the following settings: connection, infiniband, ipv4, ipv6
nmcli> set ipv4.ignore-auto-dns yes
nmcli> set ipv4.ignore-auto-routes yes
nmcli> set ipv4.never-default true
nmcli> set ipv6.ignore-auto-dns yes
```

```
nmcli> set ipv6.ignore-auto-routes yes
nmcli> set ipv6.never-default true
nmcli> save
Connection 'mlx4_ib0' (8029a0d7-8b05-49ff-a826-2a6d722025cc) successfully
updated.
nmcli> quit
```

若需要 **P_Key** 接口，则请使用 **nmcli** 创建该接口，如下：

```
~]$ nmcli con add type infiniband con-name mlx4_ib0.8002 ifname mlx4_ib0.8002
parent mlx4_ib0 p-key 0x8002
Connection 'mlx4_ib0.8002' (4a9f5509-7bd9-4e89-87e9-77751a1c54b4)
successfully added.
~]$ nmcli con modify mlx4_ib0.8002 infiniband.mtu 65520 infiniband.transport-
mode connected ipv4.ignore-auto-dns yes ipv4.ignore-auto-routes yes
ipv4.never-default true ipv6.ignore-auto-dns yes ipv6.ignore-auto-routes yes
ipv6.never-default true
```

9.9. 使用命令行配置 IPoIB

首先确定是否需要重新命名默认的 IPoIB 设备，如果是，则请按照 [第 9.3.3 节 “70-persistent-ipoib.rules 用法”](#) 一节中的说明，使用 **udev** 重命名规则重新命名这些设备。用户可强制重命名 IPoIB 设备而无需重启，方法是删除 **ib_ipoib** 内核模块，然后按以下方法重新载入：

```
~]$ rmmod ib_ipoib
~]$ modprobe ib_ipoib
```

按要求命名设备后，管理员可使用其首选编辑器创建 **ifcfg** 文件以控制这些设备。采用静态 **IPv4** 寻址的典型 IPoIB 配置文件类似如下：

```
~]$ more ifcfg-mlx4_ib0
DEVICE=mlx4_ib0
TYPE=InfiniBand
ONBOOT=yes
HWADDR=80:00:00:4c:fe:80:00:00:00:00:00:00:f4:52:14:03:00:7b:cb:a1
BOOTPROTO=none
IPADDR=172.31.0.254
PREFIX=24
NETWORK=172.31.0.0
BROADCAST=172.31.0.255
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
MTU=65520
CONNECTED_MODE=yes
NAME=mlx4_ib0
```

DEVICE 字段必须与使用 **udev** 命名规则创建的自定义名称映射。**NAME** 条目不需要与设备名称映射。如果启动 GUI 连接编辑器，则使用 **NAME** 字段为用户显示这个连接的名称。**TYPE** 字段必须为 **InfiniBand**，以便正确处理 **InfiniBand** 选项。**CONNECTED_MODE** 可以是 **yes**，也可以是 **no**，其中 **yes** 代表在通讯中使用连接模式，而 **no** 代表使用数据报模式（详情请查看 [第 9.6.2 节 “了解 IPoIB 通讯方式”](#) 一节）。

以下是 **P_Key** 接口的典型配置文件：


```

~]$ more ifcfg-mlx4_ib0.8002
DEVICE=mlx4_ib0.8002
PHYSDEV=mlx4_ib0
PKEY=yes
PKEY_ID=2
TYPE=InfiniBand
ONBOOT=yes
HWADDR=80:00:00:4c:fe:80:00:00:00:00:00:00:f4:52:14:03:00:7b:cb:a1
BOOTPROTO=none
IPADDR=172.31.2.254
PREFIX=24
NETWORK=172.31.2.0
BROADCAST=172.31.2.255
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
MTU=65520
CONNECTED_MODE=yes
NAME=mlx4_ib0.8002

```

所有 **P_Key** 接口文件都需要 **PHYSDEV** 指令，且必须是上级设备的名称。**PKEY** 指令必须为 **yes**，同时 **PKEY_ID** 必须为接口号（添加或不添加 **0x8000** 成员关系字节均可）。但该设备名称必须是 **PKEY_ID** 的四位十六进制数字代表附加使用逻辑或运算符的 **0x8000** 成员字节，如下：

```
NAME=${PHYSDEV}.$((0x8000 | $PKEY_ID))
```

默认情况下，会将该文件中的 **PKEY_ID** 视为十进制数字，并转换为示例禁止数字，然后使用逻辑或运算符与 **0x8000** 合并得到该设备的正确名称，但用户需在该数字前添加标注 **0x** 前缀，以便使用十六进制指定 **PKEY_ID**。

9.10. 配置 IPoIB 后测试 RDMA 网络

配置 IPoIB 后，可以使用 **IP** 地址指定 RDMA 设备。由于普遍使用 **IP** 地址和主机名指定机器，大多数 RDMA 应用程序也首选使用此方式，只是在某些情况下指定远程机器以及连接到该机器的本地设备时使用 IPoIB。

要测试 IPoIB 层的功能，可使用任意标准 **IP** 网络测试工具，并提供要测试 IPoIB 设备的 **IP** 地址。例如：测试 IPoIB 设备 **IP** 地址是否工作的 ping 命令。

Red Hat Enterprise Linux 中有两种不同的 RDMA 性能测试软件包，即 *qperf* 和 *perftest*。这两个软件包均可用来进一步测试 RDMA 网络性能。

但使用 *perftest* 软件包中的任意程序时，或使用 *qperf* 应用程序时，需要特别注意地址解析。即使使用 **IP** 地址或 IPoIB 设备的主机名指定远程主机，也允许测试应用程序通过不同的 RDMA 接口。理由是因为从主机名或 **IP** 地址转换为 RDMA 地址的过程允许在两台机器中使用的所有有效 RDMA 地址对。如果有多种方式可让客户端连接到服务器，那么该程序会在指定路径出问题选择使用不同的路径。例如：如果每台机器中均有两个端口连接到同一 InfiniBand 子网，并给出每台机器中第二个端口的 **IP** 地址，那么该程序就很可能发现每台机器中的第一个端口是有效的，并使用该端口。在这种情况下，可使用 *perftest* 程序的所有命令行选项告诉它们要捆绑的网卡及端口，如 [第 9.5 节“测试早期 InfiniBand RDMA 操作”](#) 中 *ibping* 所做的一样，以便保证在要求测试的具体端口中进行测试。*qperf* 捆绑端口的方式稍有不同。*qperf* 程序在一台机器中作为服务器使用，侦听所有设备（包括非 RDMA 设备）。客户端可使用服务器的有效 **IP** 或主机名连接到 *qperf*。*Qperf* 会首先尝试打开数据连接，并使用在客户端命令中给出的 **IP** 地址或主机名运行要求的测试；但如果使用该地址有任何问题，*qperf* 就会返回，并尝试着客户端和服务端之间的任意有效路径中运行。因此，要强制 *qperf* 使用具体链接进行测试，请在 *qperf* 客户端中使用 **-loc_id** 或 **-rem_id** 选项，以便强制使用具体链接进行测试。

9.11. 使用 CNI 配置 IPoIB

9.11. 使用 GUI 配置 InfiniBand

要使用图形工具配置 InfiniBand 连接，请使用 **Network Connections** 工具

过程 9.1. 添加新 InfiniBand 连接

1. 要使用图形 **Network Connections** 工具，请按 **Super** 键进入活动概述，输入 **Network Connections** 并按 **Enter**。此时会出现 **Network Connections** 工具
2. 点击 **添加** 按钮打开选择列表。选择 **InfiniBand** 然后点击 **创建**。此时会出现 **编辑 InfiniBand 连接 1** 窗口。
3. 在 **InfiniBand** 标签中，从下拉菜单中选择要使用的传输模式。
4. 输入 InfiniBand MAC 地址。
5. 检查并确定设置，然后点击 **保存** 按钮。
6. 要编辑具体 InfiniBand 设置，请查看 [第 9.11.1 节“配置 InfiniBand 标签”](#)。

过程 9.2. 编辑现有 InfiniBand 连接

按照以下步骤编辑现有 InfiniBand 连接。

1. 按 **Super** 进入活动概述页面，输入 **Network Connections** 然后按 **Enter**。此时会出现 **Network Connections** 工具。
2. 选择要编辑的连接，点击 **编辑** 按钮。
3. 选择 **常规** 标签。
4. 配置连接名称、自动连接行为及可用性设置。

编辑 对话框中的五种设置适用于所有连接类型，请查看 **常规** 标签：

- ✦ **连接名称** — 为网络连接输入描述性名称。这个名称可用于在 **网络** 窗口中列出这个连接。
- ✦ **可用时自动连接到这个网络**复选框：如果要让 **NetworkManager** 每次可用时自动连接到这个连接，则选择这个选项。详情请查看 [第 2.5.3 节“自动连接到网络”](#)。
- ✦ **所有用户都可以连接到这个网络** — 要创建可用于系统中其他用户的连接，请选中这个复选框。详情请查看 [第 2.5.4 节“系统范围及专用连接配置文件”](#)。
- ✦ **使用此连接时自动连接到 VPN** — 如果要让 **NetworkManager** 在可用时自动连接到 VPN 连接，请选择正规复选框。请在下拉菜单中选择 VPN。
- ✦ **防火墙区** — 请在下拉菜单中选择防火墙区。有关防火墙区的详情，请查看 [《Red Hat Enterprise Linux 7 安全指南》](#)。

5. 请参考 [第 9.11.1 节“配置 InfiniBand 标签”](#) 编辑具体 InfiniBand 设置。

保存新的（或修改的）连接，并做进一步的配置。

完成 InfiniBand 连接编辑后，点击 **保存** 按钮保存自定义配置。如果编辑该配置文件时正在使用该文件，则请断开连接，以便 **NetworkManager** 应用更改。如果该配置文件处于 OFF 状态，请在网络连接图标菜单中将其设定为 ON。有关使用新的或更改的连接的详情，请查看 [第 2.5.1 节“使用 GUI 连接到网络”](#)。

您可以配置现有连接，方法是在 **网络连接** 窗口中选择 该连接，并点击 **编辑** 返回 **编辑** 对话框。

然后配置：

- ✧ 该连接的 IPv4 设置，点击 **IPv4 设置** 标签，继续执行 [第 2.5.10.4 节“配置 IPv4 设置”](#)；或者，
- ✧ 该连接的 IPv6 设置，点击 **IPv6 设置** 标签，继续执行 [第 2.5.10.5 节“配置 IPv6 设置”](#)。

9.11.1. 配置 InfiniBand 标签

若已添加新的 InfiniBand 连接（步骤请参考 [过程 9.1, “添加新 InfiniBand 连接”](#)），则可以编辑 **InfiniBand** 标签设定上级接口及 InfiniBand ID。

传输模式

可从下拉菜单列表中选择数据报或连接模式。选择您的其他 IPoIB 网络正在使用的模式。

设备 MAC 地址

可使用 InfiniBand 设备的 MAC 地址可用于 InfiniBand 网络流量。若已安装 InfiniBand 硬件，则会预先填入这个硬件地址字段。

MTU

另外可设置用于通过 InfiniBand 连接发送的数据包的最大传输单元（Maximum Transmission Unit, MTU）。

9.12. 其他资料

以下信息资源为您提供 Red Hat Enterprise Linux 7 中有关 InfiniBand 及 RDMA 联网的附加资源。

9.12.1. 已安装文档

- ✧ `/usr/share/doc/initscripts-version/sysconfig.txt` — 描述配置文件及其指令。

9.12.2. 在线文档

<https://www.kernel.org/doc/Documentation/infiniband/ipoib.txt>

IPoIB 驱动程序描述，包括参考文献及相关 RFC。

部分 III. 服务器

这部分论述如何设置联网一般需要的服务器。

第 10 章 DHCP 服务器

动态主机配置协议（DHCP）是为客户的机器自动分配 TCP/IP 信息的网络协议。每个 **DHCP** 客户端都连接到中央 **DHCP** 服务器，该服务器会返回该客户端的网络配置（其中包括 **IP** 地址、网关及 **DNS** 服务器）。

10.1. 为什么使用 DHCP

DHCP 对自动配置客户端网络接口很有帮助。配置客户端系统时，可选择 **DHCP** 而不是指定 **IP** 地址、子网掩码、网关或 **DNS** 服务器。该客户端会从 **DHCP** 服务器中检索这个信息。**DHCP** 还对更改大量系统的 **IP** 地址很有帮助。使用这个工具可不必重新配置所有系统，只要在该服务器的一个配置文件中编辑一组新的 **IP** 地址即可。如果用于机构的 **DNS** 服务器有变化，会在 **DHCP** 服务器发生变化，而不是 **DHCP** 客户端发生变化。重启网络或重启客户端后更改会生效。

如果机构可正确将 **DHCP** 服务器连接到网络、笔记本电脑及其他移动计算机用户，就可以在办公室间移动这些设备。

注：**DNS** 和 **DHCP** 服务器的管理员，以及所有配置应用程序，应接受机构中使用的主机名格式。有关主机名格式的详情请查看 [第 3.1.1 节“建议到命名方法”](#)。

10.2. 配置 DHCP 服务器

dhcp 软件包包含互联网系统联盟（ISC）**DHCP** 服务器。请作为 **root** 安装该软件包：

```
~]# yum install dhcp
```

安装 *dhcp* 软件包可生成文件 `/etc/dhcp/dhcpd.conf`，该文件基本是一个空白配置文件。请作为 **root** 运行以下命令：

```
~]# cat /etc/dhcp/dhcpd.conf
#
# DHCP Server Configuration file.
#   see /usr/share/doc/dhcp*/dhcpd.conf.example
#   see dhcpd.conf(5) man page
#
```

可在 `/usr/share/doc/dhcp-version;/dhcpd.conf.example` 找到示例配置文件。使用这个文件可帮助您配置 `/etc/dhcp/dhcpd.conf`，如下所示。

DHCP 还使用 `/var/lib/dhcpd/dhcpd.leases` 文件保存客户端租用数据库。详情请查看 [第 10.2.2 节“租期数据库”](#)。

10.2.1. 配置文件

配置 **DHCP** 服务器的第一步是创建保存客户端网络信息的配置文件。使用这个文件向客户端系统声明选项。

该配置文件可包含附加标签或空白行以方便格式化。关键词区分大小写，同时将以井号（#）开始的行视为注释。

配置文件中两类语句：

- ✱ 参数 -- 说明如何执行任务，是否执行任务或向该客户端发送什么网络配置选项。
- ✱ 声明 -- 描述网络拓扑，描述客户端，提供客户端 **IP** 地址，或在一组声明中应用一组参数。

以关键字选项开始的参数请参考 [选项](#)。这些选项控制 **DHCP** 选项，必须为这些参数配置参数值，或控制 **DHCP** 服务器的行为。

大括号（{ }）前面一部分的参数（包括选项）是全局参数。全局参数适用于其后的所有内容。



重要

如果更改配置文件，更改的部分要在使用 **systemctl restart dhcpd** 重启 **DHCP** 守护进程后方可生效。



注意

与其每次更改 **DHCP** 配置文件并重启该服务，不如使用 **omshell** 命令提供一个互动的方法连接、查询并更改 **DHCP** 服务器的配置。使用 **omshell** 可在服务器运行过程中进行所有更改。有关 **omshell** 的详情请查看 **omshell** man page。

在 [例 10.1 “子网声明”](#) 中，可在下述 **host** 语句中使用以下 **routers**、**subnet-mask**、**domain-search**、**domain-name-servers** 和 **time-offset** 选项。

对所有提供的子网及连接到 **DHCP** 服务器的所有子网，必须有一个 **subnet** 声明，该声明告知 **DHCP** 守护进程如何识别该子网中的地址。即使没有动态为那个子网分配任何地址，每个子网也都需要有一个 **subnet** 声明。

在这个示例中，子网中的每个 **DHCP** 客户端都有全局选项及一个声明的 **range**。会为这些客户端分配一个 **range** 范围内的 **IP** 地址。

例 10.1. 子网声明

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    option routers                192.168.1.254;
    option subnet-mask            255.255.255.0;
    option domain-search          "example.com";
    option domain-name-servers    192.168.1.1;
    option time-offset            -18000;      # Eastern Standard
Time
    range 192.168.1.10 192.168.1.100;
}
```

要配置一个为子网中的某个系统动态租用 **IP** 地址的 **DHCP** 服务器，请修改 [例 10.2 “Range 参数”](#) 中的示例值。它会说明客户端的默认租用时间、最长租用时间及网络配置值。这个示例为客户的系统分配 **range** 范围内的 **IP** 地址 **192.168.1.10** 和 **192.168.1.100**。

例 10.2. Range 参数

```
default-lease-time 600;
max-lease-time 7200;
option subnet-mask 255.255.255.0;
option broadcast-address 192.168.1.255;
option routers 192.168.1.254;
```

```
option domain-name-servers 192.168.1.1, 192.168.1.2;
option domain-search "example.com";
subnet 192.168.1.0 netmask 255.255.255.0 {
    range 192.168.1.10 192.168.1.100;
}
```

要根据网卡的 MAC 地址为客户端分配 IP 地址，请在 **host** 声明中使用 **hardware ethernet** 参数。如例 10.3 “使用 DHCP 的静态 IP 地址”所示，**host apex** 声明指定使用 MAC 地址 **00:A0:78:8E:9E:AA** 的网卡永远接收 IP 地址 **192.168.1.4**。

注：还可以使用自选参数 **host-name** 为客户端分配主机名。

例 10.3. 使用 DHCP 的静态 IP 地址

```
host apex {
    option host-name "apex.example.com";
    hardware ethernet 00:A0:78:8E:9E:AA;
    fixed-address 192.168.1.4;
}
```

Red Hat Enterprise Linux 7 支持为 InfiniBank IPoIB 接口分配静态 IP 地址。但因为这些接口没有正常的硬件以太网地址，因此必须使用不同的方法为 IPoIB 接口指定唯一识别符。该标准是使用选项 **dhcp-client-identifier=** 结构指定 IPoIB 接口的 **dhcp-client-identifier** 字段。DHCP 服务器主机架构最多支持一个硬件以太网，在每个主机段中支持一个 **dhcp-client-identifier** 条目。但可能有多个固定地址条目，同时 DHCP 服务器会在接收 DHCP 请求后自动使用适用于该网络的地址响应。

例 10.4. 使用 DHCP 在多个接口中使用静态 IP 地址

在使用复杂配置的机器中，例如：有两个 InfiniBand 接口，且每个物理接口中都有 **P_Key** 接口，外加一个以太网连接，则可在配置时使用以下静态 IP 架构：

```
Host apex.0 {
    option host-name "apex.example.com";
    hardware ethernet 00:A0:78:8E:9E:AA;
    option dhcp-client-
identifier=ff:00:00:00:00:00:02:00:00:02:c9:00:00:02:c9:03:00:31:7b:11;
    fixed-address 172.31.0.50,172.31.2.50,172.31.1.50,172.31.3.50;
}

host apex.1 {
    option host-name "apex.example.com";
    hardware ethernet 00:A0:78:8E:9E:AB;
    option dhcp-client-
identifier=ff:00:00:00:00:00:02:00:00:02:c9:00:00:02:c9:03:00:31:7b:12;
    fixed-address 172.31.0.50,172.31.2.50,172.31.1.50,172.31.3.50;
}
```

要为您的设备找到正确的 **dhcp-client-identifier**，通常可使用前缀 **ff:00:00:00:00:00:02:00:00:02:c9:00**，然后在最后添加 8 字节 IPoIB 接口（恰好是 IPoIB 接口所在 InfiniBand 端口的 8 字节 GUID）。在有些旧的控制器中，这个前缀是错误的。如果出现那种情况，建议您在 DHCP 服务器中使用 **tcpdump** 捕获传入 IPoIB DHCP 请求，并从中获取正确的 **dhcp-client-**

identifier。例如：

```

]$ tcpdump -vv -i mlx4_ib0
tcpdump: listening on mlx4_ib0, link-type LINUX_SLL (Linux cooked),
capture size 65535 bytes
23:42:44.131447 IP (tos 0x10, ttl 128, id 0, offset 0, flags [none], proto
UDP (17), length 328)
    0.0.0.0.bootpc > 255.255.255.255.bootps: [udp sum ok] BOOTP/DHCP,
Request, length 300, htype 32, hlen 0, xid 0x975cb024, Flags [Broadcast]
(0x8000)
    Vendor-rfc1048 Extensions
        Magic Cookie 0x63825363
        DHCP-Message Option 53, length 1: Discover
        Hostname Option 12, length 10: "rdma-qe-03"
        Parameter-Request Option 55, length 18:
            Subnet-Mask, BR, Time-Zone, Classless-Static-Route
            Domain-Name, Domain-Name-Server, Hostname, YD
            YS, NTP, MTU, Option 119
            Default-Gateway, Classless-Static-Route, Classless-Static-
Route-Microsoft, Static-Route
            Option 252, NTP
            Client-ID Option 61, length 20: hardware-type 255,
00:00:00:00:00:02:00:00:02:c9:00:00:02:c9:02:00:21:ac:c1

```

上述转储显示了 Client-ID 字段。hardware-type **255** 与 ID 起始的 **ff:** 对应，然后将 ID 的剩余部分括起，因为要在 **DHCP** 配置文件中有些内容。

如 [例 10.5 “Shared-network 声明”](#) 所示，应在 **shared-network** 声明中宣布共享同一物理网络的所有子网。**shared-network** 中的参数（未包含在子网声明中的除外）都应被视为全局参数。为 **shared-network** 分配的名称必须可描述该网络，比如使用 “test-lab” 描述所有 test lab 环境中的所有子网。

例 10.5. Shared-network 声明

```

shared-network name {
    option domain-search          "test.redhat.com";
    option domain-name-servers    ns1.redhat.com, ns2.redhat.com;
    option routers                 192.168.0.254;
    #more parameters for EXAMPLE shared-network
    subnet 192.168.1.0 netmask 255.255.252.0 {
        #parameters for subnet
        range 192.168.1.1 192.168.1.254;
    }
    subnet 192.168.2.0 netmask 255.255.252.0 {
        #parameters for subnet
        range 192.168.2.1 192.168.2.254;
    }
}

```

如 [例 10.6 “组声明”](#) 所示，**group** 声明是用来在一组声明中应用全局参数。例如：共享网络、子网及主机都可分组。

例 10.6. 组声明

```
group {
    option routers                192.168.1.254;
    option subnet-mask            255.255.255.0;
    option domain-search          "example.com";
    option domain-name-servers    192.168.1.1;
    option time-offset             -18000;      # Eastern Standard Time
    host apex {
        option host-name "apex.example.com";
        hardware ethernet 00:A0:78:8E:9E:AA;
        fixed-address 192.168.1.4;
    }
    host raleigh {
        option host-name "raleigh.example.com";
        hardware ethernet 00:A1:DD:74:C3:F2;
        fixed-address 192.168.1.6;
    }
}
```



注意

可参照提供的示例配置文件，在其中添加自定义配置选项。请作为 **root** 使用以下命令将这个文件复制到正确位置：

```
~]# cp /usr/share/doc/dhcp-version_number/dhcpd.conf.example
/etc/dhcp/dhcpd.conf
```

其中 *version_number* 是 **DHCP** 版本号。

有关选项语句的完整列表及其功能，请查看 **dhcp-options(5)** man page。

10.2.2. 租期数据库

DHCP 服务器中，使用文件 **/var/lib/dhcpd/dhcpd.leases** 保存 **DHCP** 客户端租用数据库。请勿更改这个文件。该租用数据库会自动保存每个最近分配给 **IP** 地址的 **DHCP** 租用信息。该信息包括租期、分配 **IP** 地址的对象、租用起始及终止日期、以及用来检索租用信息的网卡 **MAC** 地址。

租期数据库中所用的时间是格林威治标准时间（**GMT**），不是本地时间。

会经常重新生成这个租期数据库以免其过于庞大。首先，所有已知租期都保存在临时租期数据库中。将 **dhcpd.leases** 文件重命名为 **dhcpd.leases~**，并将临时租期数据库写入 **dhcpd.leases**。

在将租用数据库重新命名为备份文件后到写入新文件前，**DHCP** 守护进程可被杀死，或者系统可能会崩溃。如果出现这种情况，**dhcpd.leases** 文件就不存在，但要求启动该服务。请勿创建新的租用文件。如果您这样做，就会丢失所有旧的租用信息，从而造成巨大影响。正确的解决方法是将 **dhcpd.leases~** 备份文件重命名为 **dhcpd.leases**，然后重启该守护进程。

10.2.3. 启动和停止服务器



重要

首次启动 **DHCP** 服务器时，如果没有 **dhcpd.leases** 文件就会失败。如果没有该文件，可使用命令 **touch /var/lib/dhcpd/dhcpd.leases** 生成该文件。如果同一服务器还作为 **DNS** 服务器运行 **BIND**，那么就不需要执行这一步操作，因为 **named** 服务会自动检查 **dhcpd.leases** 文件。

请勿在之前运行的系统中创建新的租用文件。如果这样做，就会丢失所有旧的租用信息，造成很多问题。正确的解决方法是将 **dhcpd.leases~** 备份文件重命名为 **dhcpd.leases**，然后重启该守护进程。

请使用以下命令启动 **DHCP** 服务：

```
systemctl start dhcpd.service
```

请使用以下命令停止 **DHCP** 服务器：

```
systemctl stop dhcpd.service
```

默认情况下，**DHCP** 服务不在引导时启动。有关如何将该守护进程配置为在引导时自动启动的信息，请查看 [《Red Hat Enterprise Linux 7 系统管理员指南》](#)。

如果在该系统中添加一个以上网络接口，但 **DHCP** 只应侦听其中一个接口的 **DHCP** 请求，请将 **DHCP** 配置为只侦听那个设备。**DHCP** 守护进程将只侦听可在 **/etc/dhcp/dhcpd.conf** 文件的子网声明中找到的接口。

这对使用两个网卡的防火墙机器有帮助。可将一个网卡配置为 **DHCP** 客户端，检索互联网 **IP** 地址。将另一个网卡作为 **DHCP** 服务器用于防火墙保护下的内部网络。只指定连接到内部网络的网卡，让系统更安全，因为用户无法通过互联网连接到该守护进程。

要指定命令行选项，请作为 **root** 用户复制、然后编辑 **dhcpd.service** 文件。例如：

```
~]# cp /usr/lib/systemd/system/dhcpd.service /etc/systemd/system/
~]# vi /etc/systemd/system/dhcpd.service
```

编辑 [Service] 部分中的内容：

```
ExecStart=/usr/sbin/dhcpd -f -cf /etc/dhcp/dhcpd.conf -user dhcpd -group
dhcpd --no-pid your_interface_name(s)
```

，然后作为 **root** 用户重启该服务：

```
~]# systemctl --system daemon-reload
~]# systemctl restart dhcpd
```

可将命令行选项附加到 **/etc/systemd/system/dhcpd.service** 文件 [Service] 部分的 **ExecStart=/usr/sbin/dhcpd** 命令中。这些选项包括：

- ✱ **-p portnum** — 指定 UDP 端口号，**dhcpd** 应侦听该端口。默认端口为 67。**DHCP** 服务器将响应通过大于指定 UDP 端口号的端口传送到 **DHCP** 客户端。例如：如果默认端口为 67，则该服务器应在端口 67 侦听请求，并通过端口 68 为客户端提供响应。如果在此指定端口，同时使用 **DHCP** 中继代理，则必须为 **DHCP** 中继代理指定应侦听同一端口。详情请查看 [第 10.3 节“DHCP 中继代理程序”](#)。

- ✧ **-f** -- 将该守护进程作为前端进程运行。这大多数是用于 debug。
- ✧ **-d** — 在 DHCP 服务器守护进程日志中记录标准错误描述符。这大多用于 debug。如果未指定，则会在 `/var/log/messages` 中记录日志。
- ✧ **-cf filename** — 指定配置文件位置。默认位置为 `/etc/dhcp/dhcpd.conf`。
- ✧ **-lf filename** — 指定租用数据库文件位置。如果已有租用数据库文件，关键是每次 DHCP 服务启动时使用同一文件。强烈建议只在非产品机器中进行 debug 时使用这个选项。默认位置为 `/var/lib/dhcpd/dhcpd.leases`。
- ✧ **-q** -- 启动该守护进程时不要输出完整版权信息。

10.3. DHCP 中继代理程序

DHCP 中继代理程序 (`dhcrelay`) 可让没有 DHCP 服务器的子网向其他子网中的一个或多个 DHCP 服务器发出 DHCP 和 BOOTP 请求。

DHCP 客户端请求信息时，DHCP 中继代理程序会将该请求转发到启动 DHCP 中继代理时指定的 DHCP 服务器列表。DHCP 服务器返回回复后，会向最初发送请求的网络广播或单播这个回复。

除非在 `/etc/sysconfig/dhcrelay` 文件中使用 **INTERFACES** 指令指定接口，IPv4 的 DHCP 中继代理 `dhcrelay` 侦听所有接口的 DHCPv4 和 BOOTP 请求。详情请查看 [第 10.3.1 节“将 dhcrelay 配置为 DHCPv4 and BOOTP 中继代理程序”](#)。IPv6 的 DHCP 中继代理 `dhcrelay6` 没有这个默认行为，同时必须指定侦听 DHCPv6 请求的接口。详情请查看 [第 10.3.2 节“将 dhcrelay 配置为 DHCPv6 中继代理程序”](#)。

`dhcrelay` 可作为 DHCPv4 和 BOOTP（默认）中继代理运行，或添加 **-6** 参数作为 DHCPv6 中继代理运行。要查看用法信息，请运行命令 `dhcrelay -h`。

10.3.1. 将 dhcrelay 配置为 DHCPv4 and BOOTP 中继代理程序

要在 DHCPv4 和 BOOTP 模式服务器中运行 `dhcrelay`，指定向其转发请求的服务器，请作为 **root** 用户复制并编辑 `dhcrelay.service` 文件：

```
~]# cp /lib/systemd/system/dhcrelay.service /etc/systemd/system/
~]# vi /etc/systemd/system/dhcrelay.service
```

编辑 [Service] 部分的 **ExecStart** 选项，在该行的结尾处添加一个或多个服务器 IPv4 地址，例如：

```
ExecStart=/usr/sbin/dhcrelay -d --no-pid 192.168.1.1
```

如果要指定 DHCP 中继代理侦听 DHCP 请求的接口，请使用 **-i** 参数将其添加到 **ExecStart** 选项（否则会侦听所有接口），例如：

```
ExecStart=/usr/sbin/dhcrelay -d --no-pid 192.168.1.1 -i em1
```

。其他选项请查看 `dhcrelay(8)` man page。

请作为 **root** 用户重启该服务以便更改生效：

```
~]# systemctl --system daemon-reload
~]# systemctl restart dhcrelay
```


10.3.2. 将 dhcrelay 配置为 DHCPv6 中继代理程序

要在 **DHCPv6** 模式中运行 **dhcrelay**，请添加 **-6** 参数，并指定 “lower interface”（可使用该接口接收客户端或其他中继代理的查询）和 “upper interface”（转发来自客户端和其他中继代理的查询）。作为 **root** 用户将 **dhcrelay.service** 复制到 **dhcrelay6.service**，并进行编辑：

```
~]# cp /lib/systemd/system/dhcrelay.service
/etc/systemd/system/dhcrelay6.service
~]# vi /etc/systemd/system/dhcrelay6.service
```

编辑 [Service] 部分的 **ExecStart** 选项，添加 **-6** 参数，并添加 “lower interface” 和 “upper interface” 接口，例如：

```
ExecStart=/usr/sbin/dhcrelay -d --no-pid -6 -l em1 -u em2
```

。其他选项的详情请查看 **dhcrelay(8)** man page。

请作为 **root** 用户重启该服务以便更改生效：

```
~]# systemctl --system daemon-reload
~]# systemctl restart dhcrelay6
```

10.4. 配置 DHCP 服务器

多主机 **DHCP** 服务器可提供多个网络，即多个子网。这些小节中的示例详细论述了如何将 **DHCP** 配置为提供多个网络，选择要侦听的网络接口，以及如何为在网络间移动的系统定义网络设置。

进行修改前，请备份现有 **/etc/dhcp/dhcpd.conf** 文件。

DHCP 守护进程只侦听它能够在 **/etc/dhcp/dhcpd.conf** 文件中找到子网声明的网络。

下面是一个基本 **/etc/dhcp/dhcpd.conf** 文件，适用于有两个网络接口的服务器，接口 **eth0** 用于 **10.0.0.0/24** 网络，接口 **eth1** 用于 **172.16.0.0/24** 网络。多 **subnet** 声明可让您为多个网络定义不同的设置：

```
default-lease-time 600;
max-lease-time 7200;
subnet 10.0.0.0 netmask 255.255.255.0 {
    option subnet-mask 255.255.255.0;
    option routers 10.0.0.1;
    range 10.0.0.5 10.0.0.15;
}
subnet 172.16.0.0 netmask 255.255.255.0 {
    option subnet-mask 255.255.255.0;
    option routers 172.16.0.1;
    range 172.16.0.5 172.16.0.15;
}
```

```
subnet 10.0.0.0 netmask 255.255.255.0;
```

要求为 **DHCP** 服务器提供的每个网络提供 **subnet** 声明。多子网需要多个 **subnet** 声明。如果 **DHCP** 服务器不包含处于 **subnet** 声明中的网络，则 **DHCP** 服务器不会为那个网络提供服务。

如果只有一个 **subnet** 声明，且没有处于那个子网范围中的网络接口，则 **DHCP** 守护进程会无法启

动，并在 `/var/log/messages` 中记录如下出错信息：

```
dhcpcd: No subnet declaration for eth0 (0.0.0.0).
dhcpcd: ** Ignoring requests on eth0.  If this is not what
dhcpcd:    you want, please write a subnet declaration
dhcpcd:    in your dhcpcd.conf file for the network segment
dhcpcd:    to which interface eth1 is attached.  **
dhcpcd:
dhcpcd:
dhcpcd: Not configured to listen on any interfaces!
```

option subnet-mask 255.255.255.0;

option subnet-mask 选项定义子网掩码，并覆盖 **subnet** 声明中的 **netmask** 值。在简单示例中，子网及子网掩码值相同。

option routers 10.0.0.1;

option routers 选项为该子网地定义默认网关。系统访问不同子网中的内部网络及外部网络时需要这个配置。

range 10.0.0.5 10.0.0.15;

range 选项指定大量可用 **IP** 地址。可从指定 **IP** 地址范围中为系统分配一个地址。

详情请查看 **dhcpcd.conf(5)** man page。

10.4.1. 系统配置

做任何改动前，请备份当前的 `/etc/sysconfig/dhcpd` 和 `/etc/dhcp/dhcpd.conf` 文件。

为多网络配置单一系统

下面的 `/etc/dhcp/dhcpd.conf` 示例创建两个子网，并为同一系统配置一个 **IP** 地址，具体值要看该系统连接的网络：

```
default-lease-time 600;
max-lease-time 7200;
subnet 10.0.0.0 netmask 255.255.255.0 {
    option subnet-mask 255.255.255.0;
    option routers 10.0.0.1;
    range 10.0.0.5 10.0.0.15;
}
subnet 172.16.0.0 netmask 255.255.255.0 {
    option subnet-mask 255.255.255.0;
    option routers 172.16.0.1;
    range 172.16.0.5 172.16.0.15;
}
host example0 {
    hardware ethernet 00:1A:6B:6A:2E:0B;
    fixed-address 10.0.0.20;
}
host example1 {
    hardware ethernet 00:1A:6B:6A:2E:0B;
    fixed-address 172.16.0.20;
}
```

host example0

host 声明定义单一系统的具体参数，比如 **IP** 地址。要为多台主机配置具体参数，请使用多个 **host** 声明。

大多数 **DHCP** 客户端会忽略 **host** 声明中的名称，因此该名称可以是任意值，只要与其他 **host** 声明不同即可。要为多个网络配置相同的系统，请在每个 **host** 声明中使用不同的名称，否则 **DHCP** 守护进程将无法启动。系统是根据 **hardware ethernet** 选项识别，而不是 **host** 声明中的名称识别。

hardware ethernet 00:1A:6B:6A:2E:0B;

hardware ethernet 选项可识别该系统。要找到这个地址，请运行 **ip link** 命令。

fixed-address 10.0.0.20;

fixed-address 选项会为 **hardware ethernet** 选项指定的系统分配一个有效 **IP** 地址。这个地址必须在 **range** 选项指定的 **IP** 地址范围以外。

如果 **option** 语句的结尾处不是分号，则 **DHCP** 守护进程将无法启动，并在 **/var/log/messages** 文件中记录出错信息：

```
/etc/dhcp/dhcpd.conf line 20: semicolon expected.
dhcpd: }
dhcpd: ^
dhcpd: /etc/dhcp/dhcpd.conf line 38: unexpected end of file
dhcpd:
dhcpd: ^
dhcpd: Configuration file errors encountered -- exiting
```

配置有多个网络接口的系统

下面的 **host** 声明配置有多个网络接口的单一系统，以便每个接口都使用同样的 **IP** 地址。这个配置不适用于两个网络接口同时连接到同一网络的情况：

```
host interface0 {
    hardware ethernet 00:1a:6b:6a:2e:0b;
    fixed-address 10.0.0.18;
}
host interface1 {
    hardware ethernet 00:1A:6B:6A:27:3A;
    fixed-address 10.0.0.18;
}
```

例如：**interface0** 是第一个网络接口，**interface1** 是第二个网络接口。不同的 **hardware ethernet** 选项会识别每个接口。

如果某个系统连接到另一个网络，并添加更多 **host** 声明，请记住：

- ✱ 为该主机连接的网络分配一个有效 **fixed-address**。
- ✱ 在 **host** 声明中为其分配一个区别于其他名称的名字。

如果 **host** 声明中的名称不是唯一的，则 **DHCP** 守护进程将无法启动，并在 **/var/log/messages** 中记录如下出错信息：

```
dhcpcd: /etc/dhcp/dhpcpd.conf line 31: host interface0: already exists
dhcpcd: }
dhcpcd: ^
dhcpcd: Configuration file errors encountered -- exiting
```

这个错误是由于在 `/etc/dhcp/dhpcpd.conf` 中有多个 `host interface0` 声明造成的。

10.5. 用于 IPv6 的 DHCP (DHCPv6)

ISC DHCP 包括对 IPv6 (即 DHCPv6) 的支持, 因为 4.x 发行本有 DHCPv6 服务器、客户端及中继代理功能。这些代理程序同时支持 IPv4 和 IPv6, 但每次只能管理一个协议; 在双重支持中, 它们必须分别为 IPv4 和 IPv6 启动。例如: 分别编辑 IPv4 和 IPv6 的配置文件 `/etc/dhcp/dhpcpd.conf` 和 `/etc/dhcp/dhpcpd6.conf`, 然后运行下面的命令:

```
~]# systemctl start dhcpcd
~]# systemctl start dhcpcd6
```

DHCPv6 服务器配置文件只在 `/etc/dhcp/dhpcpd6.conf` 目录中。

可在 `/usr/share/doc/dhcp-version/dhpcpd6.conf.example` 目录中找到示例服务器配置文件。

简单的 DHCPv6 服务器配置文件类似如下:

```
subnet6 2001:db8:0:1::/64 {
    range6 2001:db8:0:1::129 2001:db8:0:1::254;
    option dhcp6.name-servers fec0:0:0:1::1;
    option dhcp6.domain-search "domain.example";
}
```

10.6. 其他资料

以下资源提供有关 DHCP 的附加信息。

10.6.1. 已安装文档

- ✦ `dhcpcd(8)` man page — 论述 DHCP 守护进程的工作原理。
- ✦ `dhcpcd.conf(5)` man page — 解释如何配置 DHCP 配置文件, 包括示例。
- ✦ `dhcpcd.leases(5)` man page — 描述持久租用数据库。
- ✦ `dhcp-options(5)` man page — 解释在 `dhcpcd.conf` 中声明 DHCP 选项的语法, 包括示例。
- ✦ `dhcrelay(8)` man page — 解释 DHCP 中继代理程序及其配置选项。
- ✦ `/usr/share/doc/dhcp-version/` — 包括 DHCP 服务当前版本的示例文件、README 文件及发行笔记。

第 11 章 DNS 服务器

DNS（域名系统）是分布式数据库系统，可用于将主机名与其各自 **IP** 地址关联。对用户而言，这样可根据他们通常使用的名称指向网络中的机器，相比数字网络地址更方便记忆。对系统管理员而言，使用 **DNS** 服务器（也称**名称服务器**）可更改主机 **IP** 地址，而不会影响使用名称进行查询。**DNS** 数据库不仅可用于将 **IP** 地址解析为域名，还可在部署 **DNSSEC** 后得到更为广泛的应用。

11.1. DNS 简介

DNS 通常采用一个或多个为某些域认证的集中服务器部署。客户端主机请求来自名称服务器的信息时，通常会连接到端口 53。然后名称服务器会解析请求的名称。如果将名称服务器配置为递归名称服务器，并没有授权回答，或者没有为之前的查询缓存的回答，它会查询其他名称服务器（即 *root 名称服务器*），决定哪个是这个要查询名称的授权名称服务器，然后查询以获取请求的名称。仅作为授权配置的名称服务器若禁用递归功能，则不会代表客户端进行查询。

11.1.1. 名称服务器区域

在 **DNS** 服务器中，所有信息都保存在基本数据元素（即**资源记录**，**RR**）中。[RFC 1034](#) 给出了资源记录定义。采用树状结构管理域名。每层均使用句号（.）分开。例如：在 **root** 域中，. 表示 **DNS** 的 **root**，即层 0。域名 **com** 指的是**顶级域**（**TLD**），即 **root** 域（.）的下一层，也是层级结构的第一层。域名 **example.com** 是层级结构的第二层。

例 11.1. 简单资源记录

简单资源记录（RR）示例：

example.com.	86400	IN	A	192.0.2.1
--------------	-------	----	---	-----------

域名 **example.com** 是 **RR** 的**所有者**。值 **86400** 是**生存时间**（**TTL**）。字母 **IN** 的含义是“互联网系统”，代表 **RR** 的分类。字母 **A** 代表 **RR** 的**类型**（在这个示例中是主机地址）。主机地址 **192.0.2.1** 是包含在这个 **RR** 最后一部分的数字。这个一行的示例是一个 **RR**。一组使用同一类型、拥有者和分类的 **RR** 构成**资源记录集**（**RRSet**）。

区域是通过**区域文件**在授权名称服务器中定义，该文件包含每个区域中的资源记录定义。**Zone** 文件保存在**主名称服务器**（也称**主名称服务器**，在此可更改这些文件）及**辅名称服务器**（也称**从属名称服务器**，在此接受主名称服务器中的区域定义）。主、辅名称服务器都是这个区域的授权，并查看相同的客户端。根据具体配置，所有名称服务器都可作为主或辅服务器同时用于多个区域。

注：**DNS** 和 **DHCP** 服务器的管理员及所有部署应用程序都应接受这个在机构中使用的名称服务器格式。有关名称服务器格式详情，请查看 [第 3.1.1 节“建议到命名方法”](#)。

11.1.2. 名称服务器类型

有两种名称服务器配置类型：

授权

授权名称服务器应答属于该区域的资源记录。这个类别包括主（master）和从属（slave）名称服务器。

递归

递归名称服务器提供解析服务，但不为任何区域授权。在固定时间段内应答所有者内存中缓存的解析，该时间段由查询的资源记录指定。

虽然名称服务器可同时既是授权，又是递归，但建议不要合并使用两种配置类型。要使其工作，所有客户端应随时都可以使用授权服务器。另一方面，因为递归服务器查询比授权响应所需时间要长得多，因此递归服务器仅应适用于有限的客户端，否则会容易受到分布式拒绝服务（DDoS）攻击。

11.1.3. BIND 作为名称服务器

BIND 包含一组与 DNS 相关的程序，其中包括名为 **named** 的名称服务器、管理程序 **rndc** 及 debug 工具 **dig**。有关如何在 Red Hat Enterprise Linux 中运行服务的详情，请查看 [《Red Hat Enterprise Linux 7 系统管理员指南》](#)。

11.2. BIND

本小节论述了 **BIND**（Berkeley Internet Name Domain），Red Hat Enterprise Linux 包含的 **DNS** 服务器。着重介绍其结构和配置文件，并论述了如何对其进行本地和远程管理。

11.2.1. 空白区域

BIND 配置大量“空白区域”，以防止递归服务器向无法使用它们的互联网服务器发送不必要请求（因而造成进行查询的客户端的 SERVFAIL 响应延迟）。这些空白区域可保证返回即时且授权的 NXDOMAIN 响应。配置选项 **empty-zones-enable** 控制是否生成空白区域，还可同时使用 **disable-empty-zone** 选项在其使用的默认前缀列表中禁用一个或多个空白区域。

已增加为 [RFC 1918](#) 前缀生成的空白区域数，同时 **BIND 9.9** 或以上的用户会在未指定 **empty-zones-enable**（默认为 **yes**），以及特别将其设定为 **yes** 时都会看到 [RFC 1918](#) 空白区域。

11.2.2. 配置 DHCP 服务器

启动 **named** 服务后，它会如 [表 11.1 “named 服务配置文件”](#) 所述从文件中读取配置。

表 11.1. named 服务配置文件

路径	描述
/	配置文件
/etc/named/	主配置文件中包含的配置文件辅助目录。

配置文件由带嵌套选项的语句集合组成，这些选项使用大括号 ({ 和 }) 括起来。注：编辑此文件时必须非常小心，以避免语法错误，否则将无法启动 **named** 服务。典型的 **/etc/named.conf** 文件类似如下：

```
statement-1 ["statement-1-name"] [statement-1-class] {
    option-1;
    option-2;
    option-N;
};
statement-2 ["statement-2-name"] [statement-2-class] {
    option-1;
    option-2;
    option-N;
};
statement-N ["statement-N-name"] [statement-N-class] {
```

```
option-1;
option-2;
option-N;
};
```



注意

如果已安装 *bind-chroot* 软件包，BIND 就会在 **chroot** 环境中运行。在那种情况下，初始化脚本将使用 `mount --bind` 命令挂载上述配置文件，以便可以在这个环境以外管理该配置。不需要向 `/var/named/chroot/` 目录复制任何内容，因为会自动挂载该目录。这样可简化维护服务，因为在 **chroot** 环境中运行时，不需要对 **BIND** 配置文件进行任何特别处理。可使用 **BIND** 管理所需的一切，而无需在 **chroot** 环境中运行。

如果 `/var/named/chroot/` 中的对应挂载点为空，则以下目录会自动挂载至 `/var/named/chroot/` 目录：

- ✧ `/etc/named`
- ✧ `/etc/pki/dnssec-keys`
- ✧ `/run/named`
- ✧ `/var/named`
- ✧ `/usr/lib64/bind` 或 `/usr/lib/bind`（视具体架构而定）。

如果在 `/var/named/chroot/` 中不存在以下文件，则也会将其挂载到目标文件中：

- ✧ `/`
- ✧ `/etc/rndc.conf`
- ✧ `/etc/rndc.key`
- ✧ `/etc/named.rfc1912.zones`
- ✧ `/etc/named.dnssec.keys`
- ✧ `/etc/named.iscdlv.key`
- ✧ `/etc/named.root.key`



重要

需要为每个挂载到 **chroot** 环境中的文件生成一个备用副本后，方可编辑原始文件。也可以使用禁用“edit-a-copy”模式的编辑器。例如：要使用 Vim 编辑在 **chroot** 环境中运行的 BIND 的配置文件 `/etc/named.conf`，请作为 **root** 运行以下命令：

```
~]# vim -c "set backupcopy=yes" /etc/named.conf
```

11.2.2.1. 在 chroot 环境中安装 BIND

要安装在 **chroot** 环境中运行的 **BIND**，请作为 **root** 运行以下命令：

```
~]# yum install bind-chroot
```

要启用 **named-chroot** 服务，首先请检查是否运行 **named** 服务，方法是运行以下命令：


```
~]$ systemctl status named
```

如果该服务正在运行，则必须将其禁用。

作为 **root** 运行以下命令，禁用 **named**：

```
~]# systemctl stop named
```

```
~]# systemctl disable named
```

然后，作为 **root** 运行以下命令启用 **named-chroot** 服务：

```
~]# systemctl enable named-chroot
```

```
~]# systemctl start named-chroot
```

作为 **root** 运行以下命令，检查 **named-chroot** 服务状态：

```
~]# systemctl status named-chroot
```

11.2.2.2. 常用语句类型

以下是在 **/etc/named.conf** 中常用的语句类型：

acl

acl（访问控制列表）语句可让您定义主机组，这样可以允许或者拒绝对该名称服务器的访问。它使用以下格式：

```
acl acl-name {  
    match-element;  
    ...  
};
```

acl-name 语句名称是访问控制列表名称，*match-element* 选项通常是独立 **IP** 地址（比如 **10.0.1.1**），或者无类别域际路由选择（CIDR）网络标记（例如：**10.0.1.0/24**）。有关已经定义的关键字列表请查看 [表 11.2 “预定义访问控制列表”](#)。

表 11.2. 预定义访问控制列表

关键字	描述
any	与所有 IP 地址匹配。
localhost	与本地系统使用的 IP 地址匹配。
localnets	与所有本地系统连接的网络中的 IP 地址匹配。
none	与任何 IP 地址都不匹配。

acl 语句在与其他语句联合使用时特别有用，比如 **options**。[例 11.2 “联合使用 acl 和 Options”](#) 定义两个访问控制列表，**black-hats** 和 **red-hats**。并在为 **red-hats** 赋予普通访问授权的同时，在 **black-hats** 中添加 **black-hats**。

例 11.2. 联合使用 `acl` 和 `Options`

```
acl black-hats {
    10.0.2.0/24;
    192.168.0.0/24;
    1234:5678::9abc/24;
};
acl red-hats {
    10.0.1.0/24;
};
options {
    blackhole { black-hats; };
    allow-query { red-hats; };
    allow-query-cache { red-hats; };
};
```

`include`

`include` 语句可让您在 `/etc/named.conf` 中包含文件，这样就可将可能的敏感数据保存在有严格权限的独立文件中。其格式如下：

```
include "file-name"
```

`file-name` 语句名称是文件的绝对路径。

例 11.3. 在 `/etc/named.conf` 中添加文件

```
include "/etc/named.rfc1912.zones";
```

`options`

`options` 语句可让您定义全局服务器配置选项，也可让您设定其他语句的默认形式。您可使用它指定 `named` 工作目录位置，允许的查询类型等等。其格式如下：

```
options {
    option;
    ...
};
```

有关常用 `option` 指令列表请查看下面的 [表 11.3 “常用配置选项”](#)。

表 11.3. 常用配置选项

选项	描述
<code>allow-query</code>	指定哪些主机可以在名称服务器中查询授权资源记录。它接受访问控制列表、IP 地址集合、或者 CIDR 标记中的网络。所有主机均默认有此功能。
<code>allow-query-cache</code>	指定哪些主机可以在名称服务器中查询未授权数据，比如递归查询。默认只允许 <code>localhost</code> 和 <code>localnets</code> 。
<code>blackhole</code>	指定哪些主机不能查询名称服务器。这个选项可在某个主机或者网络向服务器提出过多请求时使用。默认选项为 <code>none</code> 。

选项	描述
directory	为 named 服务指定工作目录。默认选项为 /var/named/ 。
disable-empty-zone	用于在使用的默认前缀列表中禁用一个或多个空白区域。可在 options 语句及 view 语句中指定。可多次使用该选项。
dnssec-enable	指定是否返回与 DNSSEC 关联的资源记录。默认选项为 yes 。
dnssec-validation	指定是否提供使用 DNSSEC 认证的资源记录。默认选项为 yes 。
empty-zones-enable	控制是否生成空白区域。只能在 options 语句中指定。
forwarders	为名称服务器指定有效 IP 地址列表，所有申请应转发到此地址进行解析。
forward	指定 forwarders 指令行为。它接受以下选项： <ul style="list-style-type: none"> ✧ first — 解析该名称前，服务器将查询 forwarders 指令中列出的名称服务器。 ✧ only — 无法查询 forwarders 指令中列出的名称服务器前，该完全将不会自行解析该名称。
listen-on	指定侦听查询的 IPv4 网络接口。在 DNS 服务器中还作为网关使用。可以使用这个选项回答来自只使用单一网络的查询。默认使用所有 IPv4 接口。
listen-on-v6	指定侦听查询的 IPv4 网络接口。在 DNS 服务器中还作为网关使用。您可以使用这个选项回答来自只使用单一网络的查询。默认使用所有 IPv6 接口。
max-cache-size	指定服务器缓存使用的最大内存数。达到极限后，该服务器可让记录永久过期，这样就不会超过极限。在使用多个 view 的服务器中，该极限分别适用于每个查看缓存。默认选项为 32M 。
notify	指定更新 zone 后是否通知辅名称服务器。它接受以下选项： <ul style="list-style-type: none"> ✧ yes — 该服务器会通知所有辅名称服务器。 ✧ no — 该服务器不会通知任何辅名称服务器。 ✧ master-only — 该服务器只通知该区域的主服务器。 ✧ explicit — 该服务器只通知在 zone 语句的 also-notify 列表中指定的辅服务器。
pid-file	指定由 named 服务生成的进程 ID 文件位置。
recursion	指定是否作为递归服务器使用。默认选项为 yes 。
statistics-file	为统计文件指定备选位置。默认使用 /var/named/named.stats 文件。



注意

已将 **named** 用于运行时数据的目录从 BIND 默认位置 **/var/run/named/** 移动到新位置 **/run/named/**。结果是将 PID 文件从默认位置 **/var/run/named/named.pid** 移动到新位置 **/run/named/named.pid**。此外，已将 **session-key** 文件移动至 **/run/named/session.key**。需要在选项部分使用语句中指定这些位置。详情请查看 [例 11.4 “使用 options 语句”](#)。



重要

要防止分布式拒绝服务（distributed denial of service, DDoS）攻击，建议您使用 **allow-query-cache** 选项只为客户端的特定子集限制递归 **DNS** 服务。

完整选项列表请参考 [第 11.2.8.1 节 “已安装文档”](#) 中提供的《*BIND 9 管理员参考手册*》及 **named.conf** manual page。

例 11.4. 使用 options 语句

```
options {
    allow-query          { localhost; };
    listen-on port      53 { 127.0.0.1; };
    listen-on-v6 port   53 { ::1; };
    max-cache-size      256M;
    directory            "/var/named";
    statistics-file      "/var/named/data/named_stats.txt";

    recursion            yes;
    dnssec-enable        yes;
    dnssec-validation    yes;

    pid-file             "/run/named/named.pid";
    session-keyfile       "/run/named/session.key";
};
```

zone

zone 语句可让您定义区域的特点，比如其配置文件位置和具体区域选项，并可用来覆盖全局 **options** 语句。其格式如下：

```
zone zone-name [zone-class] {
    option;
    ...
};
```

如 [表 11.4 “Zone 语句中的常用选项”](#) 所述，**zone-name** 属性是区域的名称，**zone-class** 是区域的自选等级，**option** 是 **zone** 语句选项。

zone-name 属性特别重要，因为它是为 **\$ORIGIN** 指令分配的默认值，该指令用于 **/var/named/** 目录中的对应区域文件。**named** 守护进程可将区域名称附加到区域文件中列出的任意非全限定域名中。例如：如果 **zone** 语句定义了 **example.com** 名称空间，请使用 **example.com** 作为 **zone-name**，这样就可将其放在 **example.com** 区域文件主机名的最后。

有关区域文件的详情，请查看 [第 11.2.3 节 “编辑区域文件”](#)。

表 11.4. Zone 语句中的常用选项

选项	描述
allow-query	指定哪些客户端可请求查询这个区域的信息。这个选项会覆盖全局 allow-query 选项。默认允许所有查询请求。

选项	描述
allow-transfer	指定哪些辅服务器可请求传递区域的信息。默认允许所有传递请求。
allow-update	指定哪些主机可在其区域中动态更新信息。默认选项是拒绝所有动态更新请求。 注：允许主机更新其区域信息时应格外小心。不要在这个选项中设置 IP 地址，除非该服务器位于可信网络中。请如 第 11.2.6.3 节“事务处理签名（Transaction Signatures, TSIG）” 所述使用 TSIG 密钥。
file	指定 named 工作目录中包含区域配置数据的文件名称。
masters	指定可请求授权 zone 信息的 IP 地址。只有将区域定义为 type slave 时方可使用这个选项。
notify	指定更新 zone 后是否通知辅名称服务器。它接受以下选项： <ul style="list-style-type: none"> ✦ yes — 该服务器会通知所有辅名称服务器。 ✦ no — 该服务器不会通知任何辅名称服务器。 ✦ master-only — 该服务器只通知该区域的主服务器。 ✦ explicit — 该服务器只通知在 zone 语句的 also-notify 列表中指定的辅服务器。
type	指定区域类型。它接受以下选项： <ul style="list-style-type: none"> ✦ delegation-only — 加强基础区域的授权状态，比如 COM、NET 或者 ORG。任何没有具体说明或暗示授权的回答都将被视为 NXDOMAIN。这个选项只可用于在递归或者缓存部署中使用的 TLD（顶级域，Top-Level Domain）或者 root 区域文件。 ✦ forward — 将所有关于这个区域信息的请求转发到其他名称服务器。 ✦ hint — 用来指向 root 名称服务器的特殊区域类型，root 名称服务器是用来解析使用其他方法无法了解的 zone。hint 区域的默认配置就足够。 ✦ master — 这个 zone 授权的专用名称服务器。如果某个区域的配置文件位于这个系统中，则应将该 zone 设定为 master。 ✦ slave — 作为这个区域的专用从属服务器的名称服务器。在 masters 指令中指定主服务器。

主、辅名称服务器 `/etc/named.conf` 文件的更改包括添加、修改、或者删除 **zone** 语句，且让名称服务器可有效工作通常只需要 **zone** 语句选项的小子集。

在 [例 11.5 “主名称服务器的 Zone 语句”](#) 中，将区域识别为 **example.com**，类型设定为 **master**，并让 **named** 服务读取 `/var/named/example.com.zone` 文件。它还只允许一个辅名称服务器（**192.168.0.2**）转移到该 zone。

例 11.5. 主名称服务器的 Zone 语句

```
zone "example.com" IN {
    type master;
    file "example.com.zone";
    allow-transfer { 192.168.0.2; };
};
```

辅服务器的 **zone** 语句略微不同。类型设定为 **slave**，同时 **masters** 指令会通告 **named** 主服务器

的 IP 地址。

在 [例 11.6 “辅名称服务器的 Zone 语句”](#) 中，将 **named** 服务配置为在 **192.168.0.1** IP 地址向主服务器查询 **example.com** zone 的信息。然后将接收到的信息保存在 **/var/named/slaves/example.com.zone** 文件中。注：必须将所有从属 zone 都放在 **/var/named/slaves/** 目录中，否则该服务就无法转移到该 zone。

例 11.6. 辅名称服务器的 Zone 语句

```
zone "example.com" {
    type slave;
    file "slaves/example.com.zone";
    masters { 192.168.0.1; };
};
```

11.2.2.3. 其他语句类型

以下是在 **/etc/named.conf** 中不常用的语句类型：

controls

controls 语句可让您配置使用 **rndc** 命令管理 **named** 服务所需的各种安全要求。

有关 **rndc** 程序及其用法请参考 [第 11.2.4 节 “使用 rndc 程序”](#)。

key

key 语句可让您根据名称定义具体密钥。密钥是用来认证各种动作，比如安全更新或者使用 **rndc** 命令。**key** 有两个选项：

- » **algorithm** *algorithm-name* — 使用的算法类型（例如：**hmac-md5**）。
- » **secret** "*key-value*" — 加密密钥。

有关 **rndc** 程序及其用法请参考 [第 11.2.4 节 “使用 rndc 程序”](#)。

logging

logging 语句可让您使用各种类型的日志，也称**频道**。可以在该语句中使用 **channel** 选项，构建自定义日志类型，该类型可有自己的文件名（**file**）、大小限制（**size**）、版本号（**version**）以及重要程度（**severity**）。定义自定义频道后，可使用 **category** 选项将频道分类并在重启 **named** 服务后开始记录。

默认情况下，**named** 会向 **rsyslog** 守护进程发送标准信息，这些信息会保存在 **/var/log/messages** 文件中。BIND 中内置了几个使用不同安全等级标准频道，比如 **default_syslog**（处理信息日志消息）和 **default_debug**（具体处理 debug 消息）。默认分类称为 **default**，它使用内置频道进行一般日志记录，不需要任何特殊配置。

自定义日志记录过程会非常繁琐，且不在本章论述范围内。有关创建自定义 BIND 日志的详情请参考 [第 11.2.8.1 节 “已安装文档”](#) 中提供的《**BIND 9 管理员参考手册**》。

server

server 语句可让您指定影响 **named** 服务响应远程名称服务器方式的具体选项，特别要考虑通知和区域传送。

transfer-format 选项控制每条信息附带的资源记录数。它可以是 **one-answer**（只有一个资源记录），也可以是 **many-answers**（多条资源记录）。备注：**many-answers** 选项更有效，但旧的 BIND 版本不支持。

trusted-keys

trusted-keys 语句可让您指定安全 DNS（DNSSEC）的分类公钥。有关这个主题的详情请参考 [第 11.2.6.4 节“DNS 安全扩展（DNSSEC）”](#)。

view

view 语句可让您根据正在进行主机查询的名称服务器所在网络生成特殊视图。这可让有些主机接收某个区域的回答，而其他主机接收完全不同的信息。另外，还可让某些区域只能用于具体的可信主机，而非可信主机只能查询其他区域。

只要名称是唯一的就可以尽量使用多视图。**match-clients** 选项可让您指定在具体窗口中使用的 IP 地址。如果在窗口中使用 **options** 语句，它会覆盖已经配置的全局 **options** 语句。最后，大多数 **view** 语句包含多个可在 **match-clients** 列表中使用的 **zone** 语句。

注：**view** 语句的排列顺序很重要，因为会使用第一个语句与具体客户端 IP 地址匹配语句。有关这个主题的详情请参考 [第 11.2.6.1 节“多窗口”](#)。

11.2.2.4. 注释标签

除这些语句外，**/etc/named.conf** 文件还包含注释。**named** 服务会忽略这些注释，但可为用户提供有价值的附加性信息。以下是有效注释标签：

//

// 字符后到该行结束的所有文本都视为注释。例如：

```
notify yes; // notify all secondary nameservers
```

#

字符后到该行结束的所有文本都视为注释。例如：

```
notify yes; # notify all secondary nameservers
```

/* 和 */

所有 /* 和 */ 之间的文本都视为注释。例如：

```
notify yes; /* notify all secondary nameservers */
```

11.2.3. 编辑区域文件

如 [第 11.1.1 节“名称服务器区域”](#) 所示，区域文件中包含有关名称空间的信息。默认是将其保存在位于 **/var/named/** 的 **named** 工作目录中，同时每个区域文件都是根据 **zone** 语句中的 **file** 选项命名，通常在某种程度上与有问题的域关联，并将该文件识别为包含区域数据，比如 **example.com.zone**。

表 11.5. named 服务区域文件

路径	描述
/var/named/	named 服务的工作目录。不允许该名称服务器写入这个目录。
/var/named/slaves/	从属区域的目录。named 服务可写入这个目录。
/var/named/dynamic/	其他文件的目录，比如动态 DNS（DDNS）区域，或者管理的 DNSSEC 密钥。named 服务可写入这个目录。
/var/named/data/	各种统计和 debug 文件目录。named 服务可写入这个目录。

域文件由指令和资源记录组成。指令指定名称服务器执行任务或者在该区域中应用特殊设置；资源记录定义该区域的参数，并为每台主机分配身份识别。虽然指令是自选的，但要求使用资源记录以便为区域提供名称服务。

所有指令和资源记录都应单独使用一行。

11.2.3.1. 常用指令

指令以美元符号（即 \$）开始，后接该指令名称，且通常是在该文件的开始。以下是区域文件中的常用指令：

\$INCLUDE

\$INCLUDE 指令可让您在出现其他文件的时候包括该文件，这样其他区域设置就可以保存在不同的区域文件中。

例 11.7. 使用 \$INCLUDE 指令

```
$INCLUDE /var/named/penguin.example.com
```

\$ORIGIN

\$ORIGIN 指令可让您将区域名附加到不合格记录中，比如那些只有 hostname 的记录。注：如果在 `/etc/named.conf` 中指定该区域，则没有必要使用这个指令，因为默认使用该区域名称。

在 [例 11.8 “使用 \\$ORIGIN 指令”](#) 中，会将所有以点（即 . 符号）结尾的资源记录附加到 `example.com`。

例 11.8. 使用 \$ORIGIN 指令

```
$ORIGIN example.com.
```

\$TTL

\$TTL 指令可让您该区域的默认 *Time to Live*（TTL）值，即区域记录可在多长时间内有效。每个资源记录都包含各自的 TTL 值，这些值可覆盖这个指令。

增大这些值可让远程名称服务器将区域 信息缓存更长时间，减小该区域的查询次数，并延长传播资源记录更改所需时间。

例 11.9. 使用 \$TTL 指令


```
$TTL 1D
```

11.2.3.2. 常用资源记录

以下是经常在区域文件中使用的资源记录：

A

地址记录指定要为某个名称分配的 **IP** 地址。它使用以下格式：

```
hostname IN A IP-address
```

如省略 *hostname* 值，则该记录会指向最新指定的 *hostname*。

在 [例 11.10 “使用资源记录”](#) 中，将 **server1.example.com** 请求指向 **10.0.1.3** 或者 **10.0.1.5**。

例 11.10. 使用资源记录

```
server1  IN  A   10.0.1.3
         IN  A   10.0.1.5
```

CNAME

正规名称记录将名称彼此配对。因此，这类记录有时是指**别名记录**。它使用以下格式：

```
alias-name IN CNAME real-name
```

CNAME 记录是使用常用命名方案服务最长指向的记录，比如 Web 服务器的 **www**。但其用法有很多局限：

- ✧ CNAME 记录不应指向其他 CNAME 记录。这主要是避免可能的无线循环。
- ✧ CNAME 记录不应包含其他资源记录类型（比如 A、NS、MX 等等）。唯一例外的是登录该区域后与 DNSSEC 相关的记录（即 RRSIG、NSEC 等）
- ✧ 其他指向主机完全限定域名（FQDN）资源记录（即 NS、MX、PTR）不应指向 CNAME 记录。

在 [例 11.11 “使用 CNAME 资源记录”](#) 中，**A** 记录将主机名与 **IP** 地址绑定，而 **CNAME** 记录将常用 **www** 主机名指向它。

例 11.11. 使用 CNAME 资源记录

```
server1  IN  A       10.0.1.5
www      IN  CNAME  server1
```

MX

邮件互换记录指定发送到由这个区域控制具体名称空间的邮件应保存在哪里。它使用以下格式：

```
IN MX preference-value email-server-name
```


email-server-name 是完全限定域名 (FQDN)。*preference-value* 可使用数字为名称空间将电子邮件服务器分级, 为某些电子邮件系统赋予其他系统没有的属性。使用最小 *preference-value* 值的 **MX** 资源记录是首选系统。但多个电子邮件服务器可使用同一值以便平均分配邮件量。

在 [例 11.12 “使用 MX 资源记录”](#) 中, 第一个 **mail.example.com** 电子邮件服务器是 **mail2.example.com** 电子邮件服务器在接收来自 **example.com** 域的邮件时的首选。

例 11.12. 使用 MX 资源记录

```
example.com.  IN  MX  10  mail.example.com.
               IN  MX  20  mail2.example.com.
```

NS

名称服务器 记录可为具体区域宣布授权名称服务器。它使用以下格式：

```
ssh username@penguin.example.net
```

nameserver-name 应该是完全限定域名 (FQDN)。注：当有两个名称服务器都作为该域的授权服务器时, 哪个是主服务器, 哪个是辅服务器并不重要。它们都是授权的服务器。

例 11.13. 使用 NS 资源记录

```
IN  NS  dns1.example.com.
IN  NS  dns2.example.com.
```

PTR

指针记录指向该名称空间的另一部分。它使用以下格式：

```
last-IP-digit IN PTR FQDN-of-system
```

last-IP-digit 指令是 **IP** 地址的最后一个数字, 且 *FQDN-of-system* 是完全限定域名 (FQDN)。

PTR 记录主要用于逆向名称解析, 因为它们将 **IP** 地址反向指回到具体名称。有关使用的 **PTR** 记录的更多示例请参考 [第 11.2.3.4.2 节 “逆向名称解析区域文件”](#)。

SOA

权限启动记录发布有关名称服务器的名称空间的重要授权信息。它在指令后面, 是区域文件的第一个资源记录。它使用以下格式：

```
@  IN  SOA  primary-name-server hostmaster-email (
        serial-number
        time-to-refresh
        time-to-retry
        time-to-expire
        minimum-TTL )
```

正确的数值是：

- @ 符号将 **\$ORIGIN** 指令 (如果没有设定 **\$ORIGIN** 指令, 即区域名称) 作为这个 **SOA** 资源记录定义的名称空间。

- *primary-name-server* 指令是为这个域授权的主名称服务器的主机名。
- *hostmaster-email* 指令是与名称空间联络的个人电子邮件。
- *serial-number* 指令是个数值，每次区域文件有变化时这个值就会增加，说明 **named** 现在应该重新载入这个区域。
- *time-to-refresh* 指令是在该区域文件有任何修改时，辅名称服务器用来决定在询问主名称服务器前需要等待多长时间的时间值。
- *time-to-retry* 指令是在主名称服务器不响应的事件中，辅名称服务器用来决定在提交刷新请求前需要等待多长时间的时间值。如果主服务器在 *time-to-expire* 指令规定的时间内没有回应刷新请求，那么辅服务器会停止响应，因为授权服务器对那个名称服务器有疑问。
- 在 BIND 4 和 8 中，*minimum-TTL* 指令是其他名称服务器缓存该区域信息的时间。在 BIND 9 中，它定义为否定回答提供的缓冲时间长度。否定回答缓存最长可设定为 3 小时（即 **3H**）。

配置 BIND 时，所有时间都使用秒为单位。但在指定秒之外的时间单位时可使用缩写，比如分钟（M）、小时（H）、天（D）和周（W）。表 11.6 “秒与其他时间单位对比”为您提供秒为单位的时间以及使用其他格式的对等时间。

表 11.6. 秒与其他时间单位对比

秒	其他时间单位
60	1M
1800	30M
3600	1H
10800	3H
21600	6H
43200	12H
86400	1D
259200	3D
604800	1W
31536000	365D

例 11.14. 使用 SOA 资源记录

```
@ IN SOA dns1.example.com. hostmaster.example.com. (  
    2001062501 ; serial  
    21600      ; refresh after 6 hours  
    3600       ; retry after 1 hour  
    604800     ; expire after 1 week  
    86400 )    ; minimum TTL of 1 day
```

11.2.3.3. 注释标签

除资源记录和指令外，区域文件还包含注释。**named** 服务会忽略这些注释，但可为用户提供有价值的附加信息。分号之后到这一行结束前的所有文本都视为注释。例如：

```
604800 ; expire after 1 week
```

11.2.3.4. 用法示例

以下解释了每个选项所配置的项目：

11.2.3.4.1. 简单区域文件

[例 11.15 “简单区域文件”](#) 演示了标准指令使用及 **SOA** 值。

例 11.15. 简单区域文件

```
$ORIGIN example.com.
$TTL 86400
@           IN  SOA  dns1.example.com.  hostmaster.example.com. (
                2001062501  ; serial
                21600       ; refresh after 6 hours
                3600        ; retry after 1 hour
                604800      ; expire after 1 week
                86400 )     ; minimum TTL of 1 day
;
;
                IN  NS   dns1.example.com.
                IN  NS   dns2.example.com.
dns1           IN  A     10.0.1.1
                IN  AAAA  aaaa:bbbb::1
dns2           IN  A     10.0.1.2
                IN  AAAA  aaaa:bbbb::2
;
;
@           IN  MX     10  mail.example.com.
                IN  MX     20  mail2.example.com.
mail         IN  A      10.0.1.5
                IN  AAAA  aaaa:bbbb::5
mail2        IN  A      10.0.1.6
                IN  AAAA  aaaa:bbbb::6
;
;
; This sample zone file illustrates sharing the same IP addresses
; for multiple services:
;
services     IN  A      10.0.1.10
                IN  AAAA  aaaa:bbbb::10
                IN  A      10.0.1.11
                IN  AAAA  aaaa:bbbb::11

ftp          IN  CNAME  services.example.com.
www          IN  CNAME  services.example.com.
;
;
```

在这个示例中，将授权名称服务器设定为 **dns1.example.com** 和 **dns2.example.com**，并使用 **A** 记录分别将其绑定到 **10.0.1.1** 和 **10.0.1.2** IP 地址。

使用 **MX** 记录配置的电子邮件服务器通过 **A** 记录指向 **mail** 和 **mail2**。因为这些名称没有以点结尾，它们后面的 **\$ORIGIN** 域会将其扩展到 **mail.example.com** 和 **mail2.example.com**。

使用 **CNAME** 记录将可在标准名称中使用的服务，比如 **www.example.com** (**WWW**) 指向正确的服务器。

应使用 `/etc/named.conf` 中类似如下的 `zone` 语句将这个区域调入服务：

```
zone "example.com" IN {
    type master;
    file "example.com.zone";
    allow-update { none; };
};
```

11.2.3.4.2. 逆向名称解析区域文件

逆向名称解析区域文件是用来将 **IP** 地址转换称完全限定域名 (FQDN) 的具体名称空间。它与标准区域文件很相似，除了在连接 **IP** 地址和完全限定域名时使用 **PTR** 资源记录，如 [例 11.16 “逆向名称解析区域文件”](#) 所示。

例 11.16. 逆向名称解析区域文件

```
$ORIGIN 1.0.10.in-addr.arpa.
$TTL 86400
@ IN SOA dns1.example.com. hostmaster.example.com. (
    2001062501 ; serial
    21600      ; refresh after 6 hours
    3600       ; retry after 1 hour
    604800     ; expire after 1 week
    86400 )    ; minimum TTL of 1 day
;
@ IN NS dns1.example.com.
;
1 IN PTR dns1.example.com.
2 IN PTR dns2.example.com.
;
5 IN PTR server1.example.com.
6 IN PTR server2.example.com.
;
3 IN PTR ftp.example.com.
4 IN PTR ftp.example.com.
```

在这个示例中，**IP** 地址 **10.0.1.1** 通过 **10.0.1.6** 指向对应的完全限定域名。

应使用 `/etc/named.conf` 中类似如下的 `zone` 语句将这个区域调入服务：

```
zone "1.0.10.in-addr.arpa" IN {
    type master;
    file "example.com.rr.zone";
    allow-update { none; };
};
```

这个示例与标准 `zone` 语句相比，除区域名称外太多什么不同之处。注：逆向名称解析要求区域文件中前三个逆向解析的 **IP** 地址块后接 `.in-addr.arpa`。这样可让在逆向名称解析区域文件中使用的单独 **IP** 块与这个区域关联。

11.2.4. 使用 `rndc` 程序

rndc 程序是可用来管理 **named** 服务的命令行工具，在本地机器和远程机器中都可使用，其用法如下：

```
rndc [option...] command [command-option]
```

11.2.4.1. 配置该工具

为防止对该服务的未授权访问，必须将 **named** 配置为侦听所选端口（默认为 **953**），同时必须在该服务和 **rndc** 程序中使用同一密钥。

表 11.7. 相关文件

路径	描述
<code>/etc/named.conf</code>	named 服务的默认配置文件。
<code>/etc/rndc.conf</code>	rndc 程序的默认配置文件。
<code>/etc/rndc.key</code>	默认密钥位置。

rndc 配置文件位于 `/etc/rndc.conf`。如果没有该文件，则该程序将使用 `/etc/rndc.key` 中的密钥，该文件是在使用 **rndc-confgen -a** 命令安装的过程中自动生成。

如 [第 11.2.2.3 节“其他语句类型”](#) 所述，**named** 服务是使用 `/etc/named.conf` 配置文件中的 **controls** 语句配置的。除非使用这个语句，否则只能允许来自回送地址（**127.0.0.1**）的连接，并使用 `/etc/rndc.key` 中的密钥。

有关这个主题的详情请参考 [第 11.2.8 节“其他资料”](#) 中《*BIND 9 管理员参考手册*》列出的手册页面。



重要

要防止非授权用户向该服务发送 control 命令，请确定只有 **root** 用户可以读取 `/etc/rndc.key` 文件：

```
~]# chmod o-rwx /etc/rndc.key
```

11.2.4.2. 检查服务状态

请使用以下命令检查 **named** 服务当前的状态：

```
~]# rndc status
version: 9.7.0-P2-RedHat-9.7.0-5.P2.el6
CPUs found: 1
worker threads: 1
number of zones: 16
debug level: 0
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF
recursive clients: 0/0/1000
tcp clients: 0/100
server is up and running
```

11.2.4.3. 重新载入配置及 Zone

要重新载入配置文件及 zone，在 shell 提示下键入以下命令：

```
~]# rndc reload
server reload successful
```

这样将重新载入 zones，同时保证之前的所有缓存的响应，以便您可以在不丢失任何保存的名称解析的情况下修改 zones 文件。

请在 **reload** 命令后指定其名称即可重新载入单一 zone，例如：

```
~]# rndc reload localhost
zone reload up-to-date
```

最后，如果只想重新载入配置文件和新添加的区域，请输入：

```
~]# rndc reconfig
```



注意

如果您想要手动修改使用动态 **DNS** (DDNS) 的区域，请确定首先运行 **freeze** 命令：

```
~]# rndc freeze localhost
```

完成后，请运行 **thaw** 命令再次允许 **DDNS**，并重新载入该区域文件：

```
~]# rndc thaw localhost
The zone reload and thaw was successful.
```

11.2.4.4. 更新区域密钥

请使用 **sign** 命令更新 DNSSEC 密钥并注册该区域。例如：

```
~]# rndc sign localhost
```

注：要使用上面的命令注册区域，则必须在 zone 语句中将 **auto-dnssec** 选项设定为 **maintain**。例如：

```
zone "localhost" IN {
    type master;
    file "named.localhost";
    allow-update { none; };
    auto-dnssec maintain;
};
```

11.2.4.5. 启用 DNSSEC 验证

请作为 **root** 运行以下命令启用 DNSSEC 验证：

```
~]# rndc validation on
```

同样，使用以下命令禁用这个选项：

```
~]# rndc validation off
```

有关如何在 `/etc/named.conf` 中配置这个选项的详情，请参考 [第 11.2.2.2 节“常用语句类型”](#) 中的 `options` 语句。

[《Red Hat Enterprise Linux 7 安全指南》](#) 中有关于 DNSSEC 的完整介绍。

11.2.4.6. 启用 Query Logging

请作为 `root` 运行以下命令启用（或禁用，假设目前处于启用状态）查询日志：

```
~]# rndc querylog
```

请作为 `root` 运行 `status` 命令检查当前设置。

11.2.5. 使用 dig 程序

`dig` 程序是一个命令行工具，可让您执行 DNS 查询和 debug 名称服务器配置。其常规用法如下：

```
dig [@server] [option...] name type
```

常用 `type` 列表请参考 [第 11.2.3.2 节“常用资源记录”](#)。

11.2.5.1. 查找名称服务器

请以如下格式使用该命令查找某个具体域的名称服务器：

```
dig name NS
```

在 [例 11.17 “名称服务器查询示例”](#) 中，使用 `dig` 程序显示 `example.com` 的名称服务器。

例 11.17. 名称服务器查询示例

```
~]$ dig example.com NS

; <<>> DiG 9.7.1-P2-RedHat-9.7.1-2.P2.fc13 <<>> example.com NS
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57883
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;example.com.                IN      NS

;; ANSWER SECTION:
example.com.                 99374   IN      NS      a.iana-servers.net.
example.com.                 99374   IN      NS      b.iana-servers.net.
```

```
;; Query time: 1 msec
;; SERVER: 10.34.255.7#53(10.34.255.7)
;; WHEN: Wed Aug 18 18:04:06 2010
;; MSG SIZE rcvd: 77
```

11.2.5.2. 查找 IP 地址

请以如下格式使用该命令查找为具体域分配的 IP 地址：

```
dig name A
```

在 [例 11.18 “IP 地址查询示例”](#) 中，使用 **dig** 程序显示 **example.com** 的 IP 地址。

例 11.18. IP 地址查询示例

```
~]$ dig example.com A

; <<>> DiG 9.7.1-P2-RedHat-9.7.1-2.P2.fc13 <<>> example.com A
;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 4849
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;example.com.                IN      A

;; ANSWER SECTION:
example.com.                 155606  IN      A      192.0.32.10

;; AUTHORITY SECTION:
example.com.                 99175   IN      NS      a.iana-servers.net.
example.com.                 99175   IN      NS      b.iana-servers.net.

;; Query time: 1 msec
;; SERVER: 10.34.255.7#53(10.34.255.7)
;; WHEN: Wed Aug 18 18:07:25 2010
;; MSG SIZE rcvd: 93
```

11.2.5.3. 查找主机名

请以如下格式使用该命令为具体 IP 地址查找主机名：

```
dig -x address
```

在 [例 11.19 “主机名查询示例”](#) 中，使用 **dig** 程序显示 **192.0.32.10** 分配的主机名。

例 11.19. 主机名查询示例

```
~]$ dig -x 192.0.32.10
```



```

; <<>> DiG 9.7.1-P2-RedHat-9.7.1-2.P2.fc13 <<>> -x 192.0.32.10
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29683
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 5, ADDITIONAL: 6

;; QUESTION SECTION:
;10.32.0.192.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
10.32.0.192.in-addr.arpa. 21600 IN      PTR      www.example.com.

;; AUTHORITY SECTION:
32.0.192.in-addr.arpa. 21600 IN      NS       b.iana-servers.org.
32.0.192.in-addr.arpa. 21600 IN      NS       c.iana-servers.net.
32.0.192.in-addr.arpa. 21600 IN      NS       d.iana-servers.net.
32.0.192.in-addr.arpa. 21600 IN      NS       ns.icann.org.
32.0.192.in-addr.arpa. 21600 IN      NS       a.iana-servers.net.

;; ADDITIONAL SECTION:
a.iana-servers.net.      13688 IN      A        192.0.34.43
b.iana-servers.org.      5844  IN      A        193.0.0.236
b.iana-servers.org.      5844  IN      AAAA     2001:610:240:2::c100:ec
c.iana-servers.net.      12173 IN      A        139.91.1.10
c.iana-servers.net.      12173 IN      AAAA     2001:648:2c30::1:10
ns.icann.org.            12884 IN      A        192.0.34.126

;; Query time: 156 msec
;; SERVER: 10.34.255.7#53(10.34.255.7)
;; WHEN: Wed Aug 18 18:25:15 2010
;; MSG SIZE rcvd: 310

```

11.2.6. BIND 的高级性能

大多数 BIND 部署只使用 **named** 服务提供名称解析服务，或者作为具体域的授权。但 BIND 9 有大量高级功能，可提供更安全、有效的 **DNS** 服务。



重要

尝试使用高级功能前，比如 DNSSEC、TSIG 或者 IXFR（增量区域传送，Incremental Zone Transfer），请确定网络环境中的所有名称服务器都支持该功能，特别是当您使用旧的 BIND 版本或者非 BIND 服务器时。

[第 11.2.8.1 节 “已安装文档”](#) 中的《*BIND 9 管理员参考手册*》对所有提到的功能都有更详细的论述。

11.2.6.1. 多窗口

可根据请求的网络来源为客户端提供不同的信息，这个功能是自选的。它主要是用来拒绝来自本地网络以外的客户端的敏感 **DNS** 条目，同时允许本地网络内客户端的查询。

请在 `/etc/named.conf` 配置文件中添加 **view** 语句方可配置多视图。使用 **match-clients** 选项与 **DNS** 地址或者整个网络匹配，并为其提供特殊选项和区域数据。

11.2.6.2. 增量区域传送 (IXFR)

增量区域传送 (IXFR) 可让辅名称服务器只下载在主名称服务器中修改区域的更新部分。与标准传送过程相比，这可让通知和更新过程更有效。

备注：只有在使用动态更新修改主区域记录时方可使用 IXFR。如果手动编辑区域文件进行修改，则会使用自动区域传送 (AXFR)。

11.2.6.3. 事务处理签名 (Transaction SIGnatures, TSIG)

事务处理签名 (TSIG) 保证在允许传送前，主、辅名称服务器中都有共享密钥。这样就加强了根据标准 IP 地址方法进行的传送认证，因为攻击者不但需要访问要传送区域的 IP 地址，还需要知道密钥。

从版本 9 开始，BIND 还支持 TKEY，它是另一个认证区域传送的共享密钥方法。



重要

在使用不安全网络进行沟通时，请不要只依赖基于 IP 地址的认证方法。

11.2.6.4. DNS 安全扩展 (DNSSEC)

域名系统安全扩展 (DNSSEC) 提供 DNS 数据的原始认证，现有拒绝的认证以及数据完整性。当将某个具体域标记为安全时，会为每个验证失败的资源记录返回 **SERFVAIL** 响应。

注：可以如 [第 11.2.5 节“使用 dig 程序”](#) 所述，使用 **dig** 程序调试使用 DNSSEC 签名的域或者可识别 DNSSEC 的解析程序。有用的选项有 **+dnssec**（设定 DNSSEC OK 字节请求与 DNSSEC 关联的资源记录），**+cd**（让递归名称服务器不要验证响应），和 **+bufsize=512**（将数据包大小改为 512B 以便通过某些防火墙）。

11.2.6.5. 互联网协议版本 6 (IPv6)

使用 **AAAA** 资源记录和 **listen-on-v6** 指令可支持互联网协议版本 6 (IPv6)，如 [表 11.3 “常用配置选项”](#) 所述。

11.2.7. 常见的要避免的错误

以下是如何避免用户在配置名称服务器时通常会犯的错误的建议列表：

正确使用分号和括号

/etc/named.conf 文件中省略的分号或者不匹配的括号可造成 **named** 服务无法启动。

正确使用句号（即 . 符号）

在区域文件中，域名结尾处的句号代表完全限定域名。如果省略，**named** 服务会添加区域名称或者 **\$ORIGIN** 值完成它。

编辑区域文件时增加序列号

如果没有增加序列号，那么主名称服务器会有正确的新信息，但将永远无法将该更改通知辅名称服务器，也就不会尝试它们在那个 zone 中的数据。

要配置防火墙允许 NFS，请：

如果防火墙阻断 **named** 服务到其他名称服务器之间的连接，建议最好可随时修改防火墙设置。



警告

在 **DNS** 查询中使用固定 **UDP** 源端口是一个潜在的安全漏洞，这个漏洞可让攻击者更容易地执行缓存中毒攻击。为防止这个攻击，默认 **DNS** 发送随机短端口。将您的防火墙配置为允许来自随机 **UDP** 源端口的传出查询。默认的使用范围为 **1024** 到 **65535**。

11.2.8. 其他资料

以下资源提供有关 BIND 的附加信息。

11.2.8.1. 已安装文档

BIND 有涉及广泛的安装的文档，覆盖很多不同主题，每个文档都保存在它自己的主题目录中。下面的每一项中，都可使用在系统中安装的 *bind* 软件包的版本替换 *version*：

/usr/share/doc/bind-version/

主要目录中包含大多数最新的文档。该目录包含《*BIND 9 管理员参考手册*》，格式有 HTML 和 PDF，该文档详细介绍了 BIND 资源要求、如何配置名称服务器的不同类型；如何执行负载平衡；以及其他高级主题。

/usr/share/doc/bind-version/sample/etc/

该目录包含 **named** 配置文件示例。

rndc(8)

rndc 名称服务器控制程序的首页页中包含其用法的文档。

named(8)

互联网名称服务器 **named** 的手册页包含可用于控制 BIND 名称服务器守护进程的各类参数的文档。

lwresd(8)

轻加权解析程序守护进程 **lwresd** 的手册页中包含守护进程及其用法的文档。

named.conf(5)

该 man page 中有可在 **named** 配置文件中使用的完整列表。

rndc.conf(5)

该 man page 中有可在 **rndc** 配置文件中使用的完整列表。

11.2.8.2. 在线资源

<https://access.redhat.com/site/articles/770133>

有关在 **chroot** 环境中运行 BIND 的 Red Hat 知识库文章，其中包括与 Red Hat Enterprise Linux 6 的对比。

https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Security_Guide/

《Red Hat Enterprise Linux 7 安全指南》有关于 DNSSEC 完整论述。

<https://www.icann.org/namecollision>

ICANN 有关域名冲突的常见问题。

附录 A. 修订历史

修订 0.9-26.2	Tue Jul 5 2016	Leah Liu
完成翻译、校对		
修订 0.9-26.1	Tue Jul 5 2016	Leah Liu
与 XML 源 0.9-26 版本同步的翻译文件		
修订 0.9-26	Wed 11 Nov 2015	Jana Heves
7.2 GA 发行本		
修订 0.9-23	Mon 09 Nov 2015	Jana Heves
添加 RH 培训课程链接。		
修订 0.9-15	Tue 17 Feb 2015	Christian Huffman
7.1 GA 发行本		
修订 0.9-14	Fri Dec 05 2014	Christian Huffman
更新《网桥》、《绑定》及《成组》中的 nmtui 和 NetworkManager GUI 一节。		
修订 0.9-12	Wed Nov 05 2014	Stephen Wadeley
《IP 联网》、《802.1Q VLAN 标记》及《成组》改进。		
修订 0.9-11	Tues Oct 21 2014	Stephen Wadeley
《网桥》、《绑定》及《成组》改进。		
修订 0.9-9	Tue Sep 2 2014	Stephen Wadeley
《绑定》及《一致的网络设备命名》改进。		
修订 0.9-8	Tue July 8 2014	Stephen Wadeley
Red Hat Enterprise Linux 7.0 GA 发布《联网指南》。		
修订 0-0	Wed Dec 12 2012	Stephen Wadeley
起草《Red Hat Enterprise Linux 7 联网指南》。		

索引

符号

/etc/named.conf (见 BIND)

主名称服务器 (见 BIND)

内核模块

- bonding 模块, [使用频道绑定](#)
 - 描述, [使用频道绑定](#)
 - 绑定接口的参数, [Bonding 模块指令](#)
- 模块参数
 - bonding 模块参数, [Bonding 模块指令](#)

动态主机配置协议 (DHCP) (见 DHCP)

名称服务器 (见 DNS)

多主机 DHCP

- 主机配置, [系统配置](#)
- 服务器配置, [配置 DHCP 服务器](#)

授权名称服务器 (见 BIND)

绑定 (见 频道绑定)

资源记录 (见 BIND)

辅名称服务器 (见 BIND)

递归名称服务器 (见 BIND)

静态路由, [静态路由及默认网关](#)

频道绑定

- 描述, [使用频道绑定](#)
- 绑定接口参数, [Bonding 模块指令](#)
- 配置, [使用频道绑定](#)

频道绑定接口 (见 内核模块)

默认网关, [静态路由及默认网关](#)

B

Berkeley Internet Name Domain (见 BIND)

BIND

- 功能
 - DNS 安全扩展 (DNSSEC), [DNS 安全扩展 \(DNSSEC\)](#)
 - 事务处理签名 (TSIG), [事务处理签名 \(Transaction SIGNatures, TSIG\)](#)
 - 互联网协议版本 6 (IPv6), [互联网协议版本 6 \(IPv6\)](#)
 - 增量区域传送 (IXFR), [增量区域传送 \(IXFR\)](#)
 - 多窗口, [多窗口](#)
- 区域
 - \$INCLUDE 指令, [常用指令](#)
 - \$ORIGIN指令, [常用指令](#)
 - \$TTL 指令, [常用指令](#)
 - A (地址) 资源记录, [常用资源记录](#)
 - CNAME (正规名称) 资源记录, [常用资源记录](#)
 - MX (邮件互换) 资源记录, [常用资源记录](#)
 - NS (名称服务器) 资源记录, [常用资源记录](#)
 - PTR (指针) 资源记录, [常用资源记录](#)
 - SOA (权限启动) 资源记录, [常用资源记录](#)
 - 描述, [名称服务器区域](#)
 - 注释标签, [注释标签](#)
 - 用法示例, [简单区域文件](#), [逆向名称解析区域文件](#)
- 实用程序
 - dig, [BIND 作为名称服务器](#)
 - named, [BIND 作为名称服务器](#)
 - rndc, [BIND 作为名称服务器](#)
- 常见错误, [常见的要避免的错误](#)
- 文件
 - /etc/named.conf, [配置 DHCP 服务器](#), [配置该工具](#)
 - /etc/rndc.conf, [配置该工具](#)
 - /etc/rndc.key, [配置该工具](#)

- 目录
 - /etc/named/, [配置 DHCP 服务器](#)
 - /var/named/, [编辑区域文件](#)
 - /var/named/data/, [编辑区域文件](#)
 - /var/named/dynamic/, [编辑区域文件](#)
 - /var/named/slaves/, [编辑区域文件](#)
- 程序
 - dig, [使用 dig 程序, DNS 安全扩展 \(DNSSEC\)](#)
 - named, [配置 DHCP 服务器](#)
 - rndc, [使用 rndc 程序](#)
- 类型
 - 主 (master) 名称服务器, [名称服务器类型](#)
 - 主 (master) 名称服务器, [名称服务器区域](#)
 - 从属 (slave) 名称服务器, [名称服务器类型](#)
 - 从属 (slave) 名称服务器, [名称服务器区域](#)
 - 授权名称服务器, [名称服务器类型](#)
 - 递归名称服务器, [名称服务器类型](#)
- 资源记录, [名称服务器区域](#)
- 配置
 - acl语句, [常用语句类型](#)
 - controls 语句, [其他语句类型](#)
 - include语句, [常用语句类型](#)
 - key语句, [其他语句类型](#)
 - logging语句, [其他语句类型](#)
 - options 语句, [常用语句类型](#)
 - server语句, [其他语句类型](#)
 - trusted-keys语句, [其他语句类型](#)
 - view语句, [其他语句类型](#)
 - zone语句, [常用语句类型](#)
 - 注释标签, [注释标签](#)
- 附加资源, [在线资源](#)
 - 安装的文档, [已安装文档](#)

D

DHCP, [DHCP 服务器](#)

- dhcpd.conf, [配置文件](#)
- dhcpd.leases, [启动和停止服务器](#)
- dhcpd6.conf, [用于 IPv6 的 DHCP \(DHCPv6\)](#)
- DHCPv6, [用于 IPv6 的 DHCP \(DHCPv6\)](#)
- dhcrelay, [DHCP 中继代理程序](#)
- group, [配置文件](#)
- shared-network, [配置文件](#)
- 中继代理程序, [DHCP 中继代理程序](#)
- 使用原因, [为什么使用 DHCP](#)
- 停止服务器, [启动和停止服务器](#)
- 全局参数, [配置文件](#)
- 其他资料, [其他资料](#)
- 启动服务器, [启动和停止服务器](#)
- 命令行选项, [启动和停止服务器](#)
- 子网, [配置文件](#)
- 服务器配置, [配置 DHCP 服务器](#)
- 选项, [配置文件](#)

dhcpcd.conf, [配置文件](#)

dhcpcd.leases, [启动和停止服务器](#)

dhcrelay, [DHCP 中继代理程序](#)

dig (见 BIND)

DNS

- 定义, [DNS 服务器](#)
- (参见 BIND)

N

named (见 BIND)

NIC

- 绑定至单一频道, [使用频道绑定](#)

R

rndc (见 BIND)

root 名称服务器 (见 BIND)