Fedora Draft Documentation

Networking Guide

Configuration and Administration of networking for Fedora 20

Stephen Wadeley
Networking Guide

Fedora Draft Documentation Networking Guide
Configuration and Administration of networking for Fedora 20
Edition 20.0.1

Author
Stephen Wadeley
swadeley@redhat.com

Copyright © 2014 Red Hat, Inc. and others.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons
Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available
at http://creativecommons.org/licenses/by-sa/3.0/. The original authors of this document, and Red Hat,
designate the Fedora Project as the "Attribution Party" for purposes of CC-BY-SA. In accordance with
CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the
original version.

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, MetaMatrix, Fedora, the Infinity
Logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

For guidelines on the permitted uses of the Fedora trademarks, refer to https://fedoraproject.org/wiki/
Legal:Trademark_guidelines.

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.

All other trademarks are the property of their respective owners.

The Networking Guide documents relevant information regarding the configuration and administration
of network interfaces, networks and network services in Fedora 20. It is oriented towards system
administrators with a basic understanding of Linux and networking.

This book is based on the Deployment Guide from Red Hat Enterprise Linux 6. The chapters related to
networking were taken from the Deployment Guide to form the foundation for this book.
Preface

1. Target Audience ........................................................................................................ vii
2. About This Book ....................................................................................................... vii
3. What’s new in Fedora 20 .......................................................................................... vii
4. How to Read this Book ............................................................................................... vii
5. Document Conventions ............................................................................................ ix
   5.1. Typographic Conventions .................................................................................... ix
   5.2. Pull-quote Conventions ....................................................................................... x
   5.3. Notes and Warnings .............................................................................................. xi
6. Feedback ................................................................................................................... xi
7. Acknowledgments ....................................................................................................... xii

I. Networking

1. Introduction to Fedora Networking ......................................................................... 3
   1.1. How this Book is Structured ............................................................................... 3
   1.2. Introduction to NetworkManager ....................................................................... 3
   1.3. Installing NetworkManager ................................................................................ 3
      1.3.1. The NetworkManager Daemon ................................................................ 4
      1.3.2. Interacting with NetworkManager ........................................................... 4
   1.4. Network Configuration Using the Command Line Interface (CLI) ....................... 4
   1.5. Network Configuration Using NetworkManager's CLI (nmcli) ......................... 5
   1.6. Network Configuration Using sysconfig Files ................................................. 5
   1.7. NetworkManager and the Network Scripts ....................................................... 6
   1.8. Additional Resources ....................................................................................... 6
      1.8.1. Installed Documentation ............................................................................ 7
2. Configure Networking ............................................................................................... 9
   2.1. Static and Dynamic Interface Settings ................................................................ 9
      2.1.1. When to Use Static Network Interface Settings ........................................ 9
      2.1.2. When to Use Dynamic Interface Settings ................................................ 9
      2.1.3. Selecting Network Configuration Methods .............................................. 9
   2.2. Using NetworkManager with the GNOME Graphical User Interface ................. 9
      2.2.1. Connecting to a Network Using a GUI ..................................................... 10
      2.2.2. Configuring New and Editing Existing Connections .................................. 10
      2.2.3. Connecting to a Network Automatically .................................................... 11
      2.2.4. System-wide and Private Connection Profiles .......................................... 12
      2.2.5. Configuring a Wired (Ethernet) Connection ............................................. 13
      2.2.6. Configuring a Wi-Fi Connection ................................................................ 15
      2.2.7. Establishing a VPN Connection ................................................................. 17
      2.2.8. Establishing a Mobile Broadband Connection .......................................... 19
      2.2.9. Establishing a DSL Connection ................................................................ 22
      2.2.10. Configuring Connection Settings ............................................................ 24
   2.3. Using the Command Line Interface (CLI) .......................................................... 30
      2.3.1. Configuring a Network Interface Using ifcfg Files .................................. 30
      2.3.2. Static Routes and the Default Gateway ...................................................... 31
      2.3.3. Configuring Static Routes in ifcfg files .................................................... 32
      2.3.4. Configuring IPv6 Tokenized Interface Identifiers ...................................... 34
   2.4. Using the NetworkManager Command Line Tool, nmcli .................................. 35
      2.4.1. Understanding the nmcli Options ............................................................. 37
      2.4.2. Connecting to a Network Using nmcli ...................................................... 38
   2.5. Additional Resources ....................................................................................... 40
      2.5.1. Installed Documentation ........................................................................... 40
      2.5.2. Online Documentation ............................................................................ 40
3. Configure Host Names ............................................................... 41
  3.1. Understanding Host Names ................................................... 41
  3.2. Configuring Host Names Using hostnamectl .......................... 41
    3.2.1. View All the Host Names .............................................. 41
    3.2.2. Set All The Host Names .............................................. 41
    3.2.3. Set A Particular Host Name ........................................ 41
    3.2.4. Clear A Particular Host Name ....................................... 42
    3.2.5. Changing Host Names Remotely .................................... 42
  3.3. Additional Resources .......................................................... 42
    3.3.1. Installed Documentation ............................................. 42
    3.3.2. Online Documentation .............................................. 43

4. Configure Network Bonding ...................................................... 45
  4.1. Configure Network Bonding Using NetworkManager .................. 45
    4.1.1. Understanding the Default Behavior of Master and Slave Interfaces 45
    4.1.2. Establishing a Bond Connection .................................... 45
  4.2. Using the Command Line Interface (CLI) .............................. 48
    4.2.1. Check if Bonding Kernel Module is Installed .................... 48
    4.2.2. Create a Channel Bonding Interface ............................. 48
    4.2.3. Creating MASTER and SLAVE Interfaces .......................... 49
  4.3. Using the NetworkManager Command Line Tool, nmcli ................. 49
  4.4. Additional Resources .......................................................... 50
    4.4.1. Installed Documentation ............................................. 50
    4.4.2. Installable Documentation ........................................... 50
    4.4.3. Online Documentation .............................................. 51

5. Configure Network Teaming ...................................................... 53
  5.1. Understanding Network Teaming ........................................... 53
  5.2. Comparison of Network Teaming to Bonding .......................... 53
  5.3. Understanding the Network Teaming Daemon and the "Runners" ....... 54
  5.4. Check if the Network Teaming Daemon is Installed .................... 55
  5.5. Install the Network Teaming Daemon .................................... 55
  5.6. Converting a Bond to a Team ............................................. 55
  5.7. Selecting Interfaces to Use as Ports for a Network Team ............... 56
  5.8. Configure a Network Team ................................................... 56
    5.8.1. Creating a Network Team Using NetworkManager ................. 56
    5.8.2. Creating a Network Team Using teamd ................................ 58
    5.8.3. Creating a Network Team Using ifcfg Files ....................... 61
    5.8.4. Add a Port to a Network Team Using iputils .................... 62
    5.8.5. Listing the ports of a Team Using teamnl ........................ 62
    5.8.6. Configuring Options of a Team Using teamnl .................... 62
    5.8.7. Add an Address to a Network Team Using iputils ................ 62
    5.8.8. Bring up an Interface to a Network Team Using iputils .......... 62
    5.8.9. Viewing the Active Port Options of a Team Using teamnl ........ 62
    5.8.10. Setting the Active Port Options of a Team Using teamnl ........ 63
  5.9. Controlling teamd with teamdctl ......................................... 63
    5.9.1. Add a Port to a Network Team ..................................... 63
    5.9.2. Remove a Port From a Network Team .............................. 63
    5.9.3. Apply a Configuration to a Port in a Network Team .............. 63
    5.9.4. View the Configuration of a Port in a Network Team ............. 64
  5.10. Configure teamd Runners .................................................. 64
    5.10.1. Configure the broadcast Runner ................................... 64
    5.10.2. Configure the random Runner ...................................... 64
    5.10.3. Configure the roundrobin Runner ................................ 64
    5.10.4. Configure the activebackup Runner ............................... 65
Preface

The Networking Guide contains information on how to use the networking related features of Fedora 20.

This manual discusses many intermediate topics such as the following:

- Setting up a network (from establishing an Ethernet connection using NetworkManager to configuring channel bonding interfaces).
- Configuring DHCP, BIND, and DNS.

1. Target Audience

The Networking Guide assumes you have a basic understanding of the Fedora operating system. If you need help with the installation of this system, see the Fedora 20 Installation Guide.

2. About This Book

The Networking Guide is based on the networking material in the Deployment Guide. It also retains the information on DHCP and DNS servers from the Part II, “Servers” section. Most of the non-networking related material from the Deployment Guide guide can now be found in the Fedora 20 System Administrator's Guide except for reference material, such as that found in the appendices of the Deployment Guide. Reference material is now in a separate guide, the Fedora 20 System Administrator’s Reference Guide.

3. What's new in Fedora 20

Network Teaming has been introduced as an alternative to bonding for link aggregation. It is designed to be easy to maintain, debug and extend. For the user it offers performance and flexibility improvements and should be evaluated for all new installations.

A new command line tool, nmcli, has been introduced to allow users and scripts to interact with NetworkManager. A simple curses-based user interface for NetworkManager, nmtui, is also available.

A number of improvements have been made to NetworkManager to make it more suitable for use in server applications. In particular, NetworkManager no longer watches for configuration file changes by default, such as those made by editors or deployment tools. It allows administrators to make it aware of external changes through the nmcli connection reload command. Changes made through NetworkManager's D-Bus API or with nmcli are still effective immediately.

Not included in this guide, but of interest to network administrators, is the new Open Linux Management Infrastructure or OpenLMI project. This is an implementation of open industry standards for remote system management, which includes an agent for networking. See the Fedora 20 System Administrator's Guide for information on the OpenLMI Networking Provider.

4. How to Read this Book

This manual is divided into the following main categories:

---

Part I, “Networking”
This part describes how to configure the network on Fedora.

Chapter 1, Introduction to Fedora Networking explains the default networking service, NetworkManager, and the various graphical and command line tools that can be used to interact with NetworkManager. These include, an associated command line configuration tool, nmcli, and two graphical user interface tools, control-center and nm-connection-editor. Read this chapter to learn more about the many ways the NetworkManager daemon can be used.

Chapter 2, Configure Networking covers static and dynamic interface settings, selecting network configuration methods, using NetworkManager with graphical and command line user interfaces. Read this chapter to learn about configuring network connections.

Chapter 3, Configure Host Names covers static, pretty, and transient host names and their configuration using hostnamctl. Read this chapter to learn more about configuring host names on local and remote systems.

Chapter 4, Configure Network Bonding covers the configuring and use of bonded network connections. Read this chapter to learn about the configuring of network bonds using graphical and command line user interfaces.

Chapter 5, Configure Network Teaming covers the configuring and use of teamed network connections. Read this chapter to learn about the configuring of network teams using graphical and command line user interfaces.

Chapter 6, Configure Network Bridging covers the configuring and use of network bridges. Read this chapter to learn about the configuring of network bridges using graphical and command line user interfaces.

Chapter 7, Configure 802.1Q VLAN tagging covers the configuring and use of virtual private networks, VLANs, according to the 802.1Q standard. Read this chapter to learn about the configuring of VLANs using graphical and command line user interfaces.

Chapter 8, Consistent Network Device Naming covers consistent network device naming for network interfaces, a feature that changes the name of network interfaces on a system in order to make locating and differentiating the interfaces easier. Read this chapter to learn about this feature and how to enable or disable it.

Part II, “Servers”
This part discusses how to set up servers normally required for networking.

Chapter 9, DHCP Servers guides you through the installation of a Dynamic Host Configuration Protocol (DHCP) server and client. Read this chapter if you need to configure DHCP on your system.

Chapter 10, DNS Servers introduces you to Domain Name System (DNS), explains how to install, configure, run, and administer the BIND DNS server. Read this chapter if you need to configure a DNS server on your system.

For topics related to network configuration but not listed above, such as configuring GRUB to enable serial links and the use of virtual console terminals, see the Fedora 20 System Administrator’s Guide.

For topics related to servers but not listed above, such as configuring Network Time Protocol (NTP) and Precision Time Protocol (PTP), see the Fedora 20 System Administrator's Guide.
5. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the Liberation Fonts\(^2\) set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

5.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

**Mono-spaced Bold**

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keycaps and key combinations. For example:

To see the contents of the file `my_next_bestselling_novel` in your current working directory, enter the `cat my_next_bestselling_novel` command at the shell prompt and press **Enter** to execute the command.

The above includes a file name, a shell command and a keycap, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from keycaps by the hyphen connecting each part of a key combination. For example:

Press **Enter** to execute the command.

Press **Ctrl+Alt+F2** to switch to the first virtual terminal. Press **Ctrl+Alt+F1** to return to your X-Windows session.

The first paragraph highlights the particular keycap to press. The second highlights two key combinations (each a set of three keycaps with each set pressed simultaneously).

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **mono-spaced bold**. For example:

File-related classes include `filesystem` for file systems, `file` for files, and `dir` for directories. Each class has its own associated set of permissions.

**Proportional Bold**

This denotes words or phrases encountered on a system, including application names; dialog box text; labeled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

Choose **System → Preferences → Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).

\(^{2}\) [https://fedorahosted.org/liberation-fonts/](https://fedorahosted.org/liberation-fonts/)
To insert a special character into a gedit file, choose Applications → Accessories → Character Map from the main menu bar. Next, choose Search → Find… from the Character Map menu bar, type the name of the character in the Search field and click Next. The character you sought will be highlighted in the Character Table. Double-click this highlighted character to place it in the Text to copy field and then click the Copy button. Now switch back to your document and choose Edit → Paste from the gedit menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

**Mono-spaced Bold Italic or Proportional Bold Italic**

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

To connect to a remote machine using ssh, type `ssh username@domain.name` at a shell prompt. If the remote machine is `example.com` and your username on that machine is john, type `ssh john@example.com`.

The `mount -o remount file-system` command remounts the named file system. For example, to remount the `/home` file system, the command is `mount -o remount /home`.

To see the version of a currently installed package, use the `rpm -q package` command. It will return a result as follows: `package-version-release`.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

Publican is a **DocBook** publishing system.

### 5.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **mono-spaced roman** and presented thus:

```plaintext
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Source-code listings are also set in **mono-spaced roman** but add syntax highlighting as follows:

```java
package org.jboss.book.jca.ex1;
import javax.naming.InitialContext;
public class ExClient {
    public static void main(String args[])
        throws Exception
```
5.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

**Note**

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

**Important**

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled 'Important' will not cause data loss but may cause irritation and frustration.

**Warning**

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

6. Feedback

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in Bugzilla\(^3\) against the product Fedora Documentation.

When submitting a bug report, be sure to provide the following information:

- Manual's identifier: networking-guide
- Version number: 20

\(^3\) http://bugzilla.redhat.com/
If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

7. Acknowledgments


The authors of this guide would like to thank the following people for their valuable contributions: Adam Tkáč, Andrew Fitzsimon, Andrius Benokraitis, Brian Cleary Edward Bailey, Garrett LeSage, Jeffrey Fearn, Joe Orton, Joshua Wulf, Karsten Wade, Lucy Ringland, Marcela Mašíňová, Mark Johnson, Michael Behm, Miroslav Lichvár, Radek Vokál, Rahul Kavalapara, Rahul Sundaram, Sandra Moore, Zbyšek Mráz, Jan Včelák, Peter Hutterer and James Antill, among many others.
Part I. Networking

This part describes how to configure the network on Fedora.
Introduction to Fedora Networking

1.1. How this Book is Structured
All new material in this book has been written and arranged in such a way as to clearly separate introductory material, such as explanations of concepts and use cases, from configuration tasks. We hope that you can quickly find configuration instructions you need, while still providing some relevant explanations and conceptual material to help you understand and decide on the appropriate tasks relevant to your needs. Where material has been reused from the Deployment Guide, it has been reviewed and changed were possible to fit this idea of separating concepts from tasks.

The material is grouped according to the goal rather than the method. That is to say, instructions on how to achieve a specific task using different methods are grouped together. This is intended to make it easier for you to find the information on how to achieve a particular task or goal, and at the same time allow you to quickly see the different methods available. We will first present a graphical user interface (GUI) method, such as the use of nm-connection-editor or control-network to direct NetworkManager, then methods using the command line and configuration files. The command line can be used to issue commands and to compose or edit configuration files, therefore we will document the use of the ip commands and configuration files together. We will end each task by explaining the use of the nmcli tool where possible.

1.2. Introduction to NetworkManager
As of Fedora 20, the default networking service is provided by NetworkManager, which is a dynamic network control and configuration daemon that attempts to keep network devices and connections up and active when they are available.

NetworkManager has an associated command line configuration tool, nmcli. The GNOME graphical user interface application called control-center can be used to instruct NetworkManager to create, edit and remove connections and interfaces. The old nm-connection-editor tool is still provided for some tasks not yet handled by control-center.

NetworkManager can be used with the following types of connections: Ethernet, wireless, mobile broadband (such as cellular 3G), and DSL and PPPoE (Point-to-Point over Ethernet). In addition, NetworkManager allows for the configuration of network aliases, static routes, DNS information and VPN connections, as well as many connection-specific parameters. Finally, NetworkManager provides an API via D-Bus which allows applications to query and control network configuration and state.

1.3. Installing NetworkManager
NetworkManager is installed by default on Fedora. To ensure that it is, run the following command as the root user:

```
[~]$ yum install NetworkManager
```

---

Chapter 1. Introduction to Fedora Networking

1.3.1. The NetworkManager Daemon

The NetworkManager daemon runs with root privileges and is usually configured to start up at boot time. You can determine whether the NetworkManager daemon is running by entering this command:

```bash
~$ 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
```

The systemctl status command will report NetworkManager as **Active: inactive (dead)** if the NetworkManager service is not running. To start it for the current session run the following command as the root user:

```bash
~# systemctl start NetworkManager
```

Run the systemctl enable command to ensure that NetworkManager starts up every time the system boots:

```bash
~# systemctl enable NetworkManager
```

For more information on starting, stopping and managing services, see the Fedora 20 System Administrator's Guide.

1.3.2. Interacting with NetworkManager

Users do not interact with the NetworkManager system service directly. Instead, you perform network configuration tasks via graphical and command line user interface tools. The following tools are available in Fedora:

1. A graphical user interface tool called control-center, provided by GNOME, is available for desktop users. It incorporates a Network settings tool. It start it, press the Super key to enter the Activities Overview, type `control network` and then press Enter.

2. A command line tool, nmcli, is provided to allow users and scripts to interact with NetworkManager.

3. A graphical user interface tool, nm-connection-editor, is available for certain tasks not yet handled by control-center. It start it, press the Super key to enter the Activities Overview, type `network connections` or `nm-connection-editor` and then press Enter.

1.4. Network Configuration Using the Command Line Interface (CLI)

The commands for the ip utility, sometimes referred to as iproute2 after the package name, are documented in the man ip(8) page. If necessary, you can check that the ip utility is installed by checking its version number as follows:

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

As mentioned in the preface, we will include examples of using the command line and configuration files for each task after explaining the use of one of the graphical user interfaces to NetworkManager, namely, **control-center** and **nm-connection-editor**.
1.5. Network Configuration Using NetworkManager's CLI (nmcli)

The NetworkManager command line tool, nmcli, provides those who prefer to use the command line a way to configure networking by controlling NetworkManager. It is installed, along with NetworkManager, by default. If necessary, see Section 1.3.1, “The NetworkManager Daemon” for information on how to verify that NetworkManager is running.

As mentioned in the preface, we will include examples of using the nmcli tool for each task where possible, after explaining the use of graphical user interfaces and other command line methods. See Section 2.4, “Using the NetworkManager Command Line Tool, nmcli” for an introduction to nmcli.

1.6. Network Configuration Using sysconfig Files

The /etc/sysconfig/ directory is a location for configuration files and scripts. Most network configuration information is stored there, with the known exception of VPN, mobile broadband and PPPoE configuration, which are stored in /etc/NetworkManager/ subdirectories. Interface specific information for example, is stored in ifcfg files in the /etc/sysconfig/network-scripts/ directory.

The file /etc/sysconfig/network is for settings which are to have global effect. Information for VPNs, mobile broadband and PPPoE connections is stored in /etc/NetworkManager/system-connections/.

In Fedora, when you edit an ifcfg file, NetworkManager is not automatically aware of the change and has to be prompted to notice the change. If you use one of the tools to update NetworkManager profile settings, then NetworkManager does not implement those changes until you reconnect using that profile. For example, if configuration files have been changed using an editor, NetworkManager must be told to read the configuration files again. To do that, issue the following command as root:

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

The above command reads all connection profiles.

Changes made using tools such as nmcli, or graphical user interface tools, do not require a reload but do require the associated interface to be put down and then up again. That can be done by using commands in the following format:

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

Followed by:

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

These commands require root privileges.

NetworkManager does not trigger any of the network scripts, though the network scripts will try to trigger NetworkManager if it is running when ifup commands are used.

The ifup script is a generic script which does a few things and then calls interface-specific scripts like ifup-ethX, ifup-wireless, ifup-ppp, and so on. When a user runs ifup-eth0 manually, the following occurs:

- ifup looks for a file called /etc/sysconfig/network-scripts/ifcfg-eth0,
• if the `ifcfg` file exists, `ifup` looks for the `TYPE` key in that file to determine which type-specific script to call,

• `ifup` calls `ifup-wireless` or `ifup-eth` or `ifup-XXX` based on `TYPE`,

• the type-specific scripts do type-specific setup,

• and then the type-specific scripts let some common functions perform IP-related tasks like DHCP or static setup.

On bootup, `/etc/init.d/network` reads through all the `ifcfg` files and calls `ifup` for each of them if `ONBOOT=yes` is present.

```
1.7. NetworkManager and the Network Scripts
```

The term network scripts is a commonly used term for the script `/etc/init.d/network` and any other installed scripts it calls. The user supplied files are typically viewed as configuration, but can be also interpreted as an amendment to the scripts. With NetworkManager providing the default networking service, user shell scripts from previous releases can no longer be used.

Note that in Fedora, NetworkManager is started first, and `/etc/init.d/network` checks with NetworkManager to avoid tampering with NetworkManager’s connections. NetworkManager is intended to be the primary application using sysconfig configuration files and `/etc/init.d/network` is intended to be secondary, playing a fallback role.

The `/etc/init.d/network` script is not event-driven, it runs either:

• (a) manually (by one of the `systemctl` commands `start|stop|restart network`),

• (b) on boot and shutdown if the network service is enabled (as a result of the command `systemctl enable network`)

It is a manual process and does not react to events that happen after boot. Users can of course also call the scripts `ifup` and `ifdown` manually.

```
1.8. Additional Resources
```

The following sources of information provide additional resources regarding networking for Fedora.
1.8.1. Installed Documentation

- **NetworkManager(8)** man page — Describes the network management daemon.
- **man(1)** man page — Describes man pages and how to find them.
- */usr/share/doc/initscripts-version/sysconfig.txt* — Describes configuration files and their directives.
Configure Networking

2.1. Static and Dynamic Interface Settings

When to use static addressing and when to use dynamic addressing? These decisions are subjective, they depend on your accessed needs, your specific requirements. Having a policy, documenting it, and applying it consistently are usually more important than the specific decisions you make. In a traditional company LAN, this is an easier decision to make as you typically have fewer servers than other hosts. Provisioning and installation tools make providing static configurations to new hosts easy and using such tools will change your workflow and requirements. The following two sections are intended to provide guidance to those who have not already been through this decision making process. For more information on automated configuration and management, see the OpenLMI section in the System Administrators Guide. The System Installation Guide documents the use of kickstart which can also be used for automating the assignment of network settings.

2.1.1. When to Use Static Network Interface Settings

Use static IP addressing on those servers and devices whose network availability you want to ensure when automatic assignment methods, such as DHCP, fail. DHCP, DNS, and authentication servers are typical examples. Interfaces for out-of-band management devices are also worth configuring with static settings as these devices are supposed to work, as far as is possible, independently of other network infrastructure.

For hosts which are not considered vital, but for which static IP addressing is still considered desirable, use an automated provisioning method when possible. For example, DHCP servers can be configured to provide the same IP address to the same host every time. This method could be used for communal printers for example.

2.1.2. When to Use Dynamic Interface Settings

Enable and use dynamic assignment of IP addresses and other network information whenever there is no compelling reason not to. The time saved in planning and documenting manual settings can be better spent elsewhere. See Section 9.1, “Why Use DHCP?” for more information on this subject.

2.1.3. Selecting Network Configuration Methods

- To configure a network using graphical user interface tools, proceed to Section 2.2, “Using NetworkManager with the GNOME Graphical User Interface”

- To configure a network interface manually, see Section 2.3, “Using the Command Line Interface (CLI)”.

- To configure an interface using NetworkManager’s command line tool, nmcli, proceed to Section 2.4, “Using the NetworkManager Command Line Tool, nmcli”

2.2. Using NetworkManager with the GNOME Graphical User Interface

As of Fedora 20, NetworkManager does not have its own graphical user interface (GUI). The Network settings configuration tool is provided as part of the new GNOME control-center GUI. The old nm-connection-editor GUI is still available for certain tasks.
2.2.1. Connecting to a Network Using a GUI
Access the Network settings window of the control-center application as follow:

- Press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears. Proceed to Section 2.2.2, “Configuring New and Editing Existing Connections”

2.2.2. Configuring New and Editing Existing Connections
The Network settings window shows the connection status, its type and interface, its IP address and routing details, and so on.

![Network Settings Window](image)

Figure 2.1. Configure Networks Using the Network Settings Window

The Network settings window has a menu on the left hand side showing the available connection profiles. Below that is a plus and minus button for adding and deleting new connection profiles, and on the right a gear wheel icon will appear for editing any existing connections. To add a new connection, click the plus symbol to open the Add Network Connection window and proceed to the section called “Configuring a New Connection”.

Editing an Existing Connection
Clicking on the gear wheel icon next to existing connections in the Network settings window opens the Network details window, from where you can perform most network configuration tasks.
Configuring a New Connection

On the Network settings window, clicking on the plus sign below the menu opens the Add Network Connection window. This displays a list of connection types that can be added.

Then, to configure:

- VPN connections, click the VPN entry and proceed to Section 2.2.7, “Establishing a VPN Connection”;
- Bond connections, click the Bond entry and proceed to Section 4.1.2, “Establishing a Bond Connection”;
- Bridge connections, click the Bridge entry and proceed to Section 6.1.1, “Establishing a Bridge Connection”; or,
- VLAN connections, click the VLAN entry and proceed to Section 7.1.1, “Establishing a VLAN Connection”;
- Team connections, click the Team entry and proceed to Section 5.8.1, “Creating a Network Team Using NetworkManager”

2.2.3. Connecting to a Network Automatically

For any connection type you add or configure, you can choose whether you want NetworkManager to try to connect to that network automatically when it is available.

Procedure 2.1. Configuring NetworkManager to Connect to a Network Automatically When Detected

1. Press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears.

2. Select the type of connection from the left-hand-side menu.

3. Click on gear wheel icon in the lower right hand side corner. The Network details window appears.

4. Select the Identity menu entry on the left. The Network window changes to the identity view.
5. Select **Connect automatically** to cause **NetworkManager** to auto-connect to the connection whenever **NetworkManager** detects that it is available. Unselect the check box if you do not want **NetworkManager** to connect automatically. If the box is unselected, you will have to select that connection manually in the network applet menu to cause it to connect.

### 2.2.4. System-wide and Private Connection Profiles

**NetworkManager** stores all connection profiles. That is, connection profiles for system-wide use (system connections) as well as all user connection profiles. Access to the connection profiles is controlled by permissions which are stored by **NetworkManager**. See man `nm-settings(5)` for more information on the connection settings permissions property. The permissions correspond to the **USERS** directive in the **ifcfg** files. If the **USERS** directive is not present, it means the configuration settings will be used to create a profile available to all users. As an example, the following command in an **ifcfg** file will make the connection available only to the users listed:

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

This can also be set using graphical user interface tools. In `nm-connection-editor` there is the corresponding **All users may connect to this network** checkbox on the **General** tab and in the GNOME control-center Network settings Identity window there is the **Make available to other users** checkbox.

**NetworkManager**'s default policy enables users to create and modify user connections, but requires them to have root privileges to add, modify, or delete system connections. Profiles that should be available at boot time cannot be private because they will not be visible until the user logs in. For example, if user joe creates **joe-em2** connection with `ONBOOT=yes; USERS=joe`, it will not be available at boot time.

**Note**

Because creating a virtual private network (VPN) involves details considered confidential, it is recommended to configure your personal VPN connections as private by means of the **USERS** directive in the **ifcfg** files or by unselecting the checkboxes in the graphical user interface tools mentioned above. If you do so, then other non-root users on the system cannot access these connections, or view their settings, in any way.

**Procedure 2.2. Changing a Connection to be User-Specific instead of System-Wide, or Vice-Versa**

Depending on the system's policy, you may need root privileges on the system in order to change whether a connection is user-specific or system-wide.

1. Press the **Super** key to enter the Activities Overview, type **control network** and then press **Enter**. The **Network** settings tool appears.

2. Select the type of connection from the left-hand-side menu.

3. Click on gear wheel icon in the lower right hand side corner. The **Network** details window appears.

4. Select the **Identity** menu entry on the left. The **Network** window changes to the identity view.

5. Select the **Make available to other users** check box to cause **NetworkManager** to make the connection a system-wide connection. Depending on system policy, you may then be prompted
for the root password by the PolicyKit application. If so, enter the root password to finalize the change.

Conversely, unselect the Make available to other users check box to make the connection user-specific.

### 2.2.5. Configuring a Wired (Ethernet) Connection

To configure a wired network connection, press the Super key to enter the Activities Overview, type `control network` and then press Enter. The Network settings tool appears.

Select the Wired connection from the left-hand-side menu if it is not already highlighted.

The system creates and configures a single wired connection profile called Wired by default. A profile is a collection of settings that is applied to an interface. More than one profile can be created for an interface and applied as needed. The default profile cannot be deleted but its settings can be changed. Although you can edit the default Wired profile by clicking the gear wheel icon, creating a new wired connection profile for your custom settings is recommended. You can create a new wired connection profile by clicking the Add Profile button.

When you add a new connection by clicking the Add Profile button, NetworkManager creates a new configuration file for that connection and then opens the same dialog that is used for editing an existing connection. The difference between these dialogs is that an existing connection profile has a Details and Reset menu entry. In effect, you are always editing a connection profile; the difference only lies in whether that connection previously existed or was just created by NetworkManager when you clicked Add Profile.

#### Configuring the Connection Name, Auto-Connect Behavior, and Availability Settings

Five settings in the Editing dialog are common to all connection types, see the General tab:

- **Connection name** — Enter a descriptive name for your network connection. This name will be used to list this connection in the menu of the Network window.

- **Automatically connect to this network when it is available** — Select this box if you want NetworkManager to auto-connect to this connection when it is available. See Section 2.2.3, “Connecting to a Network Automatically” for more information.

- **All users may connect to this network** — Select this box to create a connection available to all users on the system. Changing this setting may require root privileges. See Section 2.2.4, “System-wide and Private Connection Profiles” for details. To prevent unexpected behavior during installation, ensure that this check box remains selected for any network interface that you configure.

- **Automatically connect to VPN when using this connection** — Select this box if you want NetworkManager to auto-connect to a VPN connection when it is available. Select the VPN from the dropdown menu.

- **Firewall Zone** — Select the Firewall Zone from the dropdown menu.

#### Configuring the Profile Identity

The final three configurable settings within the Identity dialog: the first below the name field is a text-entry field where you can specify a MAC (Media Access Control) address, and the second allows you to specify a cloned MAC address, and the third allows you to specify the MTU (Maximum Transmission Unit) value. Normally, you can leave the MAC address field blank and the MTU set to
automatic. These defaults will suffice unless you are associating a wired connection with a second or specific NIC, or performing advanced networking. In such cases, see the following descriptions:

**MAC Address**

Network hardware such as a Network Interface Card (NIC) has a unique MAC address (Media Access Control; also known as a hardware address) that identifies it to the system. Running the `ip addr` command will show the MAC address associated with each interface. For example, in the following `ip addr` output, the MAC address for the `eth0` interface (which is 52:54:00:26:9e:f1) immediately follows the `link/ether` keyword:

```
~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN qlen 1000
   link/ether 52:54:00:26:9e:f1 brd ff:ff:ff:ff:ff:ff
   inet 192.168.122.251/24 brd 192.168.122.255 scope global eth0
       valid_lft forever preferred_lft forever
```

A single system can have one or more NICs installed on it. The MAC address field therefore allows you to associate a specific NIC with a specific connection (or connections). As mentioned, you can determine the MAC address using the `ip addr` command, and then copy and paste that value into the MAC address text-entry field.

The cloned MAC address field is mostly for use in such situations were a network service has been restricted to a specific MAC address and you need to emulate that MAC address.

**MTU**

The MTU (Maximum Transmission Unit) value represents the size in bytes of the largest packet that the connection will use to transmit. This value defaults to 1500 when using IPv4, or a variable number 1280 or higher for IPv6, and does not generally need to be specified or changed.

**Saving Your New (or Modified) Connection and Making Further Configurations**

Once you have finished editing your wired connection, click the Apply button and NetworkManager will immediately save your customized configuration. Given a correct configuration, you can connect to your new or customized connection by selecting it from the network Notification Area applet. See Section 2.2.1, “Connecting to a Network Using a GUI” for information on using your new or altered connection.

You can further configure an existing connection by selecting it in the Network window and clicking gearwheel icon to return to the editing dialog.

Then, to configure:

- port-based Network Access Control (PNAC), click the 802.1x Security tab and proceed to Section 2.2.10.1, “Configuring 802.1x Security”;  
- IPv4 settings for the connection, click the IPv4 Settings tab and proceed to Section 2.2.10.4, “Configuring IPv4 Settings”; or,  
- IPv6 settings for the connection, click the IPv6 Settings tab and proceed to Section 2.2.10.5, “Configuring IPv6 Settings”.
2.2.6. Configuring a Wi-Fi Connection

This section explains how to use NetworkManager to configure a Wi-Fi (also known as wireless or 802.1a/b/g/n) connection to an Access Point.

To configure a mobile broadband (such as 3G) connection, see Section 2.2.8, "Establishing a Mobile Broadband Connection".

Quickly Connecting to an Available Access Point

The easiest way to connect to an available access point is to click on the network icon to activate the Notification Area applet, locate the Service Set Identifier (SSID) of the access point in the list of Wi-Fi networks, and click on it. A padlock symbol indicates the access point requires authentication. If the access point is secured, a dialog prompts you for an authentication key or password.

NetworkManager tries to auto-detect the type of security used by the access point. If there are multiple possibilities, NetworkManager guesses the security type and presents it in the Wi-Fi security dropdown menu. To see if there are multiple choices, click the Wi-Fi security dropdown menu and select the type of security the access point is using. If you are unsure, try connecting to each type in turn. Finally, enter the key or passphrase in the Password field. Certain password types, such as a 40-bit WEP or 128-bit WPA key, are invalid unless they are of a requisite length. The Connect button will remain inactive until you enter a key of the length required for the selected security type. To learn more about wireless security, see Section 2.2.10.2, "Configuring Wi-Fi Security".

If NetworkManager connects to the access point successfully, the Notification Area applet icon will change into a graphical indicator of the wireless connection's signal strength.

You can also edit the settings for one of these auto-created access point connections just as if you had added it yourself. The Wi-Fi page of the Network window has a History button. Clicking this reveals a list of all the connections you have ever tried to connect to. See the section called "Editing a Connection, or Creating a Completely New One".

Connecting to a Hidden Wi-Fi Network

All access points have a Service Set Identifier (SSID) to identify them. However, an access point may be configured not to broadcast its SSID, in which case it is hidden, and will not show up in NetworkManager's list of Available networks. You can still connect to a wireless access point that is hiding its SSID as long as you know its SSID, authentication method, and secrets.

To connect to a hidden wireless network, press the Super key to enter the Activities Overview, type control network and then press Enter. The Network window appears. Select Wi-Fi from the menu and then select Connect to Hidden Network to cause a dialog to appear. If you have connected to the hidden network before, use the Connection dropdown to select it, and click Connect. If you have not, leave the Connection dropdown as New, enter the SSID of the hidden network, select its Wi-Fi security method, enter the correct authentication secrets, and click Connect.

For more information on wireless security settings, see Section 2.2.10.2, "Configuring Wi-Fi Security".

Editing a Connection, or Creating a Completely New One

You can edit an existing connection that you have tried or succeeded in connecting to in the past by opening the Wi-Fi page of the Network dialog and selecting the gear wheel icon to the right of the Wi-Fi connection name. If the network is not currently in range, click History to display past connections. When you click the gearwheel icon the editing connection dialog appears. The Details window shows the connection details.

To configure a new connection whose SSID is in range, first attempt to connect to it by opening the Network window, selecting the Wi-Fi menu entry, and clicking the connection name (by default, the
same as the SSID). If the SSID is not in range, see the section called “Connecting to a Hidden Wi-Fi Network”. If the SSID is in range, the procedure is as follows:

1. Press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears.
2. Select the Wi-Fi menu entry.
3. Click the Wi-Fi network you want to connect to. A padlock symbol indicates a key or password is required.
4. If requested, enter the authentication details.

### Configuring the SSID, Auto-Connect Behavior, and Availability Settings

To edit a Wi-Fi connection’s settings, select Wi-Fi in the Network page and then select the gear wheel icon to the right of the Wi-Fi connection name. Select Identity. The following settings are available:

**SSID**

The Service Set Identifier (SSID) of the access point (AP).

**BSSID**

The Basic Service Set Identifier (BSSID) is the MAC address of the specific wireless access point you are connecting to when in Infrastructure mode. This field is blank by default, and you are able to connect to a wireless access point by SSID without having to specify its BSSID. If the BSSID is specified, it will force the system to associate to a specific access point only.

For ad-hoc networks, the BSSID is generated randomly by the mac80211 subsystem when the ad-hoc network is created. It is not displayed by NetworkManager.

**MAC address**

Like an Ethernet Network Interface Card (NIC), a wireless adapter has a unique MAC address (Media Access Control; also known as a hardware address) that identifies it to the system. Running the `ip addr` command will show the MAC address associated with each interface. For example, in the following `ip addr` output, the MAC address for the wlan0 interface (which is 00:1c:bf:02:f8:70) immediately follows the `link/ether` keyword:

```
~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       inet6 ::1/128 scope host
           valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN qlen 1000
   link/ether 52:54:00:26:9e:f1 brd ff:ff:ff:ff:ff:ff
   inet 192.168.122.251/24 brd 192.168.122.255 scope global eth0
   inet6 fe80::5054:ff:fe26:9ef1/64 scope link
       valid_lft forever preferred_lft forever
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
   link/ether 00:1c:bf:02:f8:70 brd ff:ff:ff:ff:ff:ff
   inet 10.200.130.67/24 brd 10.200.130.255 scope global wlan0
   inet6 fe80::21c:bfff:fe02:f870/64 scope link
       valid_lft forever preferred_lft forever
```

A single system could have one or more wireless network adapters connected to it. The MAC address field therefore allows you to associate a specific wireless adapter with a specific connection (or connections). As mentioned, you can determine the MAC address using the `ip addr` command, and then copy and paste that value into the MAC address text-entry field.
Cloned Address
A cloned MAC address to use in place of the real hardware address.

The following settings are common to all connection profiles:

- **Connect automatically** — Select this box if you want NetworkManager to auto-connect to this connection when it is available. See Section 2.2.3, “Connecting to a Network Automatically” for more information.

- **Make available to all users** — Select this box to create a connection available to all users on the system. Changing this setting may require root privileges. See Section 2.2.4, “System-wide and Private Connection Profiles” for details. To prevent unexpected behavior during installation, ensure that this check box remains selected for any network interface that you configure.

**Saving Your New (or Modified) Connection and Making Further Configurations**

Once you have finished editing the wireless connection, click the Apply button and NetworkManager will immediately save your customized configuration. Given a correct configuration, you can successfully connect to your the modified connection by selecting it from the Notification Area applet. See Section 2.2.1, “Connecting to a Network Using a GUI” for details on selecting and connecting to a network.

You can further configure an existing connection by selecting it in the Network window and clicking the gear wheel icon to reveal the connection details.

Then, to configure:

- security authentication for the wireless connection, click Security and proceed to Section 2.2.10.2, “Configuring Wi-Fi Security”;

- IPv4 settings for the connection, click IPv4 and proceed to Section 2.2.10.4, “Configuring IPv4 Settings”; or,

- IPv6 settings for the connection, click IPv6 and proceed to Section 2.2.10.5, “Configuring IPv6 Settings”.

**2.2.7. Establishing a VPN Connection**

Establishing a Virtual Private Network (VPN) enables communication between your Local Area Network (LAN), and another, remote LAN. This is done by setting up a tunnel across an intermediate network such as the Internet. The VPN tunnel that is set up typically uses authentication and encryption. After successfully establishing a VPN connection using a secure tunnel, a VPN router or gateway performs the following actions upon the packets you transmit:

1. it adds an Authentication Header for routing and authentication purposes;

2. it encrypts the packet data; and,

3. it encloses the data with an Encapsulating Security Payload (ESP), which constitutes the decryption and handling instructions.

The receiving VPN router strips the header information, decrypts the data, and routes it to its intended destination (either a workstation or other node on a network). Using a network-to-network connection, the receiving node on the local network receives the packets already decrypted and ready for processing. The encryption/decryption process in a network-to-network VPN connection is therefore transparent to clients.
Because they employ several layers of authentication and encryption, VPNs are a secure and effective means of connecting multiple remote nodes to act as a unified intranet.

**Procedure 2.3. Adding a New VPN Connection**

1. You can configure a new VPN connection by opening the **Network** window and selecting the plus symbol below the menu.

2. Press the **Super** key to enter the Activities Overview, type `control network` and then press **Enter**. The **Network** settings tool appears.

3. Select the plus symbol below the menu. The **Add Network Connection** window appears.

4. Select the **VPN** menu entry. The view now changes to offer configuring a VPN manually, or importing a VPN configuration file.

   The appropriate **NetworkManager** VPN plug-in for the VPN type you want to configure must be installed. (see *Fedora 20 System Administrator’s Guide* for more information on how to install new packages in Fedora 20).

5. Click the **Add** button to open the **Choose a VPN Connection Type** assistant.

6. Select the VPN protocol for the gateway you are connecting to from the menu. The VPN protocols available for selection in the menu correspond to the **NetworkManager** VPN plug-ins installed.

   For example, if the **NetworkManager-openswan-gnome** package is installed then the IPsec based VPN will be selectable from the menu.

7. The **Add Network Connection** window changes to present the settings customized for the type of VPN connection you selected in the previous step.

**Procedure 2.4. Editing an Existing VPN Connection**

You can configure an existing VPN connection by opening the **Network** window and selecting the name of the connection from the list. Then click the **Edit** button.

1. Press the **Super** key to enter the Activities Overview, type `control network` and then press **Enter**. The **Network** settings tool appears.

2. Select the **VPN** connection you wish to edit from the left hand menu.

3. Click the **Configure** button.

**Configuring the Connection Name, Auto-Connect Behavior, and Availability Settings**

Five settings in the **Editing** dialog are common to all connection types, see the **General** tab:

- **Connection name** — Enter a descriptive name for your network connection. This name will be used to list this connection in the menu of the **Network** window.

- **Automatically connect to this network when it is available** — Select this box if you want **NetworkManager** to auto-connect to this connection when it is available. See **Section 2.2.3, “Connecting to a Network Automatically”** for more information.

- **All users may connect to this network** — Select this box to create a connection available to all users on the system. Changing this setting may require root privileges. See **Section 2.2.4, “System-wide and Private Connection Profiles”** for details. To prevent unexpected behavior during installation, ensure that this check box remains selected for any network interface that you configure.
• **Automatically connect to VPN when using this connection** — Select this box if you want *NetworkManager* to auto-connect to a VPN connection when it is available. Select the VPN from the dropdown menu.

• **Firewall Zone** — Select the Firewall Zone from the dropdown menu.

**Configuring the VPN Tab**

**Gateway**

The name or IP address of the remote VPN gateway.

**Group name**

The name of a VPN group configured on the remote gateway.

**User password**

If required, enter the password used to authenticate with the VPN.

**Group password**

If required, enter the password used to authenticate with the VPN.

**User name**

If required, enter the user name used to authenticate with the VPN.

**Phase1 Algorithms**

If required, enter the algorithms to be used to authenticate and set up an encrypted channel.

**Phase2 Algorithms**

If required, enter the algorithms to be used for the IPSec negotiations.

**Domain**

If required, enter the Domain Name.

**Saving Your New (or Modified) Connection and Making Further Configurations**

Once you have finished editing your new VPN connection, click the **Save** button and *NetworkManager* will immediately save your customized configuration. Given a correct configuration, you can connect to your new or customized connection by selecting it from the Notification Area applet. See Section 2.2.1, “Connecting to a Network Using a GUI” for information on using your new or altered connection.

You can further configure an existing connection by selecting it in the **Network** window and clicking **Configure** to return to the **Editing** dialog.

Then, to configure:

• IPv4 settings for the connection, click the **IPv4 Settings** tab and proceed to Section 2.2.10.4, “Configuring IPv4 Settings”.

**2.2.8. Establishing a Mobile Broadband Connection**

You can use *NetworkManager*’s mobile broadband connection abilities to connect to the following 2G and 3G services:

• **2G** — GPRS (General Packet Radio Service) or EDGE (Enhanced Data Rates for GSM Evolution)

• **3G** — UMTS (Universal Mobile Telecommunications System) or HSPA (High Speed Packet Access)
Your computer must have a mobile broadband device (modem), which the system has discovered and recognized, in order to create the connection. Such a device may be built into your computer (as is the case on many notebooks and netbooks), or may be provided separately as internal or external hardware. Examples include PC card, USB Modem or Dongle, mobile or cellular telephone capable of acting as a modem.

Procedure 2.5. Adding a New Mobile Broadband Connection
You can configure a mobile broadband connection by opening the **Network Connections** tool and selecting the **Mobile Broadband** tab.

1. Press the **Super** key to enter the Activities Overview, type `nm-connection-editor` and then press **Enter**. The **Network Connections** tool appears.

2. Click the **Add** button. The **Choose a Connection Type** menu opens.

3. Select the **Mobile Broadband** menu entry.

4. Click **Create** to open the **Set up a Mobile Broadband Connection** assistant.

5. Under **Create a connection for this mobile broadband device**, choose the 2G- or 3G-capable device you want to use with the connection. If the dropdown menu is inactive, this indicates that the system was unable to detect a device capable of mobile broadband. In this case, click **Cancel**, ensure that you do have a mobile broadband-capable device attached and recognized by the computer and then retry this procedure. Click the **Continue** button.

6. Select the country where your service provider is located from the list and click the **Continue** button.

7. Select your provider from the list or enter it manually. Click the **Continue** button.

8. Select your payment plan from the dropdown menu and confirm the **Access Point Name (APN)** is correct. Click the **Continue** button.

9. Review and confirm the settings and then click the **Apply** button.

10. Edit the mobile broadband-specific settings by referring to the section called “Configuring the **Mobile Broadband Tab**”.

Procedure 2.6. Editing an Existing Mobile Broadband Connection
Follow these steps to edit an existing mobile broadband connection.

1. Press the **Super** key to enter the Activities Overview, type `nm-connection-editor` and then press **Enter**. The **Network Connections** tool appears.

2. Select the **Mobile Broadband** tab.

3. Select the connection you wish to edit and click the **Edit** button.

4. Configure the connection name, auto-connect behavior, and availability settings.

   Five settings in the **Editing** dialog are common to all connection types, see the **General** tab:

   • **Connection name** — Enter a descriptive name for your network connection. This name will be used to list this connection in the menu of the **Network** window.

   • **Automatically connect to this network when it is available** — Select this box if you want **NetworkManager** to auto-connect to this connection when it is available. See **Section 2.2.3, “Connecting to a Network Automatically”** for more information.
• **All users may connect to this network** — Select this box to create a connection available to all users on the system. Changing this setting may require root privileges. See Section 2.2.4, “System-wide and Private Connection Profiles” for details. To prevent unexpected behavior during installation, ensure that this check box remains selected for any network interface that you configure.

• **Automatically connect to VPN when using this connection** — Select this box if you want NetworkManager to auto-connect to a VPN connection when it is available. Select the VPN from the dropdown menu.

• **Firewall Zone** — Select the Firewall Zone from the dropdown menu.

5. Edit the mobile broadband-specific settings by referring to the section called “Configuring the Mobile Broadband Tab”.

---

**Saving Your New (or Modified) Connection and Making Further Configurations**

Once you have finished editing your mobile broadband connection, click the Apply button and NetworkManager will immediately save your customized configuration. Given a correct configuration, you can connect to your new or customized connection by selecting it from the Notification Area applet. See Section 2.2.1, “Connecting to a Network Using a GUI” for information on using your new or altered connection.

You can further configure an existing connection by selecting it in the Network Connections window and clicking Edit to return to the Editing dialog.

Then, to configure:

• Point-to-point settings for the connection, click the PPP Settings tab and proceed to Section 2.2.10.3, “Configuring PPP (Point-to-Point) Settings”;

• IPv4 settings for the connection, click the IPv4 Settings tab and proceed to Section 2.2.10.4, “Configuring IPv4 Settings”; or,

• IPv6 settings for the connection, click the IPv6 Settings tab and proceed to Section 2.2.10.5, “Configuring IPv6 Settings”.

---

**Configuring the Mobile Broadband Tab**

If you have already added a new mobile broadband connection using the assistant (see Procedure 2.5, “Adding a New Mobile Broadband Connection” for instructions), you can edit the Mobile Broadband tab to disable roaming if home network is not available, assign a network ID, or instruct NetworkManager to prefer a certain technology (such as 3G or 2G) when using the connection.

**Number**

The number that is dialed to establish a PPP connection with the GSM-based mobile broadband network. This field may be automatically populated during the initial installation of the broadband device. You can usually leave this field blank and enter the APN instead.

**Username**

Enter the user name used to authenticate with the network. Some providers do not provide a user name, or accept any user name when connecting to the network.
Password
Enter the password used to authenticate with the network. Some providers do not provide a password, or accept any password.

APN
Enter the Access Point Name (APN) used to establish a connection with the GSM-based network. Entering the correct APN for a connection is important because it often determines:

• how the user is billed for their network usage; and/or
• whether the user has access to the Internet, an intranet, or a subnetwork.

Network ID
Entering a Network ID causes NetworkManager to force the device to register only to a specific network. This can be used to ensure the connection does not roam when it is not possible to control roaming directly.

Type
Any — The default value of Any leaves the modem to select the fastest network.

3G (UMTS/HSPA) — Force the connection to use only 3G network technologies.

2G (GPRS/EDGE) — Force the connection to use only 2G network technologies.

Prefer 3G (UMTS/HSPA) — First attempt to connect using a 3G technology such as HSPA or UMTS, and fall back to GPRS or EDGE only upon failure.

Prefer 2G (GPRS/EDGE) — First attempt to connect using a 2G technology such as GPRS or EDGE, and fall back to HSPA or UMTS only upon failure.

Allow roaming if home network is not available
Uncheck this box if you want NetworkManager to terminate the connection rather than transition from the home network to a roaming one, thereby avoiding possible roaming charges. If the box is checked, NetworkManager will attempt to maintain a good connection by transitioning from the home network to a roaming one, and vice versa.

PIN
If your device's SIM (Subscriber Identity Module) is locked with a PIN (Personal Identification Number), enter the PIN so that NetworkManager can unlock the device. NetworkManager must unlock the SIM if a PIN is required in order to use the device for any purpose.

2.2.9. Establishing a DSL Connection
This section is intended for those installations which have a DSL card fitted within a host rather than the external combined DSL modem router combinations typical of private consumer or SOHO installations.

Procedure 2.7. Adding a New DSL Connection
You can configure a new DSL connection by opening the Network Connections window, clicking the Add button and selecting DSL from the Hardware section of the new connection list.

1. Press the Super key to enter the Activities Overview, type nm-connection-editor and then press Enter. The Network Connections tool appears.
2. Click the Add button.
3. The Choose a Connection Type list appears.
4. Select DSL and press the Create button.

5. The Editing DSL Connection window appears.

Procedure 2.8. Editing an Existing DSL Connection
You can configure an existing DSL connection by opening the Network Connections window and selecting the name of the connection from the list. Then click the Edit button.

1. Press the Super key to enter the Activities Overview, type nm-connection-editor and then press Enter. The Network Connections tool appears.

2. Select the connection you wish to edit and click the Edit button.

Configuring the Connection Name, Auto-Connect Behavior, and Availability Settings
Five settings in the Editing dialog are common to all connection types, see the General tab:

- **Connection name** — Enter a descriptive name for your network connection. This name will be used to list this connection in the menu of the Network window.

- **Automatically connect to this network when it is available** — Select this box if you want NetworkManager to auto-connect to this connection when it is available. See Section 2.2.3, “Connecting to a Network Automatically” for more information.

- **All users may connect to this network** — Select this box to create a connection available to all users on the system. Changing this setting may require root privileges. See Section 2.2.4, “System-wide and Private Connection Profiles” for details. To prevent unexpected behavior during installation, ensure that this check box remains selected for any network interface that you configure.

- **Automatically connect to VPN when using this connection** — Select this box if you want NetworkManager to auto-connect to a VPN connection when it is available. Select the VPN from the dropdown menu.

- **Firewall Zone** — Select the Firewall Zone from the dropdown menu.

Configuring the DSL Tab

**Username**
Enter the user name used to authenticate with the service provider.

**Service**
Leave blank unless otherwise directed.

**Password**
Enter the password supplied by the service provider.

Saving Your New (or Modified) Connection and Making Further Configurations
Once you have finished editing your DSL connection, click the Apply button and NetworkManager will immediately save your customized configuration. Given a correct configuration, you can connect to your new or customized connection by selecting it from the Notification Area applet. See Section 2.2.1, “Connecting to a Network Using a GUI” for information on using your new or altered connection.
You can further configure an existing connection by selecting it in the Network Connections window and clicking Edit to return to the Editing dialog.

Then, to configure:

- The MAC address and MTU settings, click the Wired tab and proceed to the section called “Configuring the Profile Identity”;
- Point-to-point settings for the connection, click the PPP Settings tab and proceed to Section 2.2.10.3, “Configuring PPP (Point-to-Point) Settings”;
- IPv4 settings for the connection, click the IPv4 Settings tab and proceed to Section 2.2.10.4, “Configuring IPv4 Settings”.

2.2.10. Configuring Connection Settings

2.2.10.1. Configuring 802.1x Security

802.1x security is the name of the IEEE standard for port-based Network Access Control (PNAC). Simply put, 802.1x security is a way of controlling access to a logical network from a physical one. All clients who want to join the logical network must authenticate with the server (a router, for example) using the correct 802.1x authentication method.

802.1x security is most often associated with securing wireless networks (WLANs), but can also be used to prevent intruders with physical access to the network (LAN) from gaining entry. In the past, DHCP servers were configured not to lease IP addresses to unauthorized users, but for various reasons this practice is both impractical and insecure, and thus is no longer recommended. Instead, 802.1x security is used to ensure a logically-secure network through port-based authentication.

802.1x provides a framework for WLAN and LAN access control and serves as an envelope for carrying one of the Extensible Authentication Protocol (EAP) types. An EAP type is a protocol that defines how WLAN security is achieved on the network.

You can configure 802.1x security for a wired or wireless connection type by opening the Network window (see Section 2.2.1, “Connecting to a Network Using a GUI”) and following the applicable procedure below. Press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears. Proceed to Procedure 2.9, “For a Wired Connection” or Procedure 2.10, “For a Wireless Connection”:

Procedure 2.9. For a Wired Connection

1. Select a Wired connection in the left pane of the Network window.

2. Either click on Add Profile to add a new network connection for which you want to configure 802.1x security, or select an existing connection and click the gear wheel icon.

3. Then select Security and set the symbolic power button to ON to enable settings configuration.

4. Proceed to Section 2.2.10.1.1, “Configuring TLS (Transport Layer Security) Settings”

Procedure 2.10. For a Wireless Connection

1. Select a Wireless connection in the left pane of the Network window and, if necessary, set the symbolic power button to ON.

2. Either select the connection name of a new connection, or click the gear wheel icon of an existing connection, for which you want to configure 802.1x security. In the case of a new connection complete any authentication steps to complete the connection and then click the gearwheel icon.
3. Then select **Security**.

4. From the dropdown menu select one of the following security methods: **LEAP**, **Dynamic WEP (802.1x)**, or **WPA & WPA2 Enterprise**.

5. Refer to **Section 2.2.10.1.1, “Configuring TLS (Transport Layer Security) Settings”** for descriptions of which extensible authentication protocol (EAP) types correspond to your selection in the **Security** dropdown menu.

### 2.2.10.1.1. Configuring TLS (Transport Layer Security) Settings

With Transport Layer Security, the client and server mutually authenticate using the TLS protocol. The server demonstrates that it holds a digital certificate, the client proves its own identity using its client-side certificate, and key information is exchanged. Once authentication is complete, the TLS tunnel is no longer used. Instead, the client and server use the exchanged keys to encrypt data using AES, TKIP or WEP.

The fact that certificates must be distributed to all clients who want to authenticate means that the EAP-TLS authentication method is very strong, but also more complicated to set up. Using TLS security requires the overhead of a public key infrastructure (PKI) to manage certificates. The benefit of using TLS security is that a compromised password does not allow access to the (W)LAN: an intruder must also have access to the authenticating client’s private key.

**NetworkManager** does not determine the version of TLS supported. **NetworkManager** gathers the parameters entered by the user and passes them to the daemon, **wpa_supplicant**, that handles the procedure. It in turn uses OpenSSL to establish the TLS tunnel. OpenSSL itself negotiates the SSL/TLS protocol version. It uses the highest version both ends support.

**Selecting an Authentication Method**

Select from one of following authentication methods:

- Select **TLS** for **Transport Layer Security** and proceed to **Section 2.2.10.1.2, “Configuring TLS Settings”**;

- Select **FAST** for **Flexible Authentication via Secure Tunneling** and proceed to **Section 2.2.10.1.4, “Configuring Tunneled TLS Settings”**;

- Select **Tunneled TLS** for **Tunneled Transport Layer Security**, otherwise known as TTLS, or EAP-TTLS and proceed to **Section 2.2.10.1.4, “Configuring Tunneled TLS Settings”**;

- Select **Protected EAP (PEAP)** for **Protected Extensible Authentication Protocol** and proceed to **Section 2.2.10.1.5, “Configuring Protected EAP (PEAP) Settings”**.

### 2.2.10.1.2. Configuring TLS Settings

**Identity**

Provide the identity of this server.

**User certificate**

Click to browse for, and select, a personal X.509 certificate file encoded with **Distinguished Encoding Rules** (DER) or **Privacy Enhanced Mail** (PEM).

**CA certificate**

Click to browse for, and select, an X.509 certificate authority certificate file encoded with **Distinguished Encoding Rules** (DER) or **Privacy Enhanced Mail** (PEM).
Private key
Click to browse for, and select, a private key file encoded with Distinguished Encoding Rules (DER), Privacy Enhanced Mail (PEM), or the Personal Information Exchange Syntax Standard (PKCS #12).

Private key password
Enter the password for the private key in the Private key field. Select Show password to make the password visible as you type it.

2.2.10.1.3. Configuring FAST Settings

Anonymous Identity
Provide the identity of this server.

PAC provisioning
Select the checkbox to enable and then select from Anonymous, Authenticated, and Both.

PAC file
Click to browse for, and select, PAC file.

Inner authentication
GTC — Generic Token Card.

Username
Enter the user name to be used in the authentication process.

Password
Enter the password to be used in the authentication process.

2.2.10.1.4. Configuring Tunneled TLS Settings

Anonymous identity
This value is used as the unencrypted identity.

CA certificate
Click to browse for, and select, a Certificate Authority's certificate.

Inner authentication
PAP — Password Authentication Protocol.

Username
Enter the user name to be used in the authentication process.

Password
Enter the password to be used in the authentication process.
2.2.10.1.5. Configuring Protected EAP (PEAP) Settings

Anonymous Identity
This value is used as the unencrypted identity.

CA certificate
Click to browse for, and select, a Certificate Authority's certificate.

PEAP version
The version of Protected EAP to use. Automatic, 0 or 1.

Inner authentication
- MD5 — Message Digest 5, a cryptographic hash function.
- GTC — Generic Token Card.

Username
Enter the user name to be used in the authentication process.

Password
Enter the password to be used in the authentication process.

2.2.10.2. Configuring Wi-Fi Security

Security
- None — Do not encrypt the Wi-Fi connection.
- WEP 40/128-bit Key — Wired Equivalent Privacy (WEP), from the IEEE 802.11 standard. Uses a single pre-shared key (PSK).
- WEP 128-bit Passphrase — An MD5 hash of the passphrase will be used to derive a WEP key.
- Dynamic WEP (802.1x) — WEP keys are changed dynamically. Use with Section 2.2.10.1.1, “Configuring TLS (Transport Layer Security) Settings”
- WPA & WPA2 Personal — Wi-Fi Protected Access (WPA), from the draft IEEE 802.11i standard. A replacement for WEP. Wi-Fi Protected Access II (WPA2), from the 802.11i-2004 standard. Personal mode uses a pre-shared key (WPA-PSK).
- WPA & WPA2 Enterprise — WPA for use with a RADIUS authentication server to provide IEEE 802.1x network access control. Use with Section 2.2.10.1.1, “Configuring TLS (Transport Layer Security) Settings”

Password
Enter the password to be used in the authentication process.

2.2.10.3. Configuring PPP (Point-to-Point) Settings

Configure Methods
Use point-to-point encryption (MPPE)  
Microsoft Point-To-Point Encryption protocol (RFC 3078\(^1\)).

Allow BSD data compression  
PPP BSD Compression Protocol (RFC 1977\(^2\)).

Allow Deflate data compression  
PPP Deflate Protocol (RFC 1979\(^3\)).

Use TCP header compression  
Compressing TCP/IP Headers for Low-Speed Serial Links (RFC 1144\(^4\)).

Send PPP echo packets  
LCP Echo-Request and Echo-Reply Codes for loopback tests (RFC 1661\(^5\)).

2.2.10.4. Configuring IPv4 Settings

The IPv4 Settings tab allows you to configure the method by which you connect to the Internet and enter IP address, route, and DNS information as required. The IPv4 Settings tab is available when you create and modify one of the following connection types: wired, wireless, mobile broadband, VPN or DSL. If you need to configure IPv6 addresses, see Section 2.2.10.5, “Configuring IPv6 Settings”. If you need to configure static routes, which can be done by clicking on the Routes button, see Section 2.2.10.6, “Configuring Routes”.

If you are using DHCP to obtain a dynamic IP address from a DHCP server, you can simply set Method to Automatic (DHCP).

Setting the Method

Available IPv4 Methods by Connection Type

When you click the Method dropdown menu, depending on the type of connection you are configuring, you are able to select one of the following IPv4 connection methods. All of the methods are listed here according to which connection type, or types, they are associated with:

Method

Automatic (DHCP) — Choose this option if the network you are connecting to uses a DHCP server to assign IP addresses. You do not need to fill in the DHCP client ID field.

Automatic (DHCP) addresses only — Choose this option if the network you are connecting to uses a DHCP server to assign IP addresses but you want to assign DNS servers manually.

Link-Local Only — Choose this option if the network you are connecting to does not have a DHCP server and you do not want to assign IP addresses manually. Random addresses will be assigned as per RFC 3927\(^6\) with prefix 169.254/16.

Shared to other computers — Choose this option if the interface you are configuring is for sharing an Internet or WAN connection. The interface is assigned an address in the

---

\(^1\) [http://www.rfc-editor.org/info/rfc3078](http://www.rfc-editor.org/info/rfc3078)  
\(^3\) [http://www.rfc-editor.org/info/rfc1979](http://www.rfc-editor.org/info/rfc1979)  
\(^4\) [http://www.rfc-editor.org/info/rfc1144](http://www.rfc-editor.org/info/rfc1144)  
\(^5\) [http://www.rfc-editor.org/info/rfc1661](http://www.rfc-editor.org/info/rfc1661)  
\(^6\) [http://www.rfc-editor.org/info/rfc3927](http://www.rfc-editor.org/info/rfc3927)
10.42.x.1/24 range, a DHCP server and DNS server are started, and the interface is connected to the default network connection on the system with network address translation (NAT).

**Disabled** — IPv4 is disabled for this connection.

**Wired, Wireless and DSL Connection Methods**

**Manual** — Choose this option if the network you are connecting to does not have a DHCP server and you want to assign IP addresses manually.

**Mobile Broadband Connection Methods**

**Automatic (PPP)** — Choose this option if the network you are connecting to uses a DHCP server to assign IP addresses.

**Automatic (PPP) addresses only** — Choose this option if the network you are connecting to uses a DHCP server to assign IP addresses but you want to assign DNS servers manually.

**VPN Connection Methods**

**Automatic (VPN)** — Choose this option if the network you are connecting to uses a DHCP server to assign IP addresses.

**Automatic (VPN) addresses only** — Choose this option if the network you are connecting to uses a DHCP server to assign IP addresses but you want to assign DNS servers manually.

**DSL Connection Methods**

**Automatic (PPPoE)** — Choose this option if the network you are connecting to uses a DHCP server to assign IP addresses.

**Automatic (PPPoE) addresses only** — Choose this option if the network you are connecting to uses a DHCP server to assign IP addresses but you want to assign DNS servers manually.

For information on configuring static routes for the network connection, go to Section 2.2.10.6, “Configuring Routes”.

### 2.2.10.5. Configuring IPv6 Settings

**Method**

**Ignore** — Choose this option if you want to ignore IPv6 settings for this connection.

**Automatic** — Choose this option to use router advertisement (RA) to create an automatic, stateless configuration.

**Automatic, addresses only** — Choose this option if the network you are connecting to uses a DHCP server to assign IP addresses but you want to assign DNS servers manually.

**Automatic, DHCP only** — Choose this option to not use RA, but request information from DHCPv6 directly to create a stateful configuration.

**Manual** — Choose this option if the network you are connecting to does not have a DHCP server and you want to assign IP addresses manually.

**Link-Local Only** — Choose this option if the network you are connecting to does not have a DHCP server and you do not want to assign IP addresses manually. Random addresses will be assigned as per RFC 4862\(^7\) with prefix FE80::0.

\(^7\) [http://www.rfc-editor.org/info/rfc4862](http://www.rfc-editor.org/info/rfc4862)
Addresses

- DNS servers — Enter a comma separated list of DNS servers.
- Search domains — Enter a comma separated list of domain controllers.

For information on configuring static routes for the network connection, go to Section 2.2.10.6, “Configuring Routes”.

2.2.10.6. Configuring Routes

A host's routing table will be automatically populated with routes to directly connected networks. The routes are learned by observing the network interfaces when they are "up". This section describes entering static routes to networks or hosts which can be reached by traversing an intermediate network or connection, such as a VPN tunnel or leased line. In order to reach a remote network or host, the system is given the address of a gateway to which traffic should be sent.

When a host's interface is configured by DHCP, an address of a gateway that leads to an upstream network or the Internet is usually assigned. This gateway is usually referred to as the default gateway as it is the gateway to use if no better route is known to the system (and present in the routing table). Network administrators often use the first host IP address in the network as the gateway address; for example, 192.168.10.1. Not to be confused by the address which represents the network itself; in this example, 192.168.10.0.

Configuring Static Routes

To set a static route, open the IPv4 or IPv6 settings window for the connection you want to configure. See Section 2.2.1, “Connecting to a Network Using a GUI” for instructions on how to do that.

Routes

- Address — Enter the IP address of a remote network, sub-net, or host.
- Netmask — The netmask or prefix length of the IP address entered above.
- Gateway — The IP address of the gateway leading to the remote network, sub-net, or host entered above.
- Metric — A network cost, that is to say a preference value to give to this route. Lower values will be preferred over higher values.

Automatic

Ensure this is ON to use manually entered routes for this connection.

Use this connection only for resources on its network

Select this check box to prevent the connection from becoming the default route. Typical examples are where a connection is a VPN tunnel or a leased line to a head office and you do not want any Internet bound traffic to pass over the connection. Selecting this option means that only traffic specifically destined for routes learned automatically over the connection or entered here manually will be routed over the connection.

2.3. Using the Command Line Interface (CLI)

2.3.1. Configuring a Network Interface Using ifcfg Files

Interface configuration files control the software interfaces for individual network devices. As the system boots, it uses these files to determine what interfaces to bring up and how to configure them.
These files are usually named `ifcfg-name`, where the suffix `name` refers to the name of the device that the configuration file controls. By convention, the `ifcfg` file's suffix, `ethX`, is the same as the string given by the `DEVICE` directive in the configuration file itself.

### Static Network Settings

To configure an interface with static network settings using `ifcfg` files, for an interface with name `eth0`, create a file with name `ifcfg-eth0` in the `/etc/sysconfig/network-scripts/` directory as follows:

```bash
DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
NETMASK=255.255.255.0
IPADDR=10.0.1.27
USERCTL=no
NM_CONTROLLED=no
```

Optionally specify the hardware or MAC address using the `HWADDR` directive. Note that this will influence the device naming procedure as explained in Chapter 8, Consistent Network Device Naming. You do not need to specify the broadcast address as this is calculated automatically by `ipcalc`.

### Dynamic Network Settings

To configure an interface with dynamic network settings using `ifcfg` files, for an interface with name `em1`, create a file with name `ifcfg-em1` in the `/etc/sysconfig/network-scripts/` directory as follows:

```bash
DEVICE=em1
BOOTPROTO= dhcp
ONBOOT=yes
NM_CONTROLLED=no
```

Optionally specify the hardware or MAC address using the `HWADDR` directive. Note that this will influence the device naming procedure as explained in Chapter 8, Consistent Network Device Naming. You do not need to specify the broadcast address as this is calculated automatically by `ipcalc`.

For a listing of the configurable parameters in an Ethernet interface configuration file see the Fedora 20 System Administrator's Reference Guide.

### 2.3.2. Static Routes and the Default Gateway

Static routes are for traffic that must not, or should not, go through the default gateway. Routing is usually handled by routing devices and therefore it is often not necessary to configure static routes on Fedora servers or clients. Exceptions include traffic that must pass through an encrypted VPN tunnel or traffic that should take a less costly route. The default gateway is for any and all traffic which is not destined for the local network and for which no preferred route is specified in the routing table. The default gateway is traditionally a dedicated network router.

#### Configuring Static Routes Using the Command Line

Use the `ip route` command to display the IP routing table. If static routes are required, they can be added to the routing table by means of the `ip route add` command and removed using the `ip route del` command. To add a static route to a host address, that is to say to a single IP address, issue the following command as root:
Chapter 2. Configure Networking

ip route add 192.0.2.1

where 192.0.2.1 is the IP address of the host in dotted decimal notation. To add a static route to a network, that is to say to an IP address representing a range of IP addresses, issue the following command as root:

ip route add 192.0.2.0/24

where 192.0.2.0 is the IP address of the network in dotted decimal notation and /24 is the network prefix. The network prefix is the number of enabled bits in the subnet mask. This format of network address slash prefix length is referred to as CIDR notation.

Static route configuration is stored per-interface in a `/etc/sysconfig/network-scripts/ route-interface` file. For example, static routes for the eth0 interface would be stored in the `/etc/sysconfig/network-scripts/route-eth0` file. The `route-interface` file has two formats: `ip` command arguments and network/netmask directives. These are described below.

### Configuring The Default Gateway

The default gateway is determined by the network scripts which parse the `/etc/sysconfig/network` file first and then the network interface `ifcfg` files for interfaces that are “up”. The `ifcfg` files are parsed in numerically ascending order, and the last GATEWAY directive to be read is used to compose a default route in the routing table.

The default route can thus be indicated by means of the GATEWAY directive and can be specified either globally or in interface-specific configuration files. Specifying the gateway globally has certain advantages in static networking environments, especially if more than one network interface is present. It can make fault finding simpler if applied consistently. There is also the GATEWAYDEV directive, which is a global option. If multiple devices specify GATEWAY, and one interface uses the GATEWAYDEV directive, that directive will take precedence. This option is not recommend as it can have unexpected consequences if an interface goes down and it can complicate fault finding.

In dynamic network environments, where mobile hosts are managed by `NetworkManager`, gateway information is likely to be interface specific and is best left to be assigned by DHCP. In special cases where it is necessary to influence `NetworkManager`'s selection of the exit interface to be used to reach a gateway, make use of the `DEFROUTE=no` command in the `ifcfg` files for those interfaces which do not lead to the default gateway.

Global default gateway configuration is stored in the `/etc/sysconfig/network` file. This file specifies gateway and host information for all network interfaces. For more information about this file and the directives it accepts, see the `Fedora 20 System Administrator's Reference Guide`.

### 2.3.3. Configuring Static Routes in ifcfg files

Static routes set using `ip` commands at the command prompt will be lost if the system is shutdown or restarted. To configure static routes to be persistent after a system restart, they must be placed in per-interface configuration files in the `/etc/sysconfig/network-scripts/` directory. The file name should be of the format `route-ethX`. There are two types of commands to use in the configuration files; `ip` commands as explained in the section called “Static Routes Using the IP Command Arguments Format” and the Network/Netmask format as explained in the section called “Network/Netmask Directives Format”.

32
Static Routes Using the IP Command Arguments Format

If required in a per-interface configuration file, for example `/etc/sysconfig/network-scripts/route-eth0`, define a route to a default gateway on the first line. This is only required if the gateway is not set via DHCP and is not set globally in the `/etc/sysconfig/network` file:

```
default via 192.168.1.1 dev interface
```

where 192.168.1.1 is the IP address of the default gateway. The interface is the interface that is connected to, or can reach, the default gateway. The dev option can be omitted, it is optional. Note that this setting takes precedence over a setting in the `/etc/sysconfig/network` file.

If a route to a remote network is required, define a static route. Each line is parsed as an individual route:

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

where 10.10.10.0/24 is the network address and netmask of the remote network or host. 192.168.1.1 and interface are the IP address and interface for the gateway leading to the remote network. Add as many static routes as required.

The following is a sample route-eth0 file using the ip command arguments format. The default gateway is 192.168.0.1, interface eth0 and a leased line or WAN connection is available at 192.168.0.10. The two static routes are for reaching the 10.10.10.0/24 network and the 172.16.1.10/32 host:

```
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
```

Static routes should only be configured for remote networks or hosts, that is to say, networks or hosts that are not directly attached to the system. Packets going to the 192.168.0.0/24 network will be directed out the interface attached to that network. Packets to unknown, remote, networks will use the default gateway. Below is an example of setting static routes to a different network, on a machine in the 192.168.0.0/24 network, and an eth1 interface (with address 10.10.10.1) in the 10.10.10.0/24 network:

```
10.10.10.0/24 via 10.10.10.1 dev eth1
```

Specifying an exit interface is optional. It can be useful if you want to force traffic out of a specific interface. For example, in the case of a VPN, you can force traffic to a remote network to pass through a tun0 interface even when the interface is in a different subnet to the destination network.

Duplicate default gateways

If the default gateway is already assigned by DHCP, the ip command arguments format can cause one of two errors during start-up, or when bringing up an interface from the down state using the ifup command: "RTNETLINK answers: File exists" or 'Error: either "to" is a duplicate, or "X.X.X.X" is a garbage.' , where X.X.X.X is the gateway, or a different IP address. These errors can also occur if you have another route to another network using the default gateway. Both of these errors are safe to ignore.
Network/Netmask Directives Format

You can also use the network/netmask directives format for `route-interface` files. The following is a template for the network/netmask format, with instructions following afterwards:

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

- **ADDRESS0=10.10.10.0** is the network address of the remote network or host to be reached.
- **NETMASK0=255.255.255.0** is the netmask for the network address defined with `ADDRESS0=10.10.10.0`.
- **GATEWAY0=192.168.1.1** is the default gateway, or an IP address that can be used to reach `ADDRESS0=10.10.10.0`.

The following is a sample `route-eth0` file using the network/netmask directives format. The default gateway is 192.168.0.1, interface eth0. The two static routes are for reaching the 10.10.10.0/24 and 172.16.1.0/24 networks. This example is not necessary as traffic trying to reach a remote network or host would use the default gateway anyway:

```
ADDRESS0=10.10.10.0
NETMASK0=255.255.255.0
GATEWAY0=192.168.0.1
ADDRESS1=172.16.1.0
NETMASK1=255.255.255.0
GATEWAY1=192.168.1.1
```

Subsequent static routes must be numbered sequentially, and must not skip any values. For example, `ADDRESS0`, `ADDRESS1`, `ADDRESS2`, and so on.

Below is an example of setting static routes to a different network, on a machine in the 192.168.0.0/24 network. The example machine has an eth0 interface in the 192.168.0.0/24 network, and an eth1 interface (with address 10.10.10.1) in the 10.10.10.0/24 network:

```
ADDRESS0=10.10.10.0
NETMASK0=255.255.255.0
GATEWAY0=10.10.10.1
```

Note that if DHCP is used, it can assign these settings automatically.

### 2.3.4. Configuring IPv6 Tokenized Interface Identifiers

In a network, servers are generally given static addresses and these are usually configured manually to avoid relying on a DHCP server which may fail or run out of addresses. The IPv6 protocol introduced Stateless Address Autoconfiguration (SLAAC) which enables clients to assign themselves an address without relying on a DHCPv6 server. SLAAC derives the IPv6 address based on the interface hardware, therefore it should not be used for servers in case the hardware is changed and the associated SLAAC generated address changes with it. In an IPv6 environment, if the network prefix is changed, or the system is moved to a new location, any manually configured static addresses would have to be edited due to the changed prefix.

To address these problems, the IETF draft `Tokenised IPv6 Identifiers`\(^8\) has been implemented in the kernel together with corresponding additions to the `ip` utility. This enables the lower 64 bit interface

\(^8\) https://tools.ietf.org/id/draft-chown-6man-tokenised-ipv6-identifiers-02.txt
Using the NetworkManager Command Line Tool, nmcli

identifier part of the IPv6 address to be based on a token, supplied by the administrator, leaving the network prefix, the higher 64 bits, to be obtained from router advertisements (RA). This means that if the network interface hardware is changed, the lower 64 bits of the address will not change, and if the system is moved to another network, the network prefix will be obtained from router advertisements automatically, thus no manual editing is required.

To configure an interface to use a tokenized IPv6 identifier, issue a command in the following format as root user:

```
~# ip token set ::1a:2b:3c:4d/64 dev eth4
```

Where ::1a:2b:3c:4d/64 is the token to be used. This setting is not persistent. To make it persistent, add the command to an init script.

Using a memorable token is possible, but is limited to the range of valid hexadecimal digits. For example, for a DNS server, which traditionally uses port 53, a token of ::53/64 could be used.

To view all the configured IPv6 tokens, issue the following command:

```
~$ ip token
  token :: dev eth0
  token :: dev eth1
  token :: dev eth2
  token :: dev eth3
  token ::1a:2b:3c:4d dev eth4
```

To view the configured IPv6 token for a specific interface, issue the following command:

```
~$ ip token get dev eth4
  token ::1a:2b:3c:4d dev eth4
```

Note that adding a token to an interface will replace a previously allocated token, and in turn invalidate the address derived from it. Supplying a new token causes a new address to be generated and applied, but this process will leave any other addresses unchanged. In other words, a new tokenized identifier only replaces a previously existing tokenized identifier, not any other IP address.

Note

Take care not to add the same token to more than one system or interface as the duplicate address detection (DAD) mechanism will not be able to resolve the problem. Once a token is set, it cannot be cleared or reset, except by rebooting the machine.

2.4. Using the NetworkManager Command Line Tool, nmcli

The command-line tool nmcli can be used by both users and scripts for controlling NetworkManager. The basic format of a command is as follows:

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

where OBJECT can be one of general, networking, radio, connection, or device. The most used options are: -t, --terse for use in scripts, the -p, --pretty option for users, and the -h, --help option. Command completion has been implemented for nmcli, so remember to press
Tab when ever you are unsure of the command options available. See the nmcli(1) man page for a complete list of the options and commands.

The nmcli tool has some built-in context sensitive help. For example, issue the following two commands and notice the difference:

```
-]$
mcli 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[ochck]                         don't check nmcli and NetworkManager versions
  -a[sk]                            ask for missing parameters
  -w[aits] <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

-]$
mcli general help
Usage: nmcli general { COMMAND | help }

COMMAND := { status | permissions | logging }

status
permissions
logging [level <log level>] [domains <log domains>]
```

In the second example above the help is related to the object general.

The nmcli-examples(5) man page has many useful examples. We show a brief selection here:

To show the overall status of NetworkManager:

```
nmcli general status
```

To control NetworkManager logging:

```
nmcli general logging
```

To show all connections:

```
nmcli connection show
```

To show only currently active connections, add the -a, --active option as follows:

```
nmcli connection show --active
```
To show devices recognized by NetworkManager and their state:

```
$ nmcli device status
```

Commands can be shortened and some options omitted. For example the command:

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

Can be reduced to the following command:

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

The `id` option can be omitted because the connection ID (name) is unambiguous for `nmcli` in this case. As you become familiar with the commands, further abbreviations can be made. For example:

```
$ nmcli connection add type ethernet
```

can be reduced to:

```
$ nmcli c a type eth
```

**Note**

Remember to use tab completion when in doubt.

### Starting and Stopping an Interface Using nmcli

The `nmcli` tool can be used to start and stop any network interface including masters. For example:

```
$ nmcli con up id bond0
$ nmcli con up id port0
$ nmcli dev disconnect iface bond0
$ nmcli dev disconnect iface eth0
```

It is recommended to use `nmcli dev disconnect iface iface-name` rather than `nmcli con down id id-string` because disconnection places the interface into a “manual” mode, in which no automatic connection will be started until the user tells NetworkManager to start a connection or until an external event like a carrier change, hibernate, or sleep, occurs.

### The nmcli Interactive Connection Editor

The `nmcli` tool has an interactive connection editor. To use it, enter the command `nmcli con edit`. You will be prompted to enter a valid connection type from the list displayed. After entering a connection type you will be placed at the `nmcli` prompt. If you are familiar with the connection types you can add a valid connection type to the command and be taken straight to the `nmcli` prompt. Type `help` at the `nmcli` prompt to see a list of valid commands.

### 2.4.1. Understanding the nmcli Options

Many of the `nmcli` commands are self-explanatory, however a few command options are worth a moments study:
Chapter 2. Configure Networking

2.4.2. Connecting to a Network Using nmcli

To list the currently available network connections, issue a command as follows:

```
~$ nmcli con show
NAME               UUID                     TYPE            TIMESTAMP-REAL
eth0               4d5c449a-a6c5-451c-8206  802-3-ethernet  Tue 22 Oct 2013 19:50:00 BST
MyWiFi             91451385-4eb8-4080-8b82  802-11-wireless Tue 22 Oct 2013 08:50:08 BST
Bond connection 1  720aab83-28dd-4590-9325  bond            never
```

Note that the NAME field in the output always denotes the connection ID (name). It is not the interface name even though it might look the same. In the example above eth0 is the connection ID given by the user to the profile applied to the interface eth0. In the second line the user has assigned the connection ID MyWiFi to the interface wlan0.

Device status can also be viewed:

```
~$ nmcli dev status
DEVICE      TYPE            STATE
wlan0       802-11-wireless  connected
bond0       bond            connecting (getting IP configuration)
eth0        ethernet        disconnected
lo          loopback        unmanaged
```

Adding an Ethernet Connection

To add an Ethernet connection with manual IP configuration, issue a command as follows:

```bash
type — The connection type.
 Allowed values are: adsl, bond, bond-slave, bridge, bridge-slave, bluetooth, cdma,
eternet, gsm, infiniband, olpc-mesh, team, team-slave, vlan, wifi, wimax.

Each connection type has type-specific command options. Press Tab to see a list of them or see
the TYPE_SPECIFIC_OPTIONS list in the nmcli(1) man page. The type option is applicable
after the following: nmcli connection add and nmcli connection edit.

con-name — The name assigned to a connection profile.
 If you do not specify a connection name, one will be generated as follows:

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

The connection name is the name of a connection profile and should not be confused with the
interface name that denotes a device (wlan0, eth0, em1, and so on). Users can however name the
connections after interfaces, but they are not the same thing. There can be multiple connection
profiles available for a device. This is particularly useful for mobile devices or when switching a
network cable back and forth between different devices. Rather than edit the configuration, create
different profiles and apply them to the interface as needed. The id command also refers to the
connection profile name.

id — An identification string assigned by the user to a connection profile.
 The ID can be used in nmcli connection commands to identify a connection. The NAME field
in the output always denotes the connection ID (name). It refers to the same connection profile
name that the con-name does.

uuid — A unique identification string assigned by the system to a connection profile.
 The UUID can be used in nmcli connection commands to identify a connection.
To add two IPv4 DNS server addresses:

```
~]$ nmcli con mod my-eth1 ipv4.dns "8.8.8.8 8.8.4.4"
```

To add two IPv6 DNS server addresses:

```
~]$ nmcli con mod my-con-em1 ipv6.dns "2001:4860:4860::8888 2001:4860:4860::8844"
```

To bring up the new connection, issue a command as follows:

```
~]$ nmcli -p con up "my-eth1" ifname eth1
```

To view detailed information about the newly configured connection, issue a command as follows:

```
~]$ nmcli -p con show my-eth1
```

To lock this connection to a specific MAC address, issue a command as follows:

```
~]$ nmcli connection edit type ethernet con-name "my-eth1" mac 00-00-5E-00-53-00
```

### Adding a Wi-Fi Connection

To view the available Wi-Fi access points, issue a command as follows:

```
~]$ nmcli dev wifi list
SSID    MODE   CHAN  RATE     SIGNAL  BARS  SECURITY
MyCafe  Infra  11    54 MB/s  39      #▄__   WPA2
NextDoor Infra  1     54 MB/s  27      #___   WPA2
```

To create a Wi-Fi connection profile with manual IP configuration, but allowing automatic DNS address assignment, issue a command as follows:

```
~]$ nmcli con add con-name MyCafe ifname wlan0 type wifi ssid MyCafe p4 192.168.100.101/24 gw4 192.168.100.1
```

To set a WPA2 password, for example “caffeine”, issue commands as follows:

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

To change Wi-Fi state, issue a command in the following format:

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

### Changing a Specific Property

To check a specific property, for example mtu, issue a command as follows:

```
~]$ nmcli connection show id 'MyCafe' | grep mtu
```
Chapter 2. Configure Networking

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>802-11-wireless.mtu</td>
<td>auto</td>
</tr>
</tbody>
</table>

To change the property of a setting, issue a command as follows:

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

To verify the change, issue a command as follows:

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

Note that NetworkManager refers to parameters such as 802-3-ethernet and 802-11-wireless as the setting, and mtu as a property of the setting. See the nm-settings(5) man page for more information on properties and their settings.

2.5. Additional Resources

The following sources of information provide additional resources regarding nmcli.

2.5.1. Installed Documentation

- ip(8) man page — Describes the ip utility's command syntax.
- nmcli(1) man page — Describes NetworkManager's command-line tool.
- nmcli-examples(5) man page — Gives examples of nmcli commands.
- nm-settings(5) man page — Describes NetworkManager properties and their settings.

2.5.2. Online Documentation

RFC 1518 — Classless Inter-Domain Routing (CIDR)
Describes the CIDR Address Assignment and Aggregation Strategy, including variable-length subnetting.

RFC 1918 — Address Allocation for Private Internets
Describes the range of IPv4 addresses reserved for private use.

RFC 3330 — Special-Use IPv4 Addresses
Describes the global and other specialized IPv4 address blocks that have been assigned by the Internet Assigned Numbers Authority (IANA).

---

9 http://www.rfc-editor.org/info/rfc1518
10 http://www.rfc-editor.org/info/rfc1918
11 http://www.rfc-editor.org/info/rfc3330
Configure Host Names

3.1. Understanding Host Names
There are three classes of host name: static, pretty, and transient.

The “static” host name is the traditional hostname, which can be chosen by the user, and is stored in the `/etc/hostname` file. The “transient” hostname is a dynamic host name maintained by the kernel. It is initialized to the static host name by default, whose value defaults to “localhost”. It can be changed by DHCP or mDNS at runtime. The “pretty” hostname is a free-form UTF8 host name for presentation to the user.

A host name can be a free-form string up to 64 characters in length, however it is recommended that the static and transient names consists only of 7 bit ASCII lower-case characters, no spaces or dots, and limits itself to the format allowed for DNS domain name labels, even though this is not a strict requirement. The `hostnamectl` tool will enforce the following: Static and transient host names to consist of a-z, A-Z, 0-9, "-", "." and "." only, to not begin or end in a dot, and to not have two dots immediately following each other. The size limit of 64 characters is enforced.

3.2. Configuring Host Names Using hostnamectl
The `hostnamectl` tool is provided for administering the three separate classes of host names in use on a given system.

3.2.1. View All the Host Names
To view all the current host names, enter the following command:

```
~]# hostnamectl status
```

The `status` option is implied by default if no option is given.

3.2.2. Set All The Host Names
To set all the host names on a system, enter the following command as `root`:

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

This will alter the pretty, static, and transient host names alike. The static and transient host names will be simplified forms of the pretty host name. Spaces will be replaced with “-” and special characters will be removed.

3.2.3. Set A Particular Host Name
To set a particular host name, enter the following command as `root` with the relevant option:

```
~]# hostnamectl set-hostname name [option...]
```
Chapter 3. Configure Host Names

Where option is one or more of: --pretty, --static, and --transient.

If the --static or --transient options are used together with the --pretty option, the static and transient host names will be simplified forms of the pretty host name. Spaces will be replaced with "-" and special characters will be removed. If the --pretty option is not given, no simplification takes place.

When setting a pretty host name, remember to use the appropriate quotation marks if the host name contains spaces or a single quotation mark. For example:

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

### 3.2.4. Clear A Particular Host Name

To clear a particular host name and allow it to revert to the default, enter the following command as root with the relevant option:

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

Where "" is a quoted empty string and where option is one or more of: --pretty, --static, and --transient.

### 3.2.5. Changing Host Names Remotely

To execute a hostnamectl command on a remote system, use the -H, --host option as follows:

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

Where hostname is the remote host you wish to configure. The username is optional. The hostnamectl tool will use SSH to connect to the remote system.

### 3.3. Additional Resources

The following sources of information provide additional resources regarding hostnamectl.

#### 3.3.1. Installed Documentation

- **hostnamectl(1) man page** — Describes hostnamectl including the commands and command options.
- **hostname(1) man page** — Contains an explanation of the hostname and domainname command.
- **hostname(5) man page** — Contains an explanation of the host name file, its contents, and use.
- **hostname(7) man page** — Contains an explanation of host name resolution.
- **machine-info(5) man page** — Describes the local machine information file and the environment variables it contains.
- **machine-id(5) man page** — Describes the local machine ID configuration file.
- **systemd-hostnamed.service(8) man page** — Describes the systemd-hostnamed system service used by hostnamectl.
3.3.2. Online Documentation

http://www.freedesktop.org/wiki/Software/systemd/hostnamed

Information on systemd-hostnamed.
Configure Network Bonding

**Fedora** allows administrators to bind multiple network interfaces together into a single, bonded, channel. Channel bonding enables two or more network interfaces to act as one, simultaneously increasing the bandwidth and providing redundancy.

### 4.1. Configure Network Bonding Using NetworkManager

#### 4.1.1. Understanding the Default Behavior of Master and Slave Interfaces

When controlling bonded slave interfaces using `NetworkManager`, and especially when fault finding, keep the following in mind:

1. Starting the master interface does not automatically start the slave interfaces.
2. Starting a slave interface always starts the master interface.
3. Stopping the master interface also stops the slave interfaces.
4. A master without slaves can start static IP connections.
5. A master without slaves waits for slaves when starting DHCP connections.
6. A master with a DHCP connection waiting for slaves completes when a slave with a carrier is added.
7. A master with a DHCP connection waiting for slaves continues waiting when a slave without a carrier is added.

#### 4.1.2. Establishing a Bond Connection

You can use the GNOME `control-center` utility to direct `NetworkManager` to create a Bond from two or more Wired or InfiniBand connections. It is not necessary to create the connections to be bonded first. They can be configured as part of the process to configure the bond. You must have the MAC addresses of the interfaces available in order to complete the configuration process.

**Procedure 4.1. Adding a New Bond Connection**

You can configure a Bond connection by opening the `Network` window, clicking the plus symbol, and selecting `Bond` from the list.

1. Press the `Super` key to enter the Activities Overview, type `control network` and then press `Enter`. The `Network` settings tool appears.
2. Click the plus symbol to open the selection list. Select `Bond`. The `Editing Bond Connection 1` window appears.
3. On the `Bond` tab, click `Add` and select the type of interface you want to use with the bond connection. Click the `Create` button. Note that the dialog to select the slave type only comes up when you create the first slave; after that, it will automatically use that same type for all further slaves.
4. The `Editing bond0 slave 1` window appears. Fill in the MAC address of the first interface to be bonded. Click the `Save` button.
5. The name of the bonded slave appears in the **Bonded connections** window. Click the **Add** button to add further slave connections.

6. Review and confirm the settings and then click the **Save** button.

7. Edit the bond-specific settings by referring to the section called “Configuring the Bond Tab” below.

**Procedure 4.2. Editing an Existing Bond Connection**

Follow these steps to edit an existing bond connection.

1. Press the **Super** key to enter the Activities Overview, type **control network** and then press **Enter**. The **Network** settings tool appears.

2. Select the connection you wish to edit and click the **Options** button.

3. Select the **General** tab.

4. Configure the connection name, auto-connect behavior, and availability settings.

   Five settings in the **Editing** dialog are common to all connection types, see the **General** tab:

   - **Connection name** — Enter a descriptive name for your network connection. This name will be used to list this connection in the menu of the **Network** window.

   - **Automatically connect to this network when it is available** — Select this box if you want **NetworkManager** to auto-connect to this connection when it is available. See Section 2.2.3, “Connecting to a Network Automatically” for more information.

   - **All users may connect to this network** — Select this box to create a connection available to all users on the system. Changing this setting may require root privileges. See Section 2.2.4, “System-wide and Private Connection Profiles” for details. To prevent unexpected behavior during installation, ensure that this check box remains selected for any network interface that you configure.

   - **Automatically connect to VPN when using this connection** — Select this box if you want **NetworkManager** to auto-connect to a VPN connection when it is available. Select the VPN from the dropdown menu.

   - **Firewall Zone** — Select the firewall zone from the dropdown menu.

5. Edit the bond-specific settings by referring to the section called “Configuring the Bond Tab” below.

**Saving Your New (or Modified) Connection and Making Further Configurations**

Once you have finished editing your bond connection, click the **Save** button and **NetworkManager** will immediately save your customized configuration. Given a correct configuration, you can connect to your new or customized connection by selecting it from the network Notification Area applet. See Section 2.2.1, “Connecting to a Network Using a GUI” for information on using your new or altered connection.

You can further configure an existing connection by selecting it in the **Network** window and clicking **Options** to return to the **Editing** dialog.

Then, to configure:

- **IPv4** settings for the connection, click the **IPv4 Settings** tab and proceed to Section 2.2.10.4, “Configuring IPv4 Settings”; or,
• IPv6 settings for the connection, click the IPv6 Settings tab and proceed to Section 2.2.10.5, “Configuring IPv6 Settings”.

Configuring the Bond Tab
If you have already added a new bond connection (refer to Procedure 4.1, “Adding a New Bond Connection” for instructions), you can edit the Bond tab to set the load sharing mode and the type of link monitoring to use to detect failures of a slave connection.

Mode
The mode that is used to share traffic over the slave connections which make up the bond. The default is Round-robin. Other load sharing modes, such as 802.3ad, can be selected by means of the drop-down list.

Link Monitoring
The method of monitoring the slaves ability to carry network traffic.

The following modes of load sharing are selectable from the Mode drop-down list:

Round-robin
Sets a round-robin policy for fault tolerance and load balancing. Transmissions are received and sent out sequentially on each bonded slave interface beginning with the first one available.

Active backup
Sets an active-backup policy for fault tolerance. Transmissions are received and sent out via the first available bonded slave interface. Another bonded slave interface is only used if the active bonded slave interface fails. Note that this is the only mode available for bonds of InfiniBand devices.

XOR
Sets an XOR (exclusive-or) policy for fault tolerance and load balancing. Using this method, the interface matches up the incoming request's MAC address with the MAC address for one of the slave NICs. Once this link is established, transmissions are sent out sequentially beginning with the first available interface.

Broadcast
Sets a broadcast policy for fault tolerance. All transmissions are sent on all slave interfaces.

802.3ad
Sets an IEEE 802.3ad dynamic link aggregation policy. Creates aggregation groups that share the same speed and duplex settings. Transmits and receives on all slaves in the active aggregator. Requires a network switch that is 802.3ad compliant.

Adaptive transmit load balancing
Sets an adaptive Transmit Load Balancing (TLB) policy for fault tolerance and load balancing. The outgoing traffic is distributed according to the current load on each slave interface. Incoming traffic is received by the current slave. If the receiving slave fails, another slave takes over the MAC address of the failed slave.

Active Load Balancing
Sets an Active Load Balancing (ALB) policy for fault tolerance and load balancing. Includes transmit and receive load balancing for IPv4 traffic. Receive load balancing is achieved through ARP negotiation.

The following types of link monitoring can be selected from the Link Monitoring drop-down list. It is a good idea to test which channel bonding module parameters work best for your bonded interfaces.
MII (Media Independent Interface)
The state of the carrier wave of the interface is monitored. This can be done by querying the
driver, by querying MII registers directly, or by using `ethtool` to query the device. Three options are
available:

**Monitoring Frequency**
The time interval, in milliseconds, between querying the driver or MII registers.

**Link up delay**
The time in milliseconds to wait before attempting to use a link that has been reported as up.
This delay can be used if some gratuitous ARP requests are lost in the period immediately
following the link being reported as “up”. This can happen during switch initialization for
example.

**Link down delay**
The time in milliseconds to wait before changing to another link when a previously active link
has been reported as “down”. This delay can be used if an attached switch takes a relatively
long time to change to backup mode.

ARP
The address resolution protocol (ARP) is used to probe one or more peers to determine how well
the link layer connections are working. It is dependent on the device driver providing the transmit
start time and the last receive time. Two options are available:

**Monitoring Frequency**
The time interval, in milliseconds, between sending ARP requests.

**ARP targets**
A comma separated list of IP addresses to send ARP requests to.

### 4.2. Using the Command Line Interface (CLI)
A bond is created using the `bonding` kernel module and a special network interface called a `channel
bonding interface`.

#### 4.2.1. Check if Bonding Kernel Module is Installed
In **Fedora**, the bonding module is loaded by default. If necessary, you can make sure that the module
is loaded by issuing the following command as `root`:

```
~]$ modprobe --first-time bonding
modprobe: ERROR: could not insert 'bonding': Module already in kernel
```

To display information about the module, issue the following command:

```
~]$ modinfo bonding
```

See the `modprobe(8)` man page for more command options.

#### 4.2.2. Create a Channel Bonding Interface
To create a channel bonding interface, create a file in the `/etc/sysconfig/network-scripts/
directory called `ifcfg-bondN`, replacing `N` with the number for the interface, such as 0.
The contents of the file can be identical to whatever type of interface is getting bonded, such as an Ethernet interface. The only difference is that the DEVICE directive is bond\textit{N}, replacing \textit{N} with the number for the interface. The NM\_CONTROLLED directive can be added to prevent NetworkManager from configuring this device.

The following is a sample channel bonding configuration file:

```
Example 4.1. Sample ifcfg-bond0 Interface Configuration File

DEVICE=bond0
IPADDR=192.168.1.1
NETMASK=255.255.255.0
ONBOOT=yes
BOOTPROTO=none
USERCTL=no
NM_CONTROLLED=no
BONDING_OPTS="bonding parameters separated by spaces"
```

**Put all Bonding Module Parameters in ifcfg-bond\textit{N} Files**

Parameters for the bonding kernel module must be specified as a space-separated list in the BONDING_OPTS="\textit{bonding parameters}" directive in the ifcfg-bond\textit{N} interface file. Do not specify options for the bonding device in /etc/modprobe.d/bonding.conf, or in the deprecated /etc/modprobe.conf file. For further instructions and advice on configuring the bonding module and to view the list of bonding parameters, see Fedora 20 System Administrator's Reference Guide.

4.2.3. Creating MASTER and SLAVE Interfaces

After the channel bonding interface is created, the network interfaces to be bound together must be configured by adding the MASTER and SLAVE directives to their configuration files. The configuration files for each of the channel-bonded interfaces can be nearly identical.

For example, if two Ethernet interfaces are being channel bonded, both eth0 and eth1 may look like the following example:

```
DEVICE=eth\textit{N}
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
NM_CONTROLLED=no
```

In this example, replace \textit{N} with the numerical value for the interface.

4.3. Using the NetworkManager Command Line Tool, nmcli

To create a bond, with name mybond0, issue a command as follows:

```
$ nmcli con add type bond con-name mybond0 ifname mybond0 mode active-backup
Connection 'mybond0' (9301ff97-abbc-4432-aad1-246d7faea7fb) successfully added.
```
Chapter 4. Configure Network Bonding

To add a slave interface, issue a command in the following form:

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

To add additional slaves, repeat the previous command with a new interface, for example:

```
~]$ nmcli con add type bond-slave ifname ens3 master mybond0
```

Connection 'bond-slave-ens3-1' (50c59350-1531-45f4-ba04-33431c16e386) successfully added.

Note that as no con-name was given for the slaves, the name was derived from the interface name by prepending the type. At time of writing, nmcli only supports Ethernet slaves.

In order to bring up a bond, the slaves must be brought up first as follows:

```
~]$ 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)
```

Now bring up the bond as follows:

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

See Section 2.4, “Using the NetworkManager Command Line Tool, nmcli” for an introduction to nmcli

4.4. Additional Resources

The following sources of information provide additional resources regarding network bonding.

4.4.1. Installed Documentation

- nmcli(1) man page — Describes NetworkManager’s command-line tool.
- nmcli-examples(5) man page — Gives examples of nmcli commands.
- nm-settings(5) man page — Description of settings and parameters of NetworkManager connections.

4.4.2. Installable Documentation

- /usr/share/doc/kernel-doc-<kernel_version>/Documentation/ — This directory, which is provided by the kernel-doc package, contains information on bonding. Before accessing the kernel documentation, you must run the following command as root:

```
~]# yum install kernel-doc
```

/usr/share/doc/kernel-doc-version/Documentation/networking/bonding.txt — Describes the Linux bonding driver.
4.4.3. Online Documentation

*Fedora 20 System Administrator's Reference Guide*

Lists all the configurable parameters in an Ethernet interface configuration file.

*Fedora 20 System Administrator's Guide*

Explains the use of bonding module directives.
Configure Network Teaming

5.1. Understanding Network Teaming

The combining or aggregating together of network links in order to provide a logical link with higher throughput, or to provide redundancy, is known by many names such as “channel bonding”, “Ethernet bonding”, “port trunking”, “channel teaming”, “NIC teaming”, “link aggregation”, and so on. This concept as originally implemented in the Linux kernel is widely referred to as “bonding”. The term Network Teaming has been chosen to refer to this new implementation of the concept. The existing bonding driver is unaffected, Network Teaming is offered as an alternative and does not replace bonding in Fedora.

Network Teaming, or Team, is designed to implement the concept in a different way by providing a small kernel driver to implement the fast handling of packet flows, and various user-space applications to do everything else in user space. The driver has an Application Programming Interface (API), referred to as “Team Netlink API”, which implements Netlink communications. User-space applications can use this API to communication with the driver. A library, referred to as “lib”, has been provided to do user space wrapping of Team Netlink communications and RT Netlink messages. An application daemon, `teamd`, which uses Libteam lib is also provided. One instance of `teamd` can control one instance of the Team driver. The daemon implements the load-balancing and active-backup logic, such as round-robin, by using additional code referred to as “runners”. By separating the code in this way, the Network Teaming implementation presents an easily extensible and scalable solution for load-balancing and redundancy requirements. For example, custom runners can be relatively easily written to implement new logic via `teamd`, and even `teamd` is optional, users can write their own application to use `libteam`.

A tool to control a running instance of `teamd` using D-bus is provided by `teamdctl`. It provides a D-Bus wrapper around the `teamd` D-Bus API. By default, `teamd` listens and communicates using Unix Domain Sockets but still monitors D-Bus. This is to insure that `teamd` can be used in environments where D-Bus is not present or not yet loaded. For example, when booting over `teamd` links D-Bus would not yet be loaded. The `teamdctl` tool can be used during run time to read the configuration, the state of link-watchers, check and change the state of ports, add and remove ports, and to change ports between active and backup states.

Team Netlink API communicates with user-space applications using Netlink messages. The user-space library `libteam` does not directly interact with the API, but uses `libnl` or `teamnl` to interact with the driver API.

To sum up, the instances of Team driver, running in the kernel, do not get configured or controlled directly. All configuration is done with the aid of user space applications, such as the `teamd` application. The application then directs the kernel driver part accordingly.

5.2. Comparison of Network Teaming to Bonding

<table>
<thead>
<tr>
<th>Feature</th>
<th>Bonding</th>
<th>Team</th>
</tr>
</thead>
<tbody>
<tr>
<td>broadcast Tx policy</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>round-robin Tx policy</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>active-backup Tx policy</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LACP (802.3ad) support</td>
<td>Yes (passive only)</td>
<td>Yes</td>
</tr>
<tr>
<td>Hash-based Tx policy</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Feature</th>
<th>Bonding</th>
<th>Team</th>
</tr>
</thead>
<tbody>
<tr>
<td>User can set hash function</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Tx load-balancing support (TLB)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LACP hash port select</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>load-balancing for LACP support</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Ethtool link monitoring</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ARP link monitoring</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NS/NA (IPv6) link monitoring</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ports up/down delays</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>port priorities and stickiness (&quot;primary&quot; option enhancement)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>separate per-port link monitoring setup</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>multiple link monitoring setup</td>
<td>Limited</td>
<td>Yes</td>
</tr>
<tr>
<td>lockless Tx/Rx path</td>
<td>No (rwlock)</td>
<td>Yes (RCU)</td>
</tr>
<tr>
<td>VLAN support</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>user-space runtime control</td>
<td>Limited</td>
<td>Full</td>
</tr>
<tr>
<td>Logic in user-space</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Extensibility</td>
<td>Hard</td>
<td>Easy</td>
</tr>
<tr>
<td>Modular design</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Performance overhead</td>
<td>Low</td>
<td>Very Low</td>
</tr>
<tr>
<td>D-Bus interface</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>multiple device stacking</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>zero config using LLDP</td>
<td>No</td>
<td>(in planning)</td>
</tr>
<tr>
<td>NetworkManager support</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### 5.3. Understanding the Network Teaming Daemon and the "Runners"

The Team daemon, teamd, uses libteam to control one instance of the team driver. This instance of the team driver enslaves instances of a hardware device driver to form a “team” of network links. The team driver presents a network interface, team0 for example, to the other parts of the kernel. The interfaces created by instances of the team driver are given names such as team0, team1, and so forth in the documentation. This is for ease of understanding and other names can be used. The logic common to all methods of teaming is implemented by teamd and those functions that are unique to the different load sharing and backup methods, such as round-robin, are implemented by separate units of code referred to as “runners”. Because words such as “module” and “mode” already have specific meanings in relation to the kernel, the word “runner” was chosen to refer to these units of code. The user specifies the runner in the teamd.conf file and the code is then compiled into an instance of teamd when the instance is created. As the code for a “runner” is compiled into an instance of teamd as it is being created, the code for a “runner” is not a plug-in, although code could be created as a plug-in for teamd should the need arise.
Draft

Check if the Network Teaming Daemon is Installed

The following runners are available at time of writing:
- broadcast (data is transmitted over all ports)
- round-robin (data is transmitted over all ports in turn)
- active-backup (one port or link is used while others are kept as a backup)
- loadbalance (with active Tx load balancing and BPF-based Tx port selectors)
- lACP (implements the 802.3ad Link Access Control Protocol)

In addition, the following link-watchers are available:
- ethtool (libteam lib uses ethtool to watch for link state changes). This is the default if no other link-watcher is specified in the configuration file.
- arp_ping (The arp_ping utility is used to monitor the presence of a far-end hardware address using ARP packets.)
- nsna_ping (Neighbor Advertisements and Neighbor Solicitation from the IPv6 Neighbor Discovery protocol are used to monitor the presence of a neighbor’s interface)

There are no restrictions in the code to prevent a particular link-watcher from being used with a particular runner, however when using the lACP runner, ethtool is the only recommend link-watcher.

5.4. Check if the Network Teaming Daemon is Installed

To check if teamd is installed, run the following command as root:

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

5.5. Install the Network Teaming Daemon

To install teamd, run the following command as root:

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

5.6. Converting a Bond to a Team

It is possible to convert existing bonding configuration files to team configuration files using the bond2team tool. It can convert bond configuration files in ifcfg format to team configuration files in either ifcfg or JSON format. Note that firewall rules, alias interfaces, and anything that might be tied to the original interface name can break after the renaming because the tool will only change the ifcfg file, nothing else.

To see some examples of the command format, issue the following command:

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

New files will be created in a directory whose name starts with /tmp/bond2slave.XXXXXX, where XXXXXXX is a random string. After creating the new configuration files, move the old bonding files to a backup folder and then move the new files to the /etc/sysconfig/network-scripts/ directory. See the bond2team(1) man page for further details.
Chapter 5. Configure Network Teaming

5.7. Selecting Interfaces to Use as Ports for a Network Team

To view the available interfaces, issue the following command:

```
$ 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
```

From the available interfaces, determine which are suitable for adding to your network team and then proceed to Section 5.8, “Configure a Network Team”

5.8. Configure a Network Team

Configure a Network Team Using NetworkManager

To configure a network team using a graphical user interface, see Section 5.8.1, “Creating a Network Team Using NetworkManager”.

Configure a Network Team Using the Command Line Interface (CLI)

To create a network team using the command line, use teamd commands. To use this method, proceed to Section 5.8.2, “Creating a Network Team Using teamd”.

To create a network team using configuration files, proceed to Section 5.8.3, “Creating a Network Team Using ifcfg Files”.

5.8.1. Creating a Network Team Using NetworkManager

5.8.1.1. Establishing a Team Connection

You can use the GNOME control-center utility to direct NetworkManager to create a team from two or more Wired or InfiniBand connections. It is not necessary to create the connections to be teamed first. They can be configured as part of the process to configure the team. You must have the MAC addresses of the interfaces available in order to complete the configuration process.

Procedure 5.1. Adding a New Team Connection

You can configure a team connection by opening the Network window, clicking the plus symbol, and selecting Team from the list.

1. Press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears.

2. Click the plus symbol to open the selection list. Select Team. The Editing Team Connection 1 window appears.

3. On the Team tab, click Add and select the type of interface you want to use with the team connection. Click the Create button. Note that the dialog to select the slave type only comes up...
when you create the first slave; after that, it will automatically use that same type for all further slaves.

4. The Editing team0 slave 1 window appears. Fill in the MAC address of the first interface to be bonded.

5. If custom port settings are to be applied, click on the Team Port tab and enter a JSON configuration string or import it from a file.

6. Click the Save button.

7. The name of the teamed slave appears in the Teamed connections window. Click the Add button to add further slave connections.

8. Review and confirm the settings and then click the Save button.

9. Edit the team-specific settings by referring to the section called “Configuring the Team Tab” below.

Procedure 5.2. Editing an Existing Team Connection
Follow these steps to edit an existing bond connection.

1. Press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears.

2. Select the connection you wish to edit and click the Options button.

3. Select the General tab.

4. Configure the connection name, auto-connect behavior, and availability settings.

Five settings in the Editing dialog are common to all connection types, see the General tab:

- **Connection name** — Enter a descriptive name for your network connection. This name will be used to list this connection in the menu of the Network window.

- **Automatically connect to this network when it is available** — Select this box if you want NetworkManager to auto-connect to this connection when it is available. See Section 2.2.3, “Connecting to a Network Automatically” for more information.

- **All users may connect to this network** — Select this box to create a connection available to all users on the system. Changing this setting may require root privileges. See Section 2.2.4, “System-wide and Private Connection Profiles” for details. To prevent unexpected behavior during installation, ensure that this check box remains selected for any network interface that you configure.

- **Automatically connect to VPN when using this connection** — Select this box if you want NetworkManager to auto-connect to a VPN connection when it is available. Select the VPN from the dropdown menu.

- **Firewall Zone** — Select the firewall zone from the dropdown menu.

5. Edit the team-specific settings by referring to the section called “Configuring the Team Tab” below.

Saving Your New (or Modified) Connection and Making Further Configurations
Once you have finished editing your team connection, click the Save button and NetworkManager will immediately save your customized configuration. Given a correct configuration, you can connect to your new or customized connection by selecting it from the network Notification Area applet. See
Chapter 5. Configure Network Teaming

Section 2.2.1, “Connecting to a Network Using a GUI” for information on using your new or altered connection.

You can further configure an existing connection by selecting it in the Network window and clicking Options to return to the Editing dialog.

Then, to configure:

• IPv4 settings for the connection, click the IPv4 Settings tab and proceed to Section 2.2.10.4, “Configuring IPv4 Settings”; or,

• IPv6 settings for the connection, click the IPv6 Settings tab and proceed to Section 2.2.10.5, “Configuring IPv6 Settings”.

Configuring the Team Tab

If you have already added a new team connection (refer to Procedure 5.1, “Adding a New Team Connection” for instructions), you can enter a custom JSON configuration string in the text box or import a configuration file. Click Save to apply the JSON configuration to the team interface.

For examples of JSON strings, see Section 5.10, “Configure teamd Runners”

5.8.1.2. Understanding the Default Behavior of Master and Slave Interfaces

When controlling teamed slave interfaces using NetworkManager, and especially when fault finding, keep the following in mind:

1. Starting the master interface does not automatically start the slave interfaces.
2. Starting a slave interface always starts the master interface.
3. Stopping the master interface also stops the slave interfaces.
4. A master without slaves can start static IP connections.
5. A master without slaves waits for slaves when starting DHCP connections.
6. A master with a DHCP connection waiting for slaves completes when a slave with a carrier is added.
7. A master with a DHCP connection waiting for slaves continues waiting when a slave without a carrier is added.

5.8.2. Creating a Network Team Using teamd

To create a network team, a JSON format configuration file is required for the virtual interface that will serve as the interface to the team of ports or links. A quick way is to copy the example configuration files and then edit them using an editor running with root privileges. To list the available example configurations, enter the following command:

```bash
~\] 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_nsna_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
```
To view one of the included files, such as `activebackup_ethtool_1.conf`, enter the following command:

```
~$ cat /usr/share/doc/teamd-*/example_configs/activebackup_ethtool_1.conf
{
  "device": "team0",
  "runner": {"name": "activebackup"},
  "link_watch": {"name": "ethtool"},
  "ports": {
    "em1": {
      "prio": -10,
      "sticky": true
    },
    "em2": {
      "prio": 100
    }
  }
}
```

Create a working directory to store `teamd` configuration files. For example, as normal user, enter a command with the following format:

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

Copy the file you have chosen to your working directory and edit it as necessary. As an example, you could use a command with the following format:

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

To edit the file to suit your environment, for example to change the interfaces to be used as ports for the network team, open the file for editing as follows:

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

Make any necessary changes and save the file. See the `vi(1)` man page for help on using the `vi` editor or use your preferred editor.

Note that it is essential that the interfaces to be used as ports within the team must not be active, that is to say, they must be "down", when enslave them into a team device. To check their status, issue the following command:

```
~$ 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: ens6: <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: ens7: <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
```

In this example we see that both the interfaces we plan to use are "UP".

To take down an interface, issue a command as root in the following format:

```
~# ip link set down ens6
```
Repeat for each interface as necessary.

To create a team interface based on the configuration file, as root user, change to the \texttt{teamd\_working\_configs} directory and then issue a command in the following format:

\begin{verbatim}
-]# 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"
\end{verbatim}

The -\texttt{g} option is for debug messages, -\texttt{f} option is to specify the configuration file to load, and the -\texttt{d} option is to make the process run as a daemon after startup. See the \texttt{teamd(8)} man page for other options.

To check the status of the team, issue the following command as root:

\begin{verbatim}
-]# 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
\end{verbatim}

To apply an address to the network team interface, team0, issue a command as root in the following format:

\begin{verbatim}
-]# ip addr add 192.168.23.2/24 dev team0
\end{verbatim}

To check the IP address of a team interface, issue a command as follows:

\begin{verbatim}
-]# 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
\end{verbatim}

To activate the team interface, or to bring it "up", issue a command as root in the following format:

\begin{verbatim}
-]# ip link set dev team0 up
\end{verbatim}

To temporarily deactivate the team interface, or to take it "down", issue a command as root in the following format:
Creating a Network Team Using ifcfg Files

To create a networking team using ifcfg files, create a file in the `/etc/sysconfig/network-scripts/` directory as follows:

```bash
DEVICE="team0"
DEVICETYPE="Team"
ONBOOT="yes"
BOOTPROTO="static"
IPADDR="192.168.11.1"
NETMASK="255.255.255.0"
TEAM_CONFIG='{"runner": {"name": "activebackup"}, "link_watch": {"name": "ethtool"}}'
NM_CONTROLLED="no"
```

This creates the interface to the team, in other words, this is the master.

To create a slave to be a member of team0, create one or more files in the `/etc/sysconfig/network-scripts/` directory as follows:

```bash
DEVICE="eth1"
HWADDR="D4:85:64:01:46:9E"
DEVICETYPE="TeamPort"
ONBOOT="yes"
TEAM_MASTER="team0"
TEAM_PORT_CONFIG='{"prio": 100}"
NM_CONTROLLED="no"
```

Add additional slave interfaces similar to the above as required, changing the DEVICE and HWADDR field to match the ports (the network devices) being enslaved. If port priority is not specified by `prio` it defaults to `0`; it accepts negative and positive values in the range `-32,767` to `+32,767`.

Specifying the hardware or MAC address using the `HWADDR` directive will influence the device naming procedure as explained in Chapter 8, Consistent Network Device Naming.

To bring up the network team, issue the following command as root:

```bash
-]# ifup team0
```

To view the network team, issue the following command:

```bash
-]# ip link set dev team0 down
```

To terminate, or kill, an instance of the team daemon, as `root` user, issue a command in the following format:

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

The `-k` option is to specify that the instance of the daemon associated with the device team0 is to be killed. See the `teamd(8)` man page for other options.

For help on command line options for `teamd`, issue the following command:

```bash
-]$ teamd -h
```

In addition, see the `teamd(8)` man page.

5.8.3. Creating a Network Team Using ifcfg Files

To create a network team using ifcfg files, create a file in the `/etc/sysconfig/network-scripts/` directory as follows:

```bash
DEVICE="team0"
DEVICETYPE="Team"
ONBOOT="yes"
BOOTPROTO="static"
IPADDR="192.168.11.1"
NETMASK="255.255.255.0"
TEAM_CONFIG='{"runner": {"name": "activebackup"}, "link_watch": {"name": "ethtool"}}'
NM_CONTROLLED="no"
```

This creates the interface to the team, in other words, this is the master.

To create a slave to be a member of team0, create one or more files in the `/etc/sysconfig/network-scripts/` directory as follows:

```bash
DEVICE="eth1"
HWADDR="D4:85:64:01:46:9E"
DEVICETYPE="TeamPort"
ONBOOT="yes"
TEAM_MASTER="team0"
TEAM_PORT_CONFIG='{"prio": 100}"
NM_CONTROLLED="no"
```

Add additional slave interfaces similar to the above as required, changing the DEVICE and HWADDR field to match the ports (the network devices) being enslaved. If port priority is not specified by `prio` it defaults to `0`; it accepts negative and positive values in the range `-32,767` to `+32,767`.

Specifying the hardware or MAC address using the `HWADDR` directive will influence the device naming procedure as explained in Chapter 8, Consistent Network Device Naming.

To bring up the network team, issue the following command as root:

```bash
-]# ifup team0
```

To view the network team, issue the following command:
Chapter 5. Configure Network Teaming

5.8.4. Add a Port to a Network Team Using iputils
To add a port em1 to a network team team0, using the ip utility, issue the following commands as root:

```bash
~$ ip link show
~# ip link set dev em1 down
~# ip link set dev em1 master team0
```

Add additional ports as required. Team driver will bring ports up automatically.

5.8.5. Listing the ports of a Team Using teamnl
To view or list the ports in a network team, using the teamnl utility, issue the following command as root:

```bash
~# teamnl team0 ports
em2: up 100 fullduplex
em1: up 100 fullduplex
```

5.8.6. Configuring Options of a Team Using teamnl
To view or list all currently available options, using the teamnl utility, issue the following command as root:

```bash
~# teamnl team0 options
```

To configure a team to use active backup mode, issue the following command as root:

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

5.8.7. Add an Address to a Network Team Using iputils
To add an address to a team team0, using the ip utility, issue the following command as root:

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

5.8.8. Bring up an Interface to a Network Team Using iputils
To active or “bring up” an interface to a network team, team0, using the ip utility, issue the following command as root:

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

5.8.9. Viewing the Active Port Options of a Team Using teamnl
To view or list the activeport option in a network team, using the teamnl utility, issue the following command as root:

```bash
~# teamnl team0 getoption activeport
0
```
5.8.10. Setting the Active Port Options of a Team Using teamnl

To set the activeport option in a network team, using the `teamnl` utility, issue the following command as root:

```
$ teamnl team0 setoption activeport 5
```

5.9. Controlling teamd with teamdctl

In order to query a running instance of teamd for statistics or configuration information, or to make changes, the control tool `teamdctl` is used.

To view the current team state of a team team0, enter the following command:

```
$ teamdctl team0 state view
```

For a more verbose output:

```
$ teamdctl team0 state view -v
```

For a complete state dump in JSON format (useful for machine processing) of team0, use the following command:

```
$ teamdctl team0 state dump
```

For a configuration dump in JSON format of team0, use the following command:

```
$ teamdctl team0 config dump
```

To view the configuration of a port em1, that is part of a team team0, enter the following command:

```
$ teamdctl team0 port config dump em1
```

5.9.1. Add a Port to a Network Team

To add a port em1 to a network team team0, issue the following command as root:

```
# teamdctl team0 port add em1
```

5.9.2. Remove a Port From a Network Team

To remove an interface em1 from a network team team0, issue the following command as root:

```
# teamdctl team0 port remove em1
```

5.9.3. Apply a Configuration to a Port in a Network Team

To apply a JSON format configuration to a port em1 in a network team team0, issue a command as root in the following format:
Chapter 5. Configure Network Teaming

5.9.4. View the Configuration of a Port in a Network Team

To copy the configuration of a port em1 in a network team team0, issue the following command as root:

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

This will dump the JSON format configuration of the port to standard output.

5.10. Configure teamd Runners

5.10.1. Configure the broadcast Runner

To configure the broadcast runner, using an editor as root, add the following to the team `ifcfg` file:

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

5.10.2. Configure the random Runner

The random runner behaves similarly to the roundrobin runner, but it does not do percentage calculations and therefore uses less system resources.

To configure the random runner, using an editor as root, add the following to the team `ifcfg` file:

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

5.10.3. Configure the roundrobin Runner

To configure the roundrobin runner, using an editor as root, add the following to the team `ifcfg` file:

```
```
5.10.4. Configure the activebackup Runner

The active backup runner can use all of the link-watchers to determine the status of links in a team. Any one of the following examples can be added to the team `ifcfg` file:

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

This example configuration uses the active-backup runner with `ethtool` as the link watcher. Port `em2` has higher priority. The sticky flag ensures that if `em1` becomes active, it stays active as long as the link remains up.

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

This example configuration adds a queue ID of 4. It uses active-backup runner with `ethtool` as the link watcher. Port `em2` has higher priority. But the sticky flag ensures that if `em1` becomes active, it will stay active as long as the link remains up.
To configure the activebackup runner using **ethtool** as the link watcher and applying a delay, using an editor as **root**, add the following to the team **ifcfg** file:

```json
{
  "device": "team0",
  "runner": {
    "name": "activebackup"
  },
  "link_watch": {
    "name": "ethtool",
    "delay_up": 2500,
    "delay_down": 1000
  },
  "ports": {
    "em1": {
      "prio": -10,
      "sticky": true
    },
    "em2": {
      "prio": 100
    }
  }
}
```

This example configuration uses the active-backup runner with **ethtool** as the link watcher. Port em2 has higher priority. But the sticky flag ensures that if em1 becomes active, it stays active while the link remains up. Link changes are not propagated to the runner immediately, but delays are applied.

### 5.10.5. Configure the loadbalance Runner

This runner can be used for two types of load balancing, active and passive. In active mode, constant re-balancing of traffic is done by using statistics of recent traffic to share out traffic as evenly as possible. In static mode, streams of traffic are distributed randomly across the available links. This has a speed advantage due to lower processing overhead. In high volume traffic applications this is often preferred as traffic usually consists of multiple stream which will be distributed randomly between the available links, in his way load sharing is accomplished without intervention by teamd.

To configure the loadbalance runner for passive transmit (Tx) load balancing, using an editor as **root**, add the following to the team **ifcfg** file:

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

Configuration for hash-based passive transmit (Tx) load balancing.

To configure the loadbalance runner for active transmit (Tx) load balancing, using an editor as **root**, add the following to the team **ifcfg** file:

```json
{
  "device": "team0",
  "runner": {
    "name": "loadbalance",
    "tx_hash": ["eth", "ipv4", "ipv6"]
  },
  "ports": {
    "em1": {},
    "em2": {}
  }
}
```
Configuration for active transmit (Tx) load balancing using basic load balancer.

5.10.6. Configure the LACP (802.3ad) Runner

To configure the LACP runner using ethtool as a link watcher, using an editor as root, add the following to the team ifcfg file:

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

Configuration for connection to a link access control protocol (LACP) capable counterpart. The LACP runner should use ethtool to monitor the status of a link. It does not make sense to use any other link monitoring method besides the ethtool because, for example in the case of arp_ping, the link would never come up. The reason is that the link has to be established first and only after that can packets, ARP included, go through. Using ethtool prevents this because it monitors each link layer individually.

Active load balancing is possible with this runner in the same way as it is done for the loadbalance runner. To enable active transmit (Tx) load balancing, add the following section:

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

5.10.7. Configure Monitoring of the Link State

The following methods of link state monitoring are available. To implement one of the methods, add the JSON format string to the team ifcfg file using an editor running with root privileges.

5.10.7.1. Configure Ethtool for link-state Monitoring

To add or edit an existing delay, in milliseconds, between the link coming up and the runner being notified about it, add or edit a section as follows:

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

To add or edit an existing delay, in milliseconds, between the link going down and the runner being notified about it, add or edit a section as follows:

```json
"link_watch": {
    "name": "ethtool",
    "delay_down": 2500
}
```
5.10.7.2. Configure ARP Ping for Link-state Monitoring

The team daemon `teamd` sends an ARP REQUEST to an address at the remote end of the link in order to determine if the link is up. The method used is the same as the `arping` utility but it does not use that utility.

Prepare a file containing the new configuration in JSON format similar to the following example:

```json
{
    "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
        }
    }
}
```

This configuration uses `arp_ping` as the link watcher. The `missed_max` option is a limit value of the maximum allowed number of missed replies (ARP replies for example). It should be chosen in conjunction with the `interval` option in order to determine the total time before a link is reported as down.

To load a new configuration for a team port `em2`, with JSON configuration string `JSON-config-string`, issue the following command as root:

```
~# port config update em2 JSON-config-string
```

Note that the old configuration will be overwritten and that any options omitted will be reset to the default values. See the `teamdctl(8)` man page for more team daemon control tool command examples.

5.10.7.3. Configure IPv6 NA/NS for Link-state Monitoring

```json
{
    "device": "team0",
    "runner": {"name": "activebackup"},
    "link_watch": {
        "name": "nsna_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
        }
    }
}
```
To configure the interval between sending NS/NA packets, add or edit a section as follows:

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

Value is positive number in milliseconds. It should be chosen in conjunction with the `missed_max` option in order to determine the total time before a link is reported as down.

To configure the maximum number of missed NS/NA reply packets to allow before reporting the link as down, add or edit a section as follows:

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

Maximum number of missed NS/NA reply packets. If this number is exceeded, the link is reported as down. The `missed_max` option is a limit value of the maximum allowed number of missed replies (ARP replies for example). It should be chosen in conjunction with the `interval` option in order to determine the total time before a link is reported as down.

To configure the host name from which to derive the IPv6 address target address for the NS/NA packets, add or edit a section as follows:

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

Host name to be converted to an IPv6 address which will be used as the target address for the NS/NA packets. An IPv6 address can be used in place of a host name.

### 5.10.8. Configure Port Selection Override

The physical port which transmits a frame is normally selected by the kernel part of the team driver, and is not relevant to the user or system administrator. The output port is selected using the policies of the selected team mode (`teamd` runner). On occasion however, it is helpful to direct certain classes of outgoing traffic to certain physical interfaces to implement slightly more complex policies. By default the team driver is multiqueue aware and 16 queues are created when the driver initializes (see /
usr/share/doc/kernel-doc-version/Documentation/networking/multiqueue.txt for details). If more or less queues are desired, the Netlink attribute `tx_queues` can be used to change this value during the team driver instance creation.

The queue ID for a port can be set by the port configuration option `queue_id` as follows:

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

These queue ID's can be used in conjunction with the `tc` utility to configure a multiqueue queue discipline and filters to bias certain traffic to be transmitted on certain slave devices. For example, if using the above configuration and wanting to force all traffic bound to 192.168.1.100 to use eth1 in the team as its output device, issue commands as root in the following format:

```
~$ 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
```

This mechanism of overriding runner selection logic in order to bind traffic to a specific port can be used with all runners.

### 5.10.9. Configure BPF-based Tx Port Selectors for Hash Computation Algorithm

The loadbalance and LACP runners uses hashes of packets to sort network traffic flow. The hash computation mechanism is based on the Berkeley Packet Filter (BPF) code. The BPF code is used to generate a hash rather than make a policy decision for incoming packets. The hash length is 8 bits giving 256 variants. This means many different socket buffers (SKB) can have the same hash and therefore pass traffic over the same link. The use of a short hash is a quick way to sort traffic into different streams for the purposes of load balancing across multiple links. In static mode, the hash is only used to decided out of which port the traffic should be sent. In active mode, the runner will continually reassign hashes to different ports in an attempt to reach a perfect balance.

The following fragment types or strings can be used for packet Tx hash computation:

- **eth** — Uses source and destination MAC addresses.
- **vlan** — Uses VLAN ID.
- **ipv4** — Uses source and destination IPv4 addresses.
- **ipv6** — Uses source and destination IPv6 addresses.
- **ip** — Uses source and destination IPv4 and IPv6 addresses.
- **l3** — Uses source and destination IPv4 and IPv6 addresses.
- **tcp** — Uses source and destination TCP ports.
- **udp** — Uses source and destination UDP ports.
- **sctp** — Uses source and destination SCTP ports.
- **l4** — Uses source and destination TCP and UDP and SCTP ports.

These strings can be used by adding a line in the following format to the load balance runner:
"tx_hash": ["eth", "ipv4", "ipv6"]

See Section 5.10.5, “Configure the loadbalance Runner” for an example.

5.11. Configure Network Teaming Using nmcli

To create a new team interface, with name team-ServerA, issue a command as follows:

```
~$ nmcli connection add type team ifname ServerA
Connection 'team-ServerA' (981eb129-1707-4a2e-a6ea-413330d96c10) successfully added.
```

As no JSON configuration file was specified the default configuration is used. Notice that the name was derived from the interface name by prepending the type. Alternatively, specify a name with **con-name** as follows:

```
~$ nmcli connection add type team con-name Team0 ifname ServerB
Connection 'Team0' (fcafb3f0-4c95-48df-9e28-7ac7213f38ba) successfully added.
```

To view the team interfaces just configured, issue a command as follows:

```
~$ nmcli connection show
NAME          UUID                                  TYPE        TIMESTAMP-REAL
Team0         fcafb3f0-4c95-48df-9e28-7ac7213f38ba  team        never
team-ServerA  981eb129-1707-4a2e-a6ea-413330d96c10  team        never
```

To load a team configuration file for a team that already exists, issue a command as follows:

```
~$ nmcli connection modify team-ServerA team.config JSON-Config
```

To add, or enslave, an interface to the team, with name team-slave-ens3, issue a command as follows:

```
~$ nmcli connection add type team-slave ifname ens3 master Team0
Connection 'team-slave-ens3' (a33d5d32-87d7-4dc4-8a27-5a44aabfa440) successfully added.
```

Notice that the name was derived from the interface name by prepending the type. Alternatively, specify a name with **con-name** as follows:

```
~$ nmcli con add type team-slave con-name Team0-slave1 ifname ens3 master Team0
Connection 'Team0-slave1' (adbf21f2-51b6-492f-8fc8-48b831383ac9) successfully added.
~$ nmcli con add type team-slave con-name Team0-slave2 ifname ens7 master Team0
Connection 'Team0-slave2' (e5317075-c0c1-472f-b25d-0433b0297ea3) successfully added.
```

At time of writing, **nmcli** only supports Ethernet slaves.

In order to bring up a team, the slaves must be brought up first as follows:

```
~$ nmcli connection up Team0-slave1
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/2)
~$ nmcli connection up Team0-slave2
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/3)
```

You can verify the team interface was brought up by the activation of the slaves, as follows:
Chapter 5. Configure Network Teaming

```bash
$ 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:ff
```

Alternatively, issue a command to bring up the team as follows:

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

See Section 2.4, “Using the NetworkManager Command Line Tool, nmcli” for an introduction to `nmcli`

### 5.12. Additional Resources

The following sources of information provide additional resources regarding network teaming.

#### 5.12.1. Installed Documentation

- `teamd(8)` man page — Describes the `teamd` service.
- `teamdctl(8)` man page — Describes the `teamd` control tool.
- `teamd.conf(5)` man page — Describes the `teamd` configuration file.
- `teamnl(8)` man page — Describes the `teamd` Netlink library.
- `bond2team(1)` man page — Describes a tool to convert bonding options to team.

#### 5.12.2. Installable Documentation

- `/usr/share/doc/kernel-doc-<kernel_version>/Documentation/` — This directory, which is provided by the `kernel-doc` package, contains information on bonding which is also relevant to teaming. Before accessing the kernel documentation, you must run the following command as root:

  ```bash
  $ yum install kernel-doc
  ```


#### 5.12.3. Online Documentation

- [www.libteam.org](http://www.libteam.org) — The upstream project.
- [www.w3schools.com/json/json_syntax.asp](http://www.w3schools.com/json/json_syntax.asp) — An explanation of JSON syntax.
Configure Network Bridging

A network bridge is a link-layer device which forwards traffic between networks based on MAC addresses. It makes forwarding decisions based on a table of MAC addresses which it builds by listening to network traffic and thereby learning what hosts are connected to each network. A software bridge can be used within a Linux host in order to emulate a hardware bridge, for example in virtualization applications for sharing a NIC with one or more virtual NICs.

Note that a bridge cannot be established over Wi-Fi networks operating in Ad-Hoc or Infrastructure modes. This is due to the IEEE 802.11 standard that specifies the use of 3-address frames in Wi-Fi for the efficient use of airtime. A system configured to be an access point (AP) running the hostapd can support the necessary 4-address frames.

6.1. Using NetworkManager

When starting a bridge interface, NetworkManager waits for at least one port to enter the “forwarding” state before beginning any network-dependent IP configuration such as DHCP or IPv6 autoconfiguration. Static IP addressing is allowed to proceed before any slaves or ports are connected or begin forwarding packets.

6.1.1. Establishing a Bridge Connection

Procedure 6.1. Adding a New Bridge Connection

1. You can configure a new Bridge connection by opening the Network window and selecting the plus symbol below the menu.

2. To use the graphical Network settings tool, press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears.

3. Select the plus symbol below the menu. The Add Network Connection window appears.

4. Select the Bridge menu entry. The Editing Bridge connection 1 window appears.

Procedure 6.2. Editing an Existing Bridge Connection

You can configure an existing bridge connection by opening the Network window and selecting the name of the connection from the list. Then click the Edit button.

1. Press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears.

2. Select the Bridge connection you wish to edit from the left hand menu.

3. Click the Options button.

Configuring the Connection Name, Auto-Connect Behavior, and Availability Settings

Five settings in the Editing dialog are common to all connection types, see the General tab:

- Connection name — Enter a descriptive name for your network connection. This name will be used to list this connection in the menu of the Network window.

- Automatically connect to this network when it is available — Select this box if you want NetworkManager to auto-connect to this connection when it is available. See Section 2.2.3, “Connecting to a Network Automatically” for more information.
• **All users may connect to this network** — Select this box to create a connection available to all users on the system. Changing this setting may require root privileges. See Section 2.2.4, “System-wide and Private Connection Profiles” for details. To prevent unexpected behavior during installation, ensure that this check box remains selected for any network interface that you configure.

• **Automatically connect to VPN when using this connection** — Select this box if you want NetworkManager to auto-connect to a VPN connection when it is available. Select the VPN from the dropdown menu.

• **Firewall Zone** — Select the Firewall Zone from the dropdown menu.

**Configuring the Bridge Tab**

**Interface name**
The name of the interface to the bridge.

**Bridged connections**
One or more slave interfaces.

**Aging time**
The time, in seconds, a MAC address is kept in the MAC address forwarding database.

**Enable STP (Spanning Tree Protocol)**
If required, select the check box to enable STP.

**Priority**
The bridge priority; the bridge with the lowest priority will be elected as the root bridge.

**Forward delay**
The time, in seconds, spent in both the Listening and Learning states before entering the Forwarding state.

**Hello time**
The time interval, in seconds, between sending configuration information in bridge protocol data units (BPU).

**Max age**
The maximum time, in seconds, to store the configuration information from BPDUs. This value should be twice the Hello Time plus 1 but less than twice the Forwarding delay minus 1.
Saving Your New (or Modified) Connection and Making Further Configurations

Once you have finished editing your new bridge connection, click the Save button and NetworkManager will immediately save your customized configuration. Given a correct configuration, you can connect to your new or customized connection by selecting it from the network Notification Area applet. See Section 2.2.1, “Connecting to a Network Using a GUI” for information on using your new or altered connection.

You can further configure an existing connection by selecting it in the Network window and clicking Configure to return to the Editing dialog.

Then, to configure:

- IPv4 settings for the connection, click the IPv4 Settings tab and proceed to Section 2.2.10.4, “Configuring IPv4 Settings”, or;

- IPv6 settings for the connection, click the IPv6 Settings tab and proceed to Section 2.2.10.5, “Configuring IPv6 Settings”.

Figure 6.1. Editing Bridge Connection 1
6.2. Using the Command Line Interface (CLI)

6.2.1. Check if Bridging Kernel Module is Installed

In Fedora, the bridging module is loaded by default. If necessary, you can make sure that the module is loaded by issuing the following command as root:

```bash
$ modprobe --first-time bridge
modprobe: ERROR: could not insert 'bridge': Module already in kernel
```

To display information about the module, issue the following command:

```bash
$ modinfo bridge
```

See the `modprobe(8)` man page for more command options.

6.2.2. Create a Network Bridge

To create a network bridge, create a file in the `/etc/sysconfig/network-scripts/` directory called `ifcfg-brN`, replacing `N` with the number for the interface, such as `0`.

The contents of the file is similar to whatever type of interface is getting bridged to, such as an Ethernet interface. The differences in this example are as follows:

- The `DEVICE` directive is given an interface name as its argument in the format `brN`, where `N` is replaced with the number of the interface.
- The `TYPE` directive is given an argument `Bridge` or `Ethernet`. This directive determines the device type and the argument is case sensitive.
- The bridge interface configuration file is given an IP address whereas the physical interface configuration file must only have a MAC address (see below).
- An extra directive, `DELAY=0`, is added to prevent the bridge from waiting while it monitors traffic, learns where hosts are located, and builds a table of MAC addresses on which to base its filtering decisions. The default delay of 30 seconds is not needed if no routing loops are possible.
- The `NM_CONTROLLED=no` should be added to the Ethernet interface to prevent NetworkManager from altering the file. It can also be added to the bridge configuration file.

The following is a sample bridge interface configuration file using a static IP address:

```
DEVICE=br0
TYPE=Bridge
IPADDR=192.168.1.1
NETMASK=255.255.255.0
ONBOOT=yes
BOOTPROTO=none
NM_CONTROLLED=no
DELAY=0
```

To complete the bridge another interface is created, or an existing interface is modified, and pointed to the bridge interface. The following is a sample Ethernet interface configuration file pointing to a bridge interface:

```
DEVICE=eth0
TYPE=Ethernet
IPADDR=192.168.2.1
NETMASK=255.255.255.0
ONBOOT=yes
BOOTPROTO=none
NM_CONTROLLED=no
DELAY=0
```
interface. Configure your physical interface in `/etc/sysconfig/network-scripts/ifcfg-ethX`, where \( X \) is a unique number corresponding to a specific interface, as follows:

```plaintext
Example 6.2. Sample ifcfg-ethX Interface Configuration File

DEVICE=ethX
TYPE=Ethernet
HWADDR=AA:BB:CC:DD:EE:FF
BOOTPROTO=none
ONBOOT=yes
NM_CONTROLLED=no
BRIDGE=br0
```

**Note**

For the `DEVICE` directive, almost any interface name could be used as it does not determine the device type. `TYPE=Ethernet` is not strictly required. If the `TYPE` directive is not set, the device is treated as an Ethernet device (unless its name explicitly matches a different interface configuration file.)

Specifying the hardware or MAC address using the `HWADDR` directive will influence the device naming procedure as explained in *Chapter 8, Consistent Network Device Naming*.

See the *Fedora 20 System Administrator’s Reference Guide* for a review of the directives and options used in network interface configuration files.

**Warning**

If you are configuring bridging on a remote host, and you are connected to that host over the physical NIC you are configuring, please consider the implications of losing connectivity before proceeding. You will lose connectivity when restarting the service and may not be able to regain connectivity if any errors have been made. Console, or out-of-band access is advised.

Restart the networking service in order for the changes to take effect. As `root` issue the following command:

```
~\]# systemctl network restart
```

An example of a network bridge formed from two or more bonded Ethernet interfaces will now be given as this is another common application in a virtualization environment. If you are not very familiar with the configuration files for bonded interfaces then please refer to *Section 4.2.2, “Create a Channel Bonding Interface”*

Create or edit two or more Ethernet interface configuration files, which are to be bonded, as follows:

```plaintext
DEVICE=ethX
TYPE=Ethernet
USERCTL=no
```
Chapter 6. Configure Network Bridging

SLAVE=yes
MASTER=bond0
BOOTPROTO=none
HWADDR=AA:BB:CC:DD:EE:FF
NM_CONTROLLED=no

Note

Using ethX as the interface name is common practice but almost any name could be used.

Create or edit one interface configuration file, `/etc/sysconfig/network-scripts/ifcfg-bond0`, as follows:

```
DEVICE=bond0
ONBOOT=yes
BONDING_OPTS='mode=1 miimon=100'
BRIDGE=brbond0
NM_CONTROLLED=no
```

For further instructions and advice on configuring the bonding module and to view the list of bonding parameters, see the *Fedora 20 System Administrator's Reference Guide*.

Create or edit one interface configuration file, `/etc/sysconfig/network-scripts/ifcfg-brbond0`, as follows:

```
DEVICE=brbond0
ONBOOT=yes
TYPE=Bridge
IPADDR=192.168.1.1
NETMASK=255.255.255.0
NM_CONTROLLED=no
```

We now have two or more interface configuration files with the `MASTER=bond0` directive. These point to the configuration file named `/etc/sysconfig/network-scripts/ifcfg-bond0`, which contains the `DEVICE=bond0` directive. This `ifcfg-bond0` in turn points to the `/etc/sysconfig/network-scripts/ifcfg-brbond0` configuration file, which contains the IP address, and acts as an interface to the virtual networks inside the host.

Restart the networking service, in order for the changes to take effect. As root issue the following command:

```
# systemctl network restart
```

6.3. Using the NetworkManager Command Line Tool, nmcli

To create a bridge, with name bridge-br0, issue a command as follows:

```
$ nmcli con add type bridge ifname br0
```

Connection 'bridge-br0' (79cf6a3e-0310-4a78-b759-bda1cc3eef8d) successfully added.

If no interface name is specified, the name will default to bridge, bridge-1, bridge-2, and so on.

To view the connections, issue the following command:
Using the NetworkManager Command Line Tool, nmcli

Using the NetworkManager Command Line Tool, nmcli

<table>
<thead>
<tr>
<th>NAME</th>
<th>UUID</th>
<th>TYPE</th>
<th>TIMESTAMP-REAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth0</td>
<td>4d5c449a-a6c5-451c-8206-3c9a4ec8bca</td>
<td>802-3-ethernet</td>
<td>Mon 21 Oct 2013 16:01:53 BST</td>
</tr>
<tr>
<td>bridge-br0</td>
<td>79cf6a3e-0310-4a78-b759-bda1cc3eef8d</td>
<td>bridge</td>
<td>never</td>
</tr>
</tbody>
</table>

Spanning tree protocol (STP) according to the IEEE 802.1D standard is enabled by default. To disable STP for this bridge, issue a command as follows:

```bash
$ nmcli con bridge-br0 stp no
```

To re-enable 802.1D STP for this bridge, issue a command as follows:

```bash
$ nmcli con bridge-br0 stp yes
```

The default bridge priority for 802.1D STP is 32768. The lower number is preferred in root bridge selection. For example, a bridge with priority of 28672 would be selected as the root bridge in preference to a bridge with priority value of 32768 (the default). To create a bridge with a non-default value, issue a command as follows:

```bash
$ nmcli con add type bridge ifname br5 stp yes priority 28672
```

Connection 'bridge-br5' (86b83ad3-b466-4795-aeb6-4a66eb1056c7) successfully added.

The allowed values are in the range 0 to 65535, but can only be set in multiples of 4096.

To change the bridge priority of an existing bridge to a non-default value, issue a command in the following format:

```bash
$ nmcli connection modify bridge-br5 bridge.priority 36864
```

The allowed values are in the range 0 to 65535, but can only be set in multiples of 4096.

Further options for 802.1D STP are listed in the bridge section of the `nmcli(1)` man page.

To add, or enslave an interface, for example eth1, to the bridge bridge-br0, issue a command as follows:

```bash
$ nmcli con add type bridge-slave ifname eth1 master bridge-br0
```

Connection 'bridge-slave-eth1' (70ffae80-7428-4d9c-8cbd-2e35de72476e) successfully added.

At time of writing, `nmcli` only supports Ethernet slaves.

To change a value using interactive mode, issue the following command:

```bash
$ nmcli connection edit bridge-br0
```

You will be placed at the `nmcli` prompt.

```
nmcli> set bridge.priority 4096
nmcli> save
Connection 'bridge-br0' (79cf6a3e-0310-4a78-b759-bda1cc3eef8d) successfully saved.
nmcli> quit
```

See Section 2.4, "Using the NetworkManager Command Line Tool, nmcli" for an introduction to `nmcli`
6.4. Additional Resources

The following sources of information provide additional resources regarding network bridging.

6.4.1. Installed Documentation

• **nmcli(1) man page** — Describes `NetworkManager`'s command-line tool.

• **nmcli-examples(5) man page** — Gives examples of `nmcli` commands.

• **nm-settings(5) man page** — Description of settings and parameters of `NetworkManager` connections.
Configure 802.1Q VLAN tagging

7.1. Configure 802.1Q VLAN Tagging Using NetworkManager

7.1.1. Establishing a VLAN Connection
You can use NetworkManager to create a VLAN using an existing interface. Currently, at time of writing, you can only make VLANS on Ethernet devices.

Procedure 7.1. Adding a New VLAN Connection
You can configure a VLAN connection by opening the Network window, clicking the plus symbol, and selecting VLAN from the list.

1. Press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears.
2. Click the plus symbol to open the selection list. Select VLAN. The Editing VLAN Connection 1 window appears.
3. On the VLAN tab, select the parent interface from the drop-down list you want to use for the VLAN connection.
4. Enter the VLAN ID
5. Enter a VLAN interface name. This is the name of the VLAN interface that will be created. For example, “eth0.1” or “vlan2”. (Normally this is either the parent interface name plus “.” and the VLAN ID, or “vlan” plus the VLAN ID.)
6. Review and confirm the settings and then click the Save button.
7. To edit the VLAN-specific settings see the section called “Configuring the VLAN Tab”.

Procedure 7.2. Editing an Existing VLAN Connection
Follow these steps to edit an existing VLAN connection.

1. Press the Super key to enter the Activities Overview, type control network and then press Enter. The Network settings tool appears.
2. Select the connection you wish to edit and click the Options button.
3. Select the General tab.
4. Configure the connection name, auto-connect behavior, and availability settings.

These settings in the Editing dialog are common to all connection types:

- **Connection name** — Enter a descriptive name for your network connection. This name will be used to list this connection in the VLAN section of the Network window.

- **Automatically connect to this network when it is available** — Select this box if you want NetworkManager to auto-connect to this connection when it is available. Refer to Section 2.2.3, "Connecting to a Network Automatically" for more information.
• **Available to all users** — Select this box to create a connection available to all users on the system. Changing this setting may require root privileges. Refer to *Section 2.2.4, “System-wide and Private Connection Profiles”* for details.

5. To edit the VLAN-specific settings see *the section called “Configuring the VLAN Tab”*.

### Saving Your New (or Modified) Connection and Making Further Configurations

Once you have finished editing your VLAN connection, click the **Save** button and **NetworkManager** will immediately save your customized configuration. Given a correct configuration, you can connect to your new or customized connection by selecting it from the Notification Area applet. See *Section 2.2.1, “Connecting to a Network Using a GUI”* for information on using your new or altered connection.

You can further configure an existing connection by selecting it in the **Network** window and clicking **Options** to return to the **Editing** dialog.

Then, to configure:

• IPv4 settings for the connection, click the **IPv4 Settings** tab and proceed to *Section 2.2.10.4, “Configuring IPv4 Settings”*.

### Configuring the VLAN Tab

If you have already added a new VLAN connection (refer to *Procedure 7.1, “Adding a New VLAN Connection”* for instructions), you can edit the **VLAN** tab to set the parent interface and the VLAN ID.

**Parent Interface**

A previously configured interface can be selected in the drop-down list.

**VLAN ID**

The identification number to be used to tag the VLAN network traffic.

**VLAN interface name**

The name of the VLAN interface that will be created. For example, "eth0.1" or "vlan2".

**Cloned MAC address**

Optionally sets an alternate MAC address to use for identifying the VLAN interface. This can be used to change the source MAC address for packets sent on this VLAN.

**MTU**

Optionally sets a Maximum Transmission Unit (MTU) size to be used for packets to be sent over the VLAN connection.

### 7.2. Configure 802.1Q VLAN Tagging Using the Command Line

In **Fedora**, the 8021q module is loaded by default. If necessary, you can make sure that the module is loaded by issuing the following command as **root**:

```
~]# modprobe --first-time 8021q
modprobe: ERROR: could not insert '8021q': Module already in kernel
```

To display information about the module, issue the following command:
See the `modprobe(8)` man page for more command options.

### 7.2.1. Setting Up 802.1Q VLAN Tagging Using ifcfg Files

1. Configure your physical interface in `/etc/sysconfig/network-scripts/ifcfg-ethX`, where X is a unique number corresponding to a specific interface, as follows:

   ```
   DEVICE=ethX
   TYPE=Ethernet
   BOOTPROTO=none
   ONBOOT=yes
   ```

2. Configure the VLAN interface configuration in the `/etc/sysconfig/network-scripts/` directory. The configuration file name should be the physical interface plus a . character plus the VLAN ID number. For example, if the VLAN ID is 192, and the physical interface is eth0, then the configuration file name should be `ifcfg-eth0.192`:

   ```
   DEVICE=ethX.192
   BOOTPROTO=none
   ONBOOT=yes
   IPADDR=192.168.1.1
   NETMASK=255.255.255.0
   USERCTL=no
   NETWORK=192.168.1.0
   VLAN=yes
   NM_CONTROLLED=no
   ```

   If there is a need to configure a second VLAN, with for example, VLAN ID 193, on the same interface, eth0, add a new file with the name `eth0.193` with the VLAN configuration details.

3. Restart the networking service in order for the changes to take effect. As root issue the following command:

   ```
   ~]# systemctl network restart
   ```

### 7.3. Configure 802.1Q VLAN Tagging Using ip Commands

To create an 802.1Q VLAN interface on Ethernet interface eth0, with name VLAN8 and ID 8, issue a command as root as follows:

```
~]$ ip link add link eth0 name eth0.8 type vlan id 8
```

To view the VLAN, issue the following command:

```
~]$ ip link
eth0.8@eth0: <ROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT
```

To remove the VLAN, issue a command as root as follows:

```
~]# ip link delete eth0.8
```
VLAN interfaces created using \texttt{ip} commands at the command prompt will be lost if the system is shutdown or restarted. To configure VLAN interfaces to be persistent after a system restart, use \texttt{ifcfg} files. See Section 7.2.1, “Setting Up 802.1Q VLAN Tagging Using ifcfg Files”.

### 7.4. Configure 802.1Q VLAN Tagging Using the NetworkManager Command Line Tool, \texttt{nmcli}

To create an 802.1Q VLAN connection profile for Ethernet interface eth0, with name VLAN10 and ID 10, issue a command as follows:

```
$ nmcli con add type vlan ifname VLAN10 dev eth0 id 10
Connection 'vlan-VLAN10' (37750b4a-8ef5-40e6-be9b-4fb21a4b6d17) successfully added.
```

Further options for the VLAN command are listed in the VLAN section of the \texttt{nmcli(1)} man page. In the man pages the device on which the VLAN is created is referred to as the parent device. In the example above the device was specified by its interface name, eth0, it can also be specified by the connections UUID or MAC address.

To create an 802.1Q VLAN connection profile with ingress priority mapping on Ethernet interface eth1, with name VLAN1 and ID 13, issue a command as follows:

```
$ nmcli con add type vlan con-name VLAN1 dev eth2 id 13 ingress "2:3,3:5"
```

To view all the parameters associated with the VLAN created above, issue a command as follows:

```
$ nmcli connection show configured vlan-VLAN10
```

To change the MTU, issue a command as follows:

```
$ nmcli connection modify vlan-VLAN10 802.mtu 1496
```

The MTU setting determines the maximum size of the network layer packet. The maximum size of the payload the link layer frame can carry in turn limits the network layer MTU. For standard Ethernet frames this means an MTU of 1500 bytes. It should not be necessary to change the MTU when setting up a VLAN as the link layer header is increased in size by 4 bytes to accommodate the 802.1Q tag.

At time of writing, \texttt{connection.interface-name} and \texttt{vlan.interface-name} have to be the same (if they are set). They must therefore be changed simultaneously using \texttt{nmcli}’s interactive mode. To change a VLAN connections name, issue commands as follows:

```
$ nmcli con edit vlan-VLAN10
nmcli> set vlan.interface-name superVLAN
nmcli> nmcli con edit vlan-VLAN10
nmcli> save
nmcli> quit
```

The \texttt{nmcli} utility can be used to set and clear \texttt{ioctl} flags which change the way the 802.1Q code functions. The following VLAN flags are supported by NetworkManager:

- 0x01 - reordering of output packet headers
- 0x02 - use GVRP protocol
- 0x04 - loose binding of the interface and its master
The state of the VLAN is synchronized to the state of the parent or master interface (the interface or device on which the VLAN is created). If the parent interface is set to the “down” administrative state then all associated VLANs are set down and all routes are flushed from the routing table. Flag 0x04 enables a loose binding mode, in which only the operational state is passed from the parent to the associated VLANs, but the VLAN device state is not changed.

To set a VLAN flag, issue a command as follows:

```
~]$ nmcli connection modify vlan-VLAN10 vlan.flags 0x01
```

See Section 2.4, “Using the NetworkManager Command Line Tool, nmcli” for an introduction to nmcli.

### 7.5. Additional Resources

The following sources of information provide additional resources regarding Network Teaming.

#### 7.5.1. Installed Documentation

- **ip-link(8) man page** — Describes the ip utility’s network device configuration commands.
- **nmcli(1) man page** — Describes NetworkManager’s command-line tool.
- **nmcli-examples(5) man page** — Gives examples of nmcli commands.
- **nm-settings(5) man page** — Description of settings and parameters of NetworkManager connections.
Chapter 8.

**Consistent Network Device Naming**

Fedora 20 provides methods for consistent and predictable network device naming for network interfaces. These features change the name of network interfaces on a system in order to make locating and differentiating the interfaces easier.

Traditionally, network interfaces in Linux are enumerated as `eth[0123...]`, but these names do not necessarily correspond to actual labels on the chassis. Modern server platforms with multiple network adapters can encounter non-deterministic and counter-intuitive naming of these interfaces. This affects both network adapters embedded on the motherboard (Lan-on-Motherboard, or LOM) and add-in (single and multiport) adapters.

In Fedora 20, *systemd* and *udevd* support a number of different naming schemes. The default is to assign fixed names based on firmware, topology, and location information. This has the advantage that the names are fully automatic, fully predictable, that they stay fixed even if hardware is added or removed (no re-enumeration takes place), and that broken hardware can be replaced seamlessly. The disadvantage is that they are sometimes harder to read than the eth0 or wlan0 everybody is used to. For example: `enp5s0`.

The following different naming schemes for network interfaces are now supported by *udevd* natively:

1. Names incorporating Firmware or BIOS provided index numbers for on-board devices (example: `eno1`)

2. Names incorporating Firmware or BIOS provided PCI Express hotplug slot index numbers (example: `ens1`)

3. Names incorporating physical location of the connector of the hardware (example: `enp2s0`)

4. Names incorporating the interface's MAC address (example: `enx78e7d1ea46da`)

5. The traditional unpredictable kernel-native ethX naming (example: `eth0`)

By default, *systemd* will name interfaces using the following policy to apply the schemes listed above:

1. Use scheme 1 if that information from the firmware is applicable and available, falling back to scheme 2;

2. Use scheme 2 if that information from the firmware is applicable and available, falling back to scheme 3;

3. Use scheme 3 if applicable, falling back to scheme 5 in all other cases;

4. Scheme 4 is not used by default, but is available if the user chooses;

5. Scheme 5 is used if all other methods fail.

This policy, the procedure outlined above, is the default. If the system has `BIOSDEVNAME` enabled, it will take precedence. If the user has added *udevd* rules which change the name of the kernel devices, those rules will take precedence too.

**8.1. Understanding the Predictable Network Interface Device Names**

The names have two character prefixes based on the type of interface:

1. `en` for Ethernet,

2. `wl` for wireless LAN (WLAN),
3. **ww** for wireless wide area network (WWAN).

The names have the following types:

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>o&lt;index&gt;</td>
<td>on-board device index number</td>
</tr>
<tr>
<td>s&lt;slot&gt;[f&lt;function&gt;][d&lt;dev_id&gt;]</td>
<td>hotplug slot index number</td>
</tr>
<tr>
<td>x&lt;MAC&gt;</td>
<td>MAC address</td>
</tr>
<tr>
<td>p&lt;bus&gt;s&lt;slot&gt;[f&lt;function&gt;][d&lt;dev_id&gt;]</td>
<td>PCI geographical location</td>
</tr>
<tr>
<td>p&lt;bus&gt;s&lt;slot&gt;[f&lt;function&gt;][u&lt;port&gt;][..][c&lt;config&gt;][i&lt;interface&gt;]</td>
<td>USB port number chain</td>
</tr>
</tbody>
</table>

- All multi-function PCI devices will carry the [f<function>] number in the device name, including the function 0 device.
- For USB devices the full chain of port numbers of hubs is composed. If the name gets longer than the maximum number of 15 characters, the name is not exported.
- The USB configuration descriptors == 1 and USB configuration descriptors == 0 values are suppressed.

### 8.2. Naming Scheme for Network Devices Available for Linux on System z

Use the bus-ID to create predictable devices names for network interfaces on Linux on System z instances. The bus-ID identifies a device in the s390 channel subsystem. A bus ID identifies the device within the scope of a Linux instance. For a CCW device, the bus ID is the device’s device number with a leading 0.n, where n is the subchannel set ID. For example, 0.1.0ab1.

Network interfaces of device type Ethernet are named as follows:

```
enccw0.0.1234
```

Use the `znetconf -c` command or the `lsconf -a` command to display available network devices and their bus-IDs.

CTC network devices of device type SLIP are named as follows:

```
slccw0.0.1234
```

Use the `znetconf -c` command or the `lsconf -a` command to display available network devices and their bus-IDs.

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enccwbus-ID</td>
<td>device type Ethernet</td>
</tr>
<tr>
<td>slccwbus-ID</td>
<td>CTC network devices of device type SLIP</td>
</tr>
</tbody>
</table>
8.3. Consistent Network Device Naming Using BIOSDEVNAME

This feature, implemented via the biosdevname program, will change the name of all embedded network interfaces, PCI card network interfaces, and virtual function network interfaces from the existing `eth[0123...]` to the new naming convention as shown in Table 8.3, “The BIOSDEVNAME naming convention”.

Table 8.3. The BIOSDEVNAME naming convention

<table>
<thead>
<tr>
<th>Device</th>
<th>Old Name</th>
<th>New Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedded network interface (LOM)</td>
<td><code>eth[0123...]</code></td>
<td><code>em[1234...]</code>¹</td>
</tr>
<tr>
<td>PCI card network interface</td>
<td><code>eth[0123...]</code></td>
<td><code>p&lt;slot&gt;p&lt;ethernet port&gt;</code>²</td>
</tr>
<tr>
<td>Virtual function</td>
<td><code>eth[0123...]</code></td>
<td><code>p&lt;slot&gt;p&lt;ethernet port&gt;_&lt;virtual interface&gt;</code>³</td>
</tr>
</tbody>
</table>

¹ New enumeration starts at 1.
² For example: p3p4
³ For example: p3p4_1

8.3.1. System Requirements

The biosdevname program uses information from the system's BIOS, specifically the type 9 (System Slot) and type 41 (Onboard Devices Extended Information) fields contained within the SMBIOS. If the system's BIOS does not have SMBIOS version 2.6 or higher and this data, the new naming convention will not be used. Most older hardware does not support this feature because of a lack of BIOSes with the correct SMBIOS version and field information. For BIOS or SMBIOS version information, contact your hardware vendor.

For this feature to take effect, the biosdevname package must also be installed. To install it, issue the following command as root:

```
-]# yum install biosdevname
```

8.3.2. Enabling and Disabling the Feature

To disable this feature, pass the following option on the boot command line, both during and after installation:

```
biosdevname=0
```

To enable this feature, pass the following option on the boot command line, both during and after installation:

```
biosdevname=1
```

Unless the system meets the minimum requirements, this option will be ignored and the system will boot with the traditional network interface name format.

If the biosdevname install option is specified, it must remain as a boot option for the lifetime of the system.
8.4. Notes for Administrators

Many system customization files can include network interface names, and thus will require updates if moving a system from the old convention to the new convention. If you use the new naming convention, you will also need to update network interface names in areas such as custom iptables rules, scripts altering irqbalance, and other similar configuration files. Also, enabling this change for installation will require modification to existing kickstart files that use device names via the `ksdevice` parameter; these kickstart files will need to be updated to use the network device’s MAC address or the network device’s new name.

8.5. Controlling the Selection of Network Device Names

Device naming can be controlled in the following manner:

By identifying the network interface device

Setting the MAC address in an `ifcfg` file using the `HWADDR` directive enables it to be identified by `udev`. The name will be taken from the string given by the `DEVICE` directive, which by convention is the same as the `ifcfg` suffix. For example, `ifcfg-eth0`.

By turning on or off `biosdevname`

The name provided by `biosdevname` will be used (if `biosdevname` can determine one).

By turning on or off the `systemd-udev` naming scheme

The name provided by `systemd-udev` will be used (if `systemd-udev` can determine one).

8.6. Understanding the Device Renaming Procedure

The device name procedure in detail is as follows:

1. A rule in `/usr/lib/udev/rules.d/60-net.rules` instructs the `udev` helper utility, `/lib/udev/rename_device`, to look into all `/etc/sysconfig/network-scripts/ifcfg-suffix` files. If it finds an `ifcfg` file with a `HWADDR` entry matching the MAC address of an interface it renames the interface to the name given in the `ifcfg` file by the `DEVICE` directive.

2. A rule in `/usr/lib/udev/rules.d/71-biosdevname.rules` instructs `biosdevname` to rename the interface according to its naming policy, provided that it was not renamed in a previous step, `biosdevname` is installed, and `biosdevname=0` was not given as a kernel command on the boot command line.

3. A rule in `/lib/udev/rules.d/75-net-description.rules` instructs `udev` to fill in `ID_NET_NAME_ONBOARD`, `ID_NET_NAME_SLOT`, `ID_NET_NAME_PATH`, `ID_NET_NAME_MAC` by examining the network interface device. Note, that some variables can be empty.

4. A rule in `/usr/lib/udev/rules.d/80-net-name-slot.rules` instructs `udev` to rename the interface, provided that it was not renamed in a previous step, and the kernel parameter `net.ifnames=0` was not given, according to the following priority: `ID_NET_NAME_ONBOARD`, `ID_NET_NAME_SLOT`, `ID_NET_NAME_PATH`. It falls through to the next in the list, if one is unset. If none of these are set, then the interface will not be renamed.

8.7. Disabling Consistent Network Device Naming

To disable consistent network device naming, choose from one of the following:
• Disable the assignment of fixed names, so that the unpredictable kernel names are used again, by
masking udev’s rule file for the default policy. This “masking” can be done by making a symbolic link
to /dev/null. As root, issue a command as follows:

```bash
~]# ln -s /dev/null /etc/udev/rules.d/80-net-name-slot.rules
```

• Create your own manual naming scheme, for example by naming your interfaces “internet0”,
“dmz0” or “lan0”. To do that create your own udev rules file and set the NAME property for the
devices. Make sure to order it before the default policy file, for example by naming it `/etc/udev/
rules.d/70-my-net-names.rules`.

• Alter the default policy file to pick a different naming scheme, for example to name all interfaces
after their MAC address by default. As root copy the default policy file as follows:

```bash
~]# cp /usr/lib/udev/rules.d/80-net-name-slot.rules /etc/udev/rules.d/80-net-name-
slot.rules
```

Edit the file in the `/etc/udev/rules.d/` directory and change the lines as necessary.

• Add the following line to the `/etc/grub.conf` file:

```bash
net.ifnames=0
```

or pass it to the kernel at boot time using the GRUB2 command line interface. For more information
about GRUB2 see `Fedora 20 System Administrator's Guide`

8.8. Additional Resources

The following sources of information provide additional resources regarding Network Teaming.

8.8.1. Installed Documentation

• `udev(7)` man page — Describes the Linux dynamic device management daemon, udevd.

• `systemd(1)` man page — Describes systemd system and service manager.

• `biosdevname(1)` man page — Describes the utility for obtaining the BIOS-given name of a device.

---

Part II. Servers

This part discusses how to set up servers normally required for networking.
DHCP Servers

Dynamic Host Configuration Protocol (DHCP) is a network protocol that automatically assigns TCP/IP information to client machines. Each DHCP client connects to the centrally located DHCP server, which returns the network configuration (including the IP address, gateway, and DNS servers) of that client.

9.1. Why Use DHCP?

DHCP is useful for automatic configuration of client network interfaces. When configuring the client system, you can choose DHCP instead of specifying an IP address, netmask, gateway, or DNS servers. The client retrieves this information from the DHCP server. DHCP is also useful if you want to change the IP addresses of a large number of systems. Instead of reconfiguring all the systems, you can just edit one configuration file on the server for the new set of IP addresses. If the DNS servers for an organization changes, the changes happen on the DHCP server, not on the DHCP clients. When you restart the network or reboot the clients, the changes go into effect.

If an organization has a functional DHCP server correctly connected to a network, laptops and other mobile computer users can move these devices from office to office.

9.2. Configuring a DHCP Server

The dhcp package contains an Internet Systems Consortium (ISC) DHCP server. First, install the package as the superuser:

```
~\# yum install dhcp
```

Installing the dhcp package creates a file, `/etc/dhcp/dhcpd.conf`, which is merely an empty configuration file. As root, issue the following command:

```
~\# cat /etc/dhcp/dhcpd.conf
```

```
# DHCP Server Configuration file.
# see /usr/share/doc/dhcp/dhcpd.conf.example
# see dhcpd.conf(5) man page
```

The example configuration file can be found at `/usr/share/doc/dhcp/dhcpd.conf.example`. You should use this file to help you configure `/etc/dhcp/dhcpd.conf`, which is explained in detail below.

DHCP also uses the file `/var/lib/dhcpd/dhcpd.leases` to store the client lease database. Refer to Section 9.2.2, “Lease Database” for more information.

9.2.1. Configuration File

The first step in configuring a DHCP server is to create the configuration file that stores the network information for the clients. Use this file to declare options and global options for client systems.

The configuration file can contain extra tabs or blank lines for easier formatting. Keywords are case-insensitive and lines beginning with a hash sign (#) are considered comments.

There are two types of statements in the configuration file:
Chapter 9. DHCP Servers

- Parameters — State how to perform a task, whether to perform a task, or what network configuration options to send to the client.

- Declarations — Describe the topology of the network, describe the clients, provide addresses for the clients, or apply a group of parameters to a group of declarations.

The parameters that start with the keyword option are referred to as options. These options control DHCP options; whereas, parameters configure values that are not optional or control how the DHCP server behaves.

Parameters (including options) declared before a section enclosed in curly brackets ({} ) are considered global parameters. Global parameters apply to all the sections below it.

**Restart the DHCP Daemon for the Changes to Take Effect**

If the configuration file is changed, the changes do not take effect until the DHCP daemon is restarted with the command `systemctl restart dhcpd`.

**Use the omshell Command**

Instead of changing a DHCP configuration file and restarting the service each time, using the omshell command provides an interactive way to connect to, query, and change the configuration of a DHCP server. By using omshell, all changes can be made while the server is running. For more information on omshell, see the omshell man page.

In Example 9.1, “Subnet Declaration”, the routers, subnet-mask, domain-search, domain-name-servers, and time-offset options are used for any host statements declared below it.

For every subnet which will be served, and for every subnet to which the DHCP server is connected, there must be one subnet declaration, which tells the DHCP daemon how to recognize that an address is on that subnet. A subnet declaration is required for each subnet even if no addresses will be dynamically allocated to that subnet.

In this example, there are global options for every DHCP client in the subnet and a range declared. Clients are assigned an IP address within the range.

**Example 9.1. Subnet Declaration**

```plaintext
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;
}
```
To configure a DHCP server that leases a dynamic IP address to a system within a subnet, modify **Example 9.2, “Range Parameter”** with your values. It declares a default lease time, maximum lease time, and network configuration values for the clients. This example assigns IP addresses in the range 192.168.1.10 and 192.168.1.100 to client systems.

**Example 9.2. Range Parameter**

```plaintext
default-lease-time 600;
max-lease-time 7200;
option subnet-mask 255.255.255.0;
option broadcast-address 192.168.1.255;
option domain-name-servers 192.168.1.1, 192.168.1.2;
subnet 192.168.1.0 netmask 255.255.255.0 {
    range 192.168.1.10 192.168.1.100;
}
```

To assign an IP address to a client based on the MAC address of the network interface card, use the **hardware ethernet** parameter within a **host** declaration. As demonstrated in **Example 9.3, “Static IP Address Using DHCP”**, the **host apex** declaration specifies that the network interface card with the MAC address 00:A0:78:8E:9E:AA always receives the IP address 192.168.1.4.

Note that you can also use the optional parameter **host-name** to assign a host name to the client.

**Example 9.3. Static IP Address Using DHCP**

```plaintext
host apex {
    option host-name "apex.example.com";
    hardware ethernet 00:A0:78:8E:9E:AA;
    fixed-address 192.168.1.4;
}
```

All subnets that share the same physical network should be declared within a **shared-network** declaration as shown in **Example 9.4, “Shared-network Declaration”**. Parameters within the **shared-network**, but outside the enclosed **subnet** declarations, are considered to be global parameters. The name of the **shared-network** must be a descriptive title for the network, such as using the title ‘test-lab’ to describe all the subnets in a test lab environment.

**Example 9.4. Shared-network Declaration**

```plaintext
shared-network name {
    option domain-name-servers ns1.redhat.com, ns2.redhat.com;
    option routers 192.168.0.254;
    subnet 192.168.1.0 netmask 255.255.252.0 {
        range 192.168.1.1 192.168.1.254;
    }
    subnet 192.168.2.0 netmask 255.255.252.0 {
        range 192.168.2.1 192.168.2.254;
    }
}
```
As demonstrated in Example 9.5, “Group Declaration”, the group declaration is used to apply global parameters to a group of declarations. For example, shared networks, subnets, and hosts can be grouped.

**Example 9.5. Group Declaration**

```plaintext
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;
    }
}
```

**Using the Example Configuration File**

The example configuration file provided can be used as a starting point and custom configuration options can be added to it. To copy it to the proper location, use the following command:

```bash
 cp /usr/share/doc/dhcp/dhcpd.conf.example /etc/dhcp/dhcpd.conf
```

For a complete list of option statements and what they do, refer to the `dhcp-options` man page.

### 9.2.2. Lease Database

On the DHCP server, the file `/var/lib/dhcpd/dhcpd.leases` stores the DHCP client lease database. Do not change this file. DHCP lease information for each recently assigned IP address is automatically stored in the lease database. The information includes the length of the lease, to whom the IP address has been assigned, the start and end dates for the lease, and the MAC address of the network interface card that was used to retrieve the lease.

All times in the lease database are in Coordinated Universal Time (UTC), not local time.

The lease database is recreated from time to time so that it is not too large. First, all known leases are saved in a temporary lease database. The `dhcpd.leases` file is renamed `dhcpd.leases~` and the temporary lease database is written to `dhcpd.leases`.

The DHCP daemon could be killed or the system could crash after the lease database has been renamed to the backup file but before the new file has been written. If this happens, the `dhcpd.leases` file does not exist, but it is required to start the service. Do not create a new lease file. If you do, all old leases are lost which causes many problems. The correct solution is to rename the `dhcpd.leases~` backup file to `dhcpd.leases` and then start the daemon.
9.2.3. Starting and Stopping the Server

**Starting the DHCP Server for the First Time**

When the DHCP server is started for the first time, it fails unless the `dhcpd.leases` file exists. Use the command `touch /var/lib/dhcpd/dhcpd.leases` to create the file if it does not exist.

If the same server is also running BIND as a DNS server, this step is not necessary, as starting the `named` service automatically checks for a `dhcpd.leases` file.

To start the DHCP service, use the following command:

```
systemctl start dhcpd.service
```

To stop the DHCP server, type:

```
systemctl stop dhcpd.service
```

By default, the DHCP service does not start at boot time. To configure the daemon to start automatically at boot time, run:

```
systemctl enable dhcpd.service
```

If more than one network interface is attached to the system, but the DHCP server should only be started on one of the interfaces, configure the DHCP server to start only on that device. In `/etc/sysconfig/dhcpd`, add the name of the interface to the list of `DHCPDARGS`:

```
# Command line options here
DHCPDARGS=eth0
```

This is useful for a firewall machine with two network cards. One network card can be configured as a DHCP client to retrieve an IP address to the Internet. The other network card can be used as a DHCP server for the internal network behind the firewall. Specifying only the network card connected to the internal network makes the system more secure because users can not connect to the daemon via the Internet.

Other command line options that can be specified in `/etc/sysconfig/dhcpd` include:

- `-p portnum` — Specifies the UDP port number on which `dhcpd` should listen. The default is port 67. The DHCP server transmits responses to the DHCP clients at a port number one greater than the UDP port specified. For example, if the default port 67 is used, the server listens on port 67 for requests and responses to the client on port 68. If a port is specified here and the DHCP relay agent is used, the same port on which the DHCP relay agent should listen must be specified. Refer to `Section 9.2.4, “DHCP Relay Agent”` for details.

- `-f` — Runs the daemon as a foreground process. This is mostly used for debugging.
• **-d** — Logs the DHCP server daemon to the standard error descriptor. This is mostly used for debugging. If this is not specified, the log is written to `/var/log/messages`.

• **-cf filename** — Specifies the location of the configuration file. The default location is `/etc/dhcp/dhcpd.conf`.

• **-lf filename** — Specifies the location of the lease database file. If a lease database file already exists, it is very important that the same file be used every time the DHCP server is started. It is strongly recommended that this option only be used for debugging purposes on non-production machines. The default location is `/var/lib/dhcpd/dhcpd.leases`.

• **-q** — Do not print the entire copyright message when starting the daemon.

### 9.2.4. DHCP Relay Agent

The DHCP Relay Agent (`dhcrelay`) allows for the relay of DHCP and BOOTP requests from a subnet with no DHCP server on it to one or more DHCP servers on other subnets.

When a DHCP client requests information, the DHCP Relay Agent forwards the request to the list of DHCP servers specified when the DHCP Relay Agent is started. When a DHCP server returns a reply, the reply is broadcast or unicast on the network that sent the original request.

The DHCP Relay Agent listens for DHCP requests on all interfaces unless the interfaces are specified in `/etc/sysconfig/dhcrelay` with the `INTERFACES` directive.

To start the DHCP Relay Agent, use the following command:

```
systemctl start dhcrelay.service
```

### 9.3. Configuring a Multihomed DHCP Server

A multihomed DHCP server serves multiple networks, that is, multiple subnets. The examples in these sections detail how to configure a DHCP server to serve multiple networks, select which network interfaces to listen on, and how to define network settings for systems that move networks.

Before making any changes, back up the existing `/etc/sysconfig/dhcpd` and `/etc/dhcp/dhcpd.conf` files.

The DHCP daemon listens on all network interfaces unless otherwise specified. Use the `/etc/sysconfig/dhcpd` file to specify which network interfaces the DHCP daemon listens on. The following `/etc/sysconfig/dhcpd` example specifies that the DHCP daemon listens on the `eth0` and `eth1` interfaces:

```
DHCPDARGS="eth0 eth1";
```

If a system has three network interfaces cards -- `eth0`, `eth1`, and `eth2` -- and it is only desired that the DHCP daemon listens on `eth0`, then only specify `eth0` in `/etc/sysconfig/dhcpd`:

```
DHCPDARGS="eth0";
```

The following is a basic `/etc/dhcp/dhcpd.conf` file, for a server that has two network interfaces, `eth0` in a 10.0.0.0/24 network, and `eth1` in a 172.16.0.0/24 network. Multiple `subnet` declarations allow different settings to be defined for multiple networks:
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;
}

A subnet declaration is required for every network your DHCP server is serving. Multiple subnets require multiple subnet declarations. If the DHCP server does not have a network interface in a range of a subnet declaration, the DHCP server does not serve that network.

If there is only one subnet declaration, and no network interfaces are in the range of that subnet, the DHCP daemon fails to start, and an error such as the following is logged to /var/log/messages:

dhcpd: No subnet declaration for eth0 (0.0.0.0).
dhcpd: ** Ignoring requests on eth0. If this is not what
dhcpd: you want, please write a subnet declaration
dhcpd: in your dhcpd.conf file for the network segment
dhcpd: to which interface eth1 is attached. **
dhcpd:
dhcpd: Not configured to listen on any interfaces!

option subnet-mask 255.255.255.0;

The option subnet-mask option defines a subnet mask, and overrides the netmask value in the subnet declaration. In simple cases, the subnet and netmask values are the same.

option routers 10.0.0.1;

The option routers option defines the default gateway for the subnet. This is required for systems to reach internal networks on a different subnet, as well as external networks.

range 10.0.0.5 10.0.0.15;

The range option specifies the pool of available IP addresses. Systems are assigned an address from the range of specified IP addresses.

For further information, see the dhcpd.conf(5) man page.

9.3.1. Host Configuration

Before making any changes, back up the existing /etc/sysconfig/dhcpd and /etc/dhcp/dhcpd.conf files.

Configuring a Single System for Multiple Networks

The following /etc/dhcp/dhcpd.conf example creates two subnets, and configures an IP address for the same system, depending on which network it connects to:

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
The host declaration defines specific parameters for a single system, such as an IP address. To configure specific parameters for multiple hosts, use multiple host declarations.

Most DHCP clients ignore the name in host declarations, and as such, this name can be anything, as long as it is unique to other host declarations. To configure the same system for multiple networks, use a different name for each host declaration, otherwise the DHCP daemon fails to start. Systems are identified by the hardware ethernet option, not the name in the host declaration.

hardware ethernet 00:1A:6B:6A:2E:0B;
The hardware ethernet option identifies the system. To find this address, run the ip link command.

fixed-address 10.0.0.20;
The fixed-address option assigns a valid IP address to the system specified by the hardware ethernet option. This address must be outside the IP address pool specified with the range option.

If option statements do not end with a semicolon, the DHCP daemon fails to start, and an error such as the following is logged to /var/log/messages:

/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

Configuring Systems with Multiple Network Interfaces
The following host declarations configure a single system, which has multiple network interfaces, so that each interface receives the same IP address. This configuration will not work if both network interfaces are connected to the same network at the same time:

host interface0 {
  hardware ethernet 00:1a:6b:6a:2e:0b;
  fixed-address 10.0.0.18;
}
For this example, `interface0` is the first network interface, and `interface1` is the second interface. The different `hardware ethernet` options identify each interface.

If such a system connects to another network, add more `host` declarations, remembering to:

- assign a valid `fixed-address` for the network the host is connecting to.
- make the name in the `host` declaration unique.

When a name given in a `host` declaration is not unique, the DHCP daemon fails to start, and an error such as the following is logged to `/var/log/messages`:

```
dhcpd: /etc/dhcp/dhcpd.conf line 31: host interface0: already exists
```

This error was caused by having multiple `host interface0` declarations defined in `/etc/dhcp/dhcpd.conf`.

### 9.4. DHCP for IPv6 (DHCPv6)

The ISC DHCP includes support for IPv6 (DHCPv6) since the 4.x release with a DHCPv6 server, client, and relay agent functionality. The agents support both IPv4 and IPv6, however the agents can only manage one protocol at a time; for dual support they must be started separately for IPv4 and IPv6.

For example, configure both DHCPv4 and DHCPv6 by editing their respective configuration files `/etc/dhcp/dhcpd.conf` and `/etc/dhcp/dhcpd6.conf` and then issue the following commands:

```
[-]# systemctl start dhcpd
[-]# systemctl start dhcpd6
```

The DHCPv6 server configuration file can be found at `/etc/dhcp/dhcpd6.conf`.

The example server configuration file can be found at `/usr/share/doc/dhcp/dhcpd6.conf.example`.

To start the DHCPv6 service, use the following command:

```
systemctl start dhcpd6
```

A simple DHCPv6 server configuration file can look like this:

```
subnet6 2001:db8:0:1::/64 {
    range6 2001:db8:0:1::129 2001:db8:0:1::254;
    option dhcp6.name-servers fec0:0:1::1;
    option dhcp6.domain-search "domain.example";
}
```

### 9.5. Additional Resources
For additional information, see *The DHCP Handbook; Ralph Droms and Ted Lemon;* 2003 or the following resources.

### 9.5.1. Installed Documentation

- **`dhcpd(8)`** man page — Describes how the DHCP daemon works.

- **`dhcpd.conf(5)`** man page — Explains how to configure the DHCP configuration file; includes some examples.

- **`dhcpd.leases(5)`** man page — Describes a persistent database of leases.

- **`dhcplib.so(5)`** man page — Explains the syntax for declaring DHCP options in `dhcplib.conf`; includes some examples.

- **`dhcrelay(8)`** man page — Explains the DHCP Relay Agent and its configuration options.

- `/usr/share/doc/dhcp/` — Contains example configuration files.

DNS Servers

DNS (Domain Name System), is a network system that associates host names with their respective IP addresses. For users, this has the advantage that they can refer to machines on the network by names that are usually easier to remember than the numerical network addresses. For system administrators, using a DNS server, also known as a nameserver, enables changing the IP address for a host without ever affecting the name-based queries.

10.1. Introduction to DNS

DNS is usually implemented using one or more centralized servers that are authoritative for certain domains. When a client host requests information from a nameserver, it usually connects to port 53. The nameserver then attempts to resolve the name requested. If it does not have an authoritative answer, or does not already have the answer cached from an earlier query, it queries other nameservers, called root nameservers, to determine which nameservers are authoritative for the name in question, and then queries them to get the requested name.

10.1.1. Nameserver Zones

In a DNS server such as BIND (Berkeley Internet Name Domain), all information is stored in basic data elements called resource records (RR). The resource record is usually a fully qualified domain name (FQDN) of a host, and is broken down into multiple sections organized into a tree-like hierarchy. This hierarchy consists of a main trunk, primary branches, secondary branches, and so on.

Example 10.1. A simple resource record

| bob.sales.example.com |

Each level of the hierarchy is divided by a period (that is, .). In Example 10.1, “A simple resource record”, com defines the top-level domain, example its subdomain, and sales the subdomain of example. In this case, bob identifies a resource record that is part of the sales.example.com domain. With the exception of the part furthest to the left (that is, bob), each of these sections is called a zone and defines a specific namespace.

Zones are defined on authoritative nameservers through the use of zone files, which contain definitions of the resource records in each zone. Zone files are stored on primary nameservers (also called master nameservers), where changes are made to the files, and secondary nameservers (also called slave nameservers), which receive zone definitions from the primary nameservers. Both primary and secondary nameservers are authoritative for the zone and look the same to clients. Depending on the configuration, any nameserver can also serve as a primary or secondary server for multiple zones at the same time.

10.1.2. Nameserver Types

There are two nameserver configuration types:

- authoritative

  Authoritative nameservers answer to resource records that are part of their zones only. This category includes both primary (master) and secondary (slave) nameservers.
Recursive nameservers offer resolution services, but they are not authoritative for any zone. Answers for all resolutions are cached in a memory for a fixed period of time, which is specified by the retrieved resource record.

Although a nameserver can be both authoritative and recursive at the same time, it is recommended not to combine the configuration types. To be able to perform their work, authoritative servers should be available to all clients all the time. On the other hand, since the recursive lookup takes far more time than authoritative responses, recursive servers should be available to a restricted number of clients only, otherwise they are prone to distributed denial of service (DDoS) attacks.

### 10.1.3. BIND as a Nameserver

BIND consists of a set of DNS-related programs. It contains a nameserver called named, an administration utility called rndc, and a debugging tool called dig. See *Fedora 20 System Administrator's Guide* for more information on how to run a service in Fedora.

### 10.2. BIND

This chapter covers BIND (Berkeley Internet Name Domain), the DNS server included in Fedora. It focuses on the structure of its configuration files, and describes how to administer it both locally and remotely.

#### 10.2.1. Empty Zones

BIND configures a number of “empty zones” to prevent recursive servers from sending unnecessary queries to Internet servers that cannot handle them (thus creating delays and SERVFAIL responses to clients who query for them). These empty zones ensure that immediate and authoritative NXDOMAIN responses are returned instead. The configuration option `empty-zones-enable` controls whether or not empty zones are created, whilst the option `disable-empty-zone` can be used in addition to disable one or more empty zones from the list of default prefixes that would be used.

The number of empty zones created for RFC 1918\(^1\) prefixes has been increased, and users of BIND 9.9 and above will see the RFC 1918\(^2\) empty zones both when `empty-zones-enable` is unspecified (defaults to yes), and when it is explicitly set to yes.

#### 10.2.2. Configuring the named Service

When the named service is started, it reads the configuration from the files as described in *Table 10.1, “The named service configuration files”*:

<table>
<thead>
<tr>
<th>Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/etc/named.conf</code></td>
<td>The main configuration file.</td>
</tr>
</tbody>
</table>

---

\(^1\) [http://www.rfc-editor.org/info/rfc1918](http://www.rfc-editor.org/info/rfc1918)

\(^2\) [http://www.rfc-editor.org/info/rfc1918](http://www.rfc-editor.org/info/rfc1918)
The configuration file consists of a collection of statements with nested options surrounded by opening and closing curly brackets (that is, `{ and `}`). Note that when editing the file, you have to be careful not to make any syntax error, otherwise the named service will not start. A typical `/etc/named.conf` file is organized as follows:

```
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;
}
```

**Running BIND in a chroot environment**

If you have installed the `bind-chroot` package, the BIND service will run in the `/var/named/chroot` environment. In that case, the initialization script will mount the above configuration files using the `mount --bind` command, so that you can manage the configuration outside this environment.

### 10.2.2.1. Installing BIND In A Chroot Environment

To install BIND to run in a chroot environment, issue the following commands as root:

```
~ ]# yum install bind-chroot
```

To enable the named-chroot service, first check if the named service is running by issuing the following command:

```
~ ]$ systemctl status named
```

If it is running, it must be disabled.

To disable named, issue the following commands as root:

```
~ ]# systemctl stop named

~ ]# systemctl disable named
```

Then, to enable the named-chroot service, issue the following commands as root:
Chapter 10. DNS Servers

108

```bash
~# systemctl enable named-chroot
```

```bash
~# systemctl start named-chroot
```

To check the status of the named-chroot service, issue the following command as root:

```bash
~# systemctl status named-chroot
```

10.2.2.2. Common Statement Types

The following types of statements are commonly used in `/etc/named.conf`:

**acl**

The `acl` (Access Control List) statement allows you to define groups of hosts, so that they can be permitted or denied access to the nameserver. It takes the following form:

```plaintext
acl acl-name {
    match-element;
    ...
};
```

The `acl-name` statement name is the name of the access control list, and the `match-element` option is usually an individual IP address (such as `10.0.1.1`) or a CIDR (Classless Inter-Domain Routing) network notation (for example, `10.0.1.0/24`). For a list of already defined keywords, see Table 10.2, “Predefined access control lists”.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>any</td>
<td>Matches every IP address.</td>
</tr>
<tr>
<td>localhost</td>
<td>Matches any IP address that is in use by the local system.</td>
</tr>
<tr>
<td>localnets</td>
<td>Matches any IP address on any network to which the local system is connected.</td>
</tr>
<tr>
<td>none</td>
<td>Does not match any IP address.</td>
</tr>
</tbody>
</table>

The `acl` statement can be especially useful with conjunction with other statements such as `options`. Example 10.2, “Using acl in conjunction with options” defines two access control lists, `black-hats` and `red-hats`, and adds `black-hats` on the blacklist while granting `red-hats` a normal access.

**Example 10.2. Using acl in conjunction with options**

```plaintext
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
The include statement allows you to include files in the /etc/named.conf, so that potentially sensitive data can be placed in a separate file with restricted permissions. It takes the following form:

```plaintext
include "file-name"
```

The file-name statement name is an absolute path to a file.

Example 10.3. Including a file to /etc/named.conf

```plaintext
include "/etc/named.rfc1912.zones";
```

options
The options statement allows you to define global server configuration options as well as to set defaults for other statements. It can be used to specify the location of the named working directory, the types of queries allowed, and much more. It takes the following form:

```plaintext
options {
  option;
  ...
};
```

For a list of frequently used option directives, see Table 10.3, "Commonly Used Configuration Options" below.

Table 10.3. Commonly Used Configuration Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow-query</td>
<td>Specifies which hosts are allowed to query the nameserver for authoritative resource records. It accepts an access control list, a collection of IP addresses, or networks in the CIDR notation. All hosts are allowed by default.</td>
</tr>
<tr>
<td>allow-query-cache</td>
<td>Specifies which hosts are allowed to query the nameserver for non-authoritative data such as recursive queries. Only localhost and localnets are allowed by default.</td>
</tr>
<tr>
<td>blackhole</td>
<td>Specifies which hosts are not allowed to query the nameserver. This option should be used when particular host or network floods the server with requests. The default option is none.</td>
</tr>
<tr>
<td>directory</td>
<td>Specifies a working directory for the named service. The default option is /var/named/.</td>
</tr>
<tr>
<td>disable-empty-zone</td>
<td>Used to disable one or more empty zones from the list of default prefixes that would be used. Can be specified in the options statement and also in view statements. It can be used multiple times.</td>
</tr>
<tr>
<td>dnssec-enable</td>
<td>Specifies whether to return DNSSEC related resource records. The default option is yes.</td>
</tr>
<tr>
<td>dnssec-validation</td>
<td>Specifies whether to prove that resource records are authentic via DNSSEC. The default option is yes.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>empty-zones-enable</td>
<td>Controls whether or not empty zones are created. Can be specified only in the options statement.</td>
</tr>
<tr>
<td>forwarders</td>
<td>Specifies a list of valid IP addresses for nameservers to which the requests should be forwarded for resolution.</td>
</tr>
<tr>
<td>forward</td>
<td>Specifies the behavior of the forwarders directive. It accepts the following options:</td>
</tr>
<tr>
<td></td>
<td>• first — The server will query the nameservers listed in the forwarders directive before attempting to resolve the name on its own.</td>
</tr>
<tr>
<td></td>
<td>• only — When unable to query the nameservers listed in the forwarders directive, the server will not attempt to resolve the name on its own.</td>
</tr>
<tr>
<td>listen-on</td>
<td>Specifies the IPv4 network interface on which to listen for queries. On a DNS server that also acts as a gateway, you can use this option to answer queries originating from a single network only. All IPv4 interfaces are used by default.</td>
</tr>
<tr>
<td>listen-on-v6</td>
<td>Specifies the IPv6 network interface on which to listen for queries. On a DNS server that also acts as a gateway, you can use this option to answer queries originating from a single network only. All IPv6 interfaces are used by default.</td>
</tr>
<tr>
<td>max-cache-size</td>
<td>Specifies the maximum amount of memory to be used for server caches. When the limit is reached, the server causes records to expire prematurely so that the limit is not exceeded. In a server with multiple views, the limit applies separately to the cache of each view. The default option is 32M.</td>
</tr>
<tr>
<td>notify</td>
<td>Specifies whether to notify the secondary nameservers when a zone is updated. It accepts the following options:</td>
</tr>
<tr>
<td></td>
<td>• yes — The server will notify all secondary nameservers.</td>
</tr>
<tr>
<td></td>
<td>• no — The server will not notify any secondary nameserver.</td>
</tr>
<tr>
<td></td>
<td>• master-only — The server will notify primary server for the zone only.</td>
</tr>
<tr>
<td></td>
<td>• explicit — The server will notify only the secondary servers that are specified in the also-notify list within a zone statement.</td>
</tr>
<tr>
<td>pid-file</td>
<td>Specifies the location of the process ID file created by the named service.</td>
</tr>
<tr>
<td>recursion</td>
<td>Specifies whether to act as a recursive server. The default option is yes.</td>
</tr>
<tr>
<td>statistics-file</td>
<td>Specifies an alternate location for statistics files. The /var/named/named.stats file is used by default.</td>
</tr>
</tbody>
</table>
Configuring the named Service

Note

The directory used by named for runtime data has been moved from the BIND default location, /var/run/named/, to a new location /run/named/. As a result, the PID file has been moved from the default location /var/run/named/named.pid to the new location /run/named/named.pid. In addition, the session-key file has been moved to /run/named/session.key. These locations need to be specified by statements in the options section. See Example 10.4, “Using the options statement”.

Restrict recursive servers to selected clients only

To prevent distributed denial of service (DDoS) attacks, it is recommended that you use the allow-query-cache option to restrict recursive DNS services for a particular subset of clients only.

Refer to the BIND 9 Administrator Reference Manual referenced in Section 10.2.8.1, “Installed Documentation”, and the named.conf manual page for a complete list of available options.

Example 10.4. Using the options statement

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

The zone statement allows you to define the characteristics of a zone, such as the location of its configuration file and zone-specific options, and can be used to override the global options statements. It takes the following form:

zone zone-name [zone-class] {
    option;
    ...  
};
The `zone-name` attribute is the name of the zone, `zone-class` is the optional class of the zone, and `option` is a `zone` statement option as described in Table 10.4, “Commonly Used Options in Zone Statements”.

The `zone-name` attribute is particularly important, as it is the default value assigned for the `$ORIGIN` directive used within the corresponding zone file located in the `/var/named/` directory. The named daemon appends the name of the zone to any non-fully qualified domain name listed in the zone file. For example, if a `zone` statement defines the namespace for `example.com`, use `example.com` as the `zone-name` so that it is placed at the end of hostnames within the `example.com` zone file.

For more information about zone files, refer to Section 10.2.3, “Editing Zone Files”.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow-query</td>
<td>Specifies which clients are allowed to request information about this zone. This option overrides global <code>allow-query</code> option. All query requests are allowed by default.</td>
</tr>
<tr>
<td>allow-transfer</td>
<td>Specifies which secondary servers are allowed to request a transfer of the zone’s information. All transfer requests are allowed by default.</td>
</tr>
<tr>
<td>allow-update</td>
<td>Specifies which hosts are allowed to dynamically update information in their zone. The default option is to deny all dynamic update requests. Note that you should be careful when allowing hosts to update information about their zone. Do not set IP addresses in this option unless the server is in the trusted network. Instead, use TSIG key as described in Section 10.2.6.3, “Transaction SIGnatures (TSIG)”.</td>
</tr>
<tr>
<td>file</td>
<td>Specifies the name of the file in the named working directory that contains the zone’s configuration data.</td>
</tr>
<tr>
<td>masters</td>
<td>Specifies from which IP addresses to request authoritative zone information. This option is used only if the zone is defined as <code>type slave</code>.</td>
</tr>
<tr>
<td>notify</td>
<td>Specifies whether to notify the secondary nameservers when a zone is updated. It accepts the following options:</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the zone type. It accepts the following options:</td>
</tr>
</tbody>
</table>

• `delegation-only` — Enforces the delegation status of infrastructure zones such as COM, NET, or ORG. Any answer that is received without an explicit or implicit delegation is treated as `NXDOMAIN`. This option is only applicable in TLDs or root zone files used in recursive or caching implementations.

• `forward` — Forwards all requests for information about this zone to other nameservers.
Configuring the named Service

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hint</td>
<td>A special type of zone used to point to the root nameservers which resolve queries when a zone is not otherwise known. No configuration beyond the default is necessary with a hint zone.</td>
</tr>
<tr>
<td>master</td>
<td>Designates the nameserver as authoritative for this zone. A zone should be set as the master if the zone’s configuration files reside on the system.</td>
</tr>
<tr>
<td>slave</td>
<td>Designates the nameserver as a slave server for this zone. Master server is specified in masters directive.</td>
</tr>
</tbody>
</table>

Most changes to the `/etc/named.conf` file of a primary or secondary nameserver involve adding, modifying, or deleting zone statements, and only a small subset of zone statement options is usually needed for a nameserver to work efficiently.

In Example 10.5, “A zone statement for a primary nameserver”, the zone is identified as `example.com`, the type is set to master, and the named service is instructed to read the `/var/named/example.com.zone` file. It also allows only a secondary nameserver (192.168.0.2) to transfer the zone.

Example 10.5. A zone statement for a primary nameserver

```plaintext
zone "example.com" IN {
  type master;
  file "example.com.zone";
  allow-transfer { 192.168.0.2; };
};
```

A secondary server’s zone statement is slightly different. The type is set to slave, and the masters directive is telling named the IP address of the master server.

In Example 10.6, “A zone statement for a secondary nameserver”, the named service is configured to query the primary server at the 192.168.0.1 IP address for information about the example.com zone. The received information is then saved to the `/var/named/slaves/example.com.zone` file. Note that you have to put all slave zones to /var/named/slaves directory, otherwise the service will fail to transfer the zone.

Example 10.6. A zone statement for a secondary nameserver

```plaintext
zone "example.com" {
  type slave;
  file "slaves/example.com.zone";
  masters { 192.168.0.1; };
};
```

10.2.2.3. Other Statement Types

The following types of statements are less commonly used in `/etc/named.conf`:

controls

The controls statement allows you to configure various security requirements necessary to use the rndc command to administer the named service.

Refer to Section 10.2.4, “Using the rndc Utility” for more information on the rndc utility and its usage.
key
The **key** statement allows you to define a particular key by name. Keys are used to authenticate various actions, such as secure updates or the use of the **rndc** command. Two options are used with **key**:

- **algorithm algorithm-name** — The type of algorithm to be used (for example, **hmac-md5**).
- **secret "key-value"** — The encrypted key.

Refer to **Section 10.2.4, “Using the rndc Utility”** for more information on the **rndc** utility and its usage.

logging
The **logging** statement allows you to use multiple types of logs, so called **channels**. By using the **channel** option within the statement, you can construct a customized type of log with its own file name (**file**), size limit (**size**), version number (**version**), and level of importance (**severity**). Once a customized channel is defined, a **category** option is used to categorize the channel and begin logging when the named service is restarted.

By default, named sends standard messages to the **rsyslog** daemon, which places them in **/var/log/messages**. Several standard channels are built into BIND with various severity levels, such as **default_syslog** (which handles informational logging messages) and **default_debug** (which specifically handles debugging messages). A default category, called **default**, uses the built-in channels to do normal logging without any special configuration.

Customizing the logging process can be a very detailed process and is beyond the scope of this chapter. For information on creating custom BIND logs, refer to the **BIND 9 Administrator Reference Manual** referenced in **Section 10.2.8.1, “Installed Documentation”**.

server
The **server** statement allows you to specify options that affect how the named service should respond to remote nameservers, especially with regard to notifications and zone transfers.

The **transfer-format** option controls the number of resource records that are sent with each message. It can be either **one-answer** (only one resource record), or **many-answers** (multiple resource records). Note that while the **many-answers** option is more efficient, it is not supported by older versions of BIND.

trusted-keys
The **trusted-keys** statement allows you to specify assorted public keys used for secure DNS (DNSSEC). Refer to **Section 10.2.6.4, “DNS Security Extensions (DNSSEC)”** for more information on this topic.

view
The **view** statement allows you to create special views depending upon which network the host querying the nameserver is on. This allows some hosts to receive one answer regarding a zone while other hosts receive totally different information. Alternatively, certain zones may only be made available to particular trusted hosts while non-trusted hosts can only make queries for other zones.

Multiple views can be used as long as their names are unique. The **match-clients** option allows you to specify the IP addresses that apply to a particular view. If the **options** statement is used within a view, it overrides the already configured global options. Finally, most **view** statements contain multiple **zone** statements that apply to the **match-clients** list.
Draft

Note that the order in which the **view** statements are listed is important, as the first statement that matches a particular client’s IP address is used. For more information on this topic, refer to Section 10.2.6.1, “Multiple Views”.

**10.2.2.4. Comment Tags**

Additionally to statements, the **/etc/named.conf** file can also contain comments. Comments are ignored by the named service, but can prove useful when providing additional information to a user. The following are valid comment tags:

//

Any text after the // characters to the end of the line is considered a comment. For example:

```
notify yes; // notify all secondary nameservers
```

#

Any text after the # character to the end of the line is considered a comment. For example:

```
notify yes; # notify all secondary nameservers
```

/* and */

Any block of text enclosed in /* and */ is considered a comment. For example:

```
notify yes; /* notify all secondary nameservers */
```

**10.2.3. Editing Zone Files**

As outlined in Section 10.1.1, “Nameserver Zones”, zone files contain information about a namespace. They are stored in the named working directory located in **/var/named/** by default, and each zone file is named according to the **file** option in the **zone** statement, usually in a way that relates to the domain in question and identifies the file as containing zone data, such as **example.com.zone**.

<table>
<thead>
<tr>
<th>Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>/var/named/</strong></td>
<td>The working directory for the named service. The nameserver is <em>not</em> allowed to write to this directory.</td>
</tr>
<tr>
<td><strong>/var/named/slaves/</strong></td>
<td>The directory for secondary zones. This directory is writable by the named service.</td>
</tr>
<tr>
<td><strong>/var/named/dynamic/</strong></td>
<td>The directory for other files, such as dynamic DNS (DDNS) zones or managed DNSSEC keys. This directory is writable by the named service.</td>
</tr>
<tr>
<td><strong>/var/named/data/</strong></td>
<td>The directory for various statistics and debugging files. This directory is writable by the named service.</td>
</tr>
</tbody>
</table>

A zone file consists of directives and resource records. Directives tell the nameserver to perform tasks or apply special settings to the zone, resource records define the parameters of the zone and assign
identities to individual hosts. While the directives are optional, the resource records are required in order to provide name service to a zone.

All directives and resource records should be entered on individual lines.

10.2.3.1. Common Directives
Directives begin with the dollar sign character (that is, $) followed by the name of the directive, and usually appear at the top of the file. The following directives are commonly used in zone files:

$INCLUDE
The $INCLUDE directive allows you to include another file at the place where it appears, so that other zone settings can be stored in a separate zone file.

Example 10.7. Using the $INCLUDE directive
$INCLUDE /var/named/penguin.example.com

$ORIGIN
The $ORIGIN directive allows you to append the domain name to unqualified records, such as those with the hostname only. Note that the use of this directive is not necessary if the zone is specified in /etc/named.conf, since the zone name is used by default.

In Example 10.8, “Using the $ORIGIN directive”, any names used in resource records that do not end in a trailing period (that is, the . character) are appended with example.com.

Example 10.8. Using the $ORIGIN directive
$ORIGIN example.com.

$TTL
The $TTL directive allows you to set the default Time to Live (TTL) value for the zone, that is, how long is a zone record valid. Each resource record can contain its own TTL value, which overrides this directive.

Increasing this value allows remote nameservers to cache the zone information for a longer period of time, reducing the number of queries for the zone and lengthening the amount of time required to proliferate resource record changes.

Example 10.9. Using the $TTL directive
$TTL 1D

10.2.3.2. Common Resource Records
The following resource records are commonly used in zone files:

A
The Address record specifies an IP address to be assigned to a name. It takes the following form:

hostname IN A IP-address
If the hostname value is omitted, the record will point to the last specified hostname.

In Example 10.10, “Using the A resource record”, the requests for server1.example.com are pointed to 10.0.1.3 or 10.0.1.5.

Example 10.10. Using the A resource record

```
server1 IN A 10.0.1.3
   IN A 10.0.1.5
```

**CNAME**

The Canonical Name record maps one name to another. Because of this, this type of record is sometimes referred to as an alias record. It takes the following form:

```
alias-name IN CNAME real-name
```

CNAME records are most commonly used to point to services that use a common naming scheme, such as www for Web servers. However, there are multiple restrictions for their usage:

- CNAME records should not point to other CNAME records. This is mainly to avoid possible infinite loops.
- CNAME records should not contain other resource record types (such as A, NS, MX, etc.). The only exception are DNSSEC related records (that is, RRSIG, NSEC, etc.) when the zone is signed.
- Other resource record that point to the fully qualified domain name (FQDN) of a host (that is, NS, MX, PTR) should not point to a CNAME record.

In Example 10.11, “Using the CNAME resource record”, the A record binds a hostname to an IP address, while the CNAME record points the commonly used www hostname to it.

Example 10.11. Using the CNAME resource record

```
server1 IN A 10.0.1.5
  IN CNAME server1
```

**MX**

The Mail Exchange record specifies where the mail sent to a particular namespace controlled by this zone should go. It takes the following form:

```
IN MX preference-value email-server-name
```

The email-server-name is a fully qualified domain name (FQDN). The preference-value allows numerical ranking of the email servers for a namespace, giving preference to some email systems over others. The MX resource record with the lowest preference-value is preferred over the others. However, multiple email servers can possess the same value to distribute email traffic evenly among them.

In Example 10.12, “Using the MX resource record”, the first mail.example.com email server is preferred to the mail2.example.com email server when receiving email destined for the example.com domain.
Example 10.12. Using the MX resource record

| example.com. | IN MX 10 mail.example.com. |
| IN MX 20 mail2.example.com. |

**NS**

The *Nameserver* record announces authoritative nameservers for a particular zone. It takes the following form:

```
IN NS nameserver-name
```

The *nameserver-name* should be a fully qualified domain name (FQDN). Note that when two nameservers are listed as authoritative for the domain, it is not important whether these nameservers are secondary nameservers, or if one of them is a primary server. They are both still considered authoritative.

Example 10.13. Using the NS resource record

```
IN NS dns1.example.com.
IN NS dns2.example.com.
```

**PTR**

The *Pointer* record points to another part of the namespace. It takes the following form:

```
last-IP-digit IN PTR FQDN-of-system
```

The *last-IP-digit* directive is the last number in an IP address, and the *FQDN-of-system* is a fully qualified domain name (FQDN).

**PTR** records are primarily used for reverse name resolution, as they point IP addresses back to a particular name. Refer to Section 10.2.3.4.2, “A Reverse Name Resolution Zone File” for more examples of **PTR** records in use.

**SOA**

The *Start of Authority* record announces important authoritative information about a namespace to the nameserver. Located after the directives, it is the first resource record in a zone file. It takes the following form:

```
@ IN SOA primary-name-server hostmaster-email ( serial-number time-to-refresh time-to-retry time-to-expire minimum-TTL )
```

The directives are as follows:

- The @ symbol places the $ORIGIN directive (or the zone's name if the $ORIGIN directive is not set) as the namespace being defined by this **SOA** resource record.
- The primary-name-server directive is the hostname of the primary nameserver that is authoritative for this domain.
• The hostmaster-email directive is the email of the person to contact about the namespace.

• The serial-number directive is a numerical value incremented every time the zone file is altered to indicate it is time for the named service to reload the zone.

• The time-to-refresh directive is the numerical value secondary nameservers use to determine how long to wait before asking the primary nameserver if any changes have been made to the zone.

• The time-to-retry directive is a numerical value used by secondary nameservers to determine the length of time to wait before issuing a refresh request in the event that the primary nameserver is not answering. If the primary server has not replied to a refresh request before the amount of time specified in the time-to-expire directive elapses, the secondary servers stop responding as an authority for requests concerning that namespace.

• In BIND 4 and 8, the minimum-TTL directive is the amount of time other nameservers cache the zone's information. In BIND 9, it defines how long negative answers are cached for. Caching of negative answers can be set to a maximum of 3 hours (that is, 3H).

When configuring BIND, all times are specified in seconds. However, it is possible to use abbreviations when specifying units of time other than seconds, such as minutes (M), hours (H), days (D), and weeks (W). Table 10.6, “Seconds compared to other time units” shows an amount of time in seconds and the equivalent time in another format.

<table>
<thead>
<tr>
<th>Seconds</th>
<th>Other Time Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>60</td>
<td>1M</td>
</tr>
<tr>
<td>1800</td>
<td>30M</td>
</tr>
<tr>
<td>3600</td>
<td>1H</td>
</tr>
<tr>
<td>10800</td>
<td>3H</td>
</tr>
<tr>
<td>21600</td>
<td>6H</td>
</tr>
<tr>
<td>43200</td>
<td>12H</td>
</tr>
<tr>
<td>86400</td>
<td>1D</td>
</tr>
<tr>
<td>259200</td>
<td>3D</td>
</tr>
<tr>
<td>604800</td>
<td>1W</td>
</tr>
<tr>
<td>31536000</td>
<td>365D</td>
</tr>
</tbody>
</table>

Example 10.14. Using the SOA resource record

```
@ 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
```

10.2.3.3. Comment Tags

Additionally to resource records and directives, a zone file can also contain comments. Comments are ignored by the named service, but can prove useful when providing additional information to the user.
Any text after the semicolon character (that is, ;) to the end of the line is considered a comment. For example:

604800 ; expire after 1 week

10.2.3.4. Example Usage
The following examples show the basic usage of zone files.

10.2.3.4.1. A Simple Zone File

*Example 10.15, “A simple zone file”* demonstrates the use of standard directives and SOA values.

```plaintext
$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.
;
;
```

In this example, the authoritative nameservers are set as dns1.example.com and dns2.example.com, and are tied to the 10.0.1.1 and 10.0.1.2 IP addresses respectively using the A record.
The email servers configured with the MX records point to mail and mail2 via A records. Since these names do not end in a trailing period (that is, the . character), the $ORIGIN domain is placed after them, expanding them to mail.example.com and mail2.example.com.

Services available at the standard names, such as www.example.com (WWW), are pointed at the appropriate servers using the CNAME record.

This zone file would be called into service with a zone statement in the /etc/named.conf similar to the following:

```
zone "example.com" IN {
    type master;
    file "example.com.zone";
    allow-update { none; };
};
```

10.2.3.4.2. A Reverse Name Resolution Zone File

A reverse name resolution zone file is used to translate an IP address in a particular namespace into an fully qualified domain name (FQDN). It looks very similar to a standard zone file, except that the PTR resource records are used to link the IP addresses to a fully qualified domain name as shown in Example 10.16, “A reverse name resolution zone file”.

```
$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.
```

In this example, IP addresses 10.0.1.1 through 10.0.1.6 are pointed to the corresponding fully qualified domain name.

This zone file would be called into service with a zone statement in the /etc/named.conf file similar to the following:

```
zone "1.0.10.in-addr.arpa" IN {
    type master;
    file "example.com.rr.zone";
    allow-update { none; };
};
```
10.2.4. Using the rndc Utility

The **rndc** utility is a command line tool that allows you to administer the named service, both locally and from a remote machine. Its usage is as follows:

```
rndc [option...] command [command-option]
```

### 10.2.4.1. Configuring the Utility

To prevent unauthorized access to the service, named must be configured to listen on the selected port (that is, **953** by default), and an identical key must be used by both the service and the **rndc** utility.

#### Table 10.7. Relevant files

<table>
<thead>
<tr>
<th>Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/etc/named.conf</code></td>
<td>The default configuration file for the named service.</td>
</tr>
<tr>
<td><code>/etc/rndc.conf</code></td>
<td>The default configuration file for the <strong>rndc</strong> utility.</td>
</tr>
<tr>
<td><code>/etc/rndc.key</code></td>
<td>The default key location.</td>
</tr>
</tbody>
</table>

The **rndc** configuration is located in **/etc/rndc.conf**. If the file does not exist, the utility will use the key located in **/etc/rndc.key**, which was generated automatically during the installation process using the **rndc-confgen -a** command.

The named service is configured using the **controls** statement in the **/etc/named.conf** configuration file as described in *Section 10.2.2.3, "Other Statement Types"*. Unless this statement is present, only the connections from the loopback address (that is, **127.0.0.1**) will be allowed, and the key located in **/etc/rndc.key** will be used.

For more information on this topic, refer to manual pages and the **BIND 9 Administrator Reference Manual** listed in *Section 10.2.8, "Additional Resources"*.

#### Set the correct permissions

To prevent unprivileged users from sending control commands to the service, make sure only root is allowed to read the **/etc/rndc.key** file:

```
~]$ chmod o-rwx /etc/rndc.key
```

### 10.2.4.2. Checking the Service Status

To check the current status of the named service, use the following command:

```
~]$ rndc status
version: 9.7.0-P2-RedHat-9.7.0-5.P2.el6
CPUs found: 1
```
10.2.4.3. Reloading the Configuration and Zones

To reload both the configuration file and zones, type the following at a shell prompt:

```
~]# rndc reload
server reload successful
```

This will reload the zones while keeping all previously cached responses, so that you can make changes to the zone files without losing all stored name resolutions.

To reload a single zone, specify its name after the `reload` command, for example:

```
~]# rndc reload localhost
zone reload up-to-date
```

Finally, to reload the configuration file and newly added zones only, type:

```
~]# rndc reconfig
```

Modifying zones with dynamic DNS

If you intend to manually modify a zone that uses Dynamic DNS (DDNS), make sure you run the `freeze` command first:

```
~]# rndc freeze localhost
```

Once you are finished, run the `thaw` command to allow the DDNS again and reload the zone:

```
~]# rndc thaw localhost
The zone reload and thaw was successful.
```

10.2.4.4. Updating Zone Keys

To update the DNSSEC keys and sign the zone, use the `sign` command. For example:

```
~]# rndc sign localhost
```

Note that to sign a zone with the above command, the `auto-dnssec` option has to be set to `maintain` in the zone statement. For instance:

```
zone "localhost" IN {
    type master;
```
10.2.4.5. Enabling the DNSSEC Validation
To enable the DNSSEC validation, type the following at a shell prompt:

```
~# rndc validation on
```

Similarly, to disable this option, type:

```
~# rndc validation off
```

Refer to the options statement described in Section 10.2.2.2, “Common Statement Types” for information on how configure this option in /etc/named.conf.

10.2.4.6. Enabling the Query Logging
To enable (or disable in case it is currently enabled) the query logging, run the following command:

```
~# rndc querylog
```

To check the current setting, use the status command as described in Section 10.2.4.2, “Checking the Service Status”.

10.2.5. Using the dig Utility
The dig utility is a command line tool that allows you to perform DNS lookups and debug a nameserver configuration. Its typical usage is as follows:

```
dig [@server] [option...] name type
```

Refer to Section 10.2.3.2, “Common Resource Records” for a list of common types.

10.2.5.1. Looking Up a Nameserver
To look up a nameserver for a particular domain, use the command in the following form:

```
dig name NS
```

In Example 10.17, “A sample nameserver lookup”, the dig utility is used to display nameservers for example.com.

```
~$ dig 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
```
10.2.5.2. Looking Up an IP Address

To look up an IP address assigned to a particular domain, use the command in the following form:

```
dig name A
```

In *Example 10.18, “A sample IP address lookup”*, the **dig** utility is used to display the IP address of `example.com`.

```
Example 10.18. A sample IP address lookup

~]$ 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
```

10.2.5.3. Looking Up a Hostname

To look up a hostname for a particular IP address, use the command in the following form:

```
dig -x address
```

In *Example 10.19, “A sample hostname lookup”*, the **dig** utility is used to display the hostname assigned to `192.0.32.10`.

```
Example 10.19. A sample hostname lookup

~]$ dig -x 192.0.32.10
```
Chapter 10. DNS Servers

10.2.6. Advanced Features of BIND

Most BIND implementations only use the named service to provide name resolution services or to act as an authority for a particular domain. However, BIND version 9 has a number of advanced features that allow for a more secure and efficient DNS service.

Make sure the feature is supported

Before attempting to use advanced features like DNSSEC, TSIG, or IXFR (Incremental Zone Transfer), make sure that the particular feature is supported by all nameservers in the network environment, especially when you use older versions of BIND or non-BIND servers.

All of the features mentioned are discussed in greater detail in the BIND 9 Administrator Reference Manual referenced in Section 10.2.8.1, “Installed Documentation”.

10.2.6.1. Multiple Views

Optionally, different information can be presented to a client depending on the network a request originates from. This is primarily used to deny sensitive DNS entries from clients outside of the local network, while allowing queries from clients inside the local network.

To configure multiple views, add the view statement to the /etc/named.conf configuration file. Use the match-clients option to match IP addresses or entire networks and give them special options and zone data.
10.2.6.2. Incremental Zone Transfers (IXFR)

*Incremental Zone Transfers (IXFR)* allow a secondary nameserver to only download the updated portions of a zone modified on a primary nameserver. Compared to the standard transfer process, this makes the notification and update process much more efficient.

Note that IXFR is only available when using dynamic updating to make changes to master zone records. If manually editing zone files to make changes, *Automatic Zone Transfer (AXFR)* is used.

10.2.6.3. Transaction SIGnatures (TSIG)

*Transaction SIGnatures (TSIG)* ensure that a shared secret key exists on both primary and secondary nameserver before allowing a transfer. This strengthens the standard IP address-based method of transfer authorization, since attackers would not only need to have access to the IP address to transfer the zone, but they would also need to know the secret key.

Since version 9, BIND also supports *TKEY*, which is another shared secret key method of authorizing zone transfers.

**Secure the transfer**

When communicating over an insecure network, do not rely on IP address-based authentication only.

10.2.6.4. DNS Security Extensions (DNSSEC)

*Domain Name System Security Extensions (DNSSEC)* provide origin authentication of DNS data, authenticated denial of existence, and data integrity. When a particular domain is marked as secure, the *SERFVAIL* response is returned for each resource record that fails the validation.

Note that to debug a DNSSEC-signed domain or a DNSSEC-aware resolver, you can use the *dig* utility as described in *Section 10.2.5, “Using the dig Utility”*. Useful options are *+dnssec* (requests DNSSEC-related resource records by setting the DNSSEC OK bit), *+cd* (tells recursive nameserver not to validate the response), and *+bufsize=512* (changes the packet size to 512B to get through some firewalls).

10.2.6.5. Internet Protocol version 6 (IPv6)

*Internet Protocol version 6 (IPv6)* is supported through the use of *AAAA* resource records, and the *listen-on-v6* directive as described in *Table 10.3, “Commonly Used Configuration Options”*.

10.2.7. Common Mistakes to Avoid

The following is a list of recommendations on how to avoid common mistakes users make when configuring a nameserver:
Use semicolons and curly brackets correctly
   An omitted semicolon or unmatched curly bracket in the /etc/named.conf file can prevent the named service from starting.

Use period (that is, the . character) correctly
   In zone files, a period at the end of a domain name denotes a fully qualified domain name. If omitted, the named service will append the name of the zone or the value of $ORIGIN to complete it.

Increment the serial number when editing a zone file
   If the serial number is not incremented, the primary nameserver will have the correct, new information, but the secondary nameservers will never be notified of the change, and will not attempt to refresh their data of that zone.

Configure the firewall
   If a firewall is blocking connections from the named service to other nameservers, the recommended practice is to change the firewall settings.

Avoid using fixed UDP source ports
   According to the recent research in DNS security, using a fixed UDP source port for DNS queries is a potential security vulnerability that could allow an attacker to conduct cache-poisoning attacks more easily. To prevent this, configure your firewall to allow queries from a random UDP source port.

10.2.8. Additional Resources
The following sources of information provide additional resources regarding BIND.

10.2.8.1. Installed Documentation
BIND features a full range of installed documentation covering many different topics, each placed in its own subject directory. For each item below, replace version with the version of the bind package installed on the system:

/usr/share/doc/bind-version/
   The main directory containing the most recent documentation.

/usr/share/doc/bind-version/arm/
   The directory containing the BIND 9 Administrator Reference Manual in HTML and SGML formats, which details BIND resource requirements, how to configure different types of nameservers, how to perform load balancing, and other advanced topics. For most new users of BIND, this is the best place to start.

/usr/share/doc/bind-version/draft/
   The directory containing assorted technical documents that review issues related to the DNS service, and propose some methods to address them.

/usr/share/doc/bind-version/misc/
   The directory designed to address specific advanced issues. Users of BIND version 8 should consult the migration document for specific changes they must make when moving to BIND
9. The **options** file lists all of the options implemented in BIND 9 that are used in `/etc/named.conf`.

`/usr/share/doc/bind-version/rfc/`

The directory providing every RFC document related to BIND.

There is also a number of man pages for the various applications and configuration files involved with BIND:

- **man** `rndc`
  The manual page for `rndc` containing the full documentation on its usage.

- **man** `named`
  The manual page for `named` containing the documentation on assorted arguments that can be used to control the BIND nameserver daemon.

- **man** `lwresd`
  The manual page for `lwresd` containing the full documentation on the lightweight resolver daemon and its usage.

- **man** `named.conf`
  The manual page with a comprehensive list of options available within the `named` configuration file.

- **man** `rndc.conf`
  The manual page with a comprehensive list of options available within the `rndc` configuration file.

### 10.2.8.2. Useful Websites

- [http://www.isc.org/software/bind](http://www.isc.org/software/bind)
  The home page of the BIND project containing information about current releases as well as a PDF version of the *BIND 9 Administrator Reference Manual*.

### 10.2.8.3. Related Books

- *DNS and BIND* by Paul Albitz and Cricket Liu; O'Reilly & Associates
  A popular reference that explains both common and esoteric BIND configuration options, and provides strategies for securing a DNS server.

- *The Concise Guide to DNS and BIND* by Nicolai Langfeldt; Que
  Looks at the connection between multiple network services and BIND, with an emphasis on task-oriented, technical topics.
Appendix A. Revision History

Revision 1-0   Tue Feb 25 2014   Stephen Wadeley swadeley@redhat.com
First version of the Fedora Networking Guide.
Index

Symbols
/etc/named.conf (see BIND)
/etc/sysconfig/dhcpd, 99

A
authoritative nameserver (see BIND)

B
Berkeley Internet Name Domain (see BIND)
BIND
additional resources
installed documentation, 128
related books, 129
useful websites, 129
common mistakes, 127
configuration
acl statement, 108
comment tags, 115
controls statement, 113
include statement, 109
key statement, 114
logging statement, 114
options statement, 109
server statement, 114
trusted-keys statement, 114
view statement, 114
zone statement, 111
directories
/etc/named/, 106
/var/named/, 115
/var/named/data/, 115
/var/named/dynamic/, 115
/var/named/slaves/, 115
features
Automatic Zone Transfer (AXFR), 127
DNS Security Extensions (DNSSEC), 127
Incremental Zone Transfer (IXFR), 127
Internet Protocol version 6 (IPv6), 127
multiple views, 126
Transaction SIGnature (TSIG), 127
files
/etc/named.conf, 106, 122
/etc/mdc.conf, 122
/etc/mdc.key, 122
resource record, 105
types
authoritative nameserver, 105
primary (master) nameserver, 105
recursive nameserver, 106
secondary (slave) nameserver, 105
utilities
dig, 106, 124, 127
named, 106, 106
rndc, 106, 122
zones
$INCLUDE directive, 116
$ORIGIN directive, 116
$TTL directive, 116
A (Address) resource record, 116
CNAME (Canonical Name) resource record, 117
comment tags, 119
description, 105
element usage, 125, 121
MX (Mail Exchange) resource record, 117
NS (Nameserver) resource record, 118
PTR (Pointer) resource record, 118
SOA (Start of Authority) resource record, 118

D
default gateway, 32
DHCP, 95
additional resources, 103
command line options, 99
dhcpd.conf, 95
dhcpd.leases, 99
dhcpd6.conf, 103
DHCPv6, 103
dhcprelay, 100
global parameters, 96
group, 98
options, 96
reasons for using, 95
Relay Agent, 100
server configuration, 95
shared-network, 97
starting the server, 99
stopping the server, 99
subnet, 96
dhcpd.conf, 95
dhcpd.leases, 99
dhcprelay, 100
dig (see BIND)
DNS
definition, 105
(see also BIND)
Dynamic Host Configuration Protocol (see DHCP)

F
feedback
contact information for this manual, xi
FQDN (see fully qualified domain name)
fully qualified domain name, 105
Index

M
Multihomed DHCP
  host configuration, 101
  server configuration, 100

N
named (see BIND)
nameserver (see DNS)

P
primary nameserver (see BIND)

R
recursive nameserver (see BIND)
resource record (see BIND)
rndc (see BIND)
root nameserver (see BIND)

S
secondary nameserver (see BIND)
static route, 31