Oracle® Solaris Administration: Network Interfaces and Network Virtualization

Part No: 821–1458–11
December 2011

ORACLE

111206@25097

Contents

Preface

Welcome to the Oracle Solaris Administration: Network Interfaces and Network Virtualization. This book is part of a fourteen-volume set that covers a significant part of the Oracle Solaris system administration information. This book assumes that you have already installed Oracle Solaris. You should be ready to configure your network or ready to configure any networking software that is required on your network.

Note – This Oracle Solaris release supports systems that use the SPARC and x86 families of processor architectures. The supported systems appear in the *Oracle Solaris OS: Hardware Compatibility Lists*. This document cites any implementation differences between the platform types.

In this document, these x86 related terms mean the following:

- x86 refers to the larger family of 64-bit and 32-bit x86 compatible products.
- x64 relates specifically to 64-bit x86 compatible CPUs.
- "32-bit x86" points out specific 32-bit information about x86 based systems.

For supported systems, see the *Oracle Solaris OS: Hardware Compatibility Lists*.

Who Should Use This Book

This book is intended for anyone responsible for administering systems that run Oracle Solaris, which are configured in a network. To use this book, you should have at least two years of UNIX system administration experience. Attending UNIX system administration training courses might be helpful.

How the System Administration Guides Are Organized

Here is a list of the topics that are covered by the System Administration Guides.

Book Title	Topics
Booting and Shutting Down Oracle Solaris on SPARC Platforms	Booting and shutting down a system, managing boot services, modifying boot behavior, booting from ZFS, managing the boot archive, and troubleshooting booting on SPARC platforms
Booting and Shutting Down Oracle Solaris on x86 Platforms	Booting and shutting down a system, managing boot services, modifying boot behavior, booting from ZFS, managing the boot archive, and troubleshooting booting on x86 platforms
Oracle Solaris Administration: Common Tasks	Using Oracle Solaris commands, booting and shutting down a system, managing user accounts and groups, managing services, hardware faults, system information, system resources, and system performance, managing software, printing, the console and terminals, and troubleshooting system and software problems
Oracle Solaris Administration: Devices and File Systems	Removable media, disks and devices, file systems, and backing up and restoring data
Oracle Solaris Administration: IP Services	TCP/IP network administration, IPv4 and IPv6 address administration, DHCP, IPsec, IKE, IP Filter, and IPQoS
Oracle Solaris Administration: Naming and Directory Services	DNS, NIS, and LDAP naming and directory services, including transitioning from NIS to LDAP
Oracle Solaris Administration: Network Interfaces and Network Virtualization	Automatic and manual IP interface configuration including WiFi wireless; administration of bridges, VLANs, aggregations, LLDP, and IPMP; virtual NICs and resource management.
Oracle Solaris Administration: Network Services	Web cache servers, time-related services, network file systems (NFS and autofs), mail, SLP, and PPP
Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management	Resource management features, which enable you to control how applications use available system resources; Oracle Solaris Zones software partitioning technology, which virtualizes operating system services to create an isolated environment for running applications; and Oracle Solaris 10 Zones, which host Oracle Solaris 10 environments running on the Oracle Solaris 11 kernel
Oracle Solaris Administration: Security Services	Auditing, device management, file security, BART, Kerberos services, PAM, Cryptographic Framework, Key Management, privileges, RBAC, SASL, Secure Shell, and virus scanning

Book Title	Topics
Oracle Solaris Administration: SMB and Windows Interoperability	SMB service, which enables you to configure an Oracle Solaris system to make SMB shares available to SMB clients; SMB client, which enables you to access SMB shares; and native identity mapping services, which enables you to map user and group identities between Oracle Solaris systems and Windows systems
Oracle Solaris Administration: ZFS File Systems	ZFS storage pool and file system creation and management, snapshots, clones, backups, using access control lists (ACLs) to protect ZFS files, using ZFS on a Solaris system with zones installed, emulated volumes, and troubleshooting and data recovery
Oracle Solaris Trusted Extensions Configuration and Administration	System installation, configuration, and administration that is specific to Trusted Extensions
Oracle Solaris 11 Security Guidelines	Securing an Oracle Solaris system, as well as usage scenarios for its security features, such as zones, ZFS, and Trusted Extensions
Transitioning From Oracle Solaris 10 to Oracle Solaris 11	Provides system administration information and examples for transitioning from Oracle Solaris 10 to Oracle Solaris 11 in the areas of installation, device, disk, and file system management, software management, networking, system management, security, virtualization, desktop features, user account management, and user environments emulated volumes, and troubleshooting and data recovery

Related Third-Party Web Site References

Third party URLs are referenced in this document and provide additional, related information.

Note – Oracle is not responsible for the availability of third-party Web sites mentioned in this document. Oracle does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Oracle will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Description	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`** `Password:`
aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX system prompt and superuser prompt for shells that are included in the Oracle Solaris OS. Note that the default system prompt that is displayed in command examples varies, depending on the Oracle Solaris release.

TABLE P–2 Shell Prompts

Shell	Prompt
Bash shell, Korn shell, and Bourne shell	$

TABLE P–2 Shell Prompts *(Continued)*

Shell	Prompt
Bash shell, Korn shell, and Bourne shell for superuser	`#`
C shell	`machine_name%`
C shell for superuser	`machine_name#`

◆ ◆ ◆ CHAPTER 1

Overview of the Networking Stack

This chapter introduces network administration in Oracle Solaris. It describes interrelationships that underlie interfaces, datalinks over which the interfaces are configured, and network devices. Support for flexible names for datalinks is also discussed at length.

Network Configuration in This Oracle Solaris Release

Note the following differences in the manner the network is configured in this release that distinguishes it from previous Oracle Solaris releases:

- Network configuration is managed by a profile. The type of configuration that is operative in a system depends on which network configuration profile is active. See Part I, "Network Auto-Magic."
- Datalinks on layer 2 of the networking stack are administered by using the `dladm` command. This command replaces previous `ifconfig` command options to configure datalink properties. Consequently, the configuration of link aggregations, VLANs, and IP tunnels have also changed. See Chapter 8, "Datalink Configuration and Administration," Chapter 12, "Administering Link Aggregations," and Chapter 13, "Administering VLANs." See also Chapter 6, "Configuring IP Tunnels," in *Oracle Solaris Administration: IP Services*.
- Datalink names are no longer bound to their hardware drivers. Thus, datalinks, by default, are assigned generic link names such as `net0`, `net1`, and so on. See "Network Devices and Datalink Names" on page 26.
- IP interfaces on layer 3 of the networking stack are administered by using the `ipadm` command. This command replaces previous `ifconfig` command options to configure IP interfaces. See Chapter 9, "Configuring an IP Interface."
- IPMP groups are implemented as IP interfaces and are therefore similarly configured with the `ipadm` command. Additionally, the `ipmpstat` is introduced that allows you to obtain IPMP-related information and statistics. See Chapter 14, "Introducing IPMP," and Chapter 15, "Administering IPMP."

- Virtualization is implemented on the network device level. Thus you can configure VNICs and manage the use of network resources for greater efficiency. See Part III, "Network Virtualization and Resource Management."

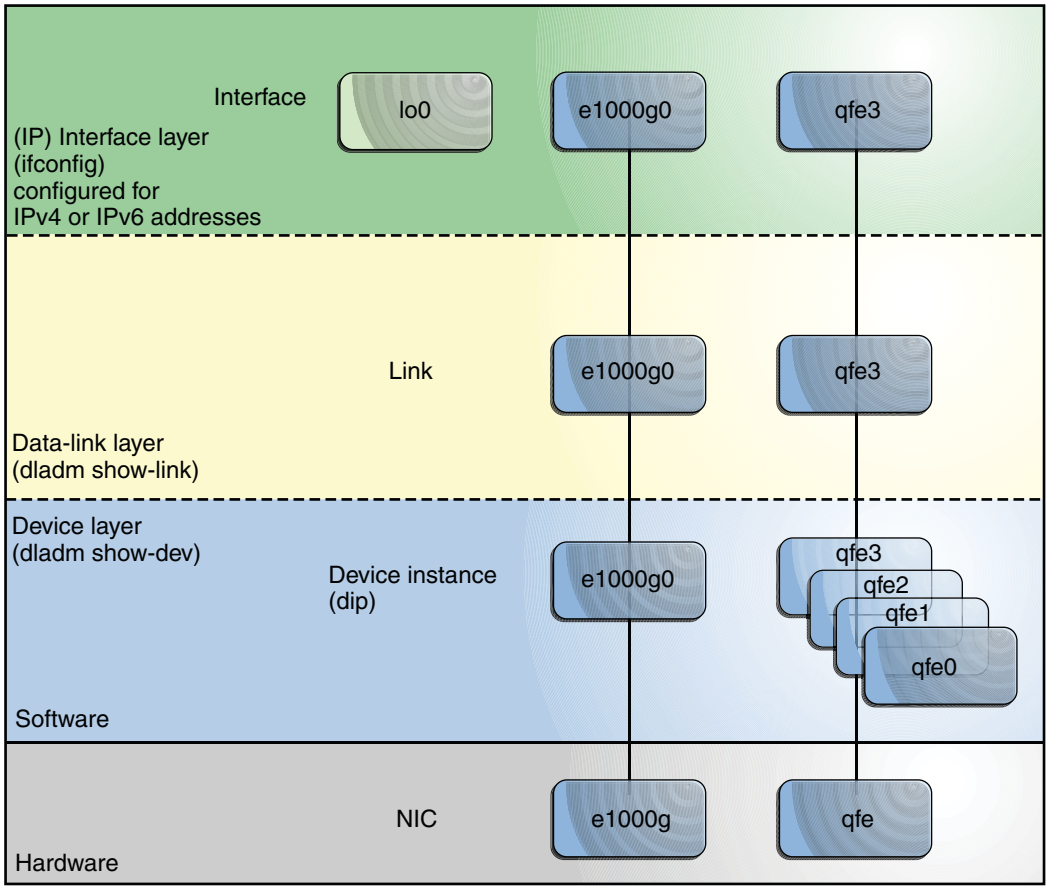
The Network Stack in Oracle Solaris

Network interfaces provide the connection between the system and the network. These interfaces are configured over datalinks, which in turn correspond to instances of hardware devices in the system. Network hardware devices are also called *network interface cards (NICs)* or *network adapters*. NICs can be built in and already present in the system when the system is purchased. However, you can also purchase separate NICs to add to the system. Certain NICs have only a single interface that resides on the card. Other brands might have multiple interfaces that you can configure to perform network operations.

In the current model of the network stack, interfaces and links on the software layer build on the devices in the hardware layer. More specifically, a hardware device instance in the hardware layer has a corresponding link on the datalink layer and a configured interface on the interface layer. This one-to-one relationship among the network device, its datalink, and the IP interface is illustrated in the figure that follows.

Note – For a fuller explanation of the TCP/IP stack, see Chapter 1, "Oracle Solaris TCP/IP Protocol Suite (Overview)," in *System Administration Guide: IP Services*.

FIGURE 1–1 Network Stack Showing Network Devices, Links, and Interfaces — Oracle Solaris 10 Model



The figure shows two NICs on the hardware layer: `e1000` with a single device instance `e1000g0`, and `qfe` with multiple device instances, `qfe0` to `qfe3`. The devices `qfe0` through `qfe2` are not used. Devices `e1000g` and `qfe3` are used and have corresponding links `e1000g` and `qfe3` on the datalink layer. In the figure, the IP interfaces are likewise named after their respective underlying hardware, `e1000g` and `qfe3`. These interfaces can be configured with IPv4 or IPv6 addresses to host both types of network traffic. Note also the presence of the loopback interface `lo0` on the interface layer. This interface is used to test, for example, that the IP stack is functioning properly.

Different administrative commands are used at each layer of the stack. For example, hardware devices that are installed on the system are listed by the `dladm show-dev` command. Information about links on the datalink layer is displayed by the `dladm show-link` command. The `ifconfig` command shows the IP interface configuration on the interface layer.

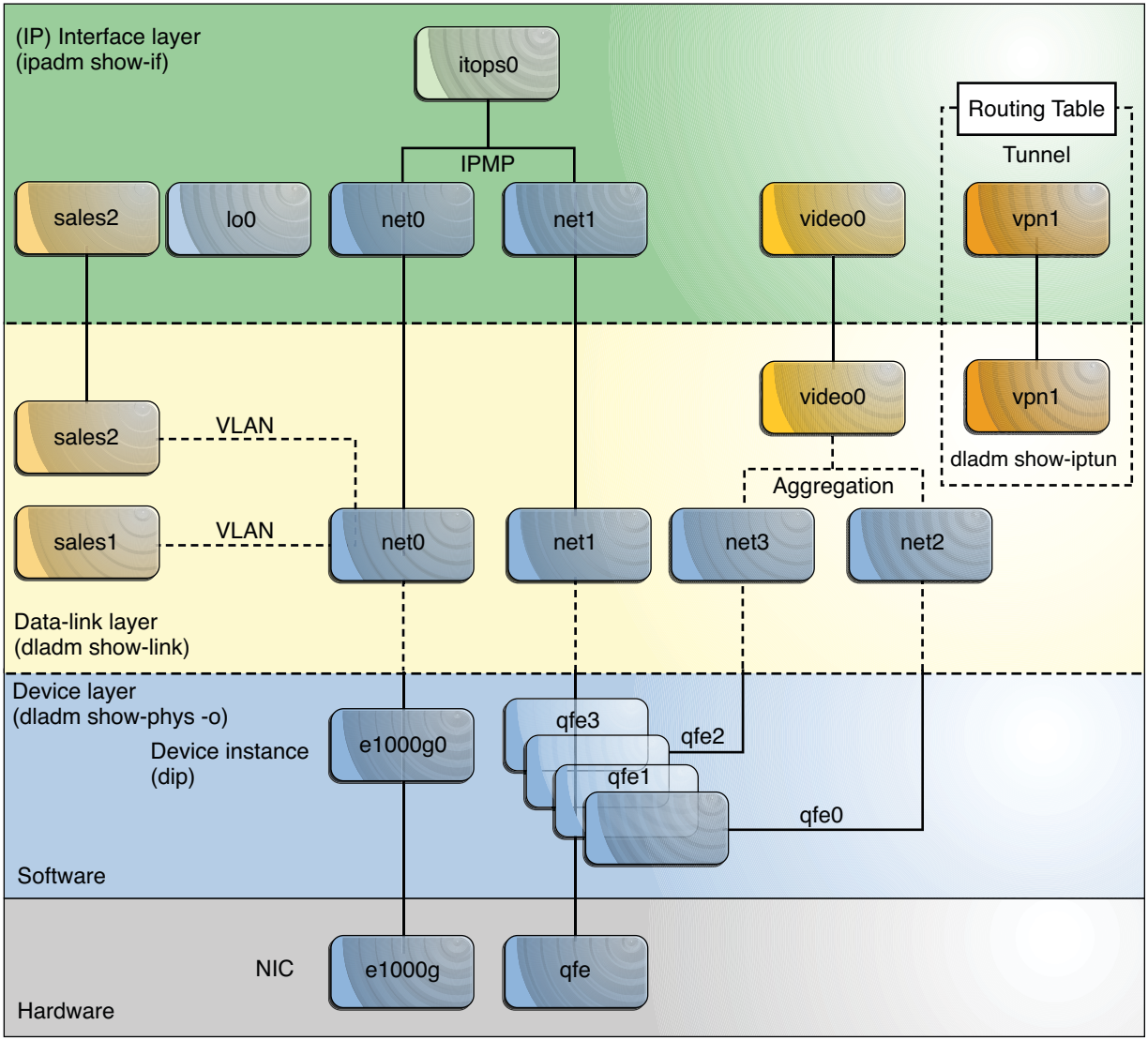
In this model, a one-to-one relationship exists that binds the device, the datalink, and the interface. This relationship means that network configuration is dependent on hardware configuration and network topology. Interfaces must be reconfigured if changes are implemented in the hardware layer, such as replacing the NIC or changing the network topology.

Oracle Solaris 11 introduces an implementation of the network stack in which the basic relationship between the hardware, datalink, and interface layers remains. However, the software layer is decoupled from the hardware layer. With this separation, network configuration on the software level is no longer bound to the chipset or the network topology in the hardware layer. This implementation makes network administration more flexible in the following ways:

- The network configuration is insulated from any changes that might occur in the hardware layer. Link and interface configurations are preserved even if the underlying hardware is removed. These same configurations can then be reapplied to any replacement NIC, provided that the two NICs are of the same type.
- The separation of the network configuration from the network hardware configuration also allows the use of customized link names in the datalink layer.
- With the abstraction of the datalink layer, multiple networking abstractions or configurations such as VLANs, VNICs, physical devices, link aggregations, and IP tunnels are unified into a common administrative entity, which is the datalink.

The following figure illustrates how these network configurations are created on the networking stack:

FIGURE 1–2 Network Stack Showing Network Devices, Links, and Interfaces — Oracle Solaris 11 Model



The configurations in this illustration are further explained in "Administration of Other Link Types" on page 31.

Network Devices and Datalink Names

From an administrative perspective, administrators create IP interfaces on top of *datalinks*. The datalink represents a link object in the second layer of the Open Systems Interconnection (OSI) model. The *physical link* is directly associated with a device and possesses a device name. The device name is essentially the device instance name, and is composed of the driver name and the device instance number. The instance number can have a value from zero to *n*, depending on how many NICs use that driver on the system.

For example, consider a Gigabit Ethernet card, which is often used as the primary NIC on both host systems and server systems. Some typical driver names for this NIC are `bge` and `e1000g`. When used as the primary NIC, the Gigabit Ethernet interface has a device name such as `bge0` or `e1000g0`. Other driver names are `nge`, `nxge`, and so on.

In this Oracle Solaris release, the device instance name continues to depend on the underlying hardware. However, datalinks on top of these devices are not similarly bound and can be given meaningful names. For example, the administrator can assign the datalink on top of device instance `e1000g0` the name `itops0`. In this Oracle Solaris release, datalinks by default are provided with generic names. To display the mapping between the datalinks with their generic names and the corresponding device instances, you use the `dladm sho-phys` subcommand.

Default Generic Link Names

When you install this Oracle Solaris release on a system for the first time, Oracle Solaris automatically provides generic link names for all the system's physical network devices. This name assignment uses the `net#` naming convention, where the # is the instance number. This instance number increments for each device, for example, `net0`, `net1`, `net2`, and so on.

Generic or flexible link names provide advantages in network configuration as shown in the following examples:

- Within a single system, dynamic reconfiguration becomes easier. The network configuration that is set for a given NIC can be inherited by a different NIC replacement.
- Zone migration becomes less complicated with regards to network setup. The zone in the migrated system preserves its network configuration if the destination system's link shares the same name with the link that has been assigned to the zone prior to migration. Thus, no additional network configuration on the zone is required after the migration.
- The generic naming scheme helps with network configuration that is specified in the System Configuration (SC) manifest. The primary network datalink is generally named `net0` for all systems. Thus, a generic SC manifest can be used for multiple systems that specify a configuration for `net0`.
- Datalink administration also becomes flexible. You can further customize the name of datalinks, for example to reflect a specific function that the datalink serves, as shown in Figure 1–2.

The following table illustrates the new correspondence between the hardware (NIC), the device instance, the link name, and the interface over the link. The names of the datalinks are automatically provided by the OS.

Hardware (NIC)	Device Instance	Link's Assigned Name	IP Interface
e1000g	e1000g0	net0	net0
qfe	qfe1	net1	net1

As the table indicates, while the device instance name remains hardware-based, the datalinks have been renamed by the OS after it is installed.

The Assignment of Generic Names to Datalinks

In Oracle Solaris, generic names are automatically assigned to all the datalinks based on specific criteria. All devices share the same prefix `net`. However, the instance numbers are assigned based on the following:

- Physical network devices are ordered according to media type, where certain types have priority over others. The media types are ordered in descending priority as follows:
  1. Ethernet
  2. IP over IB (Infiniband devices)
  3. Ethernet over IB
  4. WiFi
- After devices are grouped and sorted according to media types, these devices are further ordered based on their physical locations, where onboard devices are favored over peripheral devices.
- Devices that have higher priority based on their media type and location are assigned lower instance numbers.

Based on the criteria, Ethernet devices on a lower motherboard or ioboard, hostbridge, PCIe rootcomplex, bus, device, and function are ranked ahead of the other devices.

To display the correspondences of link names, devices, and locations, use the `dladm show-phys` command as follows:

```
# dladm show-phys -L
LINK         DEVICE       LOCATION
net0         e1000g0      MB
net1         e1000g1      MB
net2         e1000g2      MB
net3         e1000g3      MB
net4         ibp0         MB/RISER0/PCIE0/PORT1
net5         ibp1         MB/RISER0/PCIE0/PORT2
```

```
net6        eoib2       MB/RISER0/PCIE0/PORT1/cloud-nm2gw-2/1A-ETH-2
net7        eoib4       MB/RISER0/PCIE0/PORT2/cloud-nm2gw-2/1A-ETH-2
```

Customizing How Generic Link Names Are Assigned

Oracle Solaris uses the prefix `net` when assigning link names. However, any custom prefix can be used instead, such as `eth`. If you prefer, you can also disable the automatic assignment of neutral link names.

Caution – You must customize how generic link names are automatically assigned *before* you install Oracle Solaris. After installation, you cannot customize the default link names without tearing down existing configurations.

To disable automatic link naming, or to customize the prefix of link names, set the following property in the System Configuration manifests that are used by the Automated Install (AI) program.

```
<service name="network/datalink-management"
   version="1" type="service">
      <instance name="default enabled="true">
         <property_group name='linkname-policy'
            type='application'>
               <propval name='phys-prefix' type='astring'
                  value='net'/>
         </property_group>
      </instance>
</service
```

By default, the value for `phys-prefix` is set to `net`, as shown in emphasis.

- To disable automatic naming, remove any value that is set for `phys-prefix`. If you disable automatic naming, then datalink names will be based on their associated hardware drivers, such as `bge0`, `e1000g0`, and so on.
- To use a different prefix other than `net`, specify a new prefix as the value of `phys-prefix`, such as `eth`.

If the value that is provided to `phys-prefix` is invalid, then that value will be ignored. The datalinks will be named according to their associated hardware drivers, such as `bge0`, `e1000g0`, and so on. For rules about valid link names, see "Rules for Valid Link Names" on page 30.

Link Names in Upgraded Systems

In systems where this Oracle Solaris release is freshly installed, datalinks are automatically named `net0` through `net`*N-1*, where *N* represents the total number of network devices.

The case is not true if you upgrade from Oracle Solaris 11 Express. On such upgraded systems, the datalinks retain their names prior to the upgrade. These names would either be the default hardware-based names, or customized names that the administrator assigned to the datalinks before the upgrade. Further, on these upgraded systems, new network devices that are subsequently added also retain the default hardware-based names rather than receive neutral names. This behavior for upgraded systems ensures that no neutral names that are assigned by the OS become mixed with other hardware-based names or customize names assigned by the administrator before the upgrade.

In any system with this Oracle Solaris release, both hardware-based names as well as OS-supplied link names can be replaced by other names that you prefer to use. Typically, the default link names that are assigned by the OS suffice for creating the system's network configuration. However, if you select to change link names, note the important considerations discussed in the following sections.

Replacing Hardware-Based Link Names

If your system's links have hardware-based names, rename these links with at least generic names. If you retain the hardware-based names of the links, confusion might arise in later situations where these physical devices are removed or replaced.

For example, you retain the link name `bge0` that is associated with the device `bge0`. All link configurations are performed by referring to the link name. Later, you might replace the NIC `bge` with the NIC `e1000g`. To reapply the former device's link configuration to the new NIC `e1000g0`, you would need to reassign the link name `bge0` to `e1000g0`. The combination of a hardware-based link name `bge0` with a different associated NIC `e1000g0` can cause confusion. By using names that are not hardware-based, you can better distinguish the links from the associated devices.

Caution About Changing Link Names

While replacing hardware-based link names is recommended, you must still plan carefully before you rename links. Changing the device's link name does not automatically propagate the new name to all existing associated configurations. The following examples illustrate the risks when you change link names:

- Some rules in an IP Filter configuration apply to specific links. When you change a link's name, the filter rules continue to refer to the link's original name. Consequently, these rules no longer behave as expected after you rename the link. You need to adjust the filter rules to apply to the link by using the new link name.
- Consider the possibility of exporting network configuration information. As previously explained, by using the default `net#` names provided by the OS, you can migrate zones and export network configuration to another system easily. If the target system's network devices are named with generic names such as `net0`, `net1`, and others, then the zone simply inherits the network configuration of the datalink whose name matches the datalink assigned to the zone.

Thus, as a general rule, do not rename datalinks randomly. When renaming datalinks, ensure that all of the link's associated configurations continue to apply after the link name is changed. Some of the configurations that might be affected by renaming links are as follows:

- IP Filter rules
- IP configurations that are specified in configuration files such as `/etc/dhcp.*`
- Oracle Solaris 11 Zones
- `autopush` configuration

Note – No changes are required in the `autopush` configuration when you rename links. However, you must be aware of how the configuration would work with the per-link `autopush` property after the link has been renamed. For more information, see "How to Set STREAMS Modules on Datalinks" on page 164.

Rules for Valid Link Names

When you assign link names, observe the following rules:

- Link names consist of a string and a *physical point of attachment (PPA)* number.
- The name must abide by the following constraints:
  - Names consist of between 3 to 8 characters. However, names can have a maximum of 16 characters.
  - Valid characters for names are alphanumeric (a-z, 0–9) and the underscore ('_').

Caution – Do not use upper case letters on link names.

- Each datalink must have only one link name at one time.
- Each datalink must have a unique link name within the system.

Note – As an added restriction, you cannot use `lo0` as a flexible link name. This name is reserved to identify the IP loopback interface.

The function of the link within your network setup can be a useful reference when you assign link names. For example, `netmgt0` can be a link that is dedicated to network management. `Upstream2` can be the link that connects to the ISP. As a general rule to avoid confusion, do *not* assign names of known devices to your links.

Administration of Other Link Types

The separation between network configuration and network hardware configuration introduces the same flexibility to other types of link configurations. For example, virtual local area networks (VLANs), link aggregations, and IP tunnels can be assigned administratively-chosen names and then configured by referring to those names. Other related tasks, such as performing dynamic reconfiguration (DR) to replace hardware devices, are also easier to perform because no further network reconfiguration is required, provided that the network configuration was not deleted.

The following figure shows the interrelationship among devices, link types, and their corresponding interfaces.

Note – In the figure, the datalinks are named according to specific functions that they perform in the system, such as `video0` or `sales2`. The figure intends to highlight the flexibility with which you can name the datalinks. However, using the default neutral names such as `net0` as supplied by the OS is sufficient and preferable.

The figure also provides a sample of how administratively chosen names can be used in the network setup;

- VLANs are configured on the `net0` link. These VLANs, in turn, are also assigned customized names, such as `sales1` and `sales2`. The VLAN `sales2`'s IP interface is plumbed and operational.
- The device instances `qfe0` and `qfe2` are used to service video traffic. Accordingly, the corresponding links in the datalink layer are assigned the names `subvideo0` and `subvideo1`. These two links are aggregated to host video feed. The link aggregation possesses its own customized name as well, `video0`.
- Two interfaces (`net0` and `net1`) with different underlying hardware (`e1000g` and `qfe`) are grouped together as an IPMP group (`itops0`) to host email traffic.

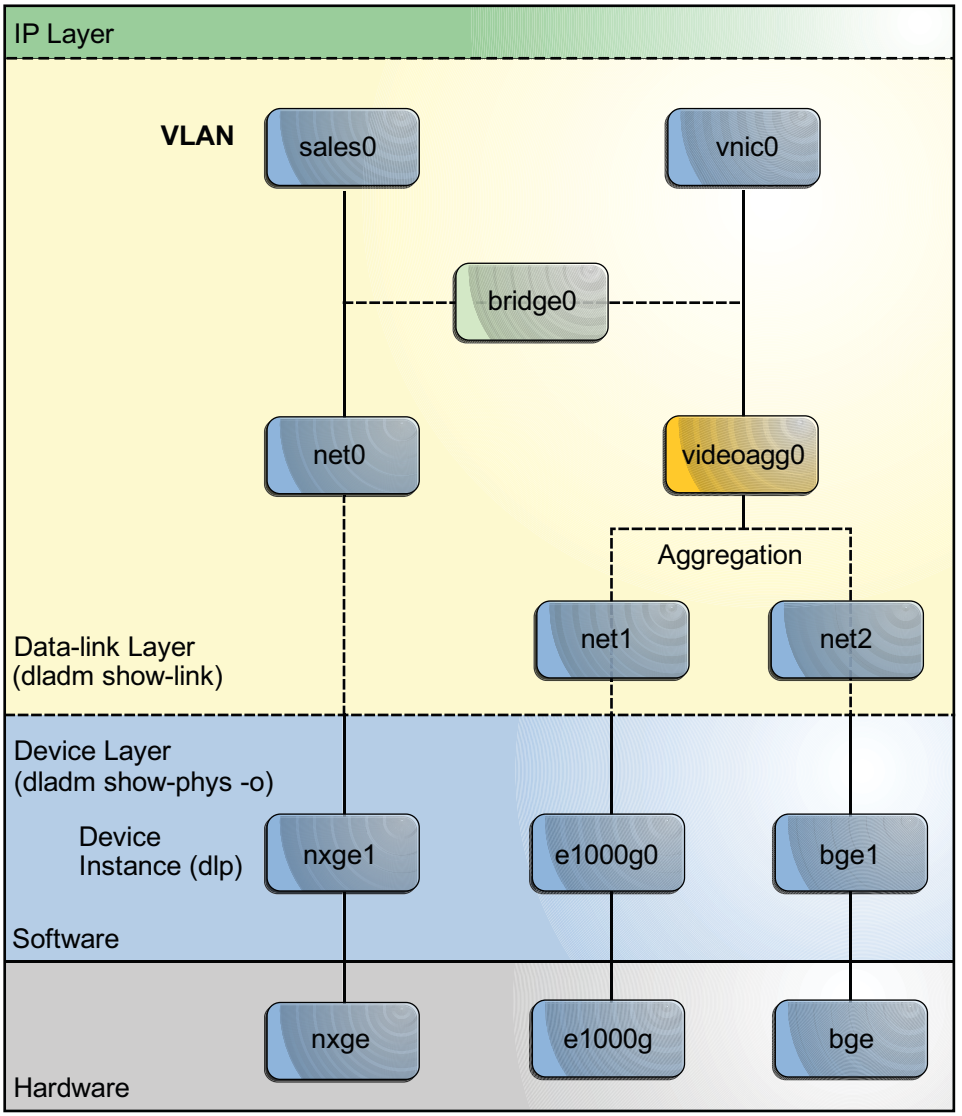
  Note – Although IPMP interfaces are not links on the datalink layer, these interfaces, like the links, can also be assigned customized names. For more information about IPMP groups, see Chapter 14, "Introducing IPMP."

- Two interfaces have no underlying devices: the tunnel `vpn1`, which is configured for VPN connections and `lo0` for IP loopback operations.

All of the link and interface configurations in this figure are independent of the configurations in the underlying hardware. For example, if the `qfe` card is replaced, the `video0` interface configuration for video traffic remains and can later be applied to a replacement NIC.

The following figure shows a bridge configuration. Two interfaces, `net0` and `videoagg0`, are configured as a bridge, `bridge0`. Packets that are received on one are forwarded to the other. After bridge configuration, both interfaces can still be used to configure VLANs and IP interfaces.

FIGURE 1–3 Bridges in the Network Stack



PART I

Network Auto-Magic

Network Auto-Magic (NWAM) is a feature of Oracle Solaris that automates the basic network configuration of your system. The topics that are covered in these chapters describe components of the NWAM architecture and how these components work together to effect automated network configuration on your Oracle Solaris system.

This documentation primarily focuses on how to manage your network configuration by using the NWAM command-line utilities. Also described is basic information about how to use the NWAM graphical user interface (GUI) to view and monitor the status of your network, as well as interact with NWAM from the desktop. Detailed instructions on monitoring and managing your network configuration by using the NWAM GUI can be found in the online help.

◆ ◆ ◆ CHAPTER 2

Introduction to NWAM

The Network Auto-Magic (NWAM) feature simplifies basic network configuration by automatically addressing basic Ethernet and WiFi configurations, such as connecting to your wired or wireless network at startup and displaying notifications about the status of your currently active network connection from the desktop. NWAM is also designed to simplify some of the more complex networking tasks, such as the creation and management of system-wide network profiles, for example, the configuration of naming services, IP Filter, and IP Security (IPsec), all of which are features of Oracle Solaris.

This chapter covers the following topics:

This chapter is intended for users and system administrators who have an understanding of basic networking concepts, as well as some experience managing network configuration by using traditional networking tools and commands. If you are ready to use NWAM to manage your network configuration, skip to Chapter 4, "NWAM Profile Configuration (Tasks)."

For basic information about administering network interfaces in Oracle Solaris, see Part II, "Datalink and Interface Configuration."

What Is an NWAM Configuration?

An NWAM configuration consists of several components that work together to effect the network configuration of a system in as automated of a manner as possible. With the primary focus on mobility, NWAM is capable of dynamically changing a system's configuration, in response to different network events, or at a user's request. NWAM includes dynamic

capabilities that address any changes in network conditions, for example, if your wired network interface becomes unplugged, or if a new wireless network becomes available.

Network configuration through NWAM is made up of properties and their values that are associated with several different types of profiles, which are also sometimes referred to as *configuration objects*.

These profiles and configuration objects include the following:

- **Network Configuration Profiles (NCPs)**

  An NCP specifies the configuration of network links and interfaces. This profile is one of the primary profile types that comprise an NWAM configuration. The second primary profile type is the Location profile.

  The system always defines an NCP called the Automatic NCP. This NCP is activated in the absence of input from the user. The Automatic NCP is created and maintained by the system and cannot be modified or removed.

  You can also create additional user-defined NCPs, as needed. For a complete description of the Automatic and user-defined NCPs, see "Description of the Automatic and User-Defined NCPs" on page 46.

- **Network Configuration Units (NCUs)**

  NCUs are the individual configuration objects that contain all of the properties that make up an NCP. The NCP is essentially a container that stores the NCUs that define it. Each NCU correlates to an individual link or interface in the system. For a complete description of an NCU, see "Description of an NCU" on page 45.

- **Locations**

  The Location profile is one of the two primary profile types that make up an NWAM configuration. The location specifies system-wide network configuration, for example, the naming services, the domain, the IP Filter, and IPsec configuration. This information consists of a set of properties that apply to system-wide network configuration. There are both system-defined and user-defined locations. For a complete description of the Location profile, see "Description of a Location Profile" on page 46.

- **External Network Modifiers (ENMs)**

  ENMs are profiles that are used to manage applications that are external to NWAM, for example the VPN application. These applications can modify and create network configuration. The `nwamd` daemon activates or deactivates an ENM, depending on conditions that are specified as a part of the ENM. For a complete description of an ENM, see "Description of an ENM" on page 47.

- **Known Wireless Local Area Networks (WLANs)**

  Known WLANs are configuration objects that NWAM uses to monitor and store information about wireless networks that are known to your system. NWAM maintains a list of all such wireless networks, then refers to this list to determine the order in which

connections to available wireless networks are attempted. For a complete description of known WLANs, see "About Known WLANs" on page 48.

NWAM Functional Components

NWAM consists of the following functional components:

- **NWAM profile repository** – The profile repository is where the NWAM configuration data is stored. Access to the profile repository is managed by the repository daemon, `netcfgd`.

  The NWAM profile repository includes a snapshot of your network configuration when NWAM is enabled. This data is preserved, in the event that you need to revert to manual configuration of your network. For more information, see "NWAM Configuration Data" on page 49.

- **Profile configuration programs (user interfaces)** – The NWAM architecture includes both a command-line interface (CLI) and a graphical user interface (GUI). These interfaces can be used to perform similar tasks, such as creating and modifying profiles, activating profiles, and querying the system for information about profiles.

  The NWAM CLI consists of two administrative commands, `netcfg` and `netadm`. The `netcfg` command enables you to create and modify profiles. This command operates in interactive mode, command-line mode, and command-file mode. The `netadm` command enables you to perform certain actions, for example, enabling or disabling a profile and listing information about profile states. For more information, see the `netcfg`(1M) and `netadm`(1M) man pages.

  For step-by-step instructions on creating and managing profiles by using the NWAM CLI, see Chapter 4, "NWAM Profile Configuration (Tasks)," and Chapter 5, "NWAM Profile Administration (Tasks)."

  The NWAM GUI can also be used to create and manage network profiles. The GUI has additional functionality that enables you to quickly view and monitor the status of network connections from the desktop. The GUI also has a notification feature that alerts you about changes in the current status of your network. The notification feature is only available in the GUI. To find about more about using the NWAM GUI, see Chapter 6, "About the NWAM Graphical User Interface," or refer to the online help. See also the `nwammgr`(1M) and the `nwammgr-properties`(1M) man pages.

- **Policy engine daemon** – The `nwamd` daemon is the policy component of NWAM. This daemon functions in multiple roles and manages your network configuration based on the profiles that are stored in the profile repository. The daemon determines which profile should be activated, depending on current network conditions, and then activates that profile. To accomplish this task, the daemon integrates information from multiple sources. The multiple roles that the `nwamd` daemon fulfills are described in detail in the section, "Overview of the NWAM Daemons" on page 65.

- **Repository daemon** – The `netcfgd` daemon controls the common profile repository that stores all of the configuration data for profiles and other configuration objects. The `netcfg` command, the NWAM GUI, and the `nwamd` daemon all interact with the `netcfgd` daemon by sending requests to access the profile repository. The repository daemon's job is to verify whether the various processes that are attempting to access the repository data have the correct authorizations. The daemon prohibits (fails) any access attempts by unauthorized processes. For more information, see "Description of the NWAM Repository Daemon (`netcfgd`)" on page 66.
- **NWAM library interface** – The `libnwam` library provides a functional interface to interact with the profile repository, thereby enabling information about profiles to be read and modified by NWAM.
- **Service Management Facility (SMF) network services** – Several network services that NWAM uses are already a part of Oracle Solaris. However, some of these existing services have been modified, and new services that are specific to NWAM, have been introduced. For more information, see "SMF Network Services" on page 67.

When to Use NWAM

Typically, if you change work environments and connection methods often (wired or wireless), you will want to take advantage of the automated network configuration capabilities of NWAM. You can use NWAM to set up user-defined profiles that enable you to connect to networks in a variety of settings, for example, the office, at home, or on the road. NWAM is a valuable tool for users of laptop models and systems that require frequent changes in network environments. In addition, the NWAM GUI makes the setting up of static IP configurations and connections to WiFi networks much easier than traditional networking tools and commands.

NWAM can be configured to adapt to changes in your network environment, such as loss of Ethernet connectivity or the addition or removal of a network interface card (NIC).

Note – You might choose to configure your network manually, for example, if you are using advanced networking features that are not currently supported by NWAM. For more information, see "Managing Network Configuration" on page 104.

How the NWAM Configuration Works

NWAM's default behavior is to perform basic configuration of your wired or wireless network "automagically", without any user interaction. The only time you are required to interact with NWAM is if you are prompted by the system for more information, for example, to provide a security key or password for a wireless network.

The automated NWAM configuration is triggered by the following events and activities:

- Connecting or disconnecting an Ethernet cable
- Connecting or disconnecting a WLAN card
- Booting a system when a wired interface, a wireless interface, or both, is available
- Resuming from suspend when a wired interface, a wireless interface, or both, is available (if supported)
- Acquiring or losing a DHCP lease

The NWAM components interact with each other in the following manner:

- At all times, one NCP and one Location profile must be active on the system.
- During a system boot, the policy engine daemon, `nwamd`, performs the following actions:
  1. Consults the service property for the currently active NCP
  2. Proceeds until one or more IP addresses have been configured
  3. Checks the conditions of the Location profiles
  4. Activates the Location profile that is specified by the policy engine
  5. Configures the network, or networks, accordingly
- As events that might trigger a change in the network configuration occur, the NWAM daemon, `nwamd`, functions in various roles and performs the following operations:
  1. As an event handler, `nwamd` detects each event as it occurs.
  2. As a profile daemon, `nwamd` consults the active profile.
  3. Depending on the change, `nwamd` might reconfigure the network, or networks, accordingly.

NWAM Default Behavior

In the absence of user-defined network profiles, `nwamd` manages network configuration based on the following three system-defined profiles:

- Automatic NCP
- Automatic location
- NoNet location

The Automatic NCP implements the following basic policy:

- Configure all available (connected) Ethernet interfaces by using DHCP.
- If no Ethernet interfaces are connected, or if none can obtain an IP address, activate one wireless interface, automatically connecting to the best available WLAN from the *Known WLAN list*. Or, wait for the user to select a wireless network to connect to.
- Until at least one IPv4 address has been obtained, the NoNet location remains active. This Location profile provides a strict set of IP Filter rules that only pass data that is relevant to IP address acquisition (DHCP and IPv6 `autoconf` messages). All of the properties of the NoNet location, with the exception of the activation conditions, can be modified.
- When at least one IPv4 address has been assigned to one of the system's interfaces, the Automatic location is activated. This Location profile has no IP Filter or IPsec rules. The Location profile applies the DNS configuration data that is obtained from the DHCP server. As with the NoNet location, all of the properties of the Automatic location, with the exception of its activation conditions, can be modified.
- The NoNet location is always applied when the system has no IPv4 addresses assigned to it. When there is at least one IPv4 address assigned, the system selects the Location profile with the activation rules that best match the current network conditions. In the absence of a better match, the system falls back to the Automatic location. For more information, see "How NWAM Profiles Are Activated" on page 54.

How NWAM Works With Other Oracle Solaris Networking Technologies

NWAM works with the following other Oracle Solaris networking technologies:

- **Network virtualization**

  NWAM works with the various Oracle Solaris network virtualization technologies as follows:

  - **Virtual machines: Oracle VM Server for SPARC (formerly Logical Domains) and Oracle VM VirtualBox**

    NWAM is supported in both Oracle Solaris hosts and guests. NWAM manages only the interfaces that belong to the specified virtual machines and does not interfere with other virtual machines.

  - **Oracle Solaris Zones and stack instances**

    NWAM works in global zones or in an exclusive stack, non-global zone.

    Note – NWAM does not work in a shared stack zone.

  - **VNICs**

Although the current NWAM implementation does not manage VNICs, manually created VNICs persist across reboots and can be created, for example, for assignment to an exclusive-stack zone.

- **Bridging technology**

  Bridging technology is a method of connecting separate network segments to enable communications between the attached nodes, as if only a single segment were in use. Although the current NWAM implementation does not actively support network configurations that use the bridging technology, you do not need to disable NWAM configuration management prior to using this technology on your system.

- **Dynamic Reconfiguration and Network Configuration Profiles**

  On systems that support dynamic reconfiguration (DR) and hot-plug capabilities, these features are readily used only if the active NCP on these systems is `DefaultFixed`.

  If the enabled NCP on these systems is `Automatic` or any other user-created NCP, then before performing any DR operations, you must first do one of the following steps:

  - Stop the network service. This action brings down all the network interfaces on the system. Therefore, you must use the system console to stop the service. After you have removed or replaced the device, restart the service.
  - Remove the IP interface from that active NCP's configuration by using the `netcfg` command. Then, you can proceed with physically removing or replacing that IP interface's underlying hardware device. If applicable, reconfigure the IP interface after DR is complete.

- **Traditional networking commands and utilities**

  At any given time, the system uses *either* traditional network configuration or NWAM network configuration. If the `DefaultFixed` NCP is enabled, the system uses traditional network configuration. The system applies the persistent configuration that is stored in the `/etc/ipadm/ipadm.conf` and `/etc/dladm/datalink.conf` files when this NCP is enabled. Also, you can use the `ipadm` and `dladm` commands to view and alter the network configuration. If an NWAM NCP is enabled, the system ignores the `/etc/ipadm/ipadm.conf` configuration, and NWAM manages the network configuration according to the policy that is specified in the active NCP.

  When NWAM manages network configuration, you can still use the command-line networking utilities, `dladm` and `ipadm`, to view the components of your current network configuration.

  Note – Making changes to network configuration by using command-line tools is not supported, as those changes might conflict with the policy that is enforced by NWAM.

- **IP Network Multipathing (IPMP)**

  NWAM does not currently support the use of IPMP. Before configuring your network to use IPMP, ensure that the `DefaultFixed` NCP is enabled.

Where to Find Network Configuration Tasks

The following table lists network configuration topics and where to go for more information.

Networking Task	For More Information
Find detailed overview information about NWAM.	Chapter 3, "NWAM Configuration and Administration (Overview)"
Create, modify, and remove profiles and configuration objects by using the NWAM CLI.	Chapter 4, "NWAM Profile Configuration (Tasks)"
View information about and administer profiles and configuration objects by using the NWAM CLI.	Chapter 5, "NWAM Profile Administration (Tasks)"
View information about your network status, switch network connections, and create and modify profiles and configuration objects by using the NWAM GUI from the desktop.	Chapter 6, "About the NWAM Graphical User Interface," and the online help
Switch between NWAM network configuration mode and traditional network configuration mode.	"Managing Network Configuration" on page 104
Manage your network configuration by using traditional networking tools and commands.	Chapter 8, "Datalink Configuration and Administration," and Chapter 9, "Configuring an IP Interface"
Configure and manage virtual networks.	Chapter 17, "Introducing Network Virtualization and Resource Control (Overview)"

◆ ◆ ◆ CHAPTER 3

NWAM Configuration and Administration (Overview)

This chapter provides background and overview information about the NWAM configuration and administration process. A detailed description of the profiles implementation that NWAM uses to simplify and automate network configuration is also provided.

This chapter covers the following topics:

Overview of NWAM Configuration

NWAM manages network configuration by storing preferred property values in the form of profiles on the system. NWAM then determines which profile should be activated, depending on current network conditions, and subsequently activates that profile. The NWAM profiles implementation is a primary component of NWAM.

What Are Network Profiles?

Network profiles are collections of properties that determine how the network is configured and how it operates, depending on current network conditions.

The following are the profile types and configuration objects that comprise NWAM configuration:

- Network Configuration Profiles (NCPs)
- Location profiles
- External Network Modifiers (ENMs)
- Known WLANs

The two primary network profile types are the NCP and the Location profile. To effect autoconfiguration of the network through NWAM, exactly one NCP and one Location profile must be active on the system at all times.

The NCP specifies the configuration of the local network, including the configuration of individual components, such as physical links and IP interfaces. Each NCP consists of individual configuration objects that are called *Network Configuration Units* (NCUs). Each NCU represents a physical link or an interface and is made up of properties that define the configuration for that link or interface. The process of configuring a user-defined NCP involves creating NCUs for that NCP. For more information, see "Description of an NCU" on page 45.

A Location profile contains system-wide network configuration information, such as the following:

- Conditions under which the Location profile is activated
- Which naming service to use
- Domain name
- Set of IP Filter rules
- IPsec policy

For more information, see "Description of a Location Profile" on page 46.

ENMs are NWAM profiles that are for external applications that are capable of creating and modifying network configuration. NWAM can be configured to activate and deactivate these external applications under conditions that you specify when you create the ENM.

Known WLANs are NWAM profiles that are used to maintain a list of known wireless networks to which you have connected previously. For more information, see "Description of an ENM" on page 47 and "About Known WLANs" on page 48.

Description of an NCP

An NCP defines the network configuration of a system. The NCUs that make up an NCP specify how to configure the various network links and interfaces, for example, which interface or interfaces should be brought up, and under what conditions that interface should be brought up, as well as how the IP address for the interface is obtained. There are two NCP types: Automatic and user-defined. The Automatic NCP is a system-defined profile that is automatically created by NWAM. This profile cannot be created, modified, or removed.

User-defined NCPs are profiles that you create to meet the needs of your particular network configuration. A user-defined NCP can be modified and removed by the user.

The Automatic NCP is a representation of all of the links and interfaces that are currently in the system. The content of the Automatic NCP changes if network devices are added or removed. However, the configuration preferences that are associated with the Automatic NCP cannot be edited. The Automatic NCP is created to provide access to a profile that utilizes DHCP and address autoconfiguration that make it possible to obtain IP addresses for the system. This profile also implements a link selection policy that favors wired links over wireless links. If the specification of an alternate IP configuration policy, or an alternate link selection policy is required, you would create additional user-defined NCPs on your system.

Description of an NCU

NCUs are the individual configuration objects that make up an NCP. NCUs represent the individual physical links and interfaces that are on a system. The process of configuring a user-defined NCP includes creating NCUs that specify how and under what conditions each link and interface should be configured.

There are two types of NCUs:

- **Link NCUs**

  Link NCUs, for example, physical devices, are Layer 2 entities in the Open Systems Interconnection (OSI) model.

- **Interface NCUs**

  Interface NCUs, specifically, IP interfaces, are Layer 3 entities in the OSI model.

Link NCUs represent data links. There are several different classes of data links:

- Physical links (Ethernet or WiFi)
- Tunnels
- Aggregations
- Virtual local area networks (VLANs)
- Virtual network interface cards (VNICs)

Note – The current NWAM implementation includes support for basic network configuration of physical links (Ethernet and WiFi) *only*. Although not actively supported by NWAM, several advanced networking technologies, such as VNICs and bridging, can be configured on your network without requiring that you disable NWAM configuration management.

However, if you configure your system to use IP Network Multipathing (IPMP), you cannot use NWAM configuration management. You must use the traditional network configuration. For instructions, see "How to Switch From Automatic Network Configuration Mode to Manual Network Configuration Mode" on page 104.

Description of the Automatic and User-Defined NCPs

The Automatic NCP is a system-defined profile that is made up of one link NCU and one interface NCU for each physical link that is present in the system. The NCU activation policy in this NCP is to prefer connected, wired links over wireless links, and to plumb both IPv4 and IPv6 on each enabled link. DHCP is used to obtain IPv4 addresses. Stateless Autoconf and DHCP are used to obtain IPv6 addresses. The Automatic NCP changes dynamically when new links are inserted or removed from the system. All NCUs that correspond to the inserted or removed link are also added or removed at the same time. The profile is updated automatically by the `nwamd` daemon.

User-defined NCPs are created and managed by the user. You must explicitly add and remove NCUs from the specified profile. You can create NCUs that do not correlate to any link that is currently present in the system. You can also remove NCUs that do not correlate to any link that is present in the system. In addition, you can determine the policy for the user-defined NCP. For example, you can allow multiple links and interfaces to be enabled on the system at a given time, as well as specify different dependency relationships between NCUs and static IP addresses.

For step-by-step instructions on creating a user-defined NCP and adding and removing NCUs to this NCP, see "Creating an NCP" on page 74.

Description of a Location Profile

A Location profile provides additional networking details after the basic IP connectivity has been established. Locations contain network configuration information that is comprised of a set of properties that relate to network configuration on a system-wide level.

A Location profile consists of certain network configuration information, for example, a naming service and firewall settings, that are applied together, when required. Also, because a location does not necessarily correspond to a physical location, you can set up several Location profiles to meet different networking needs. For example, one location can be used when you

are connected to the company intranet. Another location can be used when you are connected to the public Internet by using a wireless access point that is located in your office.

By default, two Location profiles are predefined by the system:

- **NoNet**

  The NoNet location has very specific activation conditions. This profile is applied by NWAM to a stand-alone system when no local interfaces have an assigned IP address. You can modify the NoNet location after it is activated on your system for the first time. A read-only copy of the original NoNet location is stored on the system, in case you want to restore the default settings for this location.

- **Automatic**

  The Automatic location is activated if there are networks available, but no other Location profile supersedes it. You can modify the Automatic location after it has been activated on your system for the first time. A read-only copy of the original Automatic location is stored on the system, in case you want to restore the default settings for this location.

Note – The Automatic location should not be confused with the Automatic NCP. The Automatic location is a Location profile type that defines system-wide network properties after the initial network configuration of a system takes place. The Automatic NCP specifies link and interface network configuration on a system.

User-defined locations are profiles that you create with values that you specify for system-wide network configuration. User-defined locations are identical to system-defined locations, except that a user-defined location is configured with values that you set, while system-defined locations have preset values.

For more information about creating user-defined locations, see "Creating a Location Profile" on page 81.

Description of an ENM

ENMs are profiles that pertain to applications that are external to NWAM. These applications can create and modify network configuration. ENMs are included in the NWAM design as a means of creating and removing customized network configuration that is not an NCP or a Location profile. An ENM can also be defined as a service or application that directly modifies network configuration when it is enabled or disabled. You can configure NWAM to activate and deactivate ENMs under conditions that you specify. Unlike an NCP or a Location profile, where only one of each profile type can be active on the system at any given time, multiple ENMs can potentially be active on the system at the same time. The ENMs that are active on a system at any given time are not necessarily dependent on the NCP or Location profile that is also enabled on the system at the same time.

Although there are several external applications and services for which you can create an ENM, the obvious example is the VPN application. After you install and configure VPN on your system, you can create an ENM that automatically activates and deactivates the application under the conditions that you specify.

Note – It is important to understand that NWAM does not have the capability to automatically learn about external applications that are capable of directly modifying the network configuration on a system. To manage the activation or deactivation of a VPN application, or any external application or service, you must first install the application, then create an ENM for it by using either the CLI or the NWAM GUI.

Persistent information about any network configuration that is performed by an ENM is not stored or tracked by NWAM in exactly the same way that information about an NCP or a Location profile is stored. However, NWAM is capable of noting an externally initiated network configuration, and then based on any configuration changes that are made to the system by an ENM, reevaluating which Location profile should be active, and subsequently activating that location. An example would be switching to a location that is activated conditionally when a certain IP address is in use. If the `svc:/network/physical:default` service is restarted at any time, the network configuration that is specified by the active NCP is reinstated. ENMs are restarted as well, possibly tearing down and recreating network configuration in the process.

For information about creating and modifying the properties of an ENM, see "Creating an ENM Profile" on page 86.

About Known WLANs

Known WLANs are configuration objects that NWAM uses to manage wireless networks that are known to the system. NWAM maintains a global list of these known wireless networks. This information is then used to determine the order in which NWAM attempts to connect to available wireless networks. If a wireless network that exists in the *Known WLAN list* is available, NWAM automatically connects to that network. If two or more known wireless networks are available, NWAM attempts to connect to the wireless network with the highest priority (lowest number). Any new wireless network that NWAM connects to is automatically added to the top of the known WLAN list and becomes the current highest priority wireless network.

Known WLANs are selected in priority order, with a priority that is assigned by an unsigned integer. A lower number indicates a higher priority in the known WLAN list. The first time you connect to a wireless network, NWAM automatically adds that WLAN to the list. When a new WLAN is added, it assumes the highest priority in this list. The NWAM default behavior is to prefer more recently connected WLANs over WLANs that you connected to previously. At no time can any known WLANs share the same priority. If a new WLAN is added to the list with

the same priority value as an existing WLAN, the existing entry is shifted to a lower priority value. Subsequently, the priority value of every other WLAN in the list is dynamically shifted to a lower priority value.

One or more key names can also be associated with a known WLAN. *Key names* enable you to create your own keys by using the `dladm create-secobj` command. You can then associate these keys with WLANs by adding the secure object names to the known WLAN `keyname` property. For more information, see the `dladm(1M)` man page.

For more information about using the NWAM command-line utilities to manage WLANs, see "Performing a Wireless Scan and Connecting to Available Wireless Networks" on page 113.

NWAM Configuration Data

There are effectively two configuration repositories on the system: the NWAM profile repository, which is stored in the `/etc/nwam` directory, and the traditional configuration repository, which includes the `/etc/ipadm/ipadm.conf` and `/etc/dladm/datalink.conf` files, as well as other configuration files that are associated with network services.

When NWAM manages network configuration, it works primarily from its own repository. The interface configuration that is stored in the `/etc/ipadm/ipadm.conf` file is ignored. NWAM configures physical links and interfaces directly based on the NCP data.

Location profile data is read from the NWAM profile repository. When a location is activated, this configuration is applied to the running system in most cases by setting the appropriate SMF service properties and restarting the corresponding services to apply the configuration changes. This action overwrites existing values for those service properties.

Because NWAM overwrites legacy configuration data in the process of applying Location profiles, upon startup, any configuration that might be overwritten is saved. NWAM then restores that configuration upon shutdown. Although it is not a location that can be applied as part of the NWAM operation, this data is referred to as the *Legacy location* data.

Property values for the following system-defined and user-defined network profiles are stored in the NWAM repository:

- NCPs – Contains values for the Automatic NCP, as well as any user-defined NCPs
- NCUs – Contains values for both link and interface NCUs
- Locations – Contains values for the three system-defined location types, as well as values for any user-defined locations
- ENMs – Contains information about applications
- Known WLANs – Contains information about wireless networks that you might be connected to automatically

Configuration data for each NCP is stored persistently as a file in the `/etc/nwam` directory, using the format, `ncp-`*name*. There is one file per NCP, with the entries representing each NCU. For example, the file for the Automatic NCP is named `ncp-Automatic.conf`. All NCP files are stored in the `/etc/nwam` directory.

Location properties are stored in the `/etc/nwam/loc.conf` file.

ENM properties are stored in the `/etc/nwam/enm.conf` file. Known WLANs are stored in the `/etc/nwam/known-wlan.conf` file. This file format is similar to the file format of the `/etc/dladm/datalink.conf` file.

Note – Although it is possible to modify network profiles by directly editing the files in the NWAM profile repository, the appropriate way to modify a profile is to use the `netcfg` command or the NWAM GUI configuration panels. The file format and the use of files might change in future releases. See "Setting and Changing Property Values for a Profile" on page 92.

NCU Property Values

NCUs, the individual configuration objects of an NCP, represent individual links and interfaces on a system. General properties for both NCU types (link and interface), as well as properties that are specific to each NCU type, are stored in the NWAM profile repository. The `type`, `class`, and `parent` properties are set when the NCU is created and cannot be changed later. Also, you cannot directly change an `enabled` property. The property is changed indirectly by enabling or disabling an NCU by using the `netadm` command.

The Automatic NCP consists of one link NCU for each physical link that is discovered in the system and one interface NCU that is plumbed on each link. The Automatic NCP changes dynamically upon insertion of additional physical links. As new links are inserted, a link NCU and corresponding interface NCU are created for each new link. The following tables define the values that are assigned to each NCU that makes up the Automatic NCP.

Note – The properties in this table are listed in the order in which they appear when viewing the NCU properties of the Automatic NCP. Certain values apply to each NCU type.

TABLE 3–1 Link NCU Properties for the Automatic NCP

Property	Link NCU Value
`type`	`link`
`class`	`phys`
`parent`	`Automatic`
`enabled`	`true`

TABLE 3–1 Link NCU Properties for the Automatic NCP *(Continued)*

Property	Link NCU Value
`activation-mode`	`prioritized`
`priority-group`	`0` (for 802.3 links) or `1` (for 802.11 links)
`priority-group-mode`	`shared` (for 802.3 links) or `exclusive` (for 802.11 links)
`mac-address`	Hardware-assigned
`autopush`	N/A
`MTU`	N/A

TABLE 3–2 Interface NCU Properties for the Automatic NCP

Property	Interface NCU Value
`type`	`interface`
`class`	`IP`
`parent`	`Automatic`
`enabled`	`true`
`ip-version`	`ipv4`, `ipv6`
`ipv4-addrsrc`	`dhcp`
`ipv4-static-addr`	N/A
`ipv6-addrsrc`	`dhcp`, `autoconf`
`ipv6-static-addr`	N/A

Property Values of System-Defined Locations

The following table provides the default property values for the Automatic location, which is a system-defined profile. You can modify these values, with the exception of the `activation-mode` and the `enabled` properties. The system always activates the Automatic location when at least one interface is active and no other Location profile supersedes it.

TABLE 3–3 Properties of System-Defined Locations

Property	Value
`name`	`Automatic`
`activation-mode`	`system`

TABLE 3–3 Properties of System-Defined Locations *(Continued)*

Property	Value
`enabled`	`system` modified, as required
`conditions`	N/A
`default-domain`	N/A
`nameservices`	`dns`
`nameservices-config-file`	`/etc/nsswitch.dns`
`dns-nameservice-configsrc`	`dhcp`
`dns-nameservice-domain`	N/A
`dns-nameservice-servers`	N/A
`dns-nameservice-search`	N/A
`nis-nameservice-configsrc`	N/A
`nis-nameservice-servers`	N/A
`ldap-nameservice-configsrc`	N/A
`ldap-nameservice-servers`	N/A
`nfsv4-domain`	N/A
`ipfilter-config-file`	N/A
`ipfilter-v6—config-file`	N/A
`ipnat-config-file`	N/A
`ippool-config-file`	N/A
`ike-config-file`	N/A
`ipsecpolicy-config-file`	N/A

The following table provides the predefined properties for the NoNet location. Note that you can modify these values, with the exception of the `activation-mode` and `enabled` properties. The system always enables the `NoNet` location when there are no active interfaces.

TABLE 3–4 Properties of the NoNet Location

Property	Value
`name`	`NoNet`
`activation-mode`	`system`

TABLE 3–4 Properties of the NoNet Location *(Continued)*

Property	Value
`enabled`	`system` modified, as required
`conditions`	N/A
`default-domain`	N/A
`nameservices`	`files`
`nameservices-config-file`	`/etc/nsswitch.files`
`dns-nameservice-configsrc`	N/A
`dns-nameservice-domain`	N/A
`dns-nameservice-servers`	N/A
`dns-nameservice-search`	N/A
`nis-nameservice-configsrc`	N/A
`nis-nameservice-servers`	N/A
`ldap-nameservice-configsrc`	N/A
`ldap-nameservice-servers`	N/A
`nfsv4-domain`	N/A
`ipfilter-config-file`	`/etc/nwam/loc/NoNet/ipf.conf`, which consists of IP Filter rules that block all non-loopback traffic, with the exception of a minimum amount of network traffic that is required by NWAM to perform network configuration, such as DHCP address assignment.
`ipfilter-v6–config-file`	`/etc/nwam/loc/NoNet/ipf6.conf`, which consists of IP Filter rules, as described for the `ipfilter-config-file`.
`ipnat-config-file`	N/A
`ippool-config-file`	N/A
`ike-config-file`	N/A
`ipsecpolicy-config-file`	N/A

For more information about location properties, including the properties that make up user-defined locations, see the `netcfg(1M)` man page.

How NWAM Profiles Are Activated

NCPs, Location profiles, and ENMs have `activation-mode` properties. The allowable values for each profile type differ. In addition, how the `activation-mode` property is validated differs for each profile type, as do the conditions under which each profile is activated.

For system-defined locations (Automatic and NoNet), the `activation-mode` property value is set to `system`, which means that the location can only be activated by the system, under those conditions that the system has predetermined are appropriate for the given location.

For user-defined locations, you can set the `activation-mode` and `conditions` properties to `manual`, `conditional-any`, or `conditional-all`. For more information, see "Location Activation Selection Criteria" on page 56.

A Location profile can be manually enabled by using the `netadm` command or by using the NWAM GUI. If you do not explicitly enable a location, the NWAM daemon, `nwamd`, checks the activation rules for all of the conditionally activated and system-activated Location profiles, and then chooses the location that best matches the current network environment.

NWAM uses an algorithm to determine the "best match" for a location choice. If there is no suitable match for a location, the Automatic location is then activated. Changes in the network environment cause the `nwamd` daemon to continually reassess the location selection to determine the best match. However, if you explicitly enable a Location profile by using the `netadm` command (either a location that is manually activated or a location that is conditionally activated), that location remains active until you explicitly disable it or enable a different location. In this situation, changes in the network environment do not result in a change in Location profiles, regardless of whether there might be a better match available. The fact that you explicitly specified the current location makes it, in effect, the best possible match. For instructions on activating and deactivating profiles, see "Activating and Deactivating Profiles" on page 110.

NCP Activation Policy

NWAM enables you to specify NCP policy, in terms of when NCUs are activated. The NCP policy is enforced through the use of properties and conditions that can be specified for each NCU. Examples of policies that you might specify include: "prefer wired connections over wireless connections" or "activate one interface at a time." How and when NCPs are activated is defined in the properties that are set for each NCU type.

Note – An interface NCU must always be associated with an underlying link NCU. Each interface NCU becomes active when its associated link NCU is activated. You can override the default behavior of an NCU by using the `netadm` command. However, the dependency on the underlying link NCU can never be removed. For example, if you enable an interface NCU without enabling its associated link NCU, the interface will not actually come online until the underlying NCU for that interface is activated.

Example of an NCP Policy

In the following example, NCU properties are set for when the NCP policy needs to specify that all of the available wired links are activated, and that a wireless connection should only be used if no wired connection is available.

For all physical links:

- NCU type: `link`
- NCU class: `phys`
- `activation-mode`: `prioritized`
- `priority-group`: `0` for wired; `1` for wireless
- `priority-mode`: `shared` for wired; `exclusive` for wireless

In the following example, NCU properties are set according to an NCP policy that specifies that there be only one active link on the system at any given time, and that a wired connection is preferred over a wireless connection.

For all physical links:

- NCU type: `link`
- NCU class: `phys`
- `activation-mode`: `prioritized`
- `priority-group`: `0` for wired; `1` for wireless
- `priority-mode`: `exclusive`

NCU Activation Properties

How network connections are activated is set in the link NCU properties. The following properties are used to define the NCP activation policy:

- `activation-mode` property

  This property can be set to either `manual` or `prioritized`.

  - `manual` – The NCU activation is managed by the administrator. You can use the NWAM CLI or the GUI to activate or deactivate the NCU. If an NCU's `activation-mode` is set to `manual`, values that are set for both the `priority-group` and `priority-mode` NCU properties are ignored.

- `prioritized` – The NCU is activated according to the values that are set in the `priority-group` and `priority-mode` properties for the specified NCU. The enabled property is always true for prioritized NCUs.

  Prioritized activation enables groups of links to be activated at the same time. This activation mode also enables one or more links to be preferred over other links. The `priority-group` property assigns a numeric priority level to a given link. All links at the same priority level are examined as a group. The `priority-mode` property defines how many of the group members might or must be available for the group to be activated.

- `enabled` property (`activation-mode` is set to `manual`)

  The value of this property can be `true` or `false`. You cannot set the value of this property. Rather, the value reflects the current state of a manually enabled NCU, which can be changed by using the `netadm` command or by using the NWAM GUI.

- `priority-group` property (`activation-mode` is set to `prioritized`)

  The value is numeric. Zero (`0`) indicates the highest priority. Negative values are invalid.

  Among all of the available `priority-groups`, only the NCUs in the highest available `priority-group` are activated. When more than one NCU with the same priority is available, activation behavior is defined by the `priority-mode` property. The priority number is not an absolute value. It can change, as the NCP repository is updated.

  Note – The priority order is strictly enforced.

- `priority-mode` property (`activation-mode` is set to `prioritized`)

  The property is set when a value for the `priority-group` property has been specified.

  The values for this property are as follows:

  - `exclusive` – Specifies that only one NCU in the `priority-group` can be active at any given time. NWAM activates the first available NCU within the priority group and ignores the other NCUs.
  - `shared` – Specifies that multiple NCUs in the priority group can be active at the same time. Any available NCUs in the priority group are activated.
  - `all` – Specifies that all of the NCUs in the priority group must be made available for the priority group to be considered available and thus made active.

Location Activation Selection Criteria

Each Location profile contains properties that define activation criteria. These properties specify information about the conditions under which a location is activated. NWAM continuously reevaluates the selection criteria for all the configured locations, each time determining which location has the criteria that is the best match for the current network

environment. If changes take place in the current network environment that result in a better criteria match, NWAM deactivates the current Location profile and activates the Location profile that is the better match for the new environment.

The selection criteria for when and how a location is activated are specified by the following properties:

- `activation-mode`
- `conditions`

The `activation-mode` property is set to one of the following possible values:

- `manual`
- `conditional-any`
- `conditional-all`
- `system`

Note – The `system` value of the `activation-mode` property can only be assigned to system-provided locations: the Automatic and NoNet locations. The `system` value indicates that the system determines when to activate these locations.

If the `activation-mode` property is set to `conditional-any` or `conditional-all`, the `conditions` property contains a conditional expression (or expressions) that are user-defined. Each expression contains a condition that can be assigned a boolean value, for example, "ncu ip:net0 is-not active."

If the `activation-mode` property is set to `conditional-any`, the condition is satisfied if any one of the conditions is true.

If the `activation-mode` property is set to `conditional-all`, the condition is satisfied only if *all* of the conditions are true. The criteria and operations that can be used to construct the condition strings are defined in the following table.

TABLE 3–5 Criteria and Operations for Constructing Condition Strings

Object Type/Attribute	Condition	Object
`ncu, enm, loc`	is/is-not active	name
`essid`	is/is-not contains/does-not-contain	name string
`bssid`	is/is-not	bssid string
`ip-address`	is/is-not	IPv4 or IPv6 address
`ip-address`	is-in-range/is-not-in-range	IPv4 or IPv6 address plus netmask/prefixlen

TABLE 3–5 Criteria and Operations for Constructing Condition Strings *(Continued)*

Object Type/Attribute	Condition	Object
`advertised-domain`	is/is-not contains/does-not-contain	name string
`system-domain`	is/is-not contains/does-not-contain	name string

Note – The `essid` property represents an Extended Server Set Identifier (ESSID), which is the network name of a wireless LAN (WLAN). The `bssid` property represents a Basic Service Set Identifier (BSSID), which is the MAC address of a specific wireless access point (WAP) or any access point (AP).

Note the distinction between the `advertised-domain` and the `system-domain` attributes. The advertised domain is discovered through external communications, for example, the `DNSdmain` or `NISdmain` domain names, which are advertised by a DHCP server. This attribute is useful for the conditional activation of locations, for example, if the advertised domain is `mycompany.com`, then activate the `work` location. The `system-domain` attribute is the domain that is currently assigned to the system. It is the value that is returned by the `domainname` command. This attribute is useful for the conditional activation of ENMs, as it will only become true after a location has been activated, and the system has been configured for that particular domain. For more information, see the `domainname(1M)` man page.

For more information about location properties, see "Description of a Location Profile" on page 46.

Configuring Profiles by Using the `netcfg` Command

The `netcfg` command, which is described in the `netcfg(1M)` man page, is used to configure properties and values of network profiles.

You can use the `netcfg` command to perform the following tasks:

- Create or destroy a user-defined profile.

  Note – You cannot create or destroy a system-defined profile.

- List all of the profiles that exist on a system and their property values.
- List all of the property values and resources for a specified profile.

- Display each property that is associated with a profile.
- Set or modify one or all of the properties of a specified profile.
- Export the current configuration for a user-defined profile to standard output or a file.

Note – You cannot export a system-defined profile.

- Delete any changes that were made to a profile and revert to the previous configuration for that profile.
- Verify that a profile has a valid configuration.

You can use the `netcfg` user interface in interactive mode, command-line mode, or command-file mode. Because the `netcfg` command is hierarchical, it is more easily understood when used in the interactive mode.

The concept of a *scope* is used for the `netcfg` command. When you use the command interactively, the scope you are in at any given time depends on the profile type and the task that you are performing. When you type the `netcfg` command in a terminal window, a prompt is displayed at the *global scope*.

From here, you can use the `select` or `create` subcommands to view, modify, or create the following top-level profiles:

- NCPs
- Locations
- ENMs
- Known WLANs

Before creating or selecting a profile, the `netcfg` interactive prompt is displayed in the following form:

```
netcfg>
```

After you have created or selected a profile, the `netcfg` interactive prompt is displayed as follows:

`netcfg:`*profile-type*`:`*profile-name*`>`

Note – In command-line mode, you must type the complete command on a single line. Changes that you make to a selected profile by using the `netcfg` command in command-line mode are committed to the persistent repository as soon as you finish typing the command.

For step-by-step instructions on using the `netcfg` command, see Chapter 4, "NWAM Profile Configuration (Tasks)." For more information about using the `netcfg` command, see the `netcfg(1M)` man page.

netcfg Interactive Mode

Selecting or creating a top-level profile while working in the netcfg interactive mode results in a command prompt that is displayed in the *profile scope* for Location profiles and ENMs. For example:

```
netcfg> select loc foo
netcfg:loc:foo>
```

If an NCP is selected, the command prompt is displayed is in the *NCP scope*. From the NCP scope, an NCU can be selected or created. Selecting or creating an NCU results in a profile scope prompt for the selected NCU. In this scope, all of the properties that are associated with the currently selected profile can be viewed and set, as shown in the following example where the User NCP was first selected, then an NCU was created in the NCP scope. This action resulted in the profile scope for the newly created NCU. In this scope, the properties of the NCU can be viewed or set:

```
netcfg> select ncp User
netcfg:ncp:User> create ncu phys net2
Created ncu 'net2'.  Walking properties ...
activation-mode (manual) [manual|prioritized]>
```

At any given scope, the command prompt indicates the currently selected profile. Any changes that you make to the profile in this scope can be *committed*, meaning the changes are saved to the persistent repository. Changes are implicitly committed upon exiting the scope. If you do not want to commit the changes that you made to the selected profile, you can revert to the last committed state for that profile. Doing this action reverts any changes that you made to the profile at that level. The revert and cancel subcommands work similarly.

netcfg Command-Line Mode

In command-line mode, any subcommands that affect a selected profile or property must be performed in the particular scope that the selected profile or property exists. For example, to obtain the value of a property of an NCU, you would use the get subcommand in the scope of that particular NCU. When you are in the netcfg interactive mode, it is fairly easy to understand the syntax to use for this command. However, in command-line mode, the syntax might be less obvious.

For example, to obtain the value of a property "foo," which is an attribute of an NCU called myncu in the User NCP, you would use the following syntax:

```
$ netcfg "select ncp User; select ncu ip myncu; get foo"
```

In this example, note the following information:

- Each scope is separated by a semicolon.
- The `select` subcommand is issued at each scope, once at the global scope and once at the profile scope.
- The `get` subcommand is used within the scope in which the property "foo" exists.
- Straight quotation marks are required to prevent the shell from interpreting semicolons.

`netcfg` Command-File Mode

In command-file mode, configuration information is taken from a file. The `export` subcommand is used to produce this file. The configuration can then be printed to standard output or the `-f` option can be used to specify an output file. The `export` subcommand can be used interactively also. For more information, see "`netcfg` Supported Subcommands" on page 61.

`netcfg` Supported Subcommands

The following `netcfg` subcommands are supported in interactive mode and command-line mode. Note that certain subcommands have different semantics within each scope. If a subcommand cannot be used in a certain mode, it has been noted in the subcommand's description.

- `cancel`

  Ends the current profile specification without committing the current changes to persistent storage, then proceeds to the previous scope, which is one level higher.

- `clear` *prop-name*

  Clears the value for the specified property.

- `commit`

  Commits the current profile to persistent storage. A configuration must be correct to be committed. Therefore, this operation automatically performs a `verify` on the profile or object, as well. The `commit` operation is attempted automatically upon exiting the current scope by using either the `end` or `exit` subcommand.

- `create` [`-t` *template*] *object-type* [*class*] *object-name*

  Creates an in-memory profile with the specified type and name. The `-t` *template* option specifies that the new profile be identical to *template*, where *template* is the name of an existing profile of the same type. If the `-t` option is not used, the new profile is created with the default values.

- `destroy -a`

  Removes all user-defined profiles from memory and persistent storage.

- `destroy` *object-type* [*class*] *object-name*

  Removes the specified user-defined profile from memory and persistent storage.

Caution – This operation is immediate and does not need to be committed. A destroyed profile cannot be reverted.

- `end`

  Ends the current profile specification and proceeds to the previous scope, which is one level higher. The current profile is verified and committed before ending the edit operation. If either the `verify` or `commit` operation fails, an error message is displayed. You are then given the opportunity to end the operation without committing the current changes. Or, you can remain in the current scope and continue editing the profile.

- `exit`

  Exits the `netcfg` interactive session. The current profile is verified and committed before the current session ends. If either the `verify` or `commit` operation fails, an error message is displayed. You are then given the opportunity to end the session without committing the current changes. Or, you can remain in the current scope and continue editing the profile.

- `export` [`-d`] [`-f` *output-file*] [*object-type* [*class*] *object-name*]

  Prints the current configuration at the current or specified scope to standard output or to a file that is specified with the `-f` option. The `-d` option generates the `destroy -a` subcommand as the first line of output. This subcommand produces output in a form that is suitable for use in a command file.

Note – System-defined profiles, including the Automatic NCP and the Automatic, NoNet, and Legacy locations, cannot be exported.

- `get` [`-V`] *prop-name*

  Gets the current, in-memory value of the specified property. By default, both the property name and value are printed. If the `-V` option is specified, only the property value is printed.

- `help` [*subcommand*]

  Displays general help or help about a specific subject.

- `list` [`-a`] [*object-type* [*class*] *object-name*]

  Lists all profiles, property-value pairs and resources that will be used at the current or specified scope. If the `-a` option is specified, all properties are listed, including those that will be ignored, based on current settings.

- `revert`

  Deletes any current changes that were made to a profile, then reverts to the values from persistent storage.

- `select` *object-type* [*class*] *object-name*

  Selects the object that is specified.

- `set` *prop-name=value*

  Sets the current, in-memory value of the specified property.

  If performed in command-line mode, the change is also committed immediately to persistent storage.

  The delimiter for multi-valued properties is a comma (`,`). If an individual value for a given property contains a comma, it must be preceded it with a backslash (`\`). Commas within properties that only have a single value are not interpreted as delimiters and do not need to be preceded by a backslash.

- `verify`

  Verifies that the current, in-memory profile or object has a valid configuration.

- `walkprop` [`-a`]

  "Walks" each property that is associated with the current profile. For each property, the name and current value are displayed. A prompt is provided to enable you to change the current value. If a property is not used, based on the previously specified values, the property is not displayed. For example, if the `ipv4-addrsrc` property is set to `static`, the `ipv4-addr` property is not used, and is not walked or listed, unless you specify the `-a` option.

  When used, the `-a` option iterates all available properties for the specified profile or object.

  The delimiter for multi-valued properties is a comma (`,`). If an individual value for a given property contains a comma, it must be preceded by a backslash (`\`). Commas within properties that only have a single value are not interpreted as delimiters and do not need to be preceded by a backslash.

  Note – This subcommand is meaningful when used in interactive mode only.

For task-related information, see Chapter 4, "NWAM Profile Configuration (Tasks)."

Administering Profiles by Using the `netadm` Command

The `netadm` command is used to administer and obtain the status of profiles (NCPs, locations, ENMs, and WLANs) and NCUs, the individual configuration objects that make up an NCP. In addition, you can use the `netadm` command to interact with the NWAM daemon (`nwamd`) in the absence of a GUI. For more information about `netadm`, see the `netadm(1M)` man page.

The following netadm subcommands are supported:

- enable [-p *profile-type*] [-c *ncu-class*] *profile-name*

  Enables the specified profile. If the profile name is not unique, the profile type must be specified. If the profile type is ncu, and the name is not unique, for example, if there is both a link and an interface ncu with the same name, both NCUs are enabled, unless the -c option is used to specify the NCU class.

  The profile type must be one of the following:

  - ncp
  - ncu
  - loc
  - enm
  - wlan

  The NCU class must be specified as either phys or ip.

- disable [- p *profile-type*] [-c *ncu-class*] *profile-name*

  Disables the specified profile. If the profile name is not unique, the profile type must be specified to identify the profile that is to be disabled. If the profile type is ncu and the name is not unique, for example, if there is both a link and an interface ncu with the same name, both NCUs will be disabled, unless the -c option is used to specify the NCU class.

  The profile type must be one of the following:

  - ncp
  - ncu
  - loc
  - enm
  - wlan

  The NCU class must be specified as either phys or ip.

- list [-x] [- p *profile-type*] [-c *ncu-class*] [*profile-name*]

  Lists all of the available profiles and their current state. Possible state values are listed in the following section. If a profile is specified by name, then only the current state of that profile is listed. If the profile name is not unique, all of the profiles with that given name are listed. Or, the profile type, the NCU class, or both can be included to identify a specific profile. If just the profile type is specified, all of the profiles of that type are listed.

  Listing the enabled NCP includes all of the NCUs that make up that NCP.

  If the -x option is specified, an expanded description of the state of each listed profile is also included in the output.

  Possible profile state values include the following:

  - disabled

    Indicates a manually activated profile that has not been enabled.

- `offline`

  Indicates a conditionally activated or system-activated profile that has not been enabled. The profile might not be active because its conditions have not been satisfied. Or, the profile might not be active because another profile with more specific conditions that are met has been activated instead. This condition applies to profile types that must be enabled one at a time, for example, the Location profile.

- `online`

  Indicates a conditionally activated or system-activated profile whose conditions have been met and which has been successfully enabled. Or, it might indicate a manually activated profile that has been successfully enabled at the request of the user.

- `maintenance`

  Indicates that activation of the profile was attempted, but failed.

- `initialized`

  Indicates that the profile represents a valid configuration object for which no action has yet been taken.

- `uninitialized`

  Indicates that the profile represents a configuration object that is not present on the system. For example, this state could indicate an NCU that corresponds to a physical link that was removed from the system.

- `show-events`

  Listens for a stream of events from the NWAM daemon and displays them.

- `scan-wifi` *link-name*

  Initiates a wireless scan on the link that is specified as *link-name*.

- `select-wifi` *link-name*

  Selects a wireless network to connect to from scan results on the link that is specified as *link-name*.

- `help`

  Displays a usage message with a short description of each subcommand.

For task-related information, see Chapter 5, "NWAM Profile Administration (Tasks)."

Overview of the NWAM Daemons

There are two daemons that are used by NWAM: the `nwamd` daemon and the `netcfgd` daemon. The policy engine daemon, `nwamd`, controls network autoconfiguration by functioning in multiple roles. The repository daemon, `netcfgd`, controls access to the network configuration repository.

Description of the NWAM Policy Engine Daemon (nwamd)

The nwamd daemon controls network autoconfiguration by assuming the following roles:

- **Event Collector**

  This role involves collecting link-related events that need to be detected through routing socket and sysevent registration. An example of how nwamd performs this task is that the daemon obtains an EC_DEV_ADD sysevent, which signifies that a NIC was hot-plugged into the system. All such events are packaged into the nwamd event structure and then sent to the event handling thread, which is responsible for that task.

- **Event Handler**

  This role involves running an event loop thread to respond to events of interest. The event handler operates on the state machines that are associated with the different objects that are managed by the NWAM service. In the course of handling events, the nwamd daemon detects changes in the network environment, which might trigger changes to a profile, or profiles, as a result.

- **Event Dispatcher**

  This role involves sending events to external consumers who have registered an interest in such events. Examples of event dispatching include wireless scan events that contain information about available WLANs, which is useful to the NWAM GUI. The GUI can, in turn, display the available options to the user.

- **Profile Manager**

  Management of these profiles by the nwamd daemon involves applying the network configuration, depending on the following information:

  - Which links and interfaces are activated
  - Characteristics of the connected networks
  - Contingencies and dependencies that are built into the enabled profiles
  - External events that are received

Description of the NWAM Repository Daemon (netcfgd)

The profile daemon, netcfgd, controls and manages access to a network configuration repository. The daemon is started automatically by the svc:/network/netcfg:default SMF service. The daemon ensures that any application that is attempting to read information from or write information to the repository has the following authorizations:

- solaris.network.autoconf.read
- solaris.network.autoconf.write

For more information about authorizations, see the auth_attr(4) man page. For more information about security profiles, see the prof_attr(4) man page.

For more information about the netcfgd daemon, see the netcfgd(1M) man page.

SMF Network Services

In Oracle Solaris, network configuration is implemented by multiple SMF services:

- svc:/network/loopback:default – Creates the IPv4 and IPv6 loopback interfaces.
- svc:/network/netcfg:default – This service is a prerequisite for the svc:/network/physical:default service. The service manages the network configuration repository, with its primary function being to start the netcfgd daemon.
- svc:/network/physical:default – Brings up links and plumbs IP interfaces. This service determines whether NWAM or traditional network configuration is in use, based on the currently active NCP. If NWAM is in use, the service starts the policy daemon, nwamd. If the DefaultFixed NCP is active, the service stops nwamd and applies the persistent ipadm configuration.
- svc:/network/location:default – This service is dependent on the svc:/network/physical:default service and is responsible for activating the Location profile that is selected by the nwamd daemon.

Note – The svc:/network/location:default service has a property that stores the current Location profile. Do not directly manipulate this property. Rather, use the NWAM GUI or the CLI to make these types of changes.

Overview of NWAM Security

Security for NWAM is designed to encompass the following components:

- CLI (netcfg and netadm commands)
- NWAM GUI
- NWAM profile repository daemon (netcfgd)
- Policy engine daemon (nwamd)
- NWAM library (libnwam)

The netcfgd daemon controls the repository where all of the network configuration information is stored. The netcfg command, the NWAM GUI, and the nwamd daemon all send requests to the netcfgd daemon to access the repository. These functional components make requests through the NWAM library, libnwam.

The nwamd daemon is the policy engine that receives system events, configures the network, and reads network configuration information. The NWAM GUI and the netcfg command are

configuration tools that can be used to view and modify the network configuration. These components are also used to refresh the NWAM service when a new configuration needs to be applied to the system.

Authorizations and Profiles That Are Related to NWAM

The current NWAM implementation uses the following authorizations to perform specific tasks:

- `solaris.network.autoconf.read` – Enables the reading of NWAM configuration data, which is verified by the `netcfgd` daemon
- `solaris.network.autoconf.write` – Enables the writing of NWAM configuration data, which is verified by the `netcfgd` daemon
- `solaris.network.autoconf.select` – Enables new configuration data to be applied, which is verified by the `nwamd` daemon
- `solaris.network.autconf.wlan` – Enables the writing of known WLAN configuration data

These authorizations are registered in the `auth_attr` database. For more information, see the `auth_attr(4)` man page.

Two security profiles are provided: `Network Autoconf User` and `Network Autoconf Admin`. The `User` profile has `read`, `select`, and `wlan` authorizations. The `Admin` profile adds the `write` authorization. The `Network Autoconf User` profile is assigned to the `Console User` profile. Therefore, by default, anyone who logged in to the console can view, enable, and disable profiles. Because the `Console User` is not assigned the `solaris.network.autoconf.write` authorization, this user cannot create or modify NCPs, NCUs, locations, or ENMs. However, the `Console User` can view, create, and modify WLANs.

Authorizations That Are Required to Use the NWAM User Interfaces

The NWAM commands, `netcfg` and `netadm`, can be used to view and enable NWAM profiles by anyone who has `Console User` privileges. These privileges are automatically assigned to any user who is logged in to the system from `/dev/console`.

To modify NWAM profiles by using the `netcfg` command, you need the `solaris.network.autoconf.write` authorization or the `Network Autoconf Admin` profile.

You can determine the privileges that are associated with a rights profile by using the `profiles` command with the profile name. For more information, see the `profiles(1)`man page.

For example, to determine privileges that are associated with the `Console User` rights profile, use the following command.

```
$ profiles -p "Console User" info
Found profile in files repository.
    name=Console User
    desc=Manage System as the Console User
    auths=solaris.system.shutdown,solaris.device.cdrw,solaris.smf.manage.vbiosd,
    solaris.smf.value.vbiosd
    profiles=Suspend To RAM,Suspend To Disk,Brightness,CPU Power Management,
    Network Autoconf User,Desktop Removable Media User
    help=RtConsUser.html
```

The NWAM GUI includes the following three components, which are not privileged. These components are granted authorizations, depending on how they are started and the tasks they need to perform:

- **NWAM-specific panel presence**

  This component is the panel applet in the desktop that enables a user to interact with NWAM. The panel can be run by any user and is used to monitor the autoconfiguration of the system and handle event notifications. The panel can also be used to perform some basic network configuration tasks, for example, selecting a WiFi network or manually switching locations. To perform these types of tasks, the `Network Autoconf User` rights profile is required. This rights profile is available in the default configuration, because the panel is running with the authorizations of the user who is logged in from `/dev/console`, and hence has the `Console User` profile.

- **NWAM GUI**

  The NWAM GUI is the primary means for interacting with NWAM from the desktop. The GUI is used to view the network status, to create and modify NCPs and Location profiles, and to start and stop configured ENMs. Interaction with the GUI requires four of the `solaris.network.autoconf` authorizations or the `Network Autoconf Admin` profile. By default, the `Console User` profile has sufficient authorizations to view the network status and profiles by using the GUI. In addition, you require the `solaris.network.autoconf.write` authorization or the `Network Autoconf Admin` profile to modify profiles by using the GUI.

You can obtain additional authorizations in one of the following ways:

- Assign the `Network Autoconf Admin` profile to a specific user.

  You can assign appropriate authorizations, or rights profiles, directly to a given user by editing the `/etc/user_attr` file for that user.

- Assign the `Network Autoconf Admin` profile to the `Console User`.

  You can assign this profile to the `Console User` instead of the `Network Autoconf User` profile that is assigned by default. To assign this profile, edit the entry in the `/etc/security/prof_attr` file.

◆ ◆ ◆ CHAPTER 4

NWAM Profile Configuration (Tasks)

This chapter describes the NWAM profile configuration tasks that you can perform by using the `netcfg` command. These configuration tasks include creating, modifying, and destroying profiles, as well as managing the various SMF services that control the NWAM configuration. This chapter describes how to use the `netcfg` command in both interactive mode and command-line mode.

The following topics are covered in this chapter:

For information about displaying profile states, activating and deactivating profiles, and managing known wireless networks by using the `netadm` command, see Chapter 5, "NWAM Profile Administration (Tasks)."

For information about how to interact with NWAM and how to manage your network configuration from the desktop, see Chapter 6, "About the NWAM Graphical User Interface."

For an introduction to NWAM, see Chapter 2, "Introduction to NWAM."

For detailed overview information about NWAM, including a description of the `netcfg` user interface modes, see Chapter 3, "NWAM Configuration and Administration (Overview)."

Creating Profiles

The `netcfg` command, which is described in the `netcfg(1M)` man page, is one of two administrative commands in the NWAM command-line interface.

The `netcfg` command can be used to display profile configuration data, and to display, create, and modify Known WLAN objects, by anyone who has `Console User` privileges. These privileges are automatically assigned to any user who is logged in to the system from `/dev/console`. Users who have the `Network Autoconf Admin` profile can also create and modify all types of NWAM profiles and configuration objects. For more information, see the "Overview of NWAM Security" on page 67.

You can use the `netcfg` command to select, create, modify, and destroy user-defined profiles. The command can be used in either interactive mode or command-line mode. The `netcfg` command also supports the export of profile configuration information to command files.

You can create, modify, and remove the following profiles and configuration objects:

- Network Configuration Profiles (NCPs)
- Location profiles
- External Network Modifiers (ENMs)
- Known wireless local area networks (WLANs)
- Network Configuration Units (NCUs)

Creating Profiles in Command-Line Mode

The basic command syntax to use to create a profile from the command line is as follows:

`netcfg create [ -t` *template* **`]`** *object-type* **`[`** *class* **`]`** *object-name*

`create` — Creates an in-memory profile (or configuration object) of the specified type and name.

`-t` *template* — Specifies that the new profile be identical to *template*, where *template* is the name of an existing profile of the same type. If the `-t` option is not used, the new profile is created with default values.

object-type — Specifies the type of profile to be created.

You can specify one of the following values for the *object-type* option:

- `ncp`
- `ncu`
- `loc`
- `enm`
- `wlan`

All profiles that are specified by the *object-type* option, with the exception of an `ncu`, must be created at the global scope before you can use the `netcfg select` command to select the particular object.

class Specifies the class of profile that is specified by *object-type*. This parameter is only used for the `ncu` object type, and has two possible values, `phys` or `ip`.

object-name Specifies the name of the user-defined profile. For an NCU, *object-name* is the name of the corresponding link or interface. For all the other profile types, *object-name* is any user-defined name.

For example, to create an NCP named `User`, you would type the following command:

```
$ netcfg create ncp User
```

where `ncp` is the *object-type* and `User` is the *object-name*.

Note – For the creation of NCPs, the `class` option is not required.

Optionally, you can use a copy of the Automatic NCP as your template, then make changes to that profile, as shown here:

```
$ netcfg create -t Automatic ncp
```

To create a Location profile with the name `office`, you would type the following command:

```
$ netcfg create loc office
```

Interactively Creating Profiles

You can use the `netcfg` command in interactive mode to perform the following tasks:

- Create a profile.
- Select and modify a profile.
- Verify that all of the required information about a profile is set and valid.
- Commit the changes for a new profile.
- Cancel the current profile configuration without committing any changes to persistent storage.
- Revert the changes that you made for a profile.

Creating an NCP

Creating a profile in interactive mode results in a command prompt that is in one of the following scopes:

- In the NCP scope, if an NCP is created
- In the profile scope, if a Location profile, an ENM, or a WLAN object is created

Creating an NCP or an NCU moves the focus into that object's scope, walking you through the default properties for the specified profile.

To interactively create an NCP, you begin by initiating a `netcfg` interactive session. Then, you use the `create` subcommand to create the new NCP `User`, as follows:

```
$ netcfg
netcfg> create ncp User
netcfg:ncp:User>
```

Creating NCUs for an NCP

The NCP is essentially a container that consists of a set of NCUs. All NCPs contain both link and interface NCUs. Link NCUs specify both link configuration and link selection policy. Interface NCUs specify interface configuration policy. If IP connectivity is required, both a link and an interface NCU are required. NCUs must be added or removed explicitly by using the `netcfg` command or by using the GUI.

Note – It is possible to add NCUs that do not correlate to any link that is currently installed on the system. Additionally, you can remove NCUs that map to a link that is currently installed on the system.

You can create NCUs by using the `netcfg` command in either interactive mode or command-line mode. Because creating an NCU involves several operations, it is easier and more efficient to create NCUs in interactive mode, rather than trying to construct a single-line command that creates the NCU and all of its properties. NCUs can be created when you initially create an NCP or afterward. The process of creating or modifying an NCU involves setting general NCU properties, as well as setting properties that specifically apply to each NCU type.

The properties that you are presented with during the process of creating NCUs for an NCP make the most sense based on the choices that you made during the creation of that particular NCP.

When you create an NCU interactively, `netcfg` walks through each relevant property, displaying both the default value, where a default exists, and the possible values. Pressing

Return without specifying a value applies the default value (or leaves the property empty if there is no default), or you can specify an alternate value. The properties that are displayed during the process of creating NCUs for an NCP are relevant based on the choices that you have already made. For example, if you choose `dhcp` for the `ipv4-addrsrc` property for an interface NCU, you are not prompted to specify a value for the `ipv4-addr` property.

The following table describes all of the NCU properties that you might specify when creating or modifying an NCU. Some properties apply to both NCU types. Other properties apply to either a link NCU or an interface NCU. For a complete description of all of the NCU properties, including rules and conditions that might apply when you specify these properties, see the `netcfg(1M)` man page.

TABLE 4–1 NCU Properties to Create or Modify an NCU

Property	Description	Possible Values	NCU Type
`type`	Specifies the NCU type, either link or interface.	`link` or interface	Link and interface
`class`	Specifies the NCU class.	`phys` (for link NCUs) or `ip` (for interface NCUs)	Link and interface
`parent`	Specifies the NCP to which this NCU belongs.	*parent-NCP*	Link and interface
`enabled`	Specifies whether the NCU is enabled or disabled. This property is read-only. It is only changed indirectly when you use the `netadm` command or the NWAM GUI to enable or disable the NCU.	`true` or `false`	Link and interface
`activation-mode`	Specifies the type of trigger for the automatic activation of the NCU.	`manual` or `prioritized` The default value is `manual`.	Link
`priority-group`	Specifies the group priority number.	`0` (for wired links) or `1` (for wireless links) For user-defined NCPs, different policies can be specified, for example, wireless link 1 is priority 1, wired link 1 is priority 2, and wired link 2 is priority 3. **Note –** A lower number indicates a higher priority.	Link

TABLE 4–1 NCU Properties to Create or Modify an NCU *(Continued)*

Property	Description	Possible Values	NCU Type
`priority-mode`	Specifies the mode that is used to determine the activation behavior for a priority group, if the `activation-mode` property is set to `prioritized`.	`exclusive`, `shared`, or `all` See the `netcfg(1M)` man page for the rules that apply when you specify these values.	Link
`link-mac-addr`	Specifies the MAC address that is assigned to this link. By default, NWAM uses the factory-assigned or other default MAC address. A different value can be set here to override that selection.	A string containing a 48–bit MAC address	
`link-autopush`	Identifies modules that are automatically pushed over the link when it is opened.	A list of strings (modules that are to be pushed over the link) See `autopush(1M)`.	Link
`link-mtu`	Is automatically set to the default MTU for the physical link. The value can be overridden by setting the property to a different value.	MTU size for the link	Link
`ip-version`	Specifies the version of IP to use. Multiple values can be assigned.	`ipv4` and `ipv6` The default value is `ipv4`, `ipv6`.	Interface
`ipv4-addrsrc`	Identifies the source of IPv4 addresses that are assigned to this NCU. Multiple values can be assigned.	`dhcp` and `static` The default value is `dhcp`.	Interface
`ipv6-addrsrc`	Identifies the source of IPv6 addresses assigned to this NCU. Multiple values can be assigned.	`dhcp`, `autoconf`, or `static` The default value is `dhcp`, `autoconf`.	Interface
`ipv4-addr`	Specifies one or more IPv4 addresses to be assigned to this NCU.	One or more IPv4 addresses to be assigned	Interface
`ipv6-addr`	Specifies one or more IPv6 addresses to be assigned to this NCU.	One or more IPv6 addresses to be assigned	Interface

TABLE 4–1 NCU Properties to Create or Modify an NCU *(Continued)*

Property	Description	Possible Values	NCU Type
`ipv4-default-route`	Specifies the default route for an IPv4 address.	An IPv4 address	Interface
`ipv6-default-route`	Specifies the default route for an IPv6 address.	An IPv6 address	Interface

▼ How to Interactively Create an NCP

The following procedure describes how to create an NCP in interactive mode.

Tip – The walk process that NWAM performs during the initial profile creation ensures that you are prompted for only those properties that make sense, given the choices that you made previously. Also, the `verify` subcommand that is described in this procedure verifies your configuration. If any required values are missing, you are notified. You can use the `verify` subcommand explicitly when creating or modifying a profile or implicitly by using the `commit` subcommand to save your changes.

1 Initiate an `netcfg` interactive session.

```
$ netcfg
netcfg>
```

2 Create the NCP.

```
netcfg> create ncp User
netcfg:ncp:User>
```

where `ncp` is the profile type and `User` is the profile name.

Creating the NCP automatically takes you into the NCP scope. If you were creating a location, an ENM, or a WLAN object, the command prompt would take you to the profile scope.

3 Create the link and interface NCUs for the NCP.

a. To create the link NCU, type the following command:

```
netcfg:ncp:User> create ncu phys net0
Created ncu 'net0', Walking properties ...
```

where `ncu` is the object type, `phys` is the class, and `net0` (for example purposes *only*) is the object name.

Creating an NCU moves you into that object's scope and walks you through the default properties for the object.

b. To create an interface NCU, type the following command:

```
netcfg:ncp:User> create ncu ip net0
Created ncu 'net0'. walking properties ...
```

where `ncu` is the object type, `ip` is the class, and `net0` (for example purposes *only*) is the object name.

Creating an NCU moves you into that object's scope and walks you through the default properties for the object.

During the creation of an NCU, the `class` option is used to differentiate between the two types of NCUs. This option is especially valuable in situations where different NCU types share the same name. If the `class` option is omitted, it is much more difficult to distinguish NCUs that share the same name.

4 Add the appropriate properties for the NCU that you created.

Note – Repeat Steps 3 and 4 until all of the required NCUs for the NCP are created.

---

5 During the creation of the NCU, or when setting property values for a specified NCU, use the `verify` subcommand to ensure that the changes that you made are correct.

```
netcfg:ncp:User:ncu:net0> verify
All properties verified
```

6 Commit the properties that you set for the NCU.

```
netcfg:ncp:User:ncu:net0> commit
committed changes.
```

Alternatively, you can use the `end` subcommand to perform an implicit commit, which moves the interactive session up one level to the next higher scope. In this instance, if you have completed creating the NCP and adding NCUs to it, you can exit the interactive session directly from the NCP scope.

Note –

- In interactive mode, changes are not saved to persistent storage until you commit them. When you use the `commit` subcommand, the entire profile is committed. To maintain the consistency of persistent storage, the commit operation also includes a verification step. If the verification fails, the commit also fails. If an implicit commit fails, you are given the option of ending or exiting the interactive session without committing the current changes. Or, you can remain in the current scope and continue making changes to the profile.
- To cancel the changes that you made, use the `cancel` or the `revert` subcommand.

  The `cancel` subcommand ends the current profile configuration without committing the current changes to persistent storage, then moves the interactive session up on level to the next higher scope. The `revert` subcommand undoes the changes that you made and rereads the previous configuration. When you use the `revert` subcommand, the interactive session remains in the same scope.

7 Use the `list` subcommand to display the NCP configuration.

8 When you are finished configuring the NCP, exit the interactive session.

```
netcfg:ncp:User> exit
```

Any time that you use the `exit` subcommand to end a `netcfg` interactive session, the current profile is verified and committed. If either the verification or the commit operation fails, an appropriate error message is issued, and you are given the opportunity to exit without committing the current changes. Or, you can remain in the current scope and continue making changes to the profile.

Note – To exit the scope without exiting the `netcfg` interactive session, type the `end` command:

```
netcfg:ncp:User> end
netcfg>
```

Example 4–1 Interactively Creating an NCP

In the following example, an NCP and two NCUs (one link and one interface) are created.

```
$ netcfg
netcfg> create ncp User
netcfg:ncp:User> create ncu phys net0
Created ncu 'net0', Walking properties ...
activation-mode (manual) [manual|prioritized]>
link-mac-addr>
link-autopush>
link-mtu>
netcfg:ncp:User:ncu:net0> end
Committed changes
```

```
netcfg:ncp:User> create ncu ip net0
Created ncu 'net0'. Walking properties ...
ip-version (ipv4,ipv6) [ipv4|ipv6]> ipv4
ipv4-addrsrc (dhcp) [dhcp|static]>
ipv4-default-route>
netcfg:ncp:User:ncu:net0> verify
All properties verified
netcfg:ncp:User:ncu:net0> end
Committed changes
netcfg:ncp:User> list
NCUs:
        phys    net0
        ip      net0
netcfg:ncp:User> list ncu phys net0
ncu:net0
        type                    link
        class                   phys
        parent                  "User"
        activation-mode         manual
        enabled                 true
netcfg:ncp:User> list ncu ip net0
ncu:net0
        type                    interface
        class                   ip
        parent                  "User"
        enabled                 true
        ip-version              ipv4
        ipv4-addrsrc            dhcp
        ipv6-addrsrc            dhcp,autoconf
netcfg:ncp:User> exit
$
```

In this example, because the value `ipv4` is chosen, no prompt is displayed for the `ipv6-addrsrc` property, as this property is unused. Likewise, for the `phys` NCU, the default value (manual activation) for the `priority-group` property is accepted, so no other conditionally related properties are applied.

Example 4–2 Creating an NCU for an Existing NCP

To create an NCU for an existing NCP or to modify the properties of any existing profile, use the `netcfg` command with the `select` subcommand.

In the following example, an IP NCU is created for an existing NCP. The process of modifying an existing profile in interactive mode is similar to creating a profile. The difference between the following example and Example 4–1 is that in this example, the `select` subcommand is used instead of the `create` subcommand because the NCP already exists.

```
$ netcfg
netcfg> select ncp User
netcfg:ncp:User> list
NCUs:
        phys   net0
netcfg:ncp:User> create ncu ip net0
Created ncu 'net0'. Walking properties ...
```

```
ip-version (ipv4,ipv6) [ipv4|ipv6]> ipv4
ipv4-addrsrc (dhcp) [dhcp|static]>
ipv4-default-route>
netcfg:ncp:User:ncu:net0> end
Committed changes
netcfg:ncp:User> list
NCUs:
        phys    net0
        ip      net0
netcfg:ncp:User> list ncu phys net0
ncu:net0
        type                    link
        class                   phys
        parent                  "User"
        activation-mode         manual
        enabled                 true
netcfg:ncp:User> list ncu ip net0
NCU:net0
        type                    interface
        class                   ip
        parent                  "User"
        enabled                 true
        ip-version              ipv4
        ipv4-addrsrc            dhcp
        ipv6-addrsrc            dhcp,autoconf
netcfg:ncp:User> exit
$
```

Creating a Location Profile

A Location profile contains properties that define network configuration settings that are not directly related to basic link and IP connectivity. Some examples include naming service and IP filter settings that are applied together, when required. At any given time, one Location profile and one NCP must be active on the system. There are system-defined locations and user-defined locations. System locations are the default that NWAM chooses under certain conditions, for example, if you did not specify a location, or if no manually activated locations are enabled, and none of the conditions of the conditionally activated locations has been met. System-defined locations have a `system` activation mode. User-defined locations are those that are configured to be manually or conditionally activated, according to network conditions, for example, an IP address that is obtained by a network connection.

For information about manually activating (enabling) a Location profile, see "Activating and Deactivating Profiles" on page 110.

You can create locations by using the `netcfg` command in either interactive mode or command-line mode. When you create a Location profile, you must set the properties for the location by specifying values that define the particular configuration parameters for that location. Location properties are categorized by group, where the group signifies a particular class of configuration preferences.

Location properties are also stored by NWAM in a repository. When a particular Location profile is activated, NWAM autoconfigures the network, based on the properties that are set for that location. Creating or modifying locations involves setting the various properties that define how the profile is configured, which in turn, determines how NWAM autoconfigures your network. The properties that you are presented with during the configuration process are those that make the most sense, based on the choices that you made previously.

The following table describes all of the location properties that can be specified. Note that location properties are categorized by group. For a complete description of all of the location properties, including any rules, conditions, and dependencies that might apply when you specify any of these properties, see the `netcfg(1M)` man page.

TABLE 4–2 Location Properties and Their Descriptions

Property Group and Description	Property Value and Description
Selection criteria Specifies the criteria for how and when a location is activated or deactivated.	■ `activation-mode` The possible values for the `activation-mode` property are `manual`, `conditional-any`, and `conditional-all`. ■ `conditions`
System domain Determines a host's domain name for direct use by the NIS naming service.	The `system-domain` property consists of the `default-domain` property. This property specifies the system-wide domain that is used for Remote Procedure Call (RPC) exchanges.
Name services information Specifies the naming service to use and the naming service switch configuration.	The following is a list of properties for the specified naming service: ■ `domain-name` ■ `nameservices` ■ `nameservices-config-file` ■ `dns-nameservice-configsrc` ■ `dns-nameservice-domain` ■ `dns-namservice-servers` ■ `dns-nameservice-search` ■ `dns-nameservice-sortlist` ■ `dns-nameservice-options` ■ `nis-nameservice-configsrc` ■ `nis-namservice-servers` ■ `ldap-nameservice-configsrc` ■ `ldap-namservice-servers` For more information about these properties, see the "Location Properties" section in the `netcfg(1M)` man page.

TABLE 4–2 Location Properties and Their Descriptions *(Continued)*

Property Group and Description	Property Value and Description
NFSv4 domain Specifies the NFSv4 domain.	The value that is used for the system's `nfsmapid_domain` property. This value is used to set the `nfsmapid_domain` SMF property, as described in the `nfsmapid` man page, while the location is active. If this property is not set, the system's `nfsmapid_property` is cleared when the location is active. See the `nfsmapid(1M)` man page for more information.
IP Filter configuration Specifies the parameters that are used for IP Filter configuration. For these properties, the paths to the appropriate `ipf` and `ipnat` files containing IP filter and NAT rules are specified.	■ `ipfilter-config-file` ■ `ipfilter-v6-config-file` ■ `ipnat-config-file` ■ `ippool-config-file` If a configuration file is specified, the rules that are contained in the identified file are applied to the appropriate `ipfilter` subsystem.
Configuration files for IPsec Specifies which files to use for IPsec configuration.	■ `ike-config-file` ■ `ipsecpolicy-config-file`

How to Interactively Create a Location Profile

The following procedure describes how to create a Location profile.

Tip – The walk process that NWAM performs during an initial profile creation only prompts you for those properties that make sense, given the values that you entered previously. Also, the `verify` subcommand checks to make sure your configuration is correct. If any required values are missing, you are notified. Note that you can use the `verify` subcommand explicitly when you creating or modifying a profile configuration or implicitly by using the `commit` subcommand to save your changes.

1 Initiate an `netcfg` interactive session.

```
$ netcfg
netcfg>
```

2 Create or select the location.

```
netcfg> create loc office
netcfg:loc:office>
```

In this example, the location `office` is created.

Creating the location automatically moves you to into the profile scope for this location.

3 Set the appropriate properties for the location.

4 Display the profile configuration.

For example, the following output displays the properties for the location `office`:

```
netcfg:loc:office> list
LOC:office
    activation-mode             conditional-any
    conditions                  "ncu ip:wpi0 is active"
    enabled                     false
    nameservices                dns
    nameservices-config-file    "/etc/nsswitch.dns"
    dns-nameservice-configsrc   dhcp
    ipfilter-config-file        "/export/home/test/wifi.ipf.conf"
```

5 Verify that the profile configuration is correct.

In the following example, the configuration for the location `office` is verified:

```
netcfg:loc:office> verify
All properties verified
```

6 When you complete the verification, commit the Location profile to persistent storage.

```
netcfg:loc:office> commit
Committed changes
```

Alternatively, you can use the `end` subcommand to end the session, which also saves the profile configuration.

```
netcfg:loc:office> end
Committed changes
```

Note –

- In interactive mode, changes are not saved to persistent storage until you commit them. When you use the `commit` subcommand, the entire profile is committed. To maintain the consistency of persistent storage, the commit operation also includes a verification step. If the verification fails, the commit also fails. If an implicit commit fails, you are given the option of ending or exiting the interactive session without committing the current changes. Or, you can remain in the current scope and continue making changes to the profile.
- To cancel the changes that you made, use the `cancel` subcommand.

  The `cancel` subcommand ends the current profile configuration without committing the current changes to persistent storage, then moves the interactive session up one level to the next higher scope.

---

7 Exit the interactive session.

```
netcfg> exit
Nothing to commit
$
```

Example 4–3 Interactively Creating a Location Profile

In the following example, a location named `office` is created.

```
$ netcfg
netcfg> create loc office
Created loc 'office'.  Walking properties ...
activation-mode (manual) [manual|conditional-any|conditional-all]> conditional-any
conditions> ncu ip:wpi0 is active
nameservices (dns) [dns|files|nis|ldap]>
nameservices-config-file ("/etc/nsswitch.dns")>
dns-nameservice-configsrc (dhcp) [manual|dhcp]>
nfsv4-domain>
ipfilter-config-file> /export/home/test/wifi.ipf.conf
ipfilter-v6-config-file>
ipnat-config-file>
ippool-config-file>
ike-config-file>
ipsecpolicy-config-file>
netcfg:loc:office> list
LOC:office
    activation-mode             conditional-any
    conditions                  "ncu ip:wpi0 is active"
    enabled                     false
    nameservices                dns
    nameservices-config-file    "/etc/nsswitch.dns"
    dns-nameservice-configsrc   dhcp
    ipfilter-config-file        "/export/home/test/wifi.ipf.conf"
netcfg:loc:office> verify
All properties verified
netcfg:loc:office> commit
Committed changes
netcfg> list
NCPs:
    User
    Automatic
Locations:
    Automatic
    NoNet
    test-loc
WLANs:
    sunwifi
    ibahn
    gogoinflight
    admiralsclub
    hhonors
    sjcfreewifi
netcfg> exit
Nothing to commit
$
```

In this example, the following properties were specified for the `office` location:

- The `activation-mode` property was set to `conditional-any`, which resulted in a command prompt that enabled the conditions for activation to be specified.
- The condition for activation was specified as: `ncu ip:wpi0 is active`.

Note – The `conditions` property was required because the `conditional-any` property was specified in the previous step. If, for example, the `manual` property had been specified, the `conditions` property would not be required.

- The following default values were accepted by pressing Return:
  - `nameservices`
  - `nameservices-config-file`
  - `dns-nameservice-configsrc`
  - `nfsv4-domain`
- For the `ipfilter-config-file` property, the `/export/home/test/wifi.ipf.conf` file was specified.
- The following default values were accepted by pressing Return:
  - `ipfilter-v6-config-file`
  - `ipnat-config-file`
  - `ippool-config-file`
  - `ike-config-file`
  - `ipsecpolicy-config-file`
- The `list` subcommand was used to view the properties of the Location profile.
- The `verify` subcommand was used to perform a verification of the configuration.
- The `commit` subcommand was used to commit the changes to persistent storage.
- The `list` subcommand was used again to ensure that the new location was created correctly and that it contains the correct information.
- The `exit` subcommand was used to exit the `netcfg` interactive session.

For instructions on which values can be specified for these properties, see the `netcfg(1M)` man page.

Creating an ENM Profile

ENMs pertain to the configuration of applications that are external to NWAM, for example, a VPN application. These applications can create and modify network configuration. ENMs can also be defined as services or applications that directly modify network configuration when they are activated or deactivated. You can configure NWAM to activate and deactivate ENMs under conditions that you specify. Unlike an NCP or a Location profile, where only one of each profile type can be active on a system at any given time, multiple ENMs can potentially be active on a system at the same time. The ENMs that are active on a system at any given time do not necessarily depend on the NCP or Location profile that is also active on the system at the same time.

Note – NWAM does not automatically recognize an application for which you might create an ENM. These applications must first be installed and then configured on your system before you can use the `netcfg` command to create an ENM for them.

To create an ENM, type the following command:

```
$ netcfg
netcfg> create enm my_enm
Created enm 'my_enm'.  Walking properties ...
```

where `enm` is the ENM profile and `my_enm` is the object name.

The process of creating the ENM takes you to the profile scope for the newly created ENM, and automatically begins a walk of the properties in the newly created ENM. From here, you can set properties for the ENM that dictate when and how the ENM is activated, as well as other conditions, including the ENM's start and stop method.

For further instructions on specifying ENM properties, see the `netcfg(1M)` man page.

The following table describes the properties that you might specify when creating or modifying an ENM.

Property Name	Description	Possible Values
`activation-mode`	Mode that is used to determine activation of an ENM	`conditional-any`, `conditional-all`, `manual`
`conditions`	If activation-mode is `conditional-any` or `conditional-all`, specifies the test to determine whether the ENM must be activated.	A string or strings formatted as specified in the "Condition Expressions" section of the `netcfg(1M)` man page, if the property is used.
`start`	(Optional) Absolute path to the script to be executed upon activation	Path to script, if this property is used
`stop`	(Optional) Absolute path to the script to be executed upon deactivation	Path to script, if this property is used
`fmri`	(Optional) FMRI (fault managed resource identifier) to be enabled upon ENM activation **Note –** Either an FMRI or a start script must be specified. If an FMRI is specified, both the `start` and `stop` properties are ignored.	Path to script

EXAMPLE 4–4 Interactively Creating an ENM Profile

In the following example, an ENM named `test-enm` is created in interactive mode.

```
$ netcfg
netcfg> create enm test-enm
Created enm 'testenm'.  Walking properties ...
activation-mode (manual) [manual|conditional-any|conditional-all]>
fmri> svc:/application/test-app:default
start>
stop>
netcfg:enm:test-enm> list
ENM:test-enm
    activation-mode     manual
    enabled             false
    fmri                "svc:/application/test-enm:default"
netcfg:enm:test-enm> verify
All properties verified
netcfg:enm:test-enm> end
Committed changes
netcfg> list
NCPs:
    User
    Automatic
Locations:
    Automatic
    NoNet
    test-loc
ENMs:
    test-enm
WLANs:
    sunwifi
    ibahn
    gogoinflight
    admiralsclub
    hhonors
    sjcfreewifi
netcfg> end
$
```

In this example, an ENM named `test-enm` was created with the following property values:

- The default value (`manual`) for the `activation-mode` property was accepted by pressing the Return key.
- The SMF FMRI property `svc:/application/test-enm:default` was specified as the method to use for activating and deactivating the application.

  Note that because an FMRI was specified, the `start` and `stop` method properties were bypassed.
- The `list` subcommand was used to view the properties of the ENM.
- The `verify` subcommand was used to ensure that the profile configuration is correct.
- The `end` subcommand was used to implicitly save the configuration.
- The `end` subcommand was used again to end the interactive session.

Creating WLANs

NWAM maintains a system-wide list of known WLANs. WLANs are configuration objects that contain history and configuration information for the wireless networks that you connect to from your system. This list is used to determine the order in which NWAM attempts to connect to available wireless networks. If a wireless network that exists in the Known WLAN list is available, NWAM automatically connects to that network. If two or more known networks are available, NWAM connects to the wireless network that has the highest priority (lowest number). Any new wireless network that NWAM connects to is added to the top of the Known WLAN list and becomes the new highest priority wireless network.

To create a WLAN object, type the following command:

```
$ netcfg
netcfg> create wlan mywifi
Created wlan 'mywifi'.  Walking properties ...
```

where `wlan` is the WLAN object and `mywifi` is the object name.

The process of creating a WLAN object takes you to the profile scope for the newly created WLAN, and automatically begins a walk of the properties in the newly created WLAN. From here, you can set properties for the WLAN that define its configuration.

The following table describes the properties that you might specify when creating or modifying WLANs.

Known WLAN Property	Data Type for Property
`name`	ESSID (wireless network name)
`bssids`	Base Station IDs of WLANs that your system has connected to while connected to the specified WLAN
`priority`	WLAN connection preference (lower values are preferred)
`keyslot`	Slot number (1–4) in which the WEP key is contained
`keyname`	Name of the WLAN key that is created by using the `dladm create-secobj` command.
`security-mode`	The type of encryption key in use. The type must be `none`, `wep`, or `wpa`.

EXAMPLE 4–5 Creating a WLAN

In the following example, a WLAN object named `mywifi` is created.

EXAMPLE 4–5 Creating a WLAN *(Continued)*

This example assumes that a secure object named `mywifi-key`, which contains the key that is specified by the `keyname` property for the WLAN `mywifi`, is created *before* adding the WLAN.

The priority number can change as other WLANs are added or removed. Note that no two WLANs can be assigned the same priority number. Lower numbers indicate a higher priority, in terms of which WLANs are preferred. In this example, the WLAN is assigned the priority number 100 to ensure that it has a lower priority than any other known WLANs.

When the `list` subcommand is used at the end of the procedure, the new WLAN is added to the bottom of the list, indicating that it has the lowest priority of all the existing known WLANs. If the WLAN was assigned a priority number of zero (`0`), which is the default, it would have been displayed at the top of the list, indicating the highest priority. Subsequently, the priority of all other existing WLANs would have shifted down in priority and would have been displayed in the list after the newly added WLAN.

```
$ netcfg
netcfg> create wlan mywifi
Created wlan 'mywifi'.  Walking properties ...
priority (0)> 100
bssids>
keyname> mywifi-key
keyslot>
security-mode [none|wep|wpa]> wpa
netcfg:wlan:mywifi> list
WLAN:mywifi
    priority            100
    keyname             "mywifi-key"
    security-mode       wpa
netcfg:wlan:mywifi> verify
All properties verified
netcfg:wlan:mywifi> end
Committed changes
netcfg> list
NCPs:
    User
    Automatic
Locations:
    Automatic
    NoNet
    test-loc
ENMs:
    test-enm
WLANs:
    sunwifi
    ibahn
    gogoinflight
    admiralsclub
    hhonors
    sjcfreewifi
    mywifi
netcfg> exit
Nothing to commit
```

EXAMPLE 4–5 Creating a WLAN *(Continued)*

```
$
```

Removing Profiles

You can remove all user-defined profiles or a specified user-defined profile from memory and persistent storage by using the `netcfg destroy -a` command.

Note – System-defined profiles, which include the Automatic NCP and the NoNet and Automatic Location profiles, cannot be removed.

The syntax for the `destroy` command is as follows:

`netcfg destroy` *object-type* **`[`** *class* **`]`** *object-name*

Alternatively, you can use the following command to remove all of the user-defined profiles in a system:

```
netcfg destroy -a
```

EXAMPLE 4–6 Removing All User-Defined Profiles by Using `netcfg` Command-Line Mode

To remove all of the user-defined profiles on a system, type the following command:

```
$ netcfg destroy -a
```

Because at least one profile must be active on the system at all times, and to avoid in-use errors when removing user-defined profiles, make sure that you enable the Automatic NCP before using the `destroy -a` command.

EXAMPLE 4–7 Removing a Specific User-Defined Profile by Using `netcfg` Command-Line Mode

To remove a specific user-defined profile on the system, for example the NCP named `User`, type the following command:

```
$ netcfg destroy ncp User
```

The `destroy` command can also be used to remove NCUs from an existing NCP. In the following example, an interface NCU with the name `net1` is removed from the user-defined NCP:

```
$ netcfg "select ncp User; destroy ncu ip net1"
```

To confirm that a profile has been removed, use the `list` subcommand, as shown here:

EXAMPLE 4–7 Removing a Specific User-Defined Profile by Using `netcfg` Command-Line Mode *(Continued)*

```
$ netcfg
netcfg> select ncp User
netcfg:ncp:User> list
NCUs:
        phys    net1
netcfg> exit
Nothing to commit
$
```

EXAMPLE 4–8 Interactively Removing a Profile

In the following example, an IP NCU named `net2` is removed.

```
$ netcfg list
NCPs:
        Automatic
        User
Locations:
        Automatic
        NoNet
        test
        foo
$ netcfg
netcfg> select ncp User
netcfg:ncp:User> list
NCUs:
        phys    net2
        ip      net2
netcfg:ncp:User> destroy ncu ip net2
Destroyed ncu 'net2'
netcfg:ncp:User> list
NCUs:
        phys    net2
netcfg:ncp:User> end
netcfg> exit
Nothing to commit
$
```

Setting and Changing Property Values for a Profile

Property values for new and existing user-defined profiles are set by using the `netcfg` command with the `set` subcommand. This subcommand can be used in interactive mode or in command-line mode. If a property value is set or changed in command-line mode, the change is immediately committed to persistent storage.

The syntax for the `set` subcommand is as follows:

`netcfg set` *prop-name*`=`*value1*`[,`*value2*`...]`

If you need to retrieve a specific property value, use the `netcfg get` command. For more information, see "Obtaining Values of a Specific Property" on page 97.

EXAMPLE 4–9 Setting Property Values in `netcfg` Command-Line Mode

If you are using the `netcfg` command to set a property value in command-line mode, multiple subcommands must be typed on the command line.

For example, to set the `mtu` property for a link NCU named `net1`, you would type the following command:

```
$ netcfg "select ncp User; select ncu phys net1; set mtu=1492"
```

In this example, the `select` subcommand is used to select the top-level profile, then again to select the NCU that contains the `mtu` property value that is modified.

Multiple values can be set for a given property from the command line at the same time. When setting multiple values, each value must be separated by a comma (`,`). If individual values for a specified property also contain a comma, the comma that is part of the property value must be preceded by a backslash (`\,`). Commas within properties that only have a single value are not interpreted as delimiters and therefore do not need to be preceded by a backslash.

In the following example, the `ip-version` property value for the NCU, `myncu`, in the NCP User is set:

```
$ netcfg "select ncp User; select ncu ip myncu; set ip-version=ipv4,ipv6"
```

EXAMPLE 4–10 Interactively Setting Property Values for a Profile

When interactively setting property values, you must first select a profile at the current scope, which moves the interactive session into that profile's scope. From this scope, you can select the object whose property that you want to modify. The selected profile is then loaded into memory from persistent storage. At this scope, you can modify the profile or its properties, as shown in the following example:

```
$ netcfg
netcfg> select ncp User
netcfg:ncp:User> select ncu ip iwk0
netcfg:ncp:User:ncu:iwk0> set ipv4-default-route = 129.174.7.366
```

In the following example, the `ipfilter-config-file` property of the location `foo` is set:

```
$ netcfg
netcfg> list
NCPs:
   Automatic
   User
Locations:
   Automatic
   NoNet
```

EXAMPLE 4–10 Interactively Setting Property Values for a Profile *(Continued)*

```
   foo

netcfg> select loc foo
netcfg:loc:foo> list
LOC:foo
   activation-mode              manual
   enabled                      false
   nameservices                 dns
   dns-nameservice-configsrc    dhcp
   nameservices-config-file     "/etc/nsswitch.dns"
netcfg:loc:foo> set ipfilter-config-file=/path/to/ipf-file
netcfg:loc:foo> list
LOC:foo
   activation-mode              manual
   enabled                      false
   nameservices                 dns
   dns-nameservice-configsrc    dhcp
   nameservices-config-file     "/etc/nsswitch.dns"
   ipfilter-config-file         "/path/to/ipf-file"
netcfg:loc:foo> end
Committed changes
netcfg> exit
Nothing to commit
$
```

In the following example, the `link-mtu` property of the NCU `net0` in the NCP User is modified interactively:

```
$ netcfg
netcfg> select ncp User
netcfg:ncp:User> select ncu phys net0
netcfg:ncp:User:ncu:net0> list
NCU:net0
   type                link
   class               phys
   parent              "User"
   enabled             true
   activation-mode     prioritized
   priority-mode       exclusive
   priority-group      1
netcfg:ncp:User:ncu:net0> set link-mtu=5000
netcfg:ncp:User:ncu:net0> list
NCU:net0
   type                link
   class               phys
   parent              "User"
   enabled             true
   activation-mode     prioritized
   priority-mode       exclusive
   priority-group      1
   link-mtu            5000
netcfg:ncp:User:ncu:net0> commit
Committed changes
netcfg:ncp:User:ncu:net0> exit
Nothing to commit
```

EXAMPLE 4–10 Interactively Setting Property Values for a Profile *(Continued)*

```
$
```

Querying the System for Profile Information

The `netcfg` command can be used with the `list` subcommand to list all of the profiles, property-value pairs, and resources that exist at the current or specified scope. Use the `list` subcommand to query the system for general information about all profiles or to retrieve specific information about a particular profile. The `list` subcommand can be used in either interactive mode or command-line mode.

If you need to obtain information about profiles and their current state, use the `netadm` command with the `list` subcommand. For more information, see "Displaying the Current State of a Profile" on page 108.

Listing All of the Profiles on a System

The `netcfg list` command lists all of the system-defined and user-defined profiles on a system. Note that using the `list` subcommand without any options displays all of the top-level profiles that are on a system. The command does not list the state of each profile. To display a list of the profiles and their state (online or offline), use the `netadm list` command.

To list all of the top level profiles on a system, type the following command:

```
$ netcfg list
NCPs:
        Automatic
        User
Locations:
        Automatic
        NoNet
        home
        office
ENMs:
        myvpn
        testenm
WLANs:
        workwifi
        coffeeshop
        homewifi
```

In this example, the following profiles are listed:

- NCPs

  There are two NCPs listed: the Automatic NCP, which is a system-defined profile, and a user-defined NCP, named `User`.

- Locations

  There are four Location profiles listed: two locations that are system-defined (`Automatic` and `NoNet`) and two locations that are user-defined (`home` and `office`).

- ENMs

  There are two ENMs listed: one ENM for an installed and configured VPN application, and one test ENM.

- WLANs

  There are three WLANs listed: one WLAN for work, one WLAN for the local coffee shop, and one WLAN for the user's home wireless network.

Note – Only user-defined profiles can be created, modified, or removed.

Listing All Property Values for a Specific Profile

Use the `netcfg` command with the `list` subcommand to list all of the property values for a specified profile.

The syntax for the `list` subcommand is as follows:

```
$ netcfg list [ object-type [ class ] object-name ]
```

EXAMPLE 4–11 Listing All of the Property Values of an NCU

For example, to list all of the property values for an IP NCU in the `User` NCP, you would type the following command:

```
$ netcfg "select ncp User; list ncu ip net0"
NCU:net0
        type                    interface
        class                   ip
        parent                  "User"
        enabled                 true
        ip-version              ipv4
        ipv4-addrsrc            dhcp
        ipv6-addrsrc            dhcp,autoconf
```

EXAMPLE 4–12 Listing All of the Property Values of an ENM

In the following example, all of the properties for an ENM named `myenm` are listed.

```
$ list enm myenm
ENM:myenm
activation-mode manual
enabled         true
start           "/usr/local/bin/myenm start"
stop            "/bin/alt_stop"
```

EXAMPLE 4–12 Listing All of the Property Values of an ENM *(Continued)*

In this example, the output of the `list` subcommand displays the following information:

- The activation-mode property for this ENM is set to `manual`.
- The ENM is enabled.
- The `start` and `stop` method properties have been specified, rather than using an FMRI.

Obtaining Values of a Specific Property

You can use the `netcfg` command with the `get` subcommand to obtain the specific value for a specified property. This subcommand can be used in either interactive mode or command-line mode.

The syntax for the `get` subcommand is as follows:

`netcfg get [ -V ]` *prop-name*

To obtain the value of the `ip-version` property of an NCU named `myncu`, which is a part of the `User` NCP, you would type the following command. For example:

```
$ netcfg "select ncp User; select ncu ip myncu; get -V ip-version"
ipv4
```

If the `-V` option is used with the `get` subcommand, only the property value is displayed, as shown here:

```
netcfg:ncp:User:ncu:net0> get -V activation-mode
manual
```

Otherwise, both the property and its value are displayed. For example:

```
netcfg:ncp:User:ncu:net0> get activation-mode
activation-mode         manual
```

▼ How to Interactively Obtain a Single Property Value

This procedure describes how to obtain a single property value by using the `netcfg get` command while in the `netcfg` interactive mode. In this particular procedure, some of the examples show how to obtain a single property value for an NCU in the `User` NCP. These examples are used for demonstration purposes *only*. The information that you provide when using this command would vary, depending on the profile and the property value that you attempt to retrieve.

If you want to view all of the property values for a profile, you can alternately use the `walkprop` subcommand. This subcommand walks you through all of the properties of a given profile, one

at a time, enabling you to modify one or all of the profile's properties. For more information, see "Interactively Viewing and Changing Property Values by Using the `walkprop` Subcommand " on page 99.

1 **Initiate an `netcfg` interactive session.**

```
$ netcfg
netcfg>
```

2 **Select the profile or configuration object that contains the property value that you want to obtain.**

`netcfg>` **`select`** *object-type* **`[`** *class* **`]`** *object-name*

Note – The *class* parameter is applicable *only* if you are selecting an NCU. Also, the `class` parameter must be specified if both the `phys` and `ip` class NCU share the same name. However, if the NCU name is unique, the `class` parameter is not required.

For example, to select the `User` NCP, you would type:

```
netcfg> select User NCP
```

In this example, selecting the `User` NCP moves the interactive session into the selected object's scope.

3 **(Optional) Display the components of the profile.**

```
netcfg:ncp:User> list
NCUs:
        phys    net0
        ip      net0
```

4 **Select the object that contains the property value that you want to obtain.**

In the following example, the link (`phys`) NCU `net0` in the `User` NCP is selected:

```
netcfg:ncp:User> select ncu phys net0
```

Selecting the NCU `net0` moves the interactive session to that object's scope and loads the current properties for the NCU from memory.

5 **Obtain the specified property value.**

`netcfg:ncp:User:ncu:net0>` `get` *property-value*

For example, to obtain the value of the `activation-mode` property, you would type:

```
netcfg:ncp:User:ncu:net0> get activation-mode
        activation-mode         manual
```

Next Steps At this point, you can set a new value for the property by using the `set` subcommand, or you can exit the interactive session without making any changes. Note that if you modify a property

value while in interactive mode, you must use the `commit` or `exit` subcommand to save your changes. For information about setting a property value in `netcfg` interactive mode, see "Setting and Changing Property Values for a Profile" on page 92.

Interactively Viewing and Changing Property Values by Using the `walkprop` Subcommand

The `walkprop` subcommand can be used interactively to view the properties of a profile. This subcommand "walks" you through a profile, one property at a time, displaying the name and current value for each property. An interactive command prompt is also displayed, that you can use to change the current value of the specified property. The delimiter for multi-valued properties is a comma (`,`). If an individual value for a given property contains a comma, it must be preceded it with a backslash (`\`). Commas within properties that only have a single value are not interpreted as delimiters and do not need to be preceded by a backslash.

Note – The `walkprop` subcommand is meaningful when used in interactive mode only.

EXAMPLE 4–13 Viewing and Changing Property Values for a Specific Profile

In the following example, the `activation-mode` property for the location `foo` is viewed and then changed by using the `walkprop` subcommand. Note that when using the `walkprop` subcommand, you do not need to use the `set` subcommand to set the property value.

```
$ netcfg
netcfg> select loc foo
netcfg:loc:foo> list
loc:foo
        activation-mode                 manual
        enabled                         false
        nameservices                    dns
        nameservices-config-file        "/etc/nsswitch.dns"
        dns-nameservice-configsrc       dhcp
        nfsv4-domain                    "Central.oracle.com"
netcfg:loc:foo> walkprop
activation-mode (manual) [manual|conditional-any|conditional-all]> conditional-all
conditions> advertised-domain is oracle.com
nameservices (dns) [dns|files|nis|ldap]>
nameservices-config-file ("/etc/nsswitch.dns")>
dns-nameservice-configsrc (dhcp) [manual|dhcp]>
nfsv4-domain ("Central.oracle.com")>
ipfilter-config-file>
ipfilter-v6-config-file>
ipnat-config-file>
ippool-config-file>
ike-config-file>
ipsecpolicy-config-file>
netcfg:loc:foo> list
loc:foo
        activation-mode                 conditional-all
```

EXAMPLE 4–13 Viewing and Changing Property Values for a Specific Profile *(Continued)*

```
        conditions                      "advertised-domain is oracle.com"
        enabled                         false
        nameservices                    dns
        nameservices-config-file        "/etc/nsswitch.dns"
        dns-nameservice-configsrc       dhcp
        nfsv4-domain                    "Central.oracle.com"
netcfg:loc:foo> commit
Committed changes
netcfg:loc:foo> end
netcfg> exit
$
```

Note – Only relevant properties are walked. For example, if the `ipv4-addrsrc` property is set to `static`, the `ipv4-addr` property is included in the walk. However, if `ipv4-addrsrc` is set to `dhcp`, the `ipv4-addr` property is not walked.

Exporting and Restoring a Profile Configuration

You can use the `export` subcommand to save and restore profile configurations. Exporting a profile can be useful for system administrators who are responsible for maintaining multiple servers that require identical network configurations. The `export` subcommand can be used in either interactive or command-line mode. Or, you can use the command in command-file mode to specify a file as the output of the command.

The command syntax for the `export` subcommand is as follows:

```
$ netcfg export [ -d ] [ -f output-file ] [ object-type  [ class ] object-name ]
```

Note – The `-d` and `-f` options of the `export` subcommand can be used independently of each other.

EXAMPLE 4–14 Exporting a Profile Configuration

In the following example, the `export` subcommand is used to display a system's profile configuration on the screen.

```
$ netcfg
netcfg> export
create ncp "User"
create ncu ip "net2"
set ip-version=ipv4
set ipv4-addrsrc=dhcp
set ipv6-addrsrc=dhcp,autoconf
end
```

EXAMPLE 4–14 Exporting a Profile Configuration *(Continued)*

```
create ncu phys "net2"
set activation-mode=manual
set link-mtu=5000
end
create ncu phys "wpi2"
set activation-mode=prioritized
set priority-group=1
set priority-mode=exclusive
set link-mac-addr="13:10:73:4e:2"
set link-mtu=1500
end
end
create loc "test"
set activation-mode=manual
set nameservices=dns
set nameservices-config-file="/etc/nsswitch.dns"
set dns-nameservice-configsrc=dhcp
set nfsv4-domain="domainl.oracle.com"
end
create loc "foo"
set activation-mode=conditional-all
set conditions="system-domain is oracle.com"
set nameservices=dns
set nameservices-config-file="/etc/nsswitch.dns"
set dns-nameservice-configsrc=dhcp
set nfsv4-domain="domain.oracle.com"
end
create enm "myenm"
set activation-mode=conditional-all
set conditions="ip-address is-not-in-range 1.2.3.4"
set start="/my/start/script"
set stop="/my/stop/script"
end
create wlan "mywlan"
set priority=0
set bssids="0:13:10:73:4e:2"
end
netcfg> end
$
```

EXAMPLE 4–15 Exporting a Profile Configuration in `netcfg`Interactive Mode

In the following example, the `-d` option is used with the `export` subcommand. The `-d` option adds the `destroy -a` command as the first line of the `netcfg export` output.

```
$ netcfg
netcfg> export -d
destroy -a
create ncp "User"
create ncu ip "net2"
set ip-version=ipv4
set ipv4-addrsrc=dhcp
set ipv6-addrsrc=dhcp,autoconf
end
create ncu phys "net2"
```

EXAMPLE 4–15 Exporting a Profile Configuration in `netcfg`Interactive Mode *(Continued)*

```
set activation-mode=manual
set link-mtu=5000
end
create ncu phys "wpi2"
set activation-mode=prioritized
set priority-group=1
set priority-mode=exclusive
set link-mac-addr="13:10:73:4e:2"
set link-mtu=1500
end
end
create loc "test"
set activation-mode=manual
set nameservices=dns
set nameservices-config-file="/etc/nsswitch.dns"
set dns-nameservice-configsrc=dhcp
set nfsv4-domain="domain.oracle.com"
end
create loc "foo"
set activation-mode=conditional-all
set conditions="system-domain is oracle.com"
set nameservices=dns
set nameservices-config-file="/etc/nsswitch.dns"
set dns-nameservice-configsrc=dhcp
set nfsv4-domain="domain.oracle.com"
end
create enm "myenm"
set activation-mode=conditional-all
set conditions="ip-address is-not-in-range 1.2.3.4"
set start="/my/start/script"
set stop="/my/stop/script"
end
create wlan "mywlan"
set priority=0
set bssids="0:13:10:73:4e:2"
end
netcfg> end
$
```

EXAMPLE 4–16 Exporting a Profile Configuration in `netcfg` Command-File Mode

In the following example, the configuration information for the `User` NCP is written to a file by using the `netcfg export` command with the `-f` option. The `-f` option writes the output to a new file named `user2`. The `-d` option adds the `destroy -a` command as the first line of the `netcfg export` output.

```
$ netcfg export -d -f user2 ncp User

$ ls -al
drwx------   3 root     root           4 Oct 14 10:53 .
drwxr-xr-x  37 root     root          40 Oct 14 10:06 ..

-rw-r--r--   1 root     root         352 Oct 14 10:53 user2
$
```

EXAMPLE 4–16 Exporting a Profile Configuration in `netcfg` Command-File Mode *(Continued)*

```
$ cat user2
destroy -a
create ncp "User"
create ncu ip "net2"
set ip-version=ipv4
set ipv4-addrsrc=dhcp
set ipv6-addrsrc=dhcp,autoconf
end
create ncu phys "net2"
set activation-mode=manual
set link-mtu=5000
end
create ncu phys "wpi2"
set activation-mode=prioritized
set priority-group=1
set priority-mode=exclusive
set link-mac-addr="13:10:73:4e:2"
set link-mtu=1500
end
end
create loc "test"
set activation-mode=manual
set nameservices=dns
set nameservices-config-file="/etc/nsswitch.dns"
set dns-nameservice-configsrc=dhcp
set nfsv4-domain="domain.oracle.com"
end
create loc "foo"
set activation-mode=conditional-all
set conditions="system-domain is oracle.com"
set nameservices=dns
set nameservices-config-file="/etc/nsswitch.dns"
set dns-nameservice-configsrc=dhcp
set nfsv4-domain="domain.oracle.com"
end
create enm "myenm"
set activation-mode=conditional-all
set conditions="ip-address is-not-in-range 1.2.3.4"
set start="/my/start/script"
set stop="/my/stop/script"
end
create wlan "mywlan"
set priority=0
set bssids="0:13:10:73:4e:2"
end
$
```

Restoring a User-Defined Profile

You can restore a user-defined profile by using the `netcfg` command with the `-f` option, as follows:

```
$ netcfg [ -f ] profile-name
```

For example:

```
$ netcfg -f user2
```

This command executes the command file that contains the exported configuration.

Managing Network Configuration

Network configuration management is profile-based and is managed by switching between the two network configuration modes: manual and automatic. To switch between the modes, enable the appropriate NCP. For manual network configuration, enable the `DefaultFixed` NCP. For automatic (NWAM) network configuration, enable the `Automatic` or a user-defined NCP.

▼ How to Switch From Automatic Network Configuration Mode to Manual Network Configuration Mode

If you are using advanced networking features that are not currently supported by NWAM configuration management, or if you prefer manual network configuration management, you can enable the `DefaultFixed` NCP, as shown in the following procedure.

1 Become the `root` user.

2 Enable the `DefaultFixed` NCP.

```
# netadm enable -p ncp DefaultFixed
```

3 Verify that the `network/physical:default` service has restarted and is online.

```
# svcs -xv network/physical:default
svc:/network/physical:default (physical network interface configuration)
 State: online since Fri Aug 26 16:19:18 2011
   See: man -M /usr/share/man -s 1M ipadm
   See: man -M /usr/share/man -s 5 nwam
   See: /var/svc/log/network-physical:default.log
Impact: None.
#
```

4 Verify that the `DefaultFixed` NCP is active.

```
# netadm list
netadm: DefaultFixed NCP is enabled;
automatic network management is not available.
'netadm list' is only supported when automatic network management is active.
```

Note – The `netadm` command is supported only when the network configuration is in the automatic mode. Consequently, in the manual mode, the output of the command is limited only to indicating that the `DefaultFixed` profile is enabled. No information about the other NCPs in the system is provided.

How to Switch From Manual Network Configuration Mode to Automatic Network Configuration Mode

To switch back to the automatic network configuration mode from the manual network configuration mode, enable the network configuration profile that you want to use.

1. **Become the `root` user.**

2. **Enable an NCP, for example, `Automatic`.**

```
# netadm enable -p ncp Automatic
```

3. **Verify that the `network/physical:default` service has restarted and is online.**

```
# svcs -xv network/physical:default
svc:/network/physical:default (physical network interface configuration)
 State: online since Fri Aug 26 16:19:18 2011
   See: man -M /usr/share/man -s 1M ipadm
   See: man -M /usr/share/man -s 5 nwam
   See: /var/svc/log/network-physical:default.log
Impact: None.
#
```

4. **Check the state of the NCP and the other NWAM profiles.**

```
# netadm list -x
TYPE        PROFILE        STATE          AUXILIARY STATE
ncp         Automatic      online         active
ncu:phys    net0           online         interface/link is up
ncu:ip      net0           online         interface/link is up
ncu:phys    net1           offline        interface/link is down
ncu:ip      net1           offline        conditions for activation are unmet
ncp         User           disabled       disabled by administrator
loc         Automatic      online         active
loc         NoNet          offline        conditions for activation are unmet
#
```

◆ ◆ ◆ CHAPTER 5

NWAM Profile Administration (Tasks)

This chapter describes how to use the `netadm` command to administer these profiles: NCPs, locations, ENMs, and WLANs. The `netadm` command can also be used to administer NCUs, which are the individual configuration objects that make up an NCP, and to interact with the NWAM daemon (`nwamd`) in the absence of the NWAM GUI. For more information about using the `netadm` command, see the `netadm`(1M)man page.

The following topics are covered in this chapter:

For information about creating profiles and configuring their properties by using the `netcfg` command, see Chapter 4, "NWAM Profile Configuration (Tasks)."

For information about how to interact with the NWAM configuration and how to manage your network configuration from the desktop by using the NWAM GUI, see Chapter 6, "About the NWAM Graphical User Interface."

For an introduction to NWAM, see Chapter 2, "Introduction to NWAM."

For more information about all of the NWAM components, as well as NWAM configuration details, see Chapter 3, "NWAM Configuration and Administration (Overview)."

Obtaining Information About Profile States

You can use the `netadm` command with the `list` subcommand to display all of the available profiles on a system and their current state, or to display a specific profile and its state.

The syntax for the `list` subcommand is as follows:

`netadm list` [**`-p`** *profile-type*] [**`-c`** *ncu-class*] [*profile-name*]

For example, to display all of the profiles on a system and their state, you would type the following command:

```
$ netadm list
TYPE         PROFILE        STATE
ncp          User           disabled
ncp          Automatic      online
ncu:ip       net1           offline
ncu:phys     net1           offline
ncu:ip       net0           online
ncu:phys     net0           online
loc          foo            disabled
loc          test           disabled
loc          NoNet          offline
loc          Automatic      online
$
```

In this example, every system-defined and user-defined profile that is on the system and its current state is displayed. Note that the `list` subcommand displays the enabled NCP and all of the NCUs that make up that particular NCP.

Displaying the Current State of a Profile

The profile type and NCU class can be included in the command syntax to identify a specific profile. If only a profile type is provided, all of the profiles that are of that type are displayed. If a profile is specified by name, the current state of that profile is displayed. If the profile name is not unique, all of the profiles with that name are listed.

Possible state values for each profile include the following:

`disabled` Indicates a manually activated profile that has not been enabled.

`offline` Indicates a conditionally activated or system-activated profile that has not been activated. The profile might not be active because its conditions have not been satisfied or because another profile with more specific conditions that have been met is active.

Note – The offline state occurs more often in the case of profile types that must be activated one at a time, such as the Location profile.

`online` Indicates a conditionally activated or system-activated profile that has conditions that have been met and that has been successfully activated. Or, a manually activated profile that has been successfully enabled at the user's request.

`maintenance` Indicates that the activation of the profile was attempted, but the activation failed.

`initialized` Indicates that the profile is valid, but no action has been taken on the profile.

`uninitialized` Indicates that the profile is not present in the system. For example, this state can occur when an NCU that corresponds to a physical link is removed from the system.

EXAMPLE 5–1 Displaying the Current State of a Specified Profile

The following example lists the current state of the Automatic NCP, which has been specified by name:

```
$ netadm list Automatic
TYPE         PROFILE        STATE
ncp          Automatic      online
ncu:ip       net1           offline
ncu:phys     net1           offline
ncu:ip       net0           online
ncu:phys     net0           online
loc          Automatic      online
```

In the following example, the `list` subcommand is used with the `-p` option to display all of the locations that are currently on the system:

```
$ netadm list -p loc
TYPE         PROFILE        STATE
loc          foo            disabled
loc          test           disabled
loc          NoNet          offline
loc          Automatic      online
$
```

In the following example, the `list` subcommand is used with the `-c` option to display all of the interface NCUs in the currently active NCP:

```
$ netadm list -c ip
TYPE         PROFILE         STATE
ncu:ip       net0            online
```

EXAMPLE 5–1 Displaying the Current State of a Specified Profile *(Continued)*

```
ncu:ip       net1            disabled
$
```

Auxiliary State Values

The auxiliary state of a profile provides an explanation about why a given profile is online or offline (enabled or disabled). To list auxiliary state values, use the `-x` option with the `list` subcommand, as shown in the following example:

```
$ netadm list -x
TYPE        PROFILE        STATE          AUXILIARY STATE
ncp         Automatic      disabled       disabled by administrator
ncp         User           online         active
ncu:phys    nge0           online         interface/link is up
ncu:ip      nge0           online         interface/link is up
ncu:phys    nge1           offline        interface/link is down
ncu:ip      nge1           offline        conditions for activation are unmet
loc         Automatic      offline        conditions for activation are unmet
loc         NoNet          offline        conditions for activation are unmet
loc         office         online         active
```

Auxiliary state values vary, depending on the profile type. For detailed information about auxiliary states, see the `nwamd`(1M) man page.

Activating and Deactivating Profiles

User-defined NCPs, Location profiles, and ENMs all have `activation-mode` properties. The allowable values for each profile are determined by its type.

To manually enable or disable (activate or deactivate) a profile or configuration object, use the `netadm enable` command or the `netadm disable` command. Both system-defined and user-defined profiles can be enabled and disabled, if the `activation-mode` property for the specified profile is set to `manual`. The `activation-mode` property is set when you create or modify a profile by using the `netcfg` command. For more information, see "How NWAM Profiles Are Activated" on page 54.

At any given time, there must be one active NCP and one active Location profile on the system. Enabling a different NCP or location with an `activation-mode` of `manual` implicitly deactivates the currently active NCP or Location profile. The current location can also be deactivated, if its `activation-mode` property is set to `manual`. If no other locations are available, NWAM falls back to one of the system-defined locations, either the Automatic location, if IP configuration was successful, or the NoNet location. Conditional and system locations can be manually activated, which means that the location remains active until explicitly disabled. This behavior

makes it easy to switch a conditional Location profile to "always on." Disabling the conditional location switches the system back to its normal conditional behavior. When any location is manually enabled, the system does not change the location, even if a conditionally enabled location's conditions are met.

Note – You cannot explicitly disable the NCP that is currently active on a system, as that would effectively shut down the basic network connectivity of the system. An NCP is disabled implicitly when a different NCP is manually enabled. However, there are no constraints on ENM activation. Zero or more ENMs can be active on a system at any given time. Thus, enabling or disabling an ENM has no effect on other currently active ENMs.

You can also manually enable and disable individual NCUs. Note that the specified NCU must be part of the currently active NCP and must have an `activation-mode` property of `manual`. If the NCU class is not specified, all of the NCUs (one link NCU and one interface NCU with that name) are activated or deactivated.

Activation and deactivation of objects is performed asynchronously. Therefore, the request to enable or disable might succeed, while the action (activate or deactivate) fails. A failure of this sort is reflected in the profile's state, which changes to `maintenance`, indicating that the last action taken on the profile failed. For information about displaying the state of profiles, see "Obtaining Information About Profile States" on page 108.

EXAMPLE 5–2 Enabling a Profile

The syntax to manually enable a profile is as follows:

`netadm enable` `[` **`-p`** *profile-type* `][` **`-c`** *ncu-class* `]` *profile-name*

If the profile name is not unique, for example, if there are multiple profiles with the same name, but of different types, are on the system, you must also specify the profile type.

The `-p` option can be used to specify one of the following profile types:

- `ncp`
- `ncu`
- `loc`
- `enm`

If the configuration object's type is `ncu`, the `-c` option can be used to distinguish the NCU class. The `-c` option is helpful when two NCUs with identical names are on the system.

If the `-` coption is used, it must specify either `phys` or `ip` class type.

In the following example, a location named `office` is enabled:

EXAMPLE 5–2 Enabling a Profile *(Continued)*

```
$ netadm enable -p loc office
```

where the *profile-type* is `loc`, and the *profile-name* is `office`. Note that the `-c` *ncu-class* option is not used in this example because the profile type is a location and not an NCP.

```
$ netadm enable -p ncp user
Enabling ncp 'User'
.
.
.
```

Note that when you specify profile names, the `netadm` command is case-insensitive.

EXAMPLE 5–3 Disabling a Profile

The syntax to manually disable a profile is as follows:

netadm disable [**-p** *profile-type*][**-c** *ncu-class*] *profile-name*

If the profile name is not unique, you must also specify the profile type.

The `-p` option can be used to specify one of the following profile or object types:

- `ncp`
- `ncu`
- `loc`
- `enm`

If the configuration object's type is an `ncu`, the `-c` option must also be used to distinguish the NCU class.

The NCU class must be specified as either `phys` or `ip`.

For example, to manually disable a link NCU named `net1`, you would type the following command:

```
$ netadm disable -p ncu -c phys net1
```

where the *profile-type* is `ncu`, and the *ncu-class* is `phys`, and the *profile-name* is `net1`. Note that the `-c` *ncu-class* option is used in this example because the configuration object is an NCU.

EXAMPLE 5–4 Switching Profiles

To change the active NCP and enable manual configuration, you would type the following command:

```
$ netadm enable -p ncp DefaultFixed
```

EXAMPLE 5–4 Switching Profiles *(Continued)*

Similarly, to enable automatic (NWAM) configuration with the Automatic NCP, you would type the following command:

```
$ netadm enable -p ncp Automatic
```

For more information about `netadm`, see the `netadm(1M)` man page.

Performing a Wireless Scan and Connecting to Available Wireless Networks

You can scan for and connect to available wireless networks by using the `netadm` command.

Use the `netadm scan-wifi` *link-name* command to scan a wireless link to obtain a list of available wireless networks.

Use the `netadm select-wifi` *link-name* command to select and connect to a wireless network from the scan results on the link that is specified as *link-name*. The `select-wifi` *link-name* subcommand prompts you for a WiFi selection, a key, and a key slot, if required.

Note – You must have already created a key prior to using the `netadm select-wifi` command.

You can also trigger a subsequent scan of the network to search for available wireless networks by using the `netadm scan-wifi` *link-name* command. Note that a subsequent scan might not trigger a scan event, if the new scan results are identical to the existing scan results. The `nwamd` daemon performs the scan, regardless of whether the data has changed since the last scan.

In the following example, the `netadm scan-wifi` command is used to perform a scan of the wireless link, `net1`. The `netadm select-wifi` command is then used to display a list of wireless networks from which to select. The list that is displayed is based on the results of the scan that was previously performed on `net1`.

```
$ netadm select-wifi net1
1: ESSID home BSSID 0:b:e:85:26:c0
2: ESSID neighbor1 BSSID 0:b:e:49:2f:80
3: ESSID testing BSSID 0:40:96:29:e9:d8
4: Other
Choose WLAN to connect to [1-4]: 1
$
```

In this example, the wireless network that is represented by the number `1`, selects the `home` network.

If the WLAN requires a key, you are prompted to enter the key and key slot, if WEP is specified. For example:

```
Enter WLAN key for ESSID home: mywlankey
Enter key slot [1-4]: 1
```

Troubleshooting NWAM Network Configuration

The information in this section describes how to troubleshoot NWAM network configuration issues.

Monitoring the Current State of All Network Connections

The `netadm` command can be used with the `show-events` subcommand to listen for and display events that are being monitored by the NWAM daemon, `nwamd`. This subcommand provides useful information about events that are related to the configuration process for profiles and configuration objects, as they are configured by NWAM.

The syntax for the `netadm show-events` command is as follows:

```
netadm show-events [-v]
```

In the following example, the `nwam show-events` command is used with the `-v` option to display events in verbose mode:

```
$ netadm show-events -v
EVENT                   DESCRIPTION
LINK_STATE              net0 -> state down
OBJECT_STATE            ncu link:net0 -> state online*, interface/link is down
OBJECT_STATE            ncu link:net0 -> state offline, interface/link is down
OBJECT_STATE            ncu interface:net0 -> state online*, conditions for act
OBJECT_STATE            ncu interface:net0 -> state offline, conditions for act
IF_STATE                net0 -> state (0) flags 2004801
IF_STATE                net0 -> state (0) flags 2004800
IF_STATE                net0 -> state (0) flags 1004803
IF_STATE                net0 -> state index 4 flags 0x0 address fe80::214:4fff:
IF_STATE                net0 -> state (0) flags 1004802
IF_STATE                net0 -> state index 4 flags 0x0 address 129.156.235.229
IF_STATE                net0 -> state (0) flags 1004803
IF_STATE                net0 -> state (0) flags 1004802
IF_STATE                net0 -> state (0) flags 1004803
IF_STATE                net0 -> state (0) flags 1004802
```

Troubleshooting Network Interface Configuration Issues

The `netadm list -x` command is useful for determining why a network interface might not be configured correctly. This command displays the various entities that are configured by NWAM, their current state, and the reason why these entities are in that state.

For example, if a cable is unplugged, you can use the `netadm list -x` command to determine if the link state is offline and why, for example, "link is down." Similarly, for duplicate address detection, the output of the `netadm list -x` command reveals that the physical link is online (up), but the IP interface is in a maintenance state. In this instance, the reason that is given is "Duplicate address detected."

The following is an example of the output of the `netadm list -x` command:

```
$ netadm list -x
TYPE        PROFILE        STATE          AUXILIARY STATE
ncp         Automatic      online         active
ncu:phys    net0           offline        interface/link is down
ncu:ip      net0           offline        conditions for activation are unmet
ncu:phys    net1           offline*       need WiFi network selection
ncu:ip      net1           offline        conditions for activation are unmet
ncp         User           disabled       disabled by administrator
loc         Automatic      offline        conditions for activation are unmet
loc         NoNet          online         active
loc         office         offline        conditions for activation are unmet
$
```

After determining the reason that a link or interface is offline, you can proceed to correct the problem. In the case of a duplicate IP address, you must modify the static IP address that is assigned to the specified interface by using the `netcfg` command. For instructions, see "Setting and Changing Property Values for a Profile" on page 92. After you commit the changes, run the `netadm list -x` command again to check that the interface is now configured correctly, and that its state is displayed as `online`.

Another example of why an interface might not be configured correctly is if no known WLANs are available. In this case, the WiFi link's state would be displayed as `offline`, and the reason would be "need wifi selection". Or, if a WiFi selection was made, but a key is required, the reason would be "need wifi key".

◆ ◆ ◆ CHAPTER 6

About the NWAM Graphical User Interface

This chapter provides an introduction to the NWAM graphical user interface (GUI), which includes a description of the components that make up the NWAM GUI. Basic instructions for interacting with NWAM from the desktop, controlling network connections, adding wireless networks, and creating and managing network profiles are also included in this chapter.

This chapter does not provide step-by-step instructions on managing your network exclusively by using the GUI. For detailed instructions, refer to the online help, which can be accessed by right-clicking the Network Status icon that is displayed in the panel notification area of the desktop at all times. Links within the GUI take you to pages in the online help that provide more detailed information about each topic. You can also navigate through the online help by clicking links that are displayed in the text or by clicking the various topics in the side pane.

The following topics are covered in this chapter:

Introduction to the NWAM Graphical User Interface

The NWAM graphical user interface (GUI) is the graphical equivalent to the NWAM command-line user interface. The NWAM GUI enables you to view and monitor the status of your network in the desktop, as well as interact with NWAM to manage Ethernet and wireless configuration. In addition, you can perform various networking tasks from the desktop, such as connecting to a wired or wireless network at startup and configuring new wired or wireless networks. The NWAM GUI can also be used to create and manage locations, which are profiles that simplify the complex task of system-wide network configuration. The GUI component

includes a feature that displays notifications about the current status of your network connection, as well as information about the overall condition of your network environment.

Basic feature capabilities of the NWAM GUI include the following:

- Network status notification
- Detection of hot-plugged events
- Creation and management of network profiles
- Management of wireless networks

The NWAM GUI manages network configuration the same way that the NWAM CLI does, by storing desired property values in the form of profiles on the system. The NWAM service determines which profile should be active at a given time, based on current network conditions, and then activates the most appropriate profile.

Accessing the NWAM GUI From the Desktop

There are two components that make up the NWAM GUI: the Network Status notification icon that is displayed continuously on the desktop panel and the network configuration dialogs that can be accessed both from the System → Administration menu or by right-clicking the notification icon. The NWAM GUI behaves much the same as any other application that has a continuous status notification icon, for example, the power management icon or the printer icon. These applications enable you perform certain tasks by accessing their right-click (context) menu or by using configuration dialogs that are accessed from either the icon or from various preferences menus.

The panel icon is your most frequent point of contact with NWAM. The icon shows whether you are currently connected to a wired or wireless network. By hovering your mouse over the icon, a tool tip displays additional information, such as the currently active NCP and Location profile. By right-clicking the icon, you can change basic network configuration of your system, such as connecting to a different wireless network.

Clicking (left-clicking) the panel icon opens the Network Preferences dialog. This dialog can also be opened from the System →Administration menu. Here, you can perform more detailed network configuration such as defining static IPv4 and IPv6 addresses, setting connection priorities, managing External Network Modifiers (ENMs), and creating groups of network settings for use in different locations.

Differences Between the NWAM CLI and the NWAM GUI

You can manage network configuration through NWAM by using either the CLI or the GUI. Both user interfaces can be used to manage the network configuration and interact with the NWAM configuration. Whether you choose to use the CLI or the GUI to perform a particular

task depends on the task and the given situation. For some tasks, the most logical choice is to use the NWAM GUI. An example would be checking the status of your currently active network connection or choosing a wireless network to connect to at startup. These tasks can be more easily and quickly performed by directly interacting with NWAM from the desktop through the GUI. For more complicated tasks, such as specifying a script as the start and stop method for a new ENM, you might choose to work in the command-line mode.

Although the CLI and GUI are essentially the same, the following differences should be noted:

- **Functionality differences**

  The GUI includes functionality that enables you to interact with NWAM and check network connections from the desktop. How you obtain information regarding the status of your network varies slightly between the GUI and the CLI utilities. If you are using the GUI component, notifications are displayed on the desktop as they occur. If you are using the command-line utility, you can monitor NWAM events as they occur by using the `netadm show-events` command. For more information, see "Monitoring the Current State of All Network Connections" on page 114.

  Also, to obtain information about the status of your network by using the GUI, you would visually check, hover your mouse over, or click the Network Status notification icon that is displayed on the desktop. To obtain information about the status of your network from the command line is to use the `netadm` command with the `list` subcommand. The output of this command provides information about the basic state of each network object that is configured on your system. However, the GUI provides more complete information and details about your network status, such as which wireless network you are connected to and the IP address of your network connection.

  Some commands that you can perform by using the CLI cannot be performed by using the GUI. For example, you cannot export a profile configuration by using the GUI component. To export a profile configuration, use the `netcfg export` command. For more information, see "Exporting and Restoring a Profile Configuration" on page 100.

- **Component name and term usage differences**

  In the GUI, a Network Configuration Profile (NCP) is the same as a *Network Profile*. What are called Network Configuration Units (NCUs) in the CLI are referred to as *network connections* in the GUI.

  Enabling and disabling NCPs by using the command-line interface is the same as the *Switching network profiles or connections* task if you are using the GUI.

Functional Components of the NWAM GUI

The NWAM GUI includes several functional components that are used to accomplish virtually the same tasks that you can perform by using the CLI. Table 6–1 describes each of these components. Note that some dialogs can be accessed or opened several different ways. Also, some dialogs display different information, depending on how the dialog was accessed. Specific information about these differences are noted in the related sections throughout this chapter and explained in detail in the online help.

TABLE 6–1 NWAM GUI Primary Components

Component	Function	How to Access
Network Status notification icon	Method for viewing the status of your network and interacting with NWAM from the desktop. The icon also contains a contextual menu that can be accessed to create and manage network configuration by using the GUI.	■ By viewing the icon, which is displayed on the desktop panel's notification area at all times. ■ By hovering your mouse over the icon to display a tool tip that provides information about your current network status. ■ By clicking the icon, which displays the Network Preferences dialog. ■ By right-clicking the icon, which opens it contextual menu.
Network Preferences dialog	Method for activating and managing the two primary network profile types, the system-defined Automatic profile and multiple user-defined network profiles. The Automatic and user-defined network profiles manage network configuration for individual network interfaces. This dialog is also used to configure IPv4 and IPv6 addresses for individual network interfaces and to manage favorite wireless networks.	■ By clicking the Network Status notification icon on the desktop. ■ By selecting System → Administration → Network from the Main Menu bar on the desktop panel. ■ By selecting Network Preferences from the Network Status notification's icon menu.

TABLE 6–1 NWAM GUI Primary Components *(Continued)*

Component	Function	How to Access
Network Locations dialog	Method for creating, activating, and managing the properties of system-defined and user-defined Location profiles. Locations specify certain elements of a network configuration, for example a naming service and firewall settings, that are applied together when required.	■ By choosing Network Locations from the Network Status notification icon's right-click menu. ■ Or, from the Connection Status view of the Network Preferences dialog, click the Locations button.
Join Wireless Network dialog	Method for joining wireless networks and managing a list of favorite networks. **Note –** This dialog opens automatically if you attempt to add a wireless network and more information about that network is required.	■ By selecting the Join Unlisted Wireless Network option in the notification icon's right-click menu. ■ By clicking the Join Unlisted button in the Wireless Chooser dialog. ■ By clicking a notification message that says, "No wireless networks found. Click this message to join an unlisted wireless network."
Wireless Chooser dialog	Method for choosing and connecting to a wireless network.	By clicking a notification message that says, "*interface* disconnected from *ESSID*. Click this message to view other available networks." **Note –** This dialog opens automatically whenever you have a choice of available wireless networks to join.
Network Modifiers dialog	Method for adding external network modifier applications that are capable of creating or modifying network configuration.	■ By clicking the Modifiers button in the Connection Status view of the Network Preferences dialog. ■ By right-clicking the Network Status notification icon, then selecting the Network Modifier Preferences menu item.

Interacting With NWAM From the Desktop

The Network Status notification icon, which is displayed on the desktop panel's notification area at all times is the primary method for viewing the status of your network and for interacting with automatic network configuration processes. The Network Status notification icon is also where informational messages about your network are displayed. The icon's contextual (right-click) menu enables quick access to essential network functionality. The icon's appearance indicates the overall condition of your network.

Checking the Status of Your Network Connection

The quickest way to obtain essential information about your network is to look at the Network Status notification icon that is displayed in the panel notification area of the desktop. The Network Status notification icon is the primary method for viewing the status of your currently enabled network connection and for interacting with NWAM. The icon's appearance changes, depending on the status of the currently enabled network connection. Another way you can display information about your currently enabled network connection is to hover your mouse over the Network Status notification icon. To access the notification icon's context menu, right-click the icon. From here, you can change the currently enabled network interface and view more detailed information about the wireless network, if any, you are connected to.

Note – The Network Status notification icon is only displayed on the desktop if you are using NWAM to automatically configure your network.

The following table illustrates the Network Status icon's appearance, which changes to reflect the status of the network connections that are enabled on your system.

Icon	Status	Description
	All online (Wired)	Indicates all manually enabled connections that are in the enabled network profile are online and that the required number of connections in the enabled profile group (if such a group exists) are online. The "required number" is as follows: ■ One connection if the group is of the Exclusive priority type ■ One or more connections if the group is of the Shared priority type ■ All connections in the group if the group is of the All priority type

Icon	Status	Description
	All online (Wireless)	Indicates all manually enabled connections in the enabled network profile are online and that the required number of connections in the enabled profile group (if such a group exists) are online. The required number is the same as those described for the *All online (Wired)* status. Note that at least one online connection is wireless.
	Partially online (Wired)	Indicates one or more manually enabled or priority group connections are offline, such that the status is no longer *All online*. In this example, at least one wired connection is online. The Network Status notification icon is also displayed as *Partially online* if a wireless connection is pending user input, for example choosing an available wireless network or providing a wireless network password.
	Offline (Wired)	Indicates the NWAM service is disabled or in maintenance mode.

▼ How to Show Details About an Enabled Network Connection

1. **Open the Network Preferences dialog and select Connection Status from the drop-down list, if required.**

   You can open the Network Preferences dialog in one of the following ways:

   - **Click the Network Status notification icon on the desktop panel.**
   - **Choose System → Administration → Network from the Main Menu bar on the desktop panel.**
   - **Right-click the Network Status notification icon to open its menu, then select Network Preferences.**

     For wireless network connections, the IP address, signal strength, connection speed connection status, and security type are displayed.

2. **To view or edit more properties of a specific network connection, double-click the connection in the list or select the connection from the Show drop-down menu that is located at the top of the dialog.**

Controlling Network Connections From the Desktop

By default, NWAM attempts to maintain a network connection at all times. If a wired network connection fails, an attempt is made to connect to one of your favorite wireless networks. If the attempt fails, other available wireless networks are tried, with your permission.

You can also manually switch between wired and wireless networks, as required.

Note – For all connection types, the connection behavior is set for the current session *only*. When you reboot your system or disconnect, an attempt is made to establish network connections, according to the priorities that are defined by the enabled network profile.

You can control network connections from the desktop by using the NWAM in the following ways:

- **Modify the default connection priority.**

  By default, all wired network connections take priority over all wireless network connections. That is, a wireless network connection is only attempted if a wired connection cannot be established. If more than one wireless networks are available at the current location, you are prompted to select which network to join. This behavior is defined by the Automatic network profile, which is activated by default. To enforce a different behavior, you must create and activate a different network profile.

- **Switch from a wired network to a wireless network.**

  If the Automatic network profile is enabled, disconnect any network cables from all enabled wired interfaces.

  By default, if any of your favorite wireless networks are available, an attempt will be made to join them in the order in which they appear in the favorites list. Otherwise, the Wireless Chooser dialog is displayed. In this dialog you can select which network to join.

  Note – You can change the way wireless networks are joined on the Wireless tab of the Connection Properties view.

  If a network profile other than the Automatic network profile is enabled, the method that you use to switch to a wireless network depends on the definition of that network profile.

  Choose from one of the following methods:

  - Use the Connections submenu of the Network Status notification icon to disable the wired connection and then activate a wireless connection. Note that this method is only possible if both connections have the Manual activation type.
  - Edit the enabled network profile to activate the wired connection and disable other connections, as required.

When the wireless connection is established, a notification message is displayed.

- **Switch from a wireless network to a wired network.**

  If the Automatic network profile is enabled, plug a network cable into an available wired interface.

  If a network profile other than the Automatic network profile is enabled, the method that you use to switch to a wired network depends on the definition of that network profile.

  Choose from one of the following methods:

  - Use the Connections submenu of the Networks Status notification icon to disable the wireless connection and then enable a wired connection. Note that this method is only possible if both connections have the Manual activation type.
  - Edit the enabled network profile to enable the wired connection and disable the wireless connection.

    When the wired connection is established, a notification message is displayed.

For other tasks that you can perform by using the NWAM GUI, see the online help.

Joining and Managing Favorite Wireless Networks

By default, when wireless network connections are enabled, NWAM attempts to connect to any available network in the favorites list, without asking, in the priority order in which the connections are listed. If no favorite networks are available, the Wireless Chooser dialog opens. In this dialog you can choose which wireless network to join.

You can also modify the way in which connections to wireless networks are attempted in the Wireless tab of the Network Preferences dialog's Connection Properties view. If required, you can manually connect to a different wireless network by accessing the Network Status notification icon right-click menu.

Tip – You can access the Connection Properties view for a selected network through the Network Preferences dialog. This dialog contains a drop-down list that is labeled, Show. This list enables you to switch between views for a given network. In each view, there are different tasks you can perform and information about the selected network that is specific to that view.

The following views exist for every network connection in each network profile that is on the system:

- Connection status
- Network profile
- Connection properties

For more information about working with network profiles, including a description of the Network Preferences dialog, see, "Managing Network Profiles" on page 127.

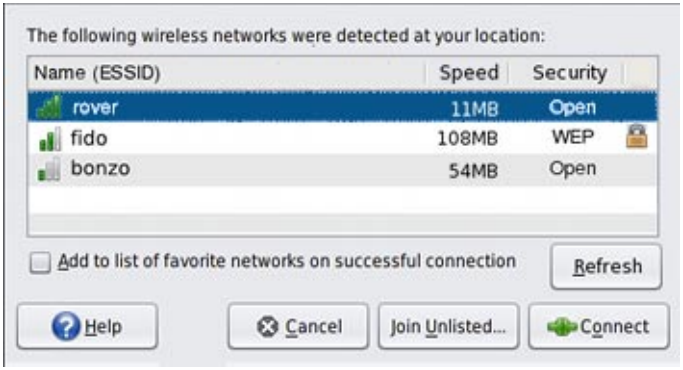
How to Join a Wireless Network

Wireless networks are joined by choosing the Join Wireless Network option that is available by right-clicking the Network Status notification icon. The Wireless Chooser dialog is where you select a wireless network to connect to, from the list of available networks that is displayed.

1 **To manually connect to a different wireless network, you can do one of the following:**

- **Select an available wireless network from the Network Status notification icon's right-click menu.**
- **Select the Join unlisted wireless network option from the Network Status notification's icon menu.**

  An unlisted wireless network is one that has been configured so that it does not broadcast its network name, yet is still available to join.
- **Select an available wireless network from the Wireless Chooser dialog. This dialog is displayed automatically, when there is a choice of available wireless networks to join.**



2 **If the Join Wireless Network dialog opens, provide all of the necessary information for the wireless network you have chosen.**

For more details about the information that you might need to provide, refer to the NWAM GUI online help.

Managing Favorite Networks

By default, when you join a wireless network for the first time, a check box that is labeled, Add to List of Favorite Networks on Successful Connection, is displayed in the Join Wireless Network dialog.

- To add the wireless network to your list of favorites, if the connection is successful, select this box. If you do not want the network to be added to your list of favorites, deselect the box. The box is selected by default.
- To add a wireless network that is not currently available, or not currently broadcasting its network name to your favorites list, go to the Wireless tab of the Connection Properties view, then click the Add button. To add the network, you will need to know its network name, security type, and security key.

Managing Network Profiles

When using the NWAM GUI, network profiles are the equivalent to the NCPs that are described in "Description of an NCP" on page 44.

A network profile specifies which network interfaces can be enabled or disabled at any given time. Using network profiles can be helpful in situations where you have more than one network interface available. For example, most modern laptop brands have both a wired and a

wireless interface. Depending on your physical location, and your work environment, you might want to use only one of those interfaces and disable the other interface for security or other reasons.

There are two network profile types that are available in the NWAM GUI, the default Automatic network profile and the user-defined network profile. You can enable and disable both types of profiles. You can modify user-defined profiles, but not the Automatic profile. You cannot create or destroy the Automatic profile by using the NWAM GUI or the CLI. However, you can create, modify, and destroy user-defined network profiles by using either the GUI or the CLI.

By default, the Automatic network profile first attempts to enable one wired connection. If that attempt fails, it then attempts to enable one wireless connection.

About the Network Preferences Dialog

The Network Preferences dialog is where individual network connections are configured and how the current state of each network connection is viewed. The dialog provides access to various views that you can switch to by using the drop-down list located at the top of the dialog.

You can open the dialog in the following ways:

- By clicking the Network Status notification icon on the desktop.
- By selecting System → Administration → Network from the Main Menu bar on the desktop panel.
- By selecting Network Preferences from the Network Status notification's icon menu.

At the top of the Network Preferences dialog is a drop-down list that is labeled, Show. This list enables you to switch between the Connection Status view, the Network Profile view, and the Connection Properties view for every network connection in each network profile.

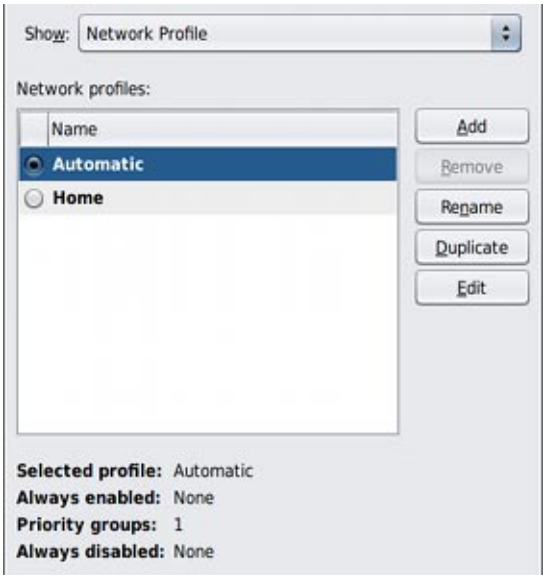
Connection Status View

- The Connection Status view displays information about each enabled network connection in the enabled network profile that has a manual activation type and each connection (whether enabled or disabled) in the active priority group. The Enabled Connections: section lists all of the enabled connections, in the same order that they are listed in the Network Profile view. See "How to Show Details About an Enabled Network Connection" on page 123.

Network Profile View

- Network profile information can be viewed in the Network profile view of the Network Preferences dialog.

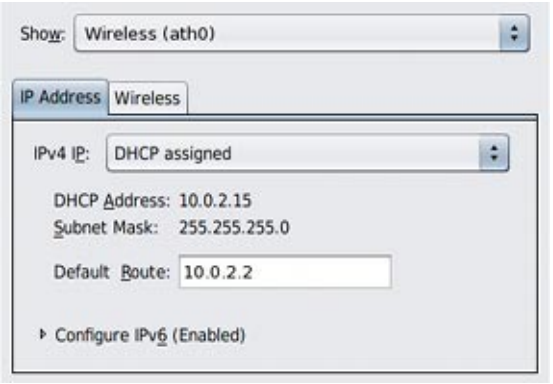
  To display this view, select Network Profile in the drop-down list that is located at the top of the Network Preferences dialog.



Connection Properties View

- The Connection Properties view enables you to view and change properties of a specified network connection. To switch to this view, select the connection name from the Show drop-down list or double-click the connection name while in either the Connection Status or the Network Profile view. A tabbed view is displayed, whereby you can view or edit the connection's properties.

  The Connection Properties view has two tabs: an IP address tab and a wireless tab. The wireless tab is only displayed if the connection type is wireless. In this IP address tab, you can configure both IPv4 and IPv6 addresses. In the wireless tab, you can configure the list of favorite networks and choose how the wireless interface connects to available networks.



Viewing Information About Network Profiles

Network profile information can be viewed in the Network profile view of the Network Preferences dialog.

To display this view, select Network Profile in the drop-down list that is located at the top of the Network Preferences dialog.

The Network Profiles list displays the name of each available network profile. The currently enabled profile is shown with a radio button indicator. By default, there is one profile, Automatic, that you can activate, but not edit or delete. However, you can create multiple additional network profiles. Network profiles that are manually created can be activated, edited, or deleted, as needed.

Below the Network Profiles list is a summary of the profile that is selected. To view the selected profile in full or edit the profile, click the Edit button.

Note – The *selected* profile might be different than the *enabled* profile.

Switching From One Network Profile to Another Network Profile

1. Open the Network Profile view of the Network Preferences dialog.
2. Select the radio button next to the network profile that you want to activate.
3. To switch network profiles, click OK or click Cancel to close the dialog without switching profiles.

Adding or Removing a Network Profile

To create or edit a network profile, select Network Profile from the drop-down list that is located at the top of the Network Preferences dialog.

- To create a new network profile, click the Add button, then type the name of the new profile.
- To duplicate an existing network profile, select the profile in the list, click the Duplicate button, then type the name of the new profile.
- To remove a network profile, select the profile in the list, then click the Remove button.

Note – You cannot remove the Automatic network profile.

For more information about editing a profile that you have added or duplicated, see "Editing Network Profiles" on page 131.

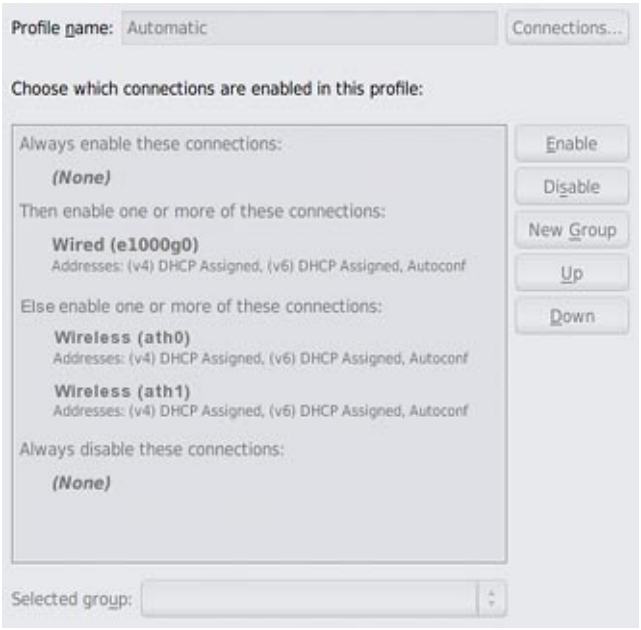
Editing Network Profiles

When you manually add a new network profile or duplicate an existing network profile, you must edit the new profile to specify those network connections that are enabled and disabled by the new profile.

Note – You can edit and remove a manually created network profile. However, you cannot edit or remove the Automatic network profile.

▼ How to Open the Network Profile Dialog

- **To edit a network profile, select the profile in the Network Profile view of the Network Preferences dialog, then click the Edit button.**



The list of network profiles consists of a minimum of two top level group descriptions. For example, the Automatic profile, which is shown in the preceding figure, contains four group descriptions that are explained in more detail in the following sections.

Note – The Automatic network profile cannot be changed or deleted. Any time the Automatic network profile is selected in the Edit Network Profile dialog, all of the profile editing buttons and drop-down lists are disabled.

For more information, see the online help.

Working With Priority Groups

A network connection in the “always enabled" group is always enabled when the selected network profile is active.

To move a network connection to the “always enabled" group, first select the connection, then do one of the following:

- Click the Enable button.
- Click the Up button until the connection moves to the “always enabled” group.

A network connection in the “always disabled" group is always disabled when the selected network profile is active.

To move a network connection to the “always disabled" group, first select the connection, then do one of the following:

- Click the Disable button.
- Click the Down button until the connection moves into the “always disabled" group.

You can create a network profile that treats one or more network interfaces as a group. If one or more of the interfaces in the highest priority group cannot be enabled, according to the group's priority type, then the group with the next highest priority is considered.

The following table describes the three different priority groups that are available.

Priority Type	Description
`Exclusive`	One connection in the group is enabled, and all the other connections are disabled. As long as at least one connection in the group is enabled (not necessarily the same one all the time), no attempt is made to enable connections in any of the lower priority groups.
`Shared`	All of the connections in the group that can be enabled are enabled. As long as at least one connection in the group remains enabled, no attempt is made to enable connections in any of the lower priority groups.
`All`	All of the connections in the group are enabled. If any of the connections are lost, all of the connections in the group are disabled. As long as all of the connections remain enabled, no attempt is made to enable connections in any of the lower priority groups.

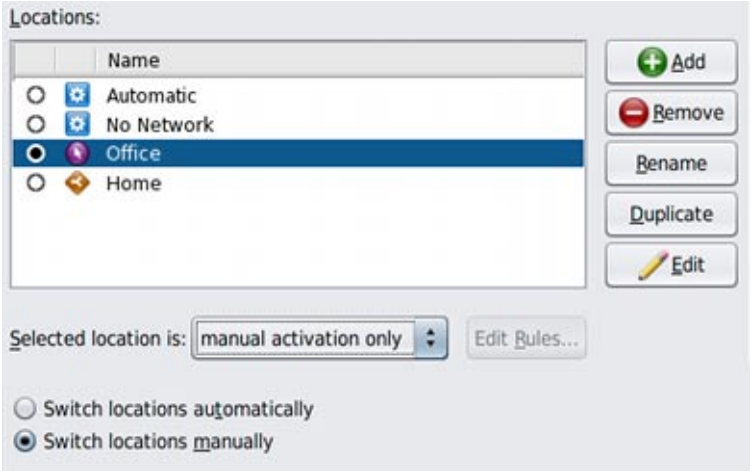
For example, the default Automatic network profile contains two exclusive priority groups. The higher priority group contains all of the *wired* network connections. The lower priority group contains all of the *wireless* network connections.

For detailed instructions on performing these and other tasks, see the online help.

Creating and Managing Locations

A location comprises certain elements of a network configuration, for example a naming service and firewall settings, that are applied together, when required. You can create multiple locations for various uses. For example, one location can be used when you are connected at the office by using the company intranet. Another location can be used at home when you are connected to the public Internet by using a wireless access point. Locations can be activated manually or automatically, according to environmental conditions, such as the IP address that is obtained by a network connection.

The Network Locations dialog is where you can to switch locations, edit location properties, create new locations, and remove locations. Note that only user-defined locations can be created and removed. The Location dialog can be opened from the Connection Status view of the Network Preferences dialog.



The Locations list is similar to the list on the Network Status notification's icon menu. Each available location, with an icon that represents its activation type, is listed.

Location types are as follows:

- System – Locations with this type are system-defined locations (Automatic and No Network), which means the system determines when to activate the location, based on current network conditions.
- Manual – Locations with this type can be manually enabled or disabled by using the Network Locations dialog or by interacting with the Network Status notification icon.
- Conditional – Locations with this type are enabled or disabled automatically, according to the rules that you specify during the creation of the location.

The activation type of a selected location is also displayed in the Selected location drop-down list. The enabled location is represented by a selected radio button that is displayed in the first column of the list.

How to Change a Location's Activation Mode

The following task describes how to change the activation mode for a location by using the NWAM GUI. If you are using the `netcfg` command, you would change the activation mode by modifying the properties of the specified location. For more information, see "Setting and Changing Property Values for a Profile" on page 92.

1. **From the Network Status notification icon's Location submenu, choose Network Locations. Or, from the Connection Status view of the Network Preferences dialog, click the Locations button.**

2. **To change the activation mode of a location, select the location in the list, then select the new activation mode from the Selected location drop-down list.**

Note – Note that when a system location is selected, the drop-down list displays Activated by system, and both the drop-down list and the Edit Rules button are disabled.

When a manual or a conditional location is selected, the drop-down list options are as follows:

- Manual activation only: This location is only enabled when it is manually selected. When this option is selected, the Edit Rules button is *disabled*.
- Activated by rules: This location is automatically selected under certain network conditions. When this option is selected, the Edit Rules button is *enabled*.

3. **(Optional) To set rules for how and when a location is activated, click the Edit Rules button.**

   For further instructions, see "Working With the Rules Dialog" in the online help.

How to Switch From One Location to Another Location

The following task describes how to switch from one location to another location by using the NWAM GUI. To switch locations by using the CLI, use the `netadm` command to activate a new location. Because exactly one location must be activated on the system at all times, activating a new location implicitly disables the currently enabled location. The same rule applies when activating a network profile. For more information about activating and deactivating locations, see "Activating and Deactivating Profiles" on page 110.

- **From the Network Status notification icon's Location submenu, choose the location that you want to activate.**

  If the Switch Locations Automatically option is selected on the Locations submenu, you cannot manually choose a location to activate. The most appropriate System or Conditional location will be activated automatically at any given time, according to changes in the network environment.

  If the Switch Locations Manually option is selected on the Location submenu, you can activate any available location, regardless of its activation type. The selected location remains activated indefinitely.

  - **Alternatively, you can switch locations in the Network Locations dialog. To do so, follow these steps:**

    a. **From the Network Status notification icon's Location submenu, select Network Locations. Or, from the Connection Status view of the Network Preferences dialog, click the Locations button.**

    b. **Select the radio button of the location to which you want to switch, then click OK.**

       - **If the Switch Locations Automatically radio button is selected in the Network Locations dialog, you cannot manually choose a location to activate. The most appropriate System or Conditional location is activated automatically at any given time, according to changes in the network environment.**

       - **If the Switch Locations Manually radio button is selected in the Network Locations dialog, you can activate any available location, regardless of its activation type. Note that location remains activated indefinitely.**

Editing Locations

Editing a location by using the NWAM GUI is the equivalent to modifying a location's properties if you are using the NWAM CLI.

To edit a location, choose Network Locations from the Network Status notification icon's Location submenu. Or, from the Connection Status view of the Network Preferences dialog, click the Locations button.

To edit the properties of a specified location, select the location in the list, then click Edit.

Alternatively, you can double-click the location in the list.

The Edit Location dialog opens, with the following two tabs available:

Name Services Enables you to configure naming services in the specified location.

Security	Enables you to select configuration files to be used by IP Filter and IPsec features, when the specified location is enabled.

To display the information to be edited, select the appropriate tab.

About External Network Modifiers

External Network Modifiers (ENMs) are profiles that are created for applications that are external to NWAM. However, these applications can create and modify network configuration. For example, VPN applications enable your network connections to communicate with a virtual private network. ENMs are configured and monitored in the NWAM GUI by using the *Network Modifiers* dialog.

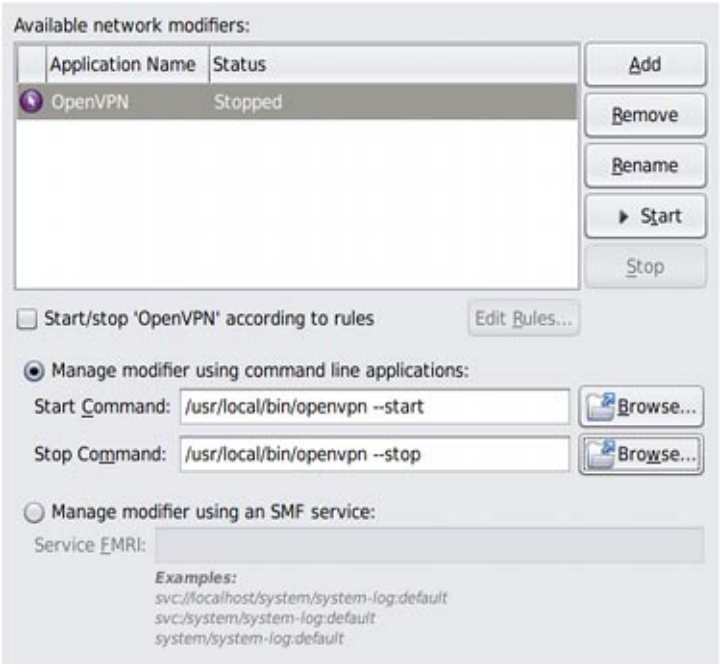
Note – Before you can manage a network modifier application or service by using the NWAM GUI, you must manually install it, then complete any initial setup, such as the installation of a certificate or shared secret.

An ENM can be started and stopped manually, as required. An ENM can also be started automatically, according to user-defined rules. To be managed by using this dialog, a network modifier application must either be implemented as a command-line tool, or as an SMF service.

To learn more about how to create and manage ENMs by using the NWAM CLI, see "Creating an ENM Profile" on page 86.

About the Network Modifiers Dialog

This dialog is used to add or remove, start and stop, and edit External Network Modifiers (ENMs), applications that are capable of creating and modifying network configuration.



Open the dialog using one of the following methods:

- Click the Modifiers button in the Connection Status view of the Network Preferences dialog.
- Right-click the Network Status notification icon, then choose the Network Modifier Preferences menu item.

The main section of the dialog is a three-column list that displays the following information for each ENM:

- Activation state (Manual or Conditional)
- User-defined name, for example, “Cisco VPN”
- Current status, “Running” or “Stopped”

The Start/Stop according to rules check box is checked if the selected network modifier application has a Conditional activation type, and unchecked if the activation type is Manual. To change the activation type, toggle the check box.

▼ How to Add a Command-Line ENM

The following procedure describes how to add a command-line ENM. For information about adding a network modifier application service, see the online help.

1. **Open the Network Modifiers dialog by using one of the following methods:**
   - From then Connection Status view of the Network Preferences dialog, click the Modifiers button.
   - Right-click the Network Status notification icon, then choose the Network Modifier Preferences menu item.
2. **Click the Add button.**
3. **Type the name of the new network modifier application.**
4. **Do one of the following:**
   - **To add a new entry that will have the Manual activation type, press Enter or Tab.**

     The two Manage modifiers radio buttons are enabled. The first of these, Command Line Applications, is selected by default. The Start and Stop command fields, and the two Browse buttons, are also enabled.
   - **To cancel your changes, press Esc.**
5. **Type the command that starts the network modifier application into the Start Command field.**

   Alternatively, you can use the Browse button to open a file chooser dialog, where you can select the command to use.

   The Start button remains disabled for the network modifier application until a valid command has been typed into this field.
6. **Type the command that stops the network modifier application into the Stop Command field.**

   Alternatively, you can use the Browse button to open a file chooser dialog, where you can select the command to use.

   The Stop button remains disabled for the network modifier application until a valid command has been typed into this field.
7. **To add this application, click OK.**

   The external network modifier is added.

PART II

Datalink and Interface Configuration

This part discusses datalink and interface configuration procedures in the context of network configuration profiles as introduced in Part I, "Network Auto-Magic." The procedures apply to any fixed profile that has been enabled or activated.

◆ ◆ ◆ CHAPTER 7

Using Datalink and Interface Configuration Commands on Profiles

This chapter describes the use of traditional configuration commands such as `dladm` and `ipadm` as they relate to profile-based network configuration.

Highlights of Profile-Based Network Configuration

In this Oracle Solaris release, network configuration is based on profiles. A system's network configuration setup is managed by a specific network configuration profile (NCP) and a corresponding location profile. For a more detailed explanation of NCPs, location profiles and other profile types, their properties, and the commands that you use to manipulate and monitor profiles, see Part I, "Network Auto-Magic."

Note – For network configuration, the principal profile types are NCPs, location profiles, external network modifiers (ENMs), and wireless local area networks (WLANs). Of these types, the main profile is the NCP. Throughout this documentation, unless specified otherwise, the term *profile* refers to the NCP.

The highlights of profile-based network configuration follow:

- Only one pair of NCP and location profiles can be active at one time to manage a system's network configuration. All other existing NCPs in the system are non-operational.
- The active NCP can either be *reactive* or *fixed*. With a reactive profile, the network configuration is monitored to adapt to changes in the system's network environment. With a fixed profile, the network configuration is instantiated but not monitored.
- The values of the different properties of an NCP constitute a policy that governs how the profile manages the network configuration.
- Changes to the NCP's properties are immediately implemented as new property values, which become part of the profile's policy that manages the network configuration.

Note – On a system that has been upgraded from the Oracle Solaris 11 11 Express release, the operational network configuration prior to the upgrade becomes the active profile after the upgrade. If the previous configuration was created by the `dladm` and `ipadm` commands, that configuration constitutes the profile `DefaultFixed`, which becomes active in the system. Otherwise, the configuration becomes the profile `Automatic` that manages the system's network configuration.

Profiles and Configuration Tools

The tools to use to customize profiles depend on the active profile. If the active profile is reactive such as `Automatic`, then you use the `netcfg` and `netadm` commands to configure and monitor the profile. If the active profile is fixed such as `DefaultFixed`, then you use the `dladm` and `ipadm` commands.

The `dladm` and `ipadm` commands are effective only on active profiles. Consequently, before you use these commands, you must make sure of the following:

- Know which profile is active to ensure that you make changes to the correct target profile by using the appropriate commands.
- Know whether the target profile is reactive or fixed to avoid causing unexpected configuration behaviors after using the commands. A reactive profile manages the network configuration differently from a fixed profile. Accordingly, the behavior of the two profiles also differs when changes are implemented.

Note – Using the `-t` option of the `dladm` and `ipadm` commands to create temporary settings can be effective only on a fixed profile. The option is not supported on reactive profiles.

Follow these two procedures to properly use the `dladm` and `ipadm` commands on profiles.

▼ How to Determine the Network Management Mode

A system's network management mode is automatic if a reactive NCP such as `Automatic` is the active NCP in the system. Use this procedure to know the network management mode before performing any network configuration. The procedure ensures that you are using the correct commands to implement configuration on the appropriate profile.

1 List the profiles in the system.

```
# netadm list -x
TYPE        PROFILE        STATE          AUXILIARY STATE
ncp         Automatic      online         active
```

```
ncu:phys    net0          online        interface/link is up
ncu:ip      net0          online        interface/link is up
ncu:phys    net1          online        interface/link is up
ncu:ip      net1          offline*      waiting for IP address to be set
ncp         testcfg       disabled      disabled by administrator
loc         Automatic     offline       conditions for activation are unmet
loc         NoNet         offline       conditions for activation are unmet
loc         Lab           online        active
loc         User          disabled      disabled by administrator
```

The output provides two pieces of information:

- The `netadm list` command is supported only if the network management mode is automatic. Therefore, the generation of a profile list indicates network management is in automatic mode. Otherwise, the `netadm list` command would have generated the following message to indicate that the DefaultFixed profile is active in the system instead.

  ```
  netadm: DefaultFixed NCP is enabled; automatic network management is not available.
  'netadm list' is only supported when automatic network management is active.
  ```

- The profile list, if generated, also identifies which specific reactive NCP is enabled by means of that NCP's `online` status. In the sample output, the `Automatic` NCP is listed as the only existing reactive NCP. Other user-created NCPs would have been included in the list if these were also present in the system.

2 Make sure that the appropriate profile is active for the configuration tools that you want to use.

For example, the `dladm` and `ipadm` commands can only be used on the `DefaulfFixed` profile. However, the `netcfg` command can only be used on reactive profiles such as `Automatic`, where network management is in automatic mode.

If the profile whose properties you want to modify with your selected configuration tools is not active, proceed to the following step to enable the proper profile. Otherwise, you can begin using the tools to configure the network.

For example, you do not want network management to be in automatic mode, but prefer to use command lines such as `dladm` and `ipadm` to configure datalinks and interfaces manually. The output in Step 1 shows that the `Automatic` profile is enabled. To use command lines for network configuration, you must therefore enable the `DefaultFixed` profile.

3 To configure a different profile, enable that profile by typing the following:

```
# netadm enable -p ncp profile-name
```

For example:

```
# netadm enable -p ncp defaultfixed
```

You also use the same command syntax if network management is in automatic mode, and you want to use a different reactive NCP. From the sample output of Step 1, suppose that you want to activate the user-created NCP `testcfg` in place of `Automatic`. You will therefore type:

```
# netadm enable -p ncp testcfg
```

Caution – The command switches active profiles. When you switch active profiles, the existing network configuration is removed, and a new configuration is created. Any persistent changes that were implemented on a previously active NCP are excluded in the new active NCP.

Next Steps

The following chapters describe procedures that you can use to perform various types of datalink and interface configurations.

- To configure datalinks, see Chapter 8, "Datalink Configuration and Administration."
- To configure IP interfaces, see Chapter 9, "Configuring an IP Interface."
- To configure wireless interfaces, see Chapter 10, "Configuring Wireless Interface Communications on Oracle Solaris."
- To configure bridges, see Chapter 11, "Administering Bridges."
- To configure link aggregations, see Chapter 12, "Administering Link Aggregations."
- To configure VLANs, see Chapter 13, "Administering VLANs."
- To configure IPMP groups, see Chapter 14, "Introducing IPMP," and Chapter 15, "Administering IPMP."
- To configure the link layer discovery protocol (LLDP), see Chapter 16, "Exchanging Network Connectivity Information With LLDP."

◆ ◆ ◆ CHAPTER 8

Datalink Configuration and Administration

This chapter discusses the `dladm` command and how the command is used to configure datalinks.

Configuration of Datalinks (Tasks)

The following tables list the different datalink configuration tasks that you can perform by using the `dladm` command. The tables also links you to the step-by-step procedures to complete the tasks.

TABLE 8–1 Performing Basic Datalink Configuration (Task Map)

Task	Description	For Instructions
Rename a datalink.	Customizes a datalink name instead of using the hardware–based name.	"How to Rename a Datalink" on page 149
Display physical attributes of a datalink.	Lists physical information that underly a datalink, including type of media, associated device instance, and other information.	"How to Display Information About Physical Attributes of Datalinks" on page 151
Display state of datalinks.	Lists information about the status of datalinks.	"How to Display Datalink Information" on page 152
Remove a datalink.	Removes a link configuration that is associated with a NIC no longer in use.	"How to Delete a Datalink" on page 152

TABLE 8–2 Setting Datalink Properties (Task Map)

Task	Description	For Instructions
Modify the MTU size.	Increases the MTU size of packet transmission to handle Jumbo frames.	"How to Enable Support for Jumbo Frames" on page 154
Modify the link speed.	Switches off higher link speed and advertises only the lower link speed to allow communications with an older system.	"How to Change Link Speed Parameters" on page 156
Display information about link properties.	Lists link properties and their current configuration; lists Ethernet parameter settings.	"How to Obtain Status Information About Datalink Properties" on page 157
Configure the driver to use DMA binding.	Sets threshold that causes the driver to switch between DMA binding and `bcopy` function during transmission.	"How to Set the `e1000g` Driver to Use Direct Memory Access Binding" on page 159
Set interrupt rates.	Manually defines rates at which interrupts are delivered by the driver instead of the rate being defined automatically.	"How to Manually Set the Interrupt Rate" on page 159
Replace a network interface card (NIC).	Changes a NIC in a system during dynamic reconfiguration (DR).	"How to Replace a Network Interface Card With Dynamic Reconfiguration" on page 161
Set per-link `autopush` properties.	Configure STREAMS module to be pushed on top of a datalink.	"How to Set STREAMS Modules on Datalinks" on page 164

The `dladm` Command

After the full implementation of the GLDv3 driver configuration framework, the `dladm` command has acquired expanded capabilities over time. The framework enhances configuration of NIC drivers as follows:

- Only a single command interface, the `dladm` command, is needed to configure network driver properties.
- A uniform syntax is used regardless of the properties: `dladm` *subcommand properties datalink*.
- Use of the `dladm` command applies to both public and private properties of the driver.
- Using the `dladm` command on a specific driver does not disrupt network connections of other NICs of similar types. Thus, you can configure datalink properties dynamically.

- Datalink configuration settings are stored in a dladm repository and persist even after you reboot the system.

To avail of the advantages previously listed when you configure datalinks, you should use dladm as the configuration tool instead of the customary tools in previous releases, such as the ndd command.

To administer datalinks, you use the following dladm subcommands:

- dladm rename-link changes the name of a datalink.
- dladm show-link displays existing datalinks in the system.
- dladm show-phys displays physical attributes of datalinks.
- dladm delete-phys deletes a datalink.
- dladm show-linkprop displays the properties that are associated with the datalink.
- dladm set-linkprop sets specified datalink properties.
- dladm reset-linkprop restores properties to their default settings.
- dladm show-ether displays Ethernet parameter settings of a datalink.

The dladm command is also used to perform other types of link administration, such as the following:

- Configuring bridges. See Chapter 11, "Administering Bridges"
- Configuring link aggregations. See Chapter 12, "Administering Link Aggregations"
- Configuring VLANs. See Chapter 13, "Administering VLANs"
- Configuring tunnels. See Chapter 6, "Configuring IP Tunnels," in *Oracle Solaris Administration: IP Services*.

For more information about the commands, see the dladm(1M) man page.

The following procedures show how to use the dladm command to configure datalinks. In most cases, datalink configuration is a part of the configuration of an IP interface over that link. Thus, where applicable, the procedures include IP interface configuration steps with the ipadm command. However, IP interface configuration and the ipadm command are discussed in further detail in Chapter 9, "Configuring an IP Interface."

▼ How to Rename a Datalink

Use this procedure if you want to change a datalink name to a customized name. For example, some of the datalinks in upgraded system might have retained legacy hardware-based names and you want to change these names to generic ones.

Before You Begin Make sure that you have studied and prepared for other steps you need to perform on associated configurations that might be affected by the change of link names. For more information, see "Link Names in Upgraded Systems" on page 28.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 If an IP interface is configured over the datalink, remove the IP interface.

```
# ipadm delete-ip interface
```

3 Change the link's current link name.

```
# dladm rename-link old-linkname new-linkname
```

old-linkname Refers to the current name of the datalink. By default, the link name is hardware-based, such as `bge0`.

new-linkname Refers to any name that you want to assign to the datalink. For rules for assigning link names, refer to "Rules for Valid Link Names" on page 30. See also "Link Names in Upgraded Systems" on page 28 for further information about renaming datalinks.

If you do not want the new link name to persist across a system reboot, then use the `-t` option immediately after the subcommand. The option renames a link temporarily. The original link name reverts when the system is rebooted.

Note – You can use `dladm rename-link` to transfer link configurations from one datalink to another. For an example, see "How to Replace a Network Interface Card With Dynamic Reconfiguration" on page 161. When you rename a link for this purpose, make sure that the link that is inheriting the configuration does not have any prior existing configurations. Otherwise, the transfer fails.

Example 8–1 Changing the System's Primary Network Interface

The following example shows how you can switch the primary network interface on your system to a second NIC by renaming datalinks. The system's primary network interface is `net0`, the generic name of the datalink on `e1000g0`. This primary network interface will be switched from using `e1000g0` as the underlying interface to `nge0`. You can use this example as part of the procedure to create a new boot environment.

```
# dladm show-phys
LINK    MEDIA       STATE    SPEED    DUPLEX    DEVICE
net0    Ethernet    up       1000     full      e1000g0
net1    Ethernet    up       1000     full      nge0
```

```
# dladm rename-link net0 oldnet0
# dladm rename-link net1 net0

# dladm show-phys
LINK      MEDIA      STATE   SPEED   DUPLEX   DEVICE
oldnet0   Ethernet   up      1000    full     e1000g0
net0      Ethernet   up      1000    full     nge0
```

How to Display Information About Physical Attributes of Datalinks

This procedure lists the steps to display information about the physical attributes of a system's datalinks.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Display information about physical attributes of datalinks currently on the system.

```
# dladm show-phys
```

You can use the `-P` with this command to also display flag status of each link. A datalink becomes unavailable if its associated hardware has been removed. Without the `-P` option, the command displays only available datalinks.

To view the `/devices` path of the datalinks, use the `-v` option.

Example 8–2 Displaying Available Datalinks

In the following example, the `-P` option includes the `FLAGS` column where unavailable links are indicated. The `r` flag for the datalink `net0` indicates the hardware that is associated with the link (`nge`) has been removed.

```
# dladm show-phys
LINK        MEDIA          STATE     SPEED     DUPLEX     DEVICE
net0        Ethernet       up        100Mb     full       e1000g0
net1        Infiniband     down      0Mb       --         ibd0
net3        Ethernet       up        100Mb     full       bge0
net4        Ethernet       --        0Mb       --         nge0
```

The following example shows the links and their physical locations that are displayed when you use the `-L` option.

```
# dladm show-phys -L
LINK     DEVICE     LOCATION
net0     bge0       MB
net2     ibp0       MB/RISER0/PCIE0/PORT1
```

```
net3      ibp1        MB/RISER0/PCIE0/PORT2
net4      eoib2       MB/RISER0/PCIE0/PORT1/cloud-nm2gw-2/1A-ETH-2
```

How to Display Datalink Information

This procedure displays the status of available links.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Display link information.**

   ```
   # dladm show-link
   ```

Example 8–3 Displaying Available Links

The following example shows persistent and available links on the system.

```
# dladm show-link -P
LINK        CLASS     BRIDGE     OVER
net0        phys      --         --
net1        phys      --         --
net2        phys      --         --
```

The `-P` option also displays any existing persistent but unavailable links. A persistent link becomes unavailable if the link is temporarily deleted. A link also becomes unavailable if the associated hardware has been removed.

How to Delete a Datalink

This procedure deletes link configurations that are associated with NICs. If you detach a NIC without intending to replace it, then you can delete the link configuration that is associated with that NIC. After you complete this procedure, the link name can be reused.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Display the datalinks on the system including those links whose hardware have been removed.**

   To include information about removed hardware, use the `-P` option.

   ```
   # dladm show-phys
   ```

3. **Remove the link configuration of the removed hardware that you do not intend to replace.**

   ```
   # dladm delete-phys link
   ```

Example 8–4 Deleting a Datalink

In the following example, the `r` flag for `net2` indicates that the link's associated hardware (`e1000g0`) has been removed. Therefore, you can also remove the link `net2` and then reassign the name to a new datalink.

```
# dladm show-phys -P
LINK        DEVICE     MEDIA        FLAGS
net0        nge0       Ethernet     -----
net1        bge0       Ethernet     -----
net2        e1000g0    Ethernet     r----

# dladm delete-phys net2
```

Setting Datalink Properties

In addition to performing basic datalink configuration, you can also use the `dladm` command to set datalink properties and customize them according to the needs of your network.

Note – Datalink properties can be customized by using the `dladm` command provided that the link's network driver has been converted to the GLDv3 framework, such as `e1000g`. To confirm whether your specific driver supports this feature, refer to the driver's man page.

Overview of Datalink Properties

Datalink properties that can be customized depend on the properties a specific NIC driver supports. Datalink properties that are configurable by using the `dladm` command fall into one of two categories:

- *Public properties* that can be applied to any driver of the given media type such as link speed, autonegotiation for Ethernet, or the MTU size that can be applied to all datalink drivers.
- *Private properties* that are particular to a certain subset of NIC drivers for a given media type. These properties can be specific to that subset because they are closely related either to the hardware that is associated with the driver or to the details of the driver implementation itself, such as debugging-related tunables.

Link properties typically have default settings. However, certain networking scenarios might require you to change specific property settings of a datalink. These property settings can be either public or private properties. For example, a NIC might be communicating with an old

switch that does not properly perform autonegotiation. Or, a switch might have been configured to support Jumbo frames. Or, driver specific properties that regulate packet transmission or packet receiving might need to be modified for the given driver. In Oracle Solaris, all of these settings can now be reset by a single administrative tool, `dladm`.

Setting Datalink Properties With the `dladm` Command

The following section provides procedures with examples to set certain datalink properties. The selected properties are public and common to all NIC drivers. A separate section describes datalink properties that are driver specific. This section is followed by procedures to configure selected private properties of the `e1000g` driver.

How to Enable Support for Jumbo Frames

Enabling support for Jumbo frames in a network setup is a common task for most network scenarios. Support for Jumbo frames requires increasing the size of a datalink's maximum transmission unit (MTU). The following procedure includes the use of customized names to identify datalinks. For an overview of customized names and their use in network configuration, see "The Network Stack in Oracle Solaris" on page 22.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 To identify the specific Ethernet device whose MTU size you need to reset, display the links in the system.

```
# dladm show-phys
```

Perform this step especially if your network configuration uses customized names for datalinks. With customized names, datalinks are no longer necessarily identified by their hardware-based names. For example, the Ethernet device is `bge0`. However, the datalink over the device is renamed `net0`. Therefore, you would need to configure the MTU size of `net0`. Refer to "IP Interface Configuration (Tasks)" on page 168 for examples of configuration tasks on datalinks that use customized names.

3 (Optional) Display the datalink's current MTU size and other properties.

- **To display a specific property of a datalink, use the following syntax:**

  `dladm show-linkprop -p` *property datalink*

  This command displays the settings of the property that you specify.

- **To display several selected properties of the datalink, use the following syntax:**

  # dladm show-link *datalink*

  This command displays datalink information, including MTU size.

4. **If an IP interface is configured over the datalink, remove the IP interface.**

   # ipadm delete-ip *interface*

5. **Change the link's MTU size to 9000, the setting for Jumbo frames.**

   # dladm set-linkprop -p mtu=9000 *datalink*

6. **Create the IP interface.**

   # ipadm create-ip *interface*

7. **Configure the IP interface.**

   # ipadm create-addr -T *addr-type* **[-a** *address***]** *addrobj*

   For more information about the `ipadm` command, see the `ipadm(1M)`.

8. **(Optional) Verify that the interface uses the new MTU size by using one of the command syntaxes in Step 3.**

   # dladm show-linkprop -p mtu *datalink*

9. **(Optional) Display the link's current Ethernet settings.**

   # dladm show-ether *datalink*

Example 8–5 Enabling Support for Jumbo Frames

The following example that enables support for Jumbo frames builds on the following scenario:

- The system has two `bge` NICs: `bge0` and `bge1`.
- The device `bge0` is used as a primary interface, while the device `bge1` is used for test purposes.
- You want to enable support for Jumbo frames on `bge1`, while you retain the default MTU size of the primary interface.
- The network configuration uses customized names for datalinks. The link name of `bge0` is `net0`. The link name of `bge1` is `net1`.

```
# dladm show-phys
LINK        MEDIA        STATE     SPEED     DUPLEX     DEVICE
net0        ether        up        100Mb     full       bge0
net1        ether        up        100Mb     full       bge1
net2        ether        up        100Mb     full       nge3

# dladm show-linkprop -p mtu net1
LINK     PROPERTY     VALUE     DEFAULT     POSSIBLE
```

```
net1     mtu          1500      1500        --

# ipadm delete-ip net1
# dladm set-linkprop -p mtu=9000 net1
# ipadm create-ip net1
# ipadm create-addr -T static -a 10.10.1.2/35 net1/v4

# dladm show-link web1
LINK     CLASS     MTU      STATE     BRIDGE     OVER
web1     phys      9000     up        --         --
```

Notice that the MTU setting is now 9000. In this example, the `dladm` command enabled you to change `net1`'s MTU size directly. The previous method that uses the `ndd` command would have required you to deleted `net0` as well, which would have unnecessarily disrupted the primary interface's operations.

▼ How to Change Link Speed Parameters

Most network setups consist of a combination of systems with varying speed capabilities. For example, the advertised speed between an older system and a newer system might need to be changed to a lower setting to allow communication. By default, all the speed and duplex capabilities of a NIC card are advertised. This procedure shows how to turn off the gigabit capabilities and advertise only the megabit capabilities.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 (Optional) Display the current status of the property you want to modify.

```
# dladm show-linkprop -p property  datalink
```

3 To advertise lower speed capabilities, turn off the higher speed capabilities to prevent them from being advertised.

```
# dladm set-linkprop -p property=value1  datalink
```

Example 8–6 Disabling Advertisement of a NIC's Gigabit Capabilities

This example shows how you can prevent the link `net1` from advertising gigabit capabilities.

```
# dladm show-linkprop -p adv_1000fdx_cap net1
LINK     PROPERTY             VALUE     DEFAULT     POSSIBLE
net1     adv_1000fdx_cap      1         --          1,0

# dladm show-linkprop -p adv_1000hdx_cap web1
LINK     PROPERTY             VALUE     DEFAULT     POSSIBLE
net1     adv_1000hdx_cap      1         --          1,0
```

The properties that advertise the link's gigabit capabilities are `adv_1000fdx_cap` and `adv_1000hdx_cap`. To disable these properties from being advertised, you would type the following commands:

```
# dladm set-linkprop -p adv_1000fdx_cap=0 net1
# dladm set-linkprop -p adv_1000hdx_cap=0 net1
```

Listing the Ethernet parameter settings would display the following output:

```
# dladm show-ether net1
LINK     PTYPE       STATE    AUTO  SPEED-DUPLEX             PAUSE
net1     current     up       yes   1G-f                     both
```

How to Obtain Status Information About Datalink Properties

You can obtain information about the datalink's properties by displaying either the Ethernet parameter settings or the link properties.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **To obtain information about the Ethernet parameter settings, use the following command:**

   # dladm show-ether [-x] *datalink*

   where the `-x` option includes additional parameter information about the link. Without the `-x` option, only the current parameter settings are displayed.

3. **To obtain information about all the properties of the link, use the following command:**

   # dladm show-linkprop *datalink*

Example 8–7 Displaying Ethernet Parameter Settings

This example displays an extended list of parameter information about a specified link.

```
# dladm show-ether -x net1
LINK     PTYPE       STATE    AUTO  SPEED-DUPLEX             PAUSE
net1     current     up       yes   1G-f                     both
--       capable     --       yes   1G-fh,100M-fh,10M-fh     both
--       adv         --       yes   100M-fh,10M-fh           both
--       peeradv     --       yes   100M-f,10M-f             both
```

With the `-x` option, the command also displays the built-in capabilities of the specified link, as well as the capabilities that are currently advertised between the host and the link partner. The following information is displayed:

- For the Ethernet device's current state, the link is up and functioning at 1 gigabits per second at full duplex. Its autonegotiation capability is enabled and has bidirectional flow control, in which both host and link partner can send and receive pause frames.
- Regardless of the current setting, the capabilities of the Ethernet device are listed. The negotiation type can be set to automatic, the device can support speeds of 1 gigabits per second, 100 megabits per second, and 10 megabits per second, at both full and half duplex. Likewise, pause frames can be received or sent in both directions between host and link partner.
- The capabilities of `net1` are advertised as follows: autonegotiation, speed-duplex, and flow control of pause frames.
- Similarly, `net1`'s link or peer partner advertises the following capabilities: autonegotiation, speed-duplex, and flow control of pause frames.

Example 8–8 Displaying Link Properties

This example shows how to list all the properties of a link. If you want to display only specific properties, you use the `-p` option with the specific properties that you want to monitor.

```
# dladm show-linkprop net1
LINK     PROPERTY             VALUE       DEFAULT     POSSIBLE
net1     speed                1000        --          --
net1     autopush             --          --          --
net1     zone                 --          --          --
net1     duplex               half        --          half,full
net1     state                unknown     up          up,down
net1     adv_autoneg_cap      1           1           1,0
net1     mtu                  1500        1500        --
net1     flowctrl             no          bi          no,tx,rx,bi
net1     adv_1000fdx_cap      1           1           1,0
net1     en_1000fdx_cap       1           1           1,0
net1     adv_1000hdx_cap      1           1           1,0
net1     en_1000hdx_cap       1           1           1,0
net1     adv_100fdx_cap       0           0           1,0
net1     en_100fdx_cap        0           0           1,0
net1     adv_100hdx_cap       0           0           1,0
net1     en_100hdx_cap        0           0           1,0
net1     adv_10fdx_cap        0           0           1,0
net1     en_10fdx_cap         0           0           1,0
net1     adv_10hdx_cap        0           0           1,0
net1     en_10hdx_cap         0           0           1,0
```

The settings for the speed and duplex capabilities of the link are manually configured on the enabled-speed properties which are labeled `en_*_cap`. For example, `en_1000fdx_cap` is the property for the gigabit full-duplex capability, and `en_100hdx_cap` is the property for the 100 megabits half-duplex capability. The settings of these enabled speed properties are advertised between the host and its link partner by corresponding advertised speed properties, which are labeled `adv_*_cap` such as `adv_1000fdx_cap` and `adv_100hdx_cap`.

Normally, the settings of a given enabled speed property and the corresponding advertised property are identical. However, if a NIC supports some advanced features such as Power Management, those features might set limits on the bits that are actually advertised between the host and its link partner. For example, with Power Management, the settings of the `adv_*_cap` properties might only be a subset of the settings of the `en_*_cap` properties. For more details about the enabled and advertised speed properties, see the `dladm(1M)` man page.

How to Set the `e1000g` Driver to Use Direct Memory Access Binding

This procedure and the next procedure show how to configure private properties. Both procedures apply to properties specific to the `e1000g` driver. However, the general steps can be used to configure private properties of other NIC drivers as well.

Bulk traffic, such as file transfers, normally involves negotiation of large packets across the network. In such cases, you can obtain better performance from the `e1000g` driver by configuring it to automatically use DMA binding, where a threshold is defined for packet fragment sizes. If a fragment size surpasses the threshold, then DMA binding is used for transmitting. If a fragment size is within the threshold, then `bcopy` mode is used, where the fragment data is copied to the preallocated transmit buffer.

To set the threshold, perform the following steps:

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Set the appropriate setting for the `_tx_bcopy_threshold` property.**

   ```
   # dladm set-linkprop -p _tx_bcopy_threshold=value  e1000g-datalink
   ```

   For this property, the valid settings for the threshold range from 60 through 2048.

   Note – As with configuring public properties, the interface must also be unplumbed before private property settings can be modified.

3. **(Optional) Verify the new threshold setting.**

   ```
   # dladm show-linkprop -p _tx_bcopy_threshold e1000g-datalink
   ```

How to Manually Set the Interrupt Rate

Parameters that regulate the rate at which interrupts are delivered by the `e1000g` driver also affect network and system performance. Typically network packets are delivered to the upper layer of the stack by generating an interrupt for every packet. In turn the interrupt rate, by default, is automatically adjusted by the GLD layer in the kernel. However, this mode might not be desirable in all network traffic conditions. For a discussion of this issue, refer to this

document (http://www.stanford.edu/class/cs240/readings/mogul.pdf) that was presented at the USENIX technical conference in 1996. Thus, in certain circumstances, setting the interrupt rate manually becomes necessary to obtain better performance.

To define the interrupt rate, you set the following parameters:

- `_intr_throttling_rate` determines the delay between interrupt assertions regardless of network traffic conditions.
- `_intr_adaptive` determines whether automatic tuning of the interrupt throttling rate is enabled. By default, this parameter is enabled.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **If necessary, identify the device whose driver property you want to modify.**

   ```
   # dladm show-phys
   ```

3. **Disable automatic tuning of the interrupt throttling rate.**

   ```
   # dladm set-linkprop -p _intr_adaptive=0 e1000g-datalink
   ```

   Note – When automatic tuning of the interrupt throttling rate is enabled, then any existing setting for the parameter `_intr_throttling_rate` is ignored.

4. **Remove any IP interface that is configured over the datalink.**

5. **Set the setting for the minimum inter interrupt level.**

   ```
   # dladm set-linkprop -p _intr_throttling_rate=value e1000g-datalink
   ```

   Note – The default setting of the `_intr_throttling_rate` parameter is 550 on SPARC based systems and 260 on x86 based systems. Setting the minimum inter-interrupt level to 0 disables the interrupt throttling logic.

6. **Configure the IP interface.**

7. **(Optional) Display the threshold's new settings.**

Example 8–9 Configuring for DMA Binding and Setting the Interrupt Throttling Rate

This example uses an x86 based system with an `e1000g` NIC. The driver is configured with a threshold setting toggle between using DMA binding or the `bcopy` mode for transmitting packets. The setting for the interrupt throttling rate is also modified. Further, the `e1000g`

datalink uses the default generic name that is assigned by the OS. Therefore, the configuration is performed on the datalink by referring to the customized name, `net0`.

```
# dladm show-phys
LINK       MEDIA        STATE     SPEED     DUPLEX     DEVICE
net0       ether        up        100Mb     full       e1000g0

# dladm show-linkprop -p _tx_bcopy_threshold net0
LINK        PROPERTY                VALUE     DEFAULT     POSSIBLE
net0        _tx_bcopy_threshold     512       512         --

# dladm show-linkprop -p _intr-throttling_rate
LINK        PROPERTY                  VALUE     DEFAULT     POSSIBLE
net0        _intr-throttling_rate     260       260         --

# ipadm delete-ip net0
# dladm set-linkprop -p _tx_bcopy_threshold=1024 net0
# dladm set-linkprop -p _intr_adaptive=0 net0
# dladm set-linkprop -p _intr-throttling_rate=1024 net0

# ipadm create-ip net0
# ipadm create-addr -T static -a 10.10.1.2/24 net0/v4addr
# dladm show-linkprop -p _tx_bocopy_threshold=1024 net0
LINK        PROPERTY                VALUE     DEFAULT     POSSIBLE
net0        _tx_bcopy_threshold     1024      512         --

# dladm show-linkprop -p _intr_adaptive net0
LINK        PROPERTY           VALUE     DEFAULT     POSSIBLE
net0        _intr-adaptive     0         1           --

# dladm show-linkprop -p _intr-throttling_rate
LINK        PROPERTY                  VALUE     DEFAULT     POSSIBLE
net0        _intr-throttling_rate     1024      260         --
```

Additional Configuration Tasks on Datalinks

This section describes other common configuration procedures that have become simplified by using the `dladm` command, such as performing dynamic reconfiguration (DR) and working with STREAMS modules.

▼ How to Replace a Network Interface Card With Dynamic Reconfiguration

This procedure applies only to systems that support dynamic reconfiguration (DR). It shows how DR is now facilitated by the separation of the network link configuration from the network hardware configuration. You no longer need to reconfigure your network links after you complete DR. Instead, you just transfer the link configurations of the removed NIC to be inherited by the replacement NIC.

Before You Begin Procedures to perform DR vary with the type of system. Make sure that you complete the following first:

- Ensure that your system supports DR.
- Ensure that your active network configuration profile is `DefaultFixed`. Refer to the section *Dynamic Reconfiguration and Network Configuration Profiles* in "How NWAM Works With Other Oracle Solaris Networking Technologies" on page 40 for information about using DR if your system's active NCP is not `DefaultFixed`.
- Consult the appropriate manual that describes DR on your system.

  To locate current documentation about DR on Sun servers from Oracle, search for `dynamic reconfiguration` on `http://www.oracle.com/technetwork/indexes/documentation/index.html`

Note – The following procedure refers only to aspects of DR that are specifically related to the use of flexible names for datalinks. The procedure does not contain the complete steps to perform DR. You must consult the appropriate DR documentation for your system.

---

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 (Optional) Display information about physical attributes of datalinks and their respective locations on the system.

```
# dladm show-phys -L
```

For more information about the type of information that is displayed by `dladm show-phys -L`, refer to the `dladm(1M)` man page.

3 Perform the DR procedures as detailed in your system's documentation to remove a NIC and then insert a replacement NIC.

Consult your system's DR documentation to perform this step.

After you have installed the replacement NIC, proceed to the next step.

4 If you inserted the replacement NIC into the same slot as the old NIC, then skip to Step 6. Otherwise, proceed to the next step.

With the new NIC using the same location which the old NIC previously occupied, the new NIC inherits the link name and configuration of the old NIC.

5 Perform one of the following steps depending on which circumstance applies.

- If the old NIC to be replaced remains in its slot in the system as an unused NIC, perform the following steps:

a. Assign a different name to the NIC to be replaced.

```
# dladm rename-link oldNIC new-name
```

oldNIC Refers to the NIC that is replaced but which you keep in the system.

new-name Refers to the new name you give to *removedNIC*. The name must not be shared by any other links in the system.

b. Assign the name of the old NIC to the replacement NIC.

```
# dladm rename-link replacementNIC oldNIC
```

replacementNIC Refers to the new NIC that you have just installed. This NIC automatically receives the default link name depending on the slot that it occupies in the system.

oldNIC Refers to the NIC that is replaced but which you keep in the system.

- If you removed the old NIC and you install the replacement NIC in a different slot but want the NIC to inherit the configurations of the old NIC, assign the name of the old NIC to the new NIC.

  ```
  # dladm rename-link replacementNIC oldNIC
  ```

6 Complete the DR process by enabling the new NIC's resources to become available for use by Oracle Solaris.

For example, you use the `cfgadm` command to configure the NIC. For more information see the `cfgadm`(1M) man page.

7 (Optional) Display link information.

For example, you can use either `dladm show-phys` or `dladm show-link` to show information about the datalinks.

Example 8–10 Performing Dynamic Reconfiguration by Installing a New Network Card

This example shows how a `bge` card with link name `net0` is replaced by a `e1000g` card. The link configurations of `net0` are transferred from `bge` to `e1000g` after `e1000g` is connected to the system.

```
# dladm show-phys -L
LINK     DEVICE     LOCATION
net0     bge0       MB
net1     ibp0       MB/RISER0/PCIE0/PORT1
net2     ibp1       MB/RISER0/PCIE0/PORT2
net3     eoib2      MB/RISER0/PCIE0/PORT1/cloud-nm2gw-2/1A-ETH-2
```

You perform the DR-specific steps such as using `cfgadm` to remove `bge` and install `e1000g` in its place. After the card is installed, the datalink of `e1000g0` automatically assumes the name `net0` and inherits the link configurations.

```
# dladm show-phys -L
LINK     DEVICE     LOCATION
net0     e1000g0    MB
net1     ibp0       MB/RISER0/PCIE0/PORT1
net2     ibp1       MB/RISER0/PCIE0/PORT2
net3     eoib2      MB/RISER0/PCIE0/PORT1/cloud-nm2gw-2/1A-ETH-2

# dladm show-link
LINK     CLASS     MTU    STATE    OVER
net0     phys      9600   up       ---
net1     phys      1500   down     ---
net2     phys      1500   down     --
net3     phys      1500   down     ---
```

Configuring STREAMS Modules on Datalinks

If necessary, you can set up to eight STREAMS modules to be pushed on top of a datalink. These modules are typically used by third-party networking software such as virtual private networks (VPNs) and firewalls. Documentation about such networking software is provided by the software vendor.

The list of STREAMS modules to push on a specific datalink is controlled by the `autopush` link property. In turn, the value of the `autopush` link property is set by using the `dladm set-linkprop` subcommand.

A separate `autopush` command can also be used to set the STREAMS `autopush` modules on a per-driver basis. However, the driver is always bound to the NIC. If the datalink's underlying NIC is removed, then the link's autopush property information becomes lost as well.

To configure the STREAMS modules to be pushed on top of a datalink, use the `dladm set-linkprop` command in preference over the `autopush` command. If both per-driver and per-link types of `autoputsh` configuration exist for a specific datalink, the per-link information that is set with `dladm set-linkprop` is used and the per-driver information is ignored.

▼ How to Set STREAMS Modules on Datalinks

The following procedure describes how to configure STREAMS modules with the `dladm set-linkprop` command.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Push the modules to the stream when the link is opened.

```
# dladm set-linkprop -p autopush=modulelist  link
```

modulelist Specifies the list of modules that you want to be automatically pushed on to the stream. A maximum of eight modules can be pushed over a link. These modules are pushed in the order that they are listed in *modulelist*. Separate the modules in the list by using dots as delimiters.

link Specifies the link on which the modules are pushed.

Example 8–11 Setting the `autopush` Link Property

In this example, you push the `vpnmod` and `bufmod` modules on top of the link `net0`. The link's underlying device is `bge0`.

```
# dladm set-linkprop -p autopush=vpnmod.bufmod net0
```

If you later replace the `bge` card with `e1000g`, you can switch to the new datalink without needing to reconfigure the `autopush` settings. The `e1000g` card automatically inherits `bge`'s link name and configuration.

▼ How to Obtain `autopush` Link Property Settings

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Display `autopush` link property settings.**

   ```
   # dladm show-linkprop -p autopush [link]
   ```

   If you do not specify *link*, then the information for all configured links is displayed.

▼ How to Remove `autopush` Link Property Settings

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Remove the `autopush` link property settings of a specific datalink.**

   ```
   # dladm reset-linkprop [-t] -p autopush link
   ```

   Use the `-t` option to remove the property settings temporarily. The settings are restored when you reboot the system.

◆ ◆ ◆ CHAPTER 9

Configuring an IP Interface

This chapter provides the procedures that are used to configure an IP interface over a datalink.

About IP Interface Configuration

After you install Oracle Solaris, you might perform the following tasks:

- Configure an IP interface over a datalink for a basic interface configuration. This chapter describes the procedures.
- Configure Wireless Interfaces. The procedures are described in Chapter 10, "Configuring Wireless Interface Communications on Oracle Solaris"
- Configure IP interfaces as members of an IPMP group. The procedures are described in Chapter 15, "Administering IPMP."

The `ipadm` Command

Advances in Oracle Solaris have surpassed the capabilities of traditional tools to efficiently administer various aspects of network configuration. The `ifconfig` command, for example, has been the customary tool to configure network interfaces. However, this command does not implement persistent configuration settings. Over time, `ifconfig` has undergone enhancements for added capabilities in network administration. However, as a consequence, the command has become complex and confusing to use.

Another issue with interface configuration and administration is the absence of simple tools to administer TCP/IP Internet protocol properties or tunables. The `ndd` command has been the prescribed customization tool for this purpose. However, like the `ifconfig` command, `ndd` does not implement persistent configuration settings. Previously, persistent settings could be simulated for a network scenario by editing the boot scripts. With the introduction of the SMF feature of Oracle Solaris, using such workarounds can become risky because of the complexities of managing SMF dependencies, particularly in the light of upgrades to the Oracle Solaris installation.

The `ipadm` command is introduced to eventually replace the `ifconfig` command for interface configuration. The command also replaces the `ndd` command to configure protocol properties.

As a tool for configuring interfaces, the `ipadm` command offers the following advantages:

- It manages IP interfaces and IP addresses more efficiently by being the tool uniquely for IP interface administration, unlike the `ifconfig` command that is used for purposes other than interface configuration.
- It provides an option to implement persistent interface and address configuration settings.

For a list of `ifconfig` options and their equivalent `ipadm` subcommands, see "`ifconfig` Command Options and `ipadm` Command Options" on page 190.

As a tool to set protocol properties, the `ipadm` command provides the following benefits:

- It can set temporary or persistent protocol properties for IP, Address Resolution Protocol (ARP), Stream Control Transmission Protocol (SCTP), and Internet Control Messaging Protocol (ICMP), as well as upper layer protocols such as TCP and User Datagram Protocol (UDP).
- It provides information about each TCP/IP parameter, such as a property's current and default setting, as well as the range of possible settings. Thus, debugging information is more easily obtained.
- The `ipadm` command also follows a consistent command syntax and therefore is easier to use.

For a list of `ndd` options and their equivalent `ipadm` subcommands, see "`ndd` Command Options and `ipadm` Command Options" on page 192.

For more details about the `ipadm` command, refer to the `ipadm(1M)` man page.

IP Interface Configuration (Tasks)

This section describes basic configuration procedures on an IP interface. The following table describes configuration tasks and maps these tasks to their corresponding procedures.

TABLE 9–1 Configuring IP Interfaces (Task Map)

Task	Description	For Instructions
Set a system to support unique MAC addresses.	Configures a SPARC based system to allow unique MAC addresses for interfaces.	"SPARC: How to Ensure That the MAC Address of an Interface Is Unique" on page 169
Perform basic IP interface configuration by using the `ipadm` command.	Creates an IP interface and assigns valid IP addresses, either static or DHCP.	"How to Configure an IP Interface" on page 171

TABLE 9–1 Configuring IP Interfaces (Task Map) *(Continued)*

Task	Description	For Instructions
Customize an IP address by using the `ipadm` command.	Sets the network ID of a given IP address.	"How to Set the Property of an IP Address" on page 175
Obtain interface information by using the `ipadm` command.	Lists different properties of interfaces, addresses, and protocols and their corresponding settings.	"How to Obtain Information About Network Interfaces" on page 185

▼ SPARC: How to Ensure That the MAC Address of an Interface Is Unique

Some applications require every interface on a host to have a unique MAC addresses. However, every SPARC based system has a system-wide MAC address, which by default is used by all interfaces. Here are two situations where you might want to configure the factory-installed MAC addresses for the interfaces on a SPARC system.

- For link aggregations, you should use the factory-set MAC addresses of the interfaces in the aggregation configuration.
- For IPMP groups, each interface in the group must have a unique MAC address. These interfaces must use their factory-installed MAC addresses.

The EEPROM parameter `local-mac-address?` determines whether all interfaces on a SPARC system use the system-wide MAC address or their unique MAC address. The next procedure shows how to use the `eeprom` command to check the current value of `local-mac-address?` and change it, if necessary.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Determine whether all interfaces on the system currently use the system-wide MAC address.

```
# eeprom local-mac-address?
local-mac-address?=false
```

In the example, the response to the `eeprom` command, `local-mac-address?=false`, indicates that all interfaces do use the system-wide MAC address. The value of `local-mac-address?=false` must be changed to `local-mac-address?=true` before the interfaces can become members of an IPMP group. You should also change `local-mac-address?=false` to `local-mac-address?=true` for aggregations.

3 If necessary, change the value of `local-mac-address?` as follows:

```
# eeprom local-mac-address?=true
```

When you reboot the system, the interfaces with factory-installed MAC addresses now use these factory settings, rather than the system-wide MAC address. Interfaces without factory-set MAC addresses continue to use the system-wide MAC address.

4 **Check the MAC addresses of all the interfaces on the system.**

Look for cases where multiple interfaces have the same MAC address. In this example, all interfaces use the system-wide MAC address `8:0:20:0:0:1`.

```
# dladm show-linkprop -p mac-address
LINK    PROPERTY        PERM VALUE              DEFAULT            POSSIBLE
net0    mac-address     rw   8:0:20:0:0:1       8:0:20:0:0:1       --
net1    mac-address     rw   8:0:20:0:0:1       8:0:20:0:0:1       --
net3    mac-address     rw   0:14:4f:45:c:2d    0:14:4f:45:c:2d    --
```

Note – Continue to the next step only if more than one network interface still has the same MAC address. Otherwise, go on to the final step.

5 **If necessary, manually configure the remaining interfaces so that all interfaces have unique MAC addresses.**

```
# dladm set-linkprop -p mac-address=mac-address  interface
```

In the example in the previous step, you would need to configure `net0` and `net1` with locally administered MAC addresses. For example, to reconfigure `net0` with the locally administered MAC address `06:05:04:03:02`, you would type the following command:

```
# dladm set-linkprop -p mac-address=06:05:04:03:02 net0
```

Refer to the `dladm(1M)` man page for details about this command.

6 **Reboot the system.**

Configuring IP Interfaces

The procedures that follow show how you use the `ipadm` command for different IP configuration needs. Although the `ifconfig` command still functions to configure interfaces, the `ipadm` command should be the preferred tool. For an overview of the `ipadm` command and its benefits, see "The `ipadm` Command" on page 167.

Note – Typically, IP interface configuration and datalink configuration occur together. Thus, where applicable, procedures that follow include datalink configuration steps with the use of the `dladm` command. For more information about using the `dladm` command to configure and administer datalinks, see Chapter 8, "Datalink Configuration and Administration."

How to Configure an IP Interface

The following procedure provides an example of performing a basic configuration of an IP interface.

Before You Begin Determine if you want to rename datalinks on the system. Typically, you use the generic names that have been assigned by default to the datalinks. To change link names, see "How to Rename a Datalink" on page 149.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 (Optional) Display information about the physical attributes of datalinks currently on the system.

```
# dladm show-phys
```

This command shows the physical network cards that are installed on your system and some of their properties. For more information about this command, see How to Display Information About Physical Attributes of Datalinks.

3 Display information about datalinks currently on the system.

```
# dladm show-link
```

This command shows the datalinks and certain properties that have been set for them, including the physical cards over which the links have been created.

4 Create the IP interface.

```
# ipadm create-interface-class  interface
```

interface-class Refers to one of three classes of interfaces that you can create:

- IP interface. This interface class is the most common that you create when you perform network configuration. To create this interface class, use the `create-ip` subcommand.
- STREAMS virtual network interface driver (VNI interface). To create this interface class, use the `create-vni` subcommand. For more information about VNI devices or interfaces, see the `vni(7d)` man page.
- IPMP interface. This interface is used when you configure IPMP groups. To create this interface class, use the `create-ipmp` subcommand. For more information about IPMP groups, see Chapter 14, "Introducing IPMP," and Chapter 15, "Administering IPMP."

interface Refers to the name of the interface. The name is identical to the name of the link over which the interface is being created.

Note – You must create the IP interface before you can assign the IP address to it.

5 Configure the IP interface with a valid IP address.

The following syntax assigns a static address to an interface. Refer to the `ipadm(1M)` man page for other options for assigning IP addresses.

```
# ipadm create-addr -T address-type -a address/prefixlen addrobj
```

`-T` *address-type* Specifies the type of IP address that is assigned to the interface, which is one of the following: `static`, `dhcp`, or `addrconf`. `Addrconf` refers to automatically generated IPv6 addresses.

`-a` Specifies the IP address to configure on the interface. You can specify either just a local address, or both a local address and a remote address in the case of tunnel configuration. Typically, you assign only a local address. In this case, you specify the address directly with the `-a` option, such as: `-a` *address*. The address is automatically considered a local address.

If you are configuring tunnels, you might be required to provide both the local address of the system and the remote address of the destination system. In this case, you must specify `local` and `remote` to distinguish the two addresses, as follows: `-a local=`*local-addr*`,remote=`*remote-addr*. For more information about configuring tunnels, see Chapter 6, "Configuring IP Tunnels," in *Oracle Solaris Administration: IP Services*.

If you are using a numeric IP address, use the format *address*/*prefixlen* for addresses in CIDR notation, for example, `1.2.3.4/24`. See the explanation for the *prefixlen* option.

Optionally, you can specify a host name for *address* instead of a numeric IP address. Using a host name is valid if a corresponding numeric IP address is defined for that host name in the `/etc/hosts` file. If no numeric IP address is defined in the file, then the numeric value is uniquely obtained by using the resolver order that is specified for `host` in the `name-service/switch` service. If multiple entries exist for a given host name, then an error is generated.

Note – During the boot process, the creation of IP addresses precedes naming services being brought online. Therefore you must ensure that any host name that is used in the network configuration must be defined in the `/etc/hosts` file.

/prefixlen — Specifies the length of the network ID that is part of the IPv4 address when you use CIDR notation. In the address `12.34.56.78/24`, `24` is the *prefixlen*. If you do not include *prefixlen*, then the netmask is computed according to the sequence listed for `netmask` in the `name-service/switch` service or by using classful address semantics.

addrobj — Specifies an identifier for the unique IP address or set of addresses that is used in the system. The addresses can be either IPv4 or IPv6 types. The identifier uses the format *interface*/*user_specified_string*.

The *interface* refers to the IP interface to which the address is assigned. The *interface* variable must reflect the name of the datalink on which the IP interface is configured.

user-specified-string refers to a string of alphanumeric characters that begins with an alphabet letter and has a maximum length of 32 characters. Subsequently, you can refer to the `addrobj` instead of the numeric IP address when you use any `ipadm` subcommand that manages addresses in the system, such as `ipadm show-addr`, or `ipadm delete-addr`.

6 (Optional) Display information about the newly configured IP interface.

You can use the following commands, depending on the information that you want to check:

- Display the general status of the interface.

  ```
  # ipadm show-if [interface]
  ```

  If you do not specify the interface, then information for all interfaces in the system is displayed.

- Display the interface's address information.

  ```
  # ipadm show-addr [addrobj]
  ```

  If you do not specify the *addrobj*, then information for all address objects in the system is displayed.

For more information about the output of the `ipadm show-*` subcommand, see "Monitoring IP Interfaces and Addresses" on page 184.

7 (Optional) Add entries for the IP addresses in the `/etc/hosts` file.

The entries in this file consist of IP addresses and the corresponding host names.

Note – This step applies only if you are configuring static IP addresses that use hostnames. If you are configuring DHCP addresses, you do not need to update the `/etc/hosts` file.

Example 9–1 Configuring a Network Interface With a Static Address

```
# dladm show-phys
LINK     MEDIA        STATE     SPEED     DUPLEX     DEVICE
net3     Ethernet     up        100Mb     full       bge3

# dladm show-link
LINK     CLASS     MTU     STATE     BRIDGE   OVER
net3     phys      1500    up        --       --

# ipadm create-ip net3
# ipadm create-addr -T static -a 192.168.84.3/24 net3/v4static

# ipadm show-if
IFNAME   CLASS        STATE     ACTIVE     OVER
lo0      loopback     ok        yes        --
net3     ip           ok        yes        --

# ipadm show-addr
ADDROBJ     TYPE       STATE     ADDR
lo0/?       static     ok        127.0.0.1/8
net3/v4     static     ok        192.168.84.3/24

# vi /etc/hosts
# Internet host table
# 127.0.0.1       localhost
10.0.0.14       myhost
192.168.84.3    campus01
```

Note that if `campus01` is already defined in the `/etc/hosts` file, you can use that host name when assigning the following address:

```
# ipadm create-addr -T static -a campus01 net3/v4static
```

Example 9–2 Automatically Configuring a Network Interface With an IP Address

This example uses the same network device as the previous example but configures the IP interface to receive its address from a DHCP server.

```
# dladm show-phys
LINK     MEDIA        STATE     SPEED     DUPLEX     DEVICE
net3     Ethernet     up        100Mb     full       bge3

# dladm show-link
LINK     CLASS     MTU     STATE     BRIDGE   OVER
net3     phys      1500    up        --       --

# ipadm create-ip net3

# ipadm create-addr -T dhcp net3/dhcp

# ipadm show-if
IFNAME   CLASS        STATE     ACTIVE     OVER
lo0      loopback     ok        yes        --
```

```
net3     ip           ok        yes        --

# ipadm show-addr net3/dhcp
ADDROBJ     TYPE       STATE     ADDR
net3/dhcp   dhcp       ok        10.8.48.242/24

# ipadm show-addr
ADDROBJ     TYPE       STATE     ADDR
lo0/?       static     ok        127.0.0.1/8
net3/dhcp   dhcp       ok        10.8.48.242/24
```

Setting IP Address Properties

The `ipadm` command enables you to set address–specific properties after these addresses are assigned to interfaces. By setting these properties, you can determine the following:

- The `prefixlen` of an address.
- Whether an IP address can be used as a source address for outbound packets.
- Whether the address belongs to a global or non–global zone.
- Whether the address is a private address.

To list the properties of an IP address, use the following syntax:

```
# ipadm show-addrprop [-p property] [addrobj]
```

The information that is displayed depends on the options that you use.

- If you do not specify a property nor an address object, then all properties of all existing addresses are displayed.
- If you specify only the property, then that property for all the addresses is displayed.
- If you specify only the address object, then all the properties of that address object are displayed.

Note – You can only set address properties one at a time.

How to Set the Property of an IP Address

This procedure shows the general steps to configure a property for an IP address.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **List the IP addresses currently in use on the system.**

   ```
   # ipadm show-addr
   ```

3 **(Optional) Determine the current setting of a specific property of an IP address that you want to change.**

```
# ipadm show-addrprop -p property addrobj
```

If you do not know the property, you can issue a general `ipadm show-addrprop` command. When you display IP addresses with this command, the addresses are displayed with the current settings of all their properties.

4 **Set the selected property to the desired value.**

```
# ipadm set-addrprop -p property=value addrobj
```

5 **View the new setting for the property.**

```
# ipadm show-addrprop -p property addrobj
```

Example 9–3 Setting the *prefixlen* Property of an Address

The `prefixlen` property refers to the netmask of an IP address. The following example changes the length of the `prefixlen` property of `net3`'s IP address. In this example, the `-t` option is used to create only a temporary change in the property. If the system is rebooted, the property's value reverts to the default setting.

```
# ipadm show-addr
ADDROBJ     TYPE       STATE     ADDR
lo0/?       static     ok        127.0.0.1/8
net3/v4     static     ok        192.168.84.3/24

# ipadm show-addrprop -p prefixlen net3/v4
ADDROBJ  PROPERTY   PERM  CURRENT  PERSISTENT  DEFAULT   POSSIBLE
net3/v4  prefixlen  rw    24       24          24        1-30,32

# ipadm set-addrprop -t -p prefixlen=8 net3/v4
# ipadm show-addrprop -p prefixlen net3/v4
ADDROBJ  PROPERTY   PERM  CURRENT  PERSISTENT  DEFAULT   POSSIBLE
net3/v4  prefixlen  rw    8        24          24        1-30,32
```

Setting IP Interface Properties

IP interfaces, like datalinks, have properties that you can customize for your specific network setting. For each interface, two sets of properties exist that apply to IPv4 and IPv6 protocols, respectively. Some properties, such as MTU, are common to both datalinks and the IP interface. Thus, you can have one MTU setting for a datalink and a different MTU setting for the interface configured over that link. Further, you can have different MTU settings that apply to IPv4 and IPv6 packets, respectively, that traverse that IP interface.

IP forwarding is an IP interface property that is typically configured in networking scenarios. The following procedure shows the steps.

Enabling Packet Forwarding

In a network, a host can receive data packets that are destined for another host system. By enabling packet forwarding in the receiving local system, that system can forward the data packet to the destination host. By default, IP forwarding is disabled. The following two procedures describe how to enable this functionality. In previous Oracle Solaris releases, the `routeadm` command was used to enable packet forwarding. The `ipadm` syntax in this procedure replaces the `routeadm` command.

Consider the following to determine whether to use the interface–based or protocol–based procedure.

- If you want to be selective in how packets are forwarded, then you enable packet forwarding on the interface. For example, you might have a system that has multiple NICs. Some NICs are connected to the external network, while other NICs are connected to the private network. You would therefore enable packet forwarding only on some of the interfaces, rather than on all interfaces. See "How to Enable IP Packet Forwarding by Setting an Interface Property" on page 177.
- If you want to implement packet forwarding globally within the system, then you enable the `forwarding` property of the protocol. For this second method, see "How to Enable Packet Forwarding by Setting the Protocol Property" on page 179.

Note – The two methods of forwarding packets are not mutually exclusive. For example, you can enable packet forwarding globally, and then customize the `forwarding` property for each interface. Thus, packet forwarding can still be selective for that particular system.

▼ How to Enable IP Packet Forwarding by Setting an Interface Property

This procedure shows how to enable packet forwarding selectively by configuring the IP forwarding property on specific interfaces.

Note – Packet forwarding involves the IP protocol. Thus, distinguishing between IP *protocol versions* is also included in the steps.

---

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Display the current setting of an interface's IP forwarding property.

```
# ipadm show-ifprop -p forwarding [-m protocol-version] interface
```

where *protocol-version* can either be `ipv4` or `ipv6`. If you do not specify the version, then the settings for both IPv4 and IPv6 protocols are displayed.

Note – To display all the valid protocol properties of a given interface, do not specify a property, as follows:

```
# ipadm show-ifprop interface
```

This syntax is also shown in Example 9–4.

3 For every interface on which you want to enable packet forwarding, type the following command:

```
# ipadm set-ifprop forwarding=on -m protocol-version interface
```

4 (Optional) Display the settings of an interface's `forwarding` property.

```
# ipadm show-ifprop -p forwarding interface
```

5 To restore an interface's `forwarding`property to its default setting, type the following command:

```
# ipadm reset-ifprop -p forwarding -m protocol-version interface
```

Example 9–4 Enabling an Interface to Forward Only IPv4 Packets

The following example shows how to implement selective packet forwarding, where forwarding of IPv4 packets is enabled only in the `net0` interface. In the other remaining interfaces of the system, packet forwarding is disabled, which is the default setting.

```
# ipadm show-ifprop -p forwarding net0
IFNAME   PROPERTY     PROTO   PERM   CURRENT   PERSISTENT   DEFAULT   POSSIBLE
net0     forwarding   ipv4    rw     off       off          off       on,off
net0     forwarding   ipv6    rw     off       --           off       on,off
```

The `ipadm show-ifprop` command syntax that uses the `-p property` option provides information only about a specific property.

```
# ipadm set-ifprop -p forwarding=on -m ipv4 net0
# ipadm show-ifprop net0
IFNAME   PROPERTY         PROTO  PERM  CURRENT  PERSISTENT  DEFAULT   POSSIBLE
...
net0     forwarding       ipv4   rw    on       on          off       on,off
...
```

The `ipadm show-ifprop` command syntax without the `-p property` option displays all the properties of an interface and their corresponding settings.

```
# ipadm reset-ifprop -p forwarding -m ipv4 net0
# ipadm show-ifprop -p forwarding -m ipv4 net0
IFNAME   PROPERTY     PROTO   PERM   CURRENT   PERSISTENT   DEFAULT   POSSIBLE
net0     forwarding   ipv4    rw     off       off          off       on,off
```

The `ipadm reset-ifprop` command syntax resets the specified property to the default setting.

How to Enable Packet Forwarding by Setting the Protocol Property

This procedure shows how to enable packet forwarding globally in the system.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Display the current setting of the IP forwarding property.**

   `# ipadm show-prop -p forwarding` *protocol-version*

   where *protocol-version* can either be `ipv4` or `ipv6`.

   Note – To display all the valid tunable properties for a given protocol and their current settings, type the following command:

   `# ipadm show-prop` *protocol*

   where *protocol* can be `ip`, `ipv4`, `ipv6`, `udp`, `tcp`, `icmp`, and `sctp`.

   This syntax is shown in Example 9–5.

   ---

3. **For every protocol version on which you want to enable forwarding, type the following command:**

   `# ipadm set-prop forwarding=on` *protocol-version*

4. **(Optional) Display the settings of the IP forwarding property by performing one of the following:**

   - To display all the properties and current settings of a protocol, type the following:

     `# ipadm show-prop` *protocol*
   - To display a specific property of a protocol, type the following:

     `# ipadm show-prop -p` *property protocol*
   - To display a specific property of a specific protocol version, type the following:

     `# ipadm show-prop -p` *property protocol-version*

5. **To reset a specific property of a protocol version to its default setting, type the following:**

   `# ipadm reset-prop -p` *property protocol-version*

Example 9–5 Enabling Forwarding for IPv4 and IPv6 Packets

The following example parallels the previous example about forwarding packets on interfaces. The two uses of `ipadm show-prop` display the settings of a specified property or all the properties of a protocol and their corresponding settings.

```
# ipadm show-prop -p forwarding ip
PROTO   PROPERTY     PERM   CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv4    forwarding   rw     off       --           off       on,off
ipv6    forwarding   rw     off       --           off       on,off
#
# ipadm set-prop -p forwarding=on ipv4
# ipadm set-prop -p forwarding=on ipv6
#
# ipadm show-prop ip
PROTO   PROPERTY     PERM   CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv4    forwarding   rw     on        on           off       on,off
ipv4    ttl          rw     255       --           255       1-255
ipv6    forwarding   rw     on        on           off       on,off
ipv6    hoplimit     rw     255       --           255       1-255#
```

Administering Protocol Properties

Aside from interfaces, the `ipadm` command can be used to configure protocol properties, also known as tunables. The `ipadm` replaces the `ndd` command which was commonly used in previous releases to set tunables. This section provides procedures and examples to customize selected TCP/IP protocol properties.

Setting TCP/IP Properties

TCP/IP properties can either be interface based or global. Properties can be applied to a specific interface, or globally to all interfaces in the zone. Global properties can have different settings in different non-global zones. For a list of supported protocol properties, refer to the `ipadm(1M)` man page.

Typically, the default settings of the TCP/IP internet protocol suffice for the network to function. However, if the default settings are insufficient for your network topology, the procedures in the following table illustrate how you can customize these TCP/IP properties.

The table describes tasks to configure certain of the protocol's properties and provides links to the respective procedures.

TABLE 9–2 Setting Selected TCP/IP Properties

Task	Description	For Instructions
Mark a port privileged.	Reserves an interface's port to restrict access to it except for the `root` user.	"How to Restrict a Port's Access to `root` User Only" on page 181
Customize the behavior of IP packets being received or transmitted on multihomed hosts.	Customizes symmetric routing in multihomed hosts.	"How to Implement Symmetric Routing on Multihomed Hosts" on page 183
Display information about a protocol's property.	Displays a protocol's property and its current setting.	"Monitoring IP Interfaces and Addresses" on page 184

Note – For procedures that use the `ipadm` tool to configure network interfaces and IP addresses, refer to "Configuring IP Interfaces" on page 170.

How to Restrict a Port's Access to `root` User Only

On transport protocols such as TCP, UDP, and SCTP, ports `1`–`1023` are default privileged ports where only processes that run with root permissions can bind to these ports. By using the `ipadm` command, you can reserve a port beyond this given default range such that it becomes a privileged port. Thus, only root processes can bind to that port. For this procedure, you use the following transport protocol properties:

- `smallest_nonpriv_port`
- `extra_priv_ports`

1 Determine if the designated port is in the range of regular ports and therefore can be used.

```
# ipadm show-prop -p smallest_nonpriv_port protocol
```

where *protocol* is the protocol type for which you want to configure a privileged port, such as IP, UDP, ICMP, and others.

In the command output, the `POSSIBLE` field shows the range of port numbers to which regular users can bind. If the designated port is within this range, then you can set it as a privileged port.

2 Verify that the port that you want to reserve is available and not already marked as a privileged port.

```
# ipadm show-prop -p extra_priv_ports protocol
```

In the command output, the `CURRENT` field indicates which ports are currently marked as privileged. If the designated port is not included under this field, then you can set it as a privileged port.

3 **Add the designated port as a privileged port.**

```
# ipadm set-prop -p extra_priv_ports=port-number  protocol
```

4 **For every additional port that you want to add or remove as privileged ports, repeat one of the following:**

- To add a ports as a privileged port, type the following syntax.

  ```
  # ipadm set-prop -p extra_priv_ports+=portnumber  protocol
  ```

  Note – By the plus sign (+) qualifier, you can assign multiple ports to become privileged ports. The plus sign qualifier enables you to build a list of these ports. Use this syntax with the qualifier to add ports to the list individually. If you do not use the qualifier, then the port that you assign replaces all the other ports that were previously listed as privileged.

- To remove a port as a privileged port, type the following syntax.

  ```
  # ipadm set-prop -p extra_priv_ports-=portnumber  protocol
  ```

  Note – By using the minus sign (-) qualifier, you can remove the port from the existing ports currently listed as privileged. Use the same syntax to remove all extra privileged ports, including the default ports.

5 **Verify the new status of the designated port.**

```
# ipadm show-prop -p extra_priv_ports protocol
```

In the command output, make sure that the designated ports are now included in the `CURRENT` field.

Example 9–6 Setting a Privileged Port

In this example, you are setting ports `3001` and `3050` as privileged ports. You also remove port `4045`, which is currently listed as a privileged port.

In the output for the `smallest_nonpriv_port` property, the `POSSIBLE` field indicates that port `1024` is the lowest non–privileged port and that the designated ports `3001` and `3050` are within the range of possible non–privileged ports to use. In the output for the `extra_priv_ports` property, ports `2049` and `4045` under the `CURRENT` field are marked as privileged. Thus, you can proceed with setting port `3001` as a privileged port.

```
# ipadm show-prop -p smallest_nonpriv_port tcp
PROTO PROPERTY                PERM   CURRENT   PERSISTENT   DEFAULT   POSSIBLE
tcp   smallest_nonpriv_port   rw     1024      --           1024      1024-32768

# ipadm show-prop -p extra_priv_ports tcp
PROTO   PROPERTY           PERM   CURRENT     PERSISTENT   DEFAULT     POSSIBLE
```

```
tcp     extra_priv_ports   rw     2049,4045   --           2049,4045   1-65535

# ipadm set-prop -p extra_priv_ports+=3001 tcp
# ipadm set-prop -p extra_priv_ports+=3050 tcp
# ipadm show-prop -p extra_priv_ports tcp
PROTO   PROPERTY           PERM   CURRENT     PERSISTENT   DEFAULT     POSSIBLE
tcp     extra_priv_ports   rw     2049,4045   3001,3050    2049,4045   1-65535
                                  3001,3050

# ipadm set-prop -p extra_priv_ports-=4045 tcp
# ipadm show-prop -p extra_priv_ports tcp
PROTO   PROPERTY           PERM   CURRENT     PERSISTENT   DEFAULT     POSSIBLE
tcp     extra_priv_ports   rw     2049,3001   3001,3050    2049,4045   1-65535
                                  3050
```

How to Implement Symmetric Routing on Multihomed Hosts

By default, a system with multiple interfaces, also called a *multihomed host*, routes its network traffic based on the longest matching route to the traffic's destination in the routing table. When multiple routes of equal length to the destination exist, Oracle Solaris applies Equal Cost Multipathing (ECMP) algorithms to spread the traffic across those routes.

Spreading the traffic in this manner is not ideal in certain cases. An IP packet might be sent through an interface on the multihomed host that is not on the same subnet as the IP source address in the packet. Further, if the outgoing packet is a response to a certain incoming request, such as an ICMP echo request, the request and the response might not traverse the same interface. Such a traffic routing configuration is called asymmetric routing. If your Internet service provider is implementing ingress filtering as described in RFC 3704 (`http://rfc-editor.org/rfc/bcp/bcp84.txt`), an asymmetric routing configuration might cause an outgoing packet to be dropped by the provider.

RFC 3704 intends to limit denial of service attacks across the Internet. To comply with this intent, your network must be configured for symmetric routing. In Oracle Solaris, the IP `hostmodel` property enables you to meet this requirement. This property controls the behavior of IP packets that are received or transmitted through a multihomed host.

The following procedure shows how to use the `ipadm` command to set the `hostmodel` property for a specific routing configuration:

1. **On the multihomed host, become an Administrator.**

2. **Configure the routing of network packets in the system.**

   ```
   # ipadm set-prop -p hostmodel=value protocol
   ```

   The property can be configured to one of the following three settings:

`strong`	Corresponds to the strong end system (ES) model as defined in RFC 1122. This setting implements symmetric routing.
`weak`	Corresponds to the weak ES model as defined in RFC 1122. With this setting, a multihomed host uses asymmetric routing.

`src-priority` Configures packet routing by using preferred routes. If multiple destination routes exist in the routing table, then the preferred routes are those that use interfaces on which the IP source address of an outgoing packet is configured. If no such routes exist, then the outgoing packet will use the longest matching route to the packet's IP destination.

3 (Optional) Check the setting of the `hostmodel` property.

```
# ipadm show-prop protocol
```

Example 9–7 Setting Symmetric Routing on a Multihomed Host

In this example, you want to enforce symmetric routing of all IP traffic in the multihomed host.

```
# ipadm set-prop -p hostmodel=strong ip
# ipadm show-prop -p hostmodel ip
PROTO  PROPERTY   PERM  CURRENT  PERSISTENT  DEFAULT   POSSIBLE
ipv6   hostmodel  rw    strong   --          weak      strong,
                                                      src-priority,
                                                      weak
ipv4   hostmodel  rw    strong   --          weak      strong,
                                                      src-priority,
                                                      weak
```

Monitoring IP Interfaces and Addresses

The `ipadm` command is also the preferred tool for monitor and obtain information about IP interfaces and their properties or parameters. The `ipadm` subcommands to obtain interface information use the following basic syntax:

`ipadm show-* [`*other-arguments***`] [`***interface***`]`**

- To obtain interface information, use `ipadm show-if`.
- To obtain address information, use `ipadm show-addr`.
- To obtain information about a specific interface property, use `ipadm show-ifprop`.
- To obtain information about a specific address property, use `ipadm show-addrprop`

This section provides several examples of using the `ipadm` command to obtain information about the network interfaces. For other types of monitoring tasks that you perform on the network, refer to Chapter 5, "Administering a TCP/IP Network," in *Oracle Solaris Administration: IP Services*.

Note – For an explanation of all the fields in the `ipadm show-*` commands, refer to the `ipadm`(1M) man page.

How to Obtain Information About Network Interfaces

This procedure describes how to display information about an interface's general status, address information, and IP properties.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 To obtain status information about an interface, type the following command:

```
# ipadm show-if [interface]
```

If you do not specify an interface, then the information covers all the interfaces on the system.

The fields in the command output refer to the following:

`IFNAME` Refers to the interface whose information is being displayed.

`CLASS` Refers to the class of interface, which can be one of four:

- `ip` refers to an IP interface
- `ipmp` refers to an IPMP interface
- `vni` refers to a virtual interface
- `loopback` refers to a loopback interface, which is automatically created. Except for the loopback interface, you can manually create the remaining 3 interface classes.

`STATE` Refers to the status of the interface, which can either be `ok`, `offline`, `failed`, `down`, or `disabled`.

The status `failed` applies to IPMP groups and can refer to a datalink or an IP interface that is down and cannot host traffic. If the IP interface belongs to an IPMP group, then the IPMP interface can continue to receive and send traffic by using other active IP interfaces in the group.

The status `down` refers to an IP interface that is switched offline by the administrator.

The status `disable` refers to the IP interface that is unplumbed by using the `ipadm disable-if` command.

`ACTIVE` Indicates whether the interface is being used to host traffic, and is set either to `yes`or `no`.

`OVER` Applies only to the IPMP class of interfaces and refers to the underlying interfaces that constitute the IPMP interface or group.

3 To obtain address information for the interface, type the following command:

```
# ipadm show-addr [addrobj]
```

If you do not specify an address identifier, then address information is provided for all the address identifiers on the system.

The fields in the command output refer to the following:

ADDROBJ	Specifies the address object whose address is being listed.
TYPE	Indicates whether the IP address is `static`, `dhcp`, or `addrconf`. The `addrconf` setting indicates that the address was obtained by using stateless or stateful address configuration.
STATE	Describes the address object in its actual active configuration. For a full list of these values, see the `ipadm(1M)` man page.
ADDR	Specifies the IP address that is configured over the interface. The address can be IPv4 or IPv6. A tunnel interface will display both local and remote addresses. For more information about tunnels, see Chapter 6, "Configuring IP Tunnels," in *Oracle Solaris Administration: IP Services*.

4 To obtain information about interface properties, type the following command:

```
# ipadm show-ifprop [-p property] interface
```

If you do not specify a property, then all the properties and their settings are displayed.

The fields in the command output refer to the following:

IFNAME	Refers to the interface whose information is being displayed.
PROPERTY	Refers to the property of the interface. An interface can have several properties.
PROTO	Refers to the protocol to which the property applies, and which can either be IPv4 or IPv6.
PERM	Refers to the allowed permissions of a given property, which can be read only, write only, or both.
CURRENT	Indicates the current setting of the property in active configuration.
PERSISTENT	Refers to the setting of the property that is reapplied when the system is rebooted.
DEFAULT	Indicates the default setting of the specified property.
POSSIBLE	Refers to a list of values that can be assigned to the specified property. For numeric settings, a range of acceptable values is displayed.

Note – If any field value is unknown, such as when an interface does not support the property whose information is being requested, the setting is displayed as a question mark (`?`).

5 To obtain information about an address property, type the following command:

```
# ipadm show-addrprop [-p property,...] [addrobj]
```

The information that is displayed depends on the options that you use.

- If you do not specify a property, then all properties are listed.
- If you specify only the property, then that property for all the addresses is displayed.
- If you specify only the address object, then the properties of all existing addresses on the system are displayed.

The fields in the command output refer to the following:

`ADDROBJ`	Refers to the address object whose properties are being listed.
`PROPERTY`	Refers to the property of the address object. An address object can have several properties.
`PERM`	Refers to the allowed permissions of a given property, which can be read only, write only, or both.
`CURRENT`	Refers to the actual setting of the property in the present configuration.
`PERSISTENT`	Refers to the setting of the property that is reapplied when the system is rebooted.
`DEFAULT`	Indicates the default setting of the specified property.
`POSSIBLE`	Refers to a list of settings that can be assigned to the specified property. For numeric settings, a range of acceptable values is displayed.

Example 9–8 Using the `ipadm` Command to Monitor Interfaces

This set of examples shows the types of information that can be obtained by using the `ipadm show-*` subcommands. First, general interface information is displayed. Then, address information is provided. Finally, information about a specific property, the `MTU` of the interface `net1`, is provided. The examples include tunnel interfaces as well as interfaces that use a customized name.

```
# ipadm show-if
IFNAME       CLASS        STATE     ACTIVE     OVER
lo0          loopback     ok        yes        --
net0         ip           ok        yes        --
net1         ip           ok        yes        --
tun0         ip           ok        yes        --
```

```
# ipadm show-addr
ADDROBJ           TYPE       STATE     ADDR
lo0/?             static     ok        127.0.0.1/8
net0/v4           static     ok        192.168.84.3/24
tun0/v4tunaddr    static     ok        173.129.134.1-->173.129.134.2
```

Note that an address object that is listed as *interface*/? indicates that the address was configured on the interface by an application that did not use `libipadm` APIs. Such applications are not under the control of the `ipadm` command, which requires that the address object name use the format *interface*/*user-defined-string*. For examples of assigning IP addresses, see "How to Configure an IP Interface" on page 171.

```
# ipadm show-ifprop -p mtu net1
IFNAME  PROPERTY  PROTO  PERM  CURRENT  PERSISTENT  DEFAULT  POSSIBLE
net1    mtu       ipv4   rw    1500      --         1500     68-1500
net1    mtu       ipv6   rw    1500      --         1500     1280-1500

# ipadm show-addrprop net1/v4
ADDROBJ          PROPERTY    PERM  CURRENT          PERSISTENT  DEFAULT          POSSIBLE
net1/v4   broadcast   r-    192.168.84.255  --          192.168.84.255   --
net1/v4   deprecated  rw    off             --          off              on,off
net1/v4   prefixlen   rw    24              24          24               1-30,32
net1/v4   private     rw    off             --          off              on,off
net1/v4   transmit    rw    on              --          on               on,off
net1/v4   zone        rw    global          --          global           --
```

Troubleshooting Interface Configuration

This section discusses common problems that you might encounter while using the `ipadm` command to configure IP interfaces.

The `ipadm` command does not work.

Manual IP interface configuration with the `dladm` and `ipadm` commands work only on fixed-type network configuration profiles (NCP), such as `DefaultFixed`. If the active NCP in the system is an automatic-type profile, switch to a fixed-type profile before using the `dladm` and `ipadm` commands.

```
# netadm list
TYPE    PROFILE        STATE
ncp     DefaultFixed   disabled
ncp     Automatic      online
loc     Automatic      offline
loc     NoNet          offline
...

# netadm enable -p ncp defaultfixed
```

IP address cannot be assigned with the `ipadm create-addr` command.

With the traditional `ifconfig` command, you can plumb and assign an IP address with a single command syntax. When using the `ipadm create-addr` command to configure an IP address, you must first create the IP interface with a separate command.

```
# ipadm create-ip interface
# ipadm create-addr -T addr-type -a address addrobj
```

The message `cannot create address object: Invalid argument provided` is displayed during IP address configuration.

The address object identifies a specific IP address bound to an IP interface. The address object is a unique identifier for each IP address on the IP interface. You must specify a different address object to identify a second IP address that you want to assign to the same IP interface. If you want to use the same address object name, then you must delete the first instance of the address object before assigning it to identify a different IP address.

```
# ipadm show-addr
ADDROBJ   TYPE     STATE   ADR
lo0       static   ok      127.0.0.1/10
net0/v4   static   ok      192.168.10.1

# ipadm create-addr -T static -a 192.168.10.5 net0/v4b
```

or

```
# ipadm show-addr
ADDROBJ   TYPE     STATE   ADR
lo0       static   ok      127.0.0.1/10
net0/v4   static   ok      192.168.10.1

# ipadm delete-addr net0/v4
# ipadm create-addr -T static -a 192.168.10.5 net0/v4
```

The message cannot create address: Persistent operation on temporary object during IP interface configuration

The ipadm command creates persistent configuration. If the IP interface that you are configuring was created as a temporary interface, then you cannot use the ipadm command to configure persistent settings on the interface. After you verify that an interface that you are configuring is temporary, delete that interface, re-create it as a persistent object, then resume configuring.

```
# ipadm show-if -o all
IFNAME   CLASS      STATE   ACTIVE   CURRENT        PERSISTENT   OVER
lo0      loopback   ok      yes      -m46-v------   46--         --
net0     ip         ok      yes      bm4---------   ----         --
```

The absence of the 4 flag for IPv4 configuration or 6 flag for IPv6 configuration on the PERSISTENT field indicates that net0 was created as a temporary interface.

```
# ipadm delete-ip net0
# ipadm create-ip net0
# # ipadm create-addr -T static -a 192.168.1.10 net0/v4
```

Comparison Tables: ipadm Command and Other Networking Commands

The ipadm command is the preferred tool to use for all configuration tasks on IP interfaces. This command replaces the commands in previous releases that were used for network configuration, such as the ifconfig and ndd commands. The following tables list selected command options of these previous tools and their equivalents in the ipadm command.

Note – These tables do not provide a comprehensive list of ipadm options. For a full list, see ipadm(1M) man page.

---

ifconfig Command Options and ipadm Command Options

The following table shows the ifconfig command options and the approximate corresponding ipadm subcommands.

TABLE 9–3 Syntax Mapping Between the ifconfig and ipadm Commands

ifconfig Command	**ipadm Command**
plumb/unplumb	ipadm create-ip ipadm create-vni ipadm create-ipmp ipadm enable-addr ipadm delete-ip ipadm delete-vni ipadm delete-ipmp ipadm disable-addr
[*address*[/*prefix-length*] [*dest-address*]] [addif *address*[*prefix-length*]] [removeif *address*[*prefix-length*]][netmask *mask*][destination *dest-address*]{auto-dhcp|dhcp}[primary][wait *seconds*]extend | release | start	ipadm create-addr -T static ipadm create-addr -T dhcp ipadm create-addr -T addrconf ipadm show-addr ipadm delete-addr ipadm refresh-addr
[deprecated | -deprecated] [preferred | -preferred] [private | -private] [zone *zonename* | -zones | -all-zones][xmit | -xmit]	ipadm set-addprop ipadm reset-addprop ipadm show-addprop
up	ipadm up-addr
down	ipadm down-addr
[metric *n*] [mtu *n*] [nud | -nud] [arp | -arp] [usesrc [*name* | none] [router | -router]	ipadm set-ifprop ipadm show-ifprop ipadm reset-ifprop
[ipmp] [group [*name* | ""]] standby | -standby] [failover | -failover]	ipadm create-ipmp ipadm delete-ipmp ipadm add-ipmp ipadm remove-ipmp ipadm set-ifprop -p [standby] [group]

TABLE 9–3 Syntax Mapping Between the ifconfig and ipadm Commands *(Continued)*

ifconfig Command	ipadm Command
[tdst *tunnel-dest-addr*] [tsrc *tunnel-srcs-addr*] [encaplimit *n*| -encaplimit] [thoplimit *n*]	dladm *-iptun set of commands. For more details, see the dladm(1M) man page and "Tunnel Configuration and Administration With the dladm Command" in *Oracle Solaris Administration: IP Services*.
[auth_algs *authentication algorithm*] [encr_algs *encryption algorithm*] [encr_auth_algs *encryption authentication algorithm*]	ipsecconf For details, see the ipsecconf(1M) and Chapter 15, "Configuring IPsec (Tasks)," in *Oracle Solaris Administration: IP Services*.
[auth_revarp] [ether [*address*]] [index *if-index*] [subnet *subnet-address*] [broadcast *broadcast-address*] [token *address/prefix-length*] dhcp options – inform, ping, release, status, drop	Equivalent subcommands currently unavailable.
modlist] [modinsert *mod_name@pos*] [modremove *mod_name@pos*]	Equivalent subcommands currently unavailable.

ndd Command Options and ipadm Command Options

The following table shows the ndd command options and the approximate corresponding ipadm subcommands.

TABLE 9–4 Syntax Mapping Between the ndd and ipadm Commands

ndd Command	ipadm Command
Retrieving Properties	

TABLE 9–4 Syntax Mapping Between the ndd and ipadm Commands *(Continued)*

ndd Command	ipadm Command
bash-3.2# **ndd -get /dev/ip ?** ip_def_ttl (read and write) ip6_def_hops (read and write) ip_forward_directed_broadcasts (read and write) ip_forwarding (read and write) ... ... bash-3.2# **ndd -get /dev/ip ** **ip_def_ttl** 100 bash-3.2# **ndd -get /dev/ip ** **ip6_def_hops** 255 bash-3.2# **ndd -get /dev/tcp ?** tcp_cwnd_max (read and write) tcp_strong_iss (read and write) tcp_time_wait_interval (read and write) tcp_tstamp_always (read and write) tcp_tstamp_if_wscale (read and write) ... ... bash-3.2# **ndd -get /dev/tcp ecn** 1 bash-3.2# **ndd -get /dev/tcp sack** 2	bash-3.2# **ipadm show-prop ip** PROTO PROPERTY PERM CURRENT PERSISTENT DEFAULT POSSIBLE ipv4 forwarding rw off -- off on,off ipv4 ttl rw 255 -- 255 1-255 ipv6 forwarding rw off -- off on,off ipv6 hoplimit rw 255 -- 255 1-255 ... bash-3.2# **ipadm show-prop -p ttl,hoplimit ip** PROTO PROPERTY PERM CURRENT PERSISTENT DEFAULT POSSIBLE ipv4 ttl rw 255 -- 255 1-255 ipv6 hoplimit rw 255 -- 255 1-255 bash-3.2# **ipadm show-prop tcp** PROTO PROPERTY PERM CURRENT PERSISTENT DEFAULT POSSIBLE tcp ecn rw passive -- passive never,passive, active tcp extra_ rw 2049 2049,4045 2049,4045 1-65535 priv_ports tcp largest_ rw 65535 -- 65535 1024-65535 anon_port tcp recv_ rw 128000 -- 128000 2048-1073741824 maxbuf tcp sack rw active -- active never,passive, active tcp send_ rw 49152 -- 49152 4096-1073741824 maxbuf tcp smallest_ rw 32768 -- 32768 1024-65535 anon_port tcp smallest_ rw 1024 -- 1024 1024-32768 nonpriv_port ... ... bash-3.2# **ipadm show-prop -p ecn,sack tcp** PROTO PROPERTY PERM CURRENT PERSISTENT DEFAULT POSSIBLE tcp ecn rw passive -- passive never,passive,active tcp sack rw active -- active never,passive,active
Setting Properties	

TABLE 9–4 Syntax Mapping Between the ndd and ipadm Commands *(Continued)*

ndd Command	ipadm Command
bash-3.2# **ndd -set /dev/ip ** **ip_def_ttl** 64 bash-3.2# **ndd -get /dev/ip ** **ip_def_ttl** 64	bash-3.2# **ipadm set-prop -p ttl=64 ipv4** bash-3.2# **ipadm show-prop -p ttl ip** PROTO PROPERTY FAMILY PERM VALUE DEFAULT POSSIBLE ip ttl inet rw 64 255 1-255 PROTO PROPERTY PERM CURRENT PERSISTENT DEFAULT POSSIBLE ipv4 ttl rw 64 64 255 1-255 bash-3.2# **ipadm reset-prop -p ttl ip** bash-3.2# **ipadm show-prop -p ttl ip** PROTO PROPERTY PERM CURRENT PERSISTENT DEFAULT POSSIBLE ipv4 ttl rw 255 255 255 1-255

◆ ◆ ◆ CHAPTER 10

Configuring Wireless Interface Communications on Oracle Solaris

This chapter explains how to configure and use wireless interface communications on a laptop that runs Oracle Solaris. The following topics are covered:

- Communicating over WiFi Interfaces
- Finding a WiFi Network
- Connecting and Using WiFi on Oracle Solaris Systems
- Secure WiFi Communications

WiFi Communications Task Map

Task	Description	For Instructions
Plan for WiFi communications on your system.	Set up your laptop or wireless network configuration, optionally including a router, in a location that supports WiFi	"How to Prepare a System for WiFi Communications" on page 197
Connect to a WiFi network	Set up and establish communications with a local WiFi network	"How to Connect to a WiFi Network" on page 198
Monitor communications on the WiFi link	Use standard Oracle Solaris networking tools to check the state of WiFi link	"How to Monitor the WiFi Link" on page 202
Establish secure WiFi communications	Create a WEP key and use it establish connections with a secure WiFi network	"How to Set Up an Encrypted WiFi Network Connection" on page 204

Communicating Over WiFi Interfaces

The IEEE 802.11 specifications define wireless communications for local area networks. These specifications and the networks they describe are referred to collectively as *WiFi*, a term that is trademarked by the Wi-Fi Alliance trade group. WiFi networks are reasonably easy to configure by both providers and prospective clients. Therefore, they are increasingly popular and in common use throughout the world. WiFi networks use the same radio wave technology as cellular phones, televisions, and radios.

Oracle Solaris contains features that enable you to configure a system as a WiFi client. This section explains how to use the WiFi connectivity options of the `dladm` command to connect a laptop or home computer to a local WiFi network.

Note – Oracle Solaris does not contain features for configuring WiFi servers or access points.

Finding a WiFi Network

WiFi networks typically come in three varieties:

- Commercially available WiFi networks
- Municipal WiFi networks
- Private WiFi networks

A location that is served by WiFi is referred to as a *hot spot*. Each hot spot includes an access point. The *access point* is a router with a "wired" connection to the Internet, for example, Ethernet or DSL. The Internet connection is usually through a wireless Internet service provider (WISP) or traditional ISP.

Commercial WiFi Networks

Many hotels and cafes offer wireless Internet connections as a service to their customers with laptop computers. These commercial hot spots have access points within their facilities. The access points are routers with wired connections to a WISP that serves commercial locations. Typical WISPs include independent providers and cellular phone companies.

You can use a laptop that runs Oracle Solaris to connect to a WiFi network that is offered by a hotel or other commercial hot spot. Ask for instructions at the hot spot for connecting to the WiFi network. Typically, the connection process involves supplying a key to a browser that you launch upon login. You might have to pay a fee to the hotel or WISP in order to use the network.

Commercial locations that are Internet hot spots usually advertise this capability to their patrons. You can also find lists of wireless hot spots from various web sites, for example, Wi-FiHotSpotList.com (`http://www.wi-fihotspotlist.com`).

Municipal WiFi Networks

Cities throughout the world have constructed free municipal WiFi networks, which their citizens can access from systems in their homes. Municipal WiFi uses radio transmitters on telephone poles or other outdoor locations to form a "mesh" over the area that the network serves. These transmitters are the access points to the municipal WiFi network. If your area is served by a municipal WiFi network, your home might be included in the network's mesh.

Access to municipal WiFi is usually free. You can access the municipal network from a properly equipped laptop or personal computer that runs Oracle Solaris. You do not need a home router to access the municipal network from your system. However, configuring a home router is recommended for areas where the signal from the municipal network is weak. Home routers are also recommended if you require secure connections over the WiFi network. For more information, see "Secure WiFi Communications" on page 203.

Private WiFi Networks

Because WiFi networks are relatively easy to configure, companies and universities use private WiFi networks with access limited to employees or students. Private WiFi networks typically require you to supply a key when you connect or run a secure VPN after you connect. You need a properly equipped laptop or PC that runs Oracle Solaris and permission to use the security features in order to connect to the private network.

Planning for WiFi Communications

Before you can connect your system to a WiFi network, complete the following instructions.

▼ How to Prepare a System for WiFi Communications

1 Equip your system with a supported WiFi interface.

Your system must have a WiFi card that is supported by Oracle Solaris, such as cards that support the Atheros chip sets. For a list of currently supported drivers and chip sets, refer to Wireless Networking for OpenSolaris (`http://hub.opensolaris.org/bin/view/Community+Group+laptop/wireless`).

If the interface is not already present on the system, follow the manufacturer's instructions for installing the interface card. You configure the interface software during the procedure "How to Connect to a WiFi Network" on page 198.

2 Locate your system in a place that is served by a WiFi network, either commercial, municipal, or private.

Your system must be near the access point for the network, which is normally not a consideration for a commercial or private network hot spot. However, if you plan to use a free municipal network, your location must be near the transmitter access point.

3 **(Optional) Set up a wireless router to serve as an additional access point.**

Set up your own router if no WiFi network is available at your location. For example, if you have a DSL line, connect the wireless router to the DSL router. Then the wireless router becomes the access point for your wireless devices.

Connecting and Using WiFi on Oracle Solaris Systems

This section contains tasks for establishing and monitoring WiFi connections for a laptop or desktop computer that runs Oracle Solaris.

▼ How to Connect to a WiFi Network

Before You Begin The following procedure assumes that you have followed the instructions in "How to Prepare a System for WiFi Communications" on page 197.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Check for available links.**

```
# dladm show-link
LINK        CLASS     MTU    STATE    BRIDGE     OVER
ath0        phys      1500   up       --         --
e1000g0     phys      1500   up       --         --
```

In this example, the output indicates that two links are available. The `ath0` link supports WiFi communications. The `e1000g` link is for attaching the system to a wired network.

3 **Configure the WiFi interface.**

Use the following steps to configure the interface:

- Create the interface that supports WiFi:

```
# ipadm create-ip ath0
```

- Verify that the link has been plumbed:

```
# ipadm show-if
IFNAME      CLASS        STATE     ACTIVE     OVER
lo0         loopback     ok        yes        --
e1000g0     ip           ok        yes        --
ath0        ip           ok        yes        --
```

4 **Check for available networks.**

```
# dladm scan-wifi
 LINK       ESSID         BSSID/IBSSID       SEC     STRENGTH   MODE   SPEED
 ath0       net1         00:0e:38:49:01:d0  none     good       g      54Mb
```

```
ath0        net2          00:0e:38:49:02:f0  none     very weak  g      54Mb
ath0        net3          00:0d:ed:a5:47:e0  none     very good  g      54Mb
```

The example output of the `scan-wifi` command displays information about the available WiFi networks at the current location. The information in the output includes:

`LINK`	Link name to be used in the WiFi connection.
`ESSID`	Extended Service Set ID. The `ESSID` is the name of the WiFi network, such as `net1`, `net2`, and `net3` in the example output.
`BSSID/IBSSID`	Basic Service Set ID, the unique identifier for a particular `ESSID`. The `BSSID` is the 48-bit MAC address of the nearby access point that serves the network with a particular `ESSID`.
`SEC`	Type of security that is needed to access the network. The values are `none` or `WEP`. For information about WEP, refer to "Secure WiFi Communications" on page 203.
`STRENGTH`	Strength of the radio signals from the WiFi networks that are available at your location.
`MODE`	Version of the 802 .11 protocol that is run by the network. The modes are a, b, or g, or these modes in combination.
`SPEED`	Speed in megabits per second of the particular network.

5 Connect to a WiFi network.

Do either of the following:

- Connect to the unsecured WiFi network with the strongest signal.

  ```
  # dladm connect-wifi
  ```

- Connect to an unsecured network by specifying its `ESSID`.

  ```
  # dladm connect-wifi -e ESSID
  ```

  The `connect-wifi` subcommand of `dladm` has several more options for connecting to a WiFi network. For complete details, refer to the `dladm(1M)` man page.

6 Configure an IP address for the interface.

Do either of the following:

- Obtain an IP address from a DHCP server.

  ```
  # ipadm create-addr -T dhcp addrobj
  ```

  where *addrobj* uses the naming convention *interface*/*user–defined–string*.

  If the WiFi network does not support DHCP, you receive the following message:

```
ipadm: interface: interface does not exist or cannot be managed using DHCP
```

- Configure a static IP address:

  Use this option if you have a dedicated IP address for the system.

  ```
  # ipadm create-addr -T static -a address addrobj
  ```

7 Check the status of the WiFi network to which the system is connected.

```
# dladm show-wifi
LINK       STATUS       ESSID         SEC     STRENGTH   MODE   SPEED
ath0       connected    net3          none    very good  g      36Mb
```

In this example, the output indicates that the system is now connected to the `net3` network. The earlier `scan-wifi` output indicated that `net3` had the strongest signal among the available networks. The `dladm show-wifi` command automatically chooses the WiFi network with strongest signal, unless you directly specify a different network.

8 Access the Internet through the WiFi network.

Do either of the following, depending on the network to which the system is connected:

- If the access point offers free service, you can now run a browser or an application of your choice.
- If the access point is in a commercial hot spot that requires a fee, follow the instructions provided at the current location. Typically, you run a browser, supply a key, and give credit card information to the network provider.

9 Conclude the session.

Do one of the following:

- Terminate the WiFi session but leave the system running.

  ```
  # dladm disconnect-wifi
  ```

- Terminate a particular WiFi session when more than one session is currently running.

  ```
  # dladm disconnect-wifi link
  ```

  where *link* represents the interface that was used for the session.

- Cleanly shut down the system while the WiFi session is running.

  ```
  # shutdown -g0 -i5
  ```

  You do not need to explicitly disconnect the WiFi session prior to turning off the system through the `shutdown` command.

Example 10–1 Connecting to a Specific WiFi Network

The following example shows a typical scenario that you might encounter when using a laptop that runs Oracle Solaris in an Internet coffee house.

Learn whether a WiFi link is available.

```
# dladm show-wifi
ath0            type: non-vlan    mtu: 1500         device: ath0
```

The `ath0` link is installed on the laptop. Configure the `ath0` interface, and verify that it is up.

```
# ipadm create-ip ath0
IFNAME     STATE    CURRENT     PERSISTENT
lo0        ok       -m-v-----46 ---
ath0       ok       bm-------46 -46
```

Display the available WiFi links at your location.

```
# dladm scan-wifi
 LINK       ESSID         BSSID/IBSSID       SEC     STRENGTH   MODE   SPEED
 ath0       net1         00:0e:38:49:01:d0  none     weak       g      54Mb
 ath0       net2         00:0e:38:49:02:f0  none     very weak  g      54Mb
 ath0       net3         00:0d:ed:a5:47:e0  wep      very good  g      54Mb
 ath0       citinet      00:40:96:2a:56:b5  none     good       b      11Mb
```

The output indicates that `net3` has the best signal. `net3` requires a key, for which the provider for the coffee house charges a fee. `citinet` is a free network provided by the local town.

Connect to the `citinet` network.

```
# dladm connect-wifi -e citinet
```

The `-e` option of `connect-wifi` takes the `ESSID` of the preferred WiFi network as its argument. The argument in this command is `citinet`, the `ESSID` of the free local network. The `dladm connect-wifi` command offers several options for connecting to the WiFi network. For more information, refer to the `dladm(1M)` man page.

Configure an IP address for the WiFi interface.

```
# ipadm create-addr -T static -a 10.192.16.3/8 ath0/v4
# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
e1000g0/v4        static   ok           129.146.69.34/24
ath0/v4static     static   ok           10.192.16.3/8
lo0/v6            static   ok           ::1/128
```

This example assumes that you have the static IP address `10.192.16.3/24` configured on your laptop.

```
# dladm show-wifi
LINK       STATUS        ESSID         SEC     STRENGTH   MODE   SPEED
ath0       connected     citinet       none    good       g      11Mb
```

The output indicates that the laptop is now connected to network `citinet`.

```
# firefox
```

The home page for the Firefox browser displays.

Run a browser or other application to commence your work over the WiFi network.

```
# dladm disconnect-wifi
# dladm show-wifi
LINK        STATUS        ESSID         SEC     STRENGTH   MODE   SPEED
ath0        disconnected   --            --       --        --       --
```

The output of `show-wifi` verifies that you have disconnected the `ath0` link from the WiFi network.

How to Monitor the WiFi Link

This procedure shows how to monitor the status of a WiFi link through standard networking tools, and change link properties through the `linkprop` subcommand.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Connect to the WiFi network, as described in "How to Connect to a WiFi Network" on page 198.

3 View the properties of the link.

Use the following syntax:

`# dladm show-linkprop` *interface*

For example, you would use the following syntax to show the status of the connection established over the `ath0` link:

```
# dladm show-linkprop ath0
PROPERTY        VALUE           DEFAULT         POSSIBLE
channel         5               --              --
powermode       off             off             off,fast,max
radio           ?               on              on,off
speed           36               --              1,2,5.5,6,9,11,12,18,24,36,48,54
```

4 Set a fixed speed for the link.

Caution – Oracle Solaris automatically chooses the optimal speed for the WiFi connection. Modifying the initial speed of the link might cause reduced performance or prevent the establishment of certain WiFi connections.

You can modify the link speed to one of the possible values for speed that is listed in the `show-linkprop` output.

```
# dladm set-linkprop -p speed=value link
```

5 Check the packet flow over the link.

```
# netstat -I ath0 -i 5
   input   ath0      output       input  (Total)    output
packets errs  packets errs  colls  packets errs  packets errs  colls
317     0     106     0     0      2905    0     571     0     0
14      0     0       0     0      20      0     0       0     0
7       0     0       0     0      16      0     1       0     0
5       0     0       0     0      9       0     0       0     0
304     0     10      0     0      631     0     316     0     0
338     0     9       0     0      722     0     381     0     0
294     0     7       0     0      670     0     371     0     0
306     0     5       0     0      649     0     338     0     0
289     0     5       0     0      597     0     301     0     0
```

Example 10–2 Set the Speed of a Link

This example shows how to set the speed of a link after you have connected to a WiFi network

```
# dladm show-linkprop -p speed ath0
PROPERTY        VALUE           DEFAULT         POSSIBLE
speed           24              --              1,2,5,6,9,11,12,18,24,36,48,54
# dladm set-linkprop -p speed=36 ath0

# dladm show-linkprop -p speed ath0
PROPERTY        VALUE           DEFAULT         POSSIBLE
speed           36              --              1,2,5,6,9,11,12,18,24,36,48,54
```

Secure WiFi Communications

Radio wave technology makes WiFi networks readily available and often freely accessible to users in many locations. As a result, connecting to a WiFi network can be an insecure undertaking. However, certain types of WiFi connections are more secure:

- Connecting to a private, restricted-access WiFi network

  Private networks, such as internal networks established by corporations or universities, restrict access to their networks to users who can provide the correct security challenge. Potential users must supply a key during the connection sequence or log in to the network through a secure VPN.
- Encrypting your connection to the WiFi network

You can encrypt communications between your system and a WiFi network by using a secure key. Your access point to the WiFi network must be a router in your home or office with a secure key-generating feature. Your system and the router establish and then share the key before creating the secure connection.

The `dladm` command can use a Wired Equivalent Privacy (WEP) key for encrypting connections through the access point. The WEP protocol is defined in IEEE 802.11 specifications for wireless connections. For complete details on the WEP-related options of the `dladm` command, refer to the `dladm(1M)` man page.

How to Set Up an Encrypted WiFi Network Connection

The next procedure shows how to set up secure communications between a system and a router in the home. Many wireless and wired routers for the home have an encryption feature that can generate a secure key. This procedure assumes that you use such a router and have its documentation available. The procedure also assumes that your system is already plugged into the router.

1. **Start the software for configuring the home router.**

   Refer to the manufacturer's documentation for instructions. Router manufacturers typically offer an internal web site or a graphical user interface for router configuration.

2. **Generate the value for the WEP key.**

   Follow the manufacturer's instructions for creating a secure key for the router. The router configuration GUI might ask you to supply a passphrase of your choice for the key. The software then uses the passphrase to generate a hexadecimal string, typically 5 bytes or 13 bytes in length. This string becomes the value to be used for the WEP key.

3. **Apply and save the key configuration.**

   Refer to the manufacturer's documentation for instructions.

4. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

5. **Create a secure object that contains the WEP key.**

   Open a terminal window on the system and type the following:

   ```
   # dladm create-secobj -c wep keyname
   ```

   where *keyname* represents the name you want to give to the key.

6 Supply the value for the WEP key to the secure object.

The `create-secobj` subcommand then runs a script that requests the value for the key.

```
provide value for keyname: 5 or 13 byte key
confirm value for keyname: retype key
```

This value is the key that was generated by the router. The script accepts either a five byte or thirteen byte string, in ASCII or in hexadecimal for the key value.

7 View the contents of the key that you just created.

```
# dladm show-secobj
OBJECT              CLASS
keyname               wep
```

where *keyname* is the name for the secure object.

8 Make an encrypted connection to the WiFi network.

```
# dladm connect-wifi -e network -k keyname interface
```

9 Verify that the connection is secure.

```
# dladm show-wifi
LINK       STATUS        ESSID         SEC     STRENGTH   MODE   SPEED
ath0       connected     net1          wep     good       g      11Mb
```

The `wep` value under the `SEC` heading indicates that WEP encryption is in place for the connection.

Example 10–3 Setting Up Encrypted WiFi Communications

This example assumes that you have already done the following:

- Connected your system to a home router that can create a WEP key
- Followed the router manufacturer's documentation and created the WEP key
- Saved the key so that you can use it to create the secure object on your system

```
# dladm create-secobj -c wep mykey
provide value for mykey: *****
confirm value for mkey: *****
```

When you supply the WEP key generated that is by the router, asterisks mask the value that you type.

```
# dladm show-secobj
OBJECT              CLASS
 mykey              wep
# dladm connect-wifi -e citinet -k mykey ath0
```

This command establishes an encrypted connection to the WiFi network `citinet`, using the secure object `mykey`.

```
# dladm show-wifi
LINK        STATUS        ESSID         SEC     STRENGTH   MODE   SPEED
ath0        connected     citinet       wep     good       g      36Mb
```

This output verifies that you are connected to `citinet` through WEP encryption.

◆ ◆ ◆ CHAPTER 11

Administering Bridges

This chapter describes bridges and how to administer them.

This chapter covers the following topics:

Bridging Overview

Bridges are used to connect separate network segments. When connected by a bridge, the attached network segments communicate as if they were a single network segment. Bridging is implemented at the datalink layer (L2) of the networking stack. Bridges use a packet-forwarding mechanism to connect subnetworks together.

While bridging and routing can both be used to distribute information about the locations of resources on the network, they differ in several ways. Routing is implemented at the IP layer (L3) and uses routing protocols. No routing protocols are used on the datalink layer. Instead, the destinations of forwarded packets are determined by examining the network traffic that is received on the links that are attached to the bridge.

When a packet is received, its source address is examined. The packet's source address associates the node from which the packet was sent to the link on which it is received. Thereafter, when a received packet uses that same address as the destination address, the bridge forwards the packet over the link to that address.

The link associated with a source address might be an intermediate link that is connected to another bridge in the bridged subnetwork. Over time, all of the bridges within the bridged subnetwork "learn" which of the links sends a packet toward a given node. Thus, the packet's destination address is used to direct the packet to its final destination by means of hop-by-hop bridging.

A local “link-down” notification indicates that all nodes on a given link are no longer reachable. In this situation, packet forwarding to the link is halted and all forwarding entries over the link are flushed. Forwarding entries are also aged away over time. When a link is restored, packets that are received over the link are treated as new. The “learning” process based on a packet's source address begins again. This process enables the bridge to properly forward packets over that link when the address is used as the destination address.

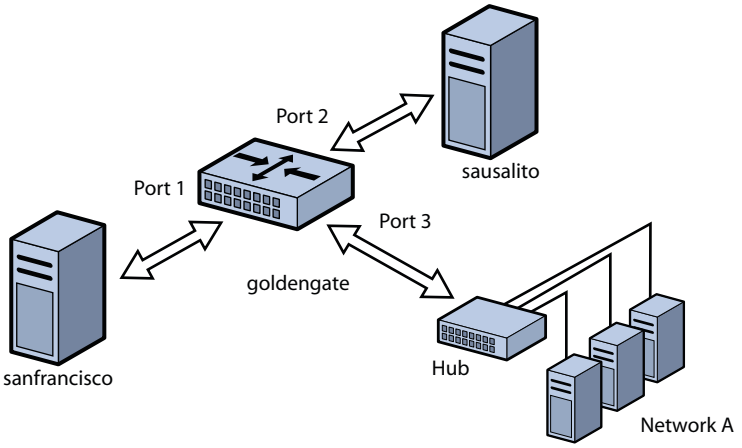
To forward packets to their destinations, bridges must listen in promiscuous mode on every link that is attached to the bridge. Listening in promiscuous mode causes bridges to become vulnerable to the occurrences of forwarding loops, in which packets circle forever at full line rate. Thus, bridging uses the Spanning Tree Protocol (STP) mechanism to prevent network loops that would render the subnetworks unusable.

In addition to using STP and the Rapid Spanning Tree Protocol (RSTP) for bridges, Oracle Solaris supports the TRILL protection enhancement. STP is used by default, but you can use TRILL by specifying the `-P trill` option for the bridging commands.

Using a bridge configuration simplifies the administration of the various nodes in the network by connecting them into a single network. By connecting these segments through a bridge, all the nodes share a single broadcast network. Thus, each node can reach the others by using network protocols such as IP rather than by using routers to forward traffic across network segments. If you do not use a bridge, you must configure IP routing to permit the forwarding of IP traffic between nodes.

The following figure shows a simple bridged network configuration. The bridge, `goldengate`, is an Oracle Solaris system that has bridging configured. `sanfrancisco` and `sausalito` are systems that are physically connected to the bridge. Network A uses a hub that is physically connected to the bridge on one side and to computer systems on the other side. The bridge ports are links, such as `bge0`, `bge1`, and `bge2`.

FIGURE 11–1 Simple Bridged Network



Bridge networks can be formed into rings that physically connect several bridges together. Such configurations are common in networks. This type of configuration could cause problems with old packets saturating the network links by endlessly looping around the ring. To protect against such looping conditions, Oracle Solaris bridges implement both the STP and TRILL protocols. Note that most hardware bridges also implement STP loop protection.

The following figure shows a bridged network that is configured in a ring. The configuration shows three bridges. Two systems are physically connected to `westminster`. One system is physically connected to `waterloo`. And one system is physically connected to `tower`. Each of the bridges are physically connected to each other through the bridge ports.

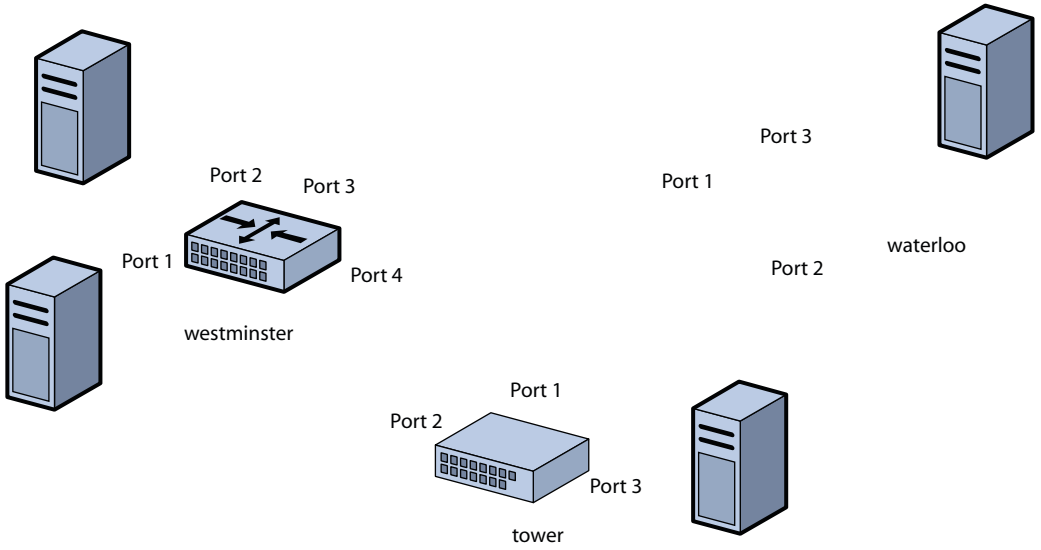
When STP or RSTP is used for loop protection, the physical loop is mitigated by preventing one of the connections in the loop from forwarding packets. The figure shows that the physical link between the `westminster` and `tower` bridges is not used to forward packets.

Note that by shutting down usable physical links to perform loop protection, STP and RSTP cost you bandwidth.

Unlike STP and RSTP, TRILL does not shut down physical links to prevent loops. Instead, TRILL computes the shortest-path information for each TRILL node in the network and uses that information to forward packets to individual destinations.

As a result, TRILL enables the system to leave *all* links in use at all times. Loops are not a problem as they are handled similarly to the way that IP handles loops. Namely, TRILL creates routes as needed and uses forwarding hop limits to avoid problems that are caused by momentary loop states.

FIGURE 11–2 Bridged Network Ring



Caution – Do *not* set `local-mac-address?=false` on SPARC platforms, or the systems will errantly use the same MAC address on multiple ports and on the same network.

Note – *Do not* configure a link into a bridge when the highest possible levels of performance are required. Bridging *requires* that the underlying interfaces are in promiscuous mode, which disables a number of important optimizations that are in the hardware, driver, and other layers of the system. The disabling of these performance enhancements is an unavoidable consequence of the bridging mechanism.

You can use a bridge on a system where *some* of the system's links are not bridged and are thus not subject to those constraints. These performance issues only affect those links that are configured to be part of a bridge.

For information about STP, see IEEE 802.1D-1998. For information about RSTP, see IEEE 820.1Q-2004. For information about TRILL, see the Internet Engineering Task Force (IETF) TRILL draft documents (`http://tools.ietf.org/wg/trill`).

Link Properties

These link properties can be shown and modified by the `dladm show-linkprop`, `dladm set-linkprop`, and `reset-linkprop` commands:

`default_tag` Define the default virtual local area network (VLAN) ID for untagged packets that are sent to and from the link. Valid values are from `0` to `4094`. The default value is `1`. Only non-VLAN and non-virtual network interface card (VNIC) type links have this property. Setting this value to `0` disables the forwarding of untagged packets to and from the port. (This is a MAC property.)

Note – This property is also used outside the scope of bridging to specify the IEEE Port VLAN Identifier (PVID) for the link. When `default_tag` is non-zero, you cannot create a VLAN that has that same ID on the link because the base link itself automatically represents the PVID.

For example, if PVID is set to 5 on `net0`, you cannot create a VLAN with ID 5 on `net0`. To specify VLAN 5 in this situation, use `net0`.

You cannot set `default_tag` to be equal to the ID of any existing VLAN that is created on that link. For instance, the following command creates VLAN 22 on `net0`:

```
# dladm create-vlan -l net0 -v 22 myvlan0
```

In this situation, you cannot set `default_tag` to 22, which would make both `net0` and `myvlan0` represent the same VLAN.

By setting `default_tag` to 0, you enable untagged packets on `net0` to be unassociated with any VLAN at all. This situation prevents such packets from being forwarded by a configured bridge.

`forward` Enable and disable traffic forwarding through the bridge. This property exists on all links except for VNIC links. Valid values are `1` (true) and `0` (false). The default value is `1`. When disabled, a VLAN that is associated with a link instance will not forward traffic through the bridge. Disabling forwarding is equivalent to removing the VLAN from the "allowed set" for a traditional bridge. This means that VLAN-based I/O to the underlying link from local clients continues, but no bridge-based forwarding is performed.

`stp` Enable and disable STP and RSTP. Valid values are `1` (true) and `0` (false). The default value is `1`, which enables STP and RSTP. When set to `0`, the link does not use any type of Spanning Tree Protocol and is placed into forwarding mode at all times. The forwarding mode uses bridge protocol data unit (BPDU) guarding. Disable STP and RSTP when you want to configure point-to-point links that are connected to end nodes. Only non-VLAN and non-VNIC type links have this property.

`stp_cost`	Represent STP and RSTP cost values for using the link. Valid values are from `1` to `65535`. The default value is `0`, which is used to signal that cost is automatically computed by link type. The following values represent the cost for several link types: `100` for 10 Mbps, `19` for 100 Mbps, `4` for 1 Gbps, and `2` for 10 Gbps.
`stp_edge`	Specify whether the port is connected to other bridges. Valid values are `1` (true) and `0` (false). The default value is `1`. If set to `0`, the daemon assumes that the port is connected to other bridges even if no BPDUs of any type are seen.
`stp_p2p`	Specify the connection mode type. Valid values are `true`, `false`, and `auto`. The default value is `auto`, which automatically discovers point-to-point connections. Specify `true` to force to point-to-point mode, and specify `false` to force normal multipoint mode.
`stp_priority`	Set the STP and RSTP port priority value. Valid values are from `0` to `255`. The default value is `128`. The STP and RSTP port priority value is used to determine the preferred root port of a bridge by prepending the value to the port identifier. The lower the numerical value is, the higher the priority.

STP Daemon

Each bridge that you create by using the `dladm create-bridge` command is represented as an identically named SMF instance of `svc:/network/bridge`. Each instance runs a copy of the `/usr/lib/bridged` daemon, which implements the STP.

The following command example creates a bridge called `pontevecchio`:

```
# dladm create-bridge pontevecchio
```

The system creates an SMF service called `svc:/network/bridge:pontevecchio` and an observability node called `/dev/net/pontevecchio0`.

For safety purposes, all ports run standard STP by default. A bridge that does not run some form of bridging protocol, such as STP, can form long-lasting forwarding loops in the network. Because Ethernet has no hop-count or TTL on packets, any such loops are fatal to the network.

When you know that a particular port is not connected to another bridge (for example, a direct point-to-point connection to a host system), you can administratively disable STP for that port. Even if all ports on a bridge have STP disabled, the STP daemon still runs. The daemon continues to run for the following reasons:

- To handle any new ports that are added
- To implement BPDU guarding
- To enable or disable forwarding on the ports, as necessary

When a port has STP disabled, the `bridged` daemon continues to listen for BPDUs (BPDU guarding). The daemon uses `syslog` to flag any errors and disables forwarding on the port to indicate a serious network misconfiguration. The link is reenabled when link status goes down and comes up again, or when you manually remove the link and re-add it.

If you disable the SMF service instance for a bridge, bridge forwarding stops on those ports as the STP daemon is stopped. If the instance is restarted, STP starts from its initial state.

TRILL Daemon

Each bridge that you create by using the `dladm create-bridge -P trill` command is represented as an identically named SMF instance of `svc:/network/bridge` and `svc:/network/routing/trill`. Each instance of `svc:/network/routing/trill` runs a copy of the `/usr/lib/trilld` daemon, which implements the TRILL protocol.

The following command example creates a bridge called `bridgeofsighs`:

```
# dladm create-bridge -P trill bridgeofsighs
```

The system creates two SMF services called `svc:/network/bridge:bridgeofsighs` and `svc:/network/routing/trill:bridgeofsighs`. In addition, the system creates an observability node called `/dev/net/bridgeofsighs0`.

Debugging Bridges

Each bridge instance is assigned an "observability node," which appears in the `/dev/net/` directory and is named by the bridge name plus a trailing `0`.

The observability node is intended for use with the `snoop` and `wireshark` utilities. This node behaves like a standard Ethernet interface, except for the transmission of packets, which are silently dropped. You cannot plumb IP on top of an observability node, and you cannot perform bind requests (`DL_BIND_REQ`) unless you use the passive option.

When used, the observability node makes a single unmodified copy of every packet handled by the bridge available to the user. This behavior is similar to a "monitoring" port on a traditional bridge, and is subject to the usual DLPI "promiscuous mode" rules. You can use `pfmod` or features in the `snoop` and `wireshark` utilities to filter based on VLAN ID.

The delivered packets represent the data that is received by the bridge.

Caution – In the cases where the bridging process adds, removes, or modifies a VLAN tag, the data shown describes the state prior to this process taking place. This rare situation might be confusing if there are distinct `default_tag` values used on different links.

To see the packets that are transmitted and received on a particular link (after the bridging process is complete), run `snoop` on the individual links rather than on the bridge observability node.

For information about observability nodes, see "Observability Features for Network Virtualization and Resource Control" on page 328.

Other Bridge Behaviors

The following sections describe how link behavior changes when bridges are used in the configuration.

For information about standard link behavior, see "Administering Virtual Local Area Networks" on page 237.

DLPI Behavior

The following describes the differences in link behavior when a bridge is enabled:

- Link up (`DL_NOTE_LINK_UP`) and link down (`DL_NOTE_LINK_DOWN`) notifications are delivered in the aggregate. This means that when all external links are showing link-down status, the upper-level clients that are using the MAC layers will also see link-down events. When any external link on the bridge shows link-up status, all upper-level clients see link-up.

  This aggregate link-up and link-down reporting is performed for the following reasons:

  - When link-down is seen, nodes on the link are no longer reachable. This is not true when the bridging code can still send and receive packets through another link. Administrative applications that need the actual status of links can use the existing MAC-layer kernel statistics to reveal the status. These applications are unlike ordinary clients, such as IP, in that they report hardware status information and do not get involved in forwarding.
  - When all external links are down, the status shows through as though the bridge itself were shut down. In this special case, the system recognizes that nothing could possibly be reachable. The trade-off is that bridges cannot be used to allow local-only communication in the case where all interfaces are "real" (not virtual) and all are disconnected.

- All link-specific features are made generic. Links that support special hardware acceleration features are unable to use those features because actual output link determination is not made entirely by the client. The bridge forwarding function must choose an output link based on the destination MAC address, and this output link can be any link on the bridge.

VLAN Administration

By default, VLANs that are configured on the system are forwarded among all the ports on a bridge instance. When you invoke the `dladm create-vlan` or `dladm create-vnic -v` command, and the underlying link is part of a bridge, that command will also enable forwarding of the specified VLAN on that bridge link.

To configure a VLAN on a link and disable forwarding to or from other links on the bridge, you must disable forwarding by setting the `forward` property with the `dladm set-linkprop` command.

Use the `dladm create-vlan` command to automatically enable the VLAN for bridging when the underlying link is configured as part of a bridge.

VLANs are ignored in the standards-compliant STP. The bridging protocol computes just one loop-free topology by using tag-free BPDU messages, and uses this tree to enable and disable links. You must configure any duplicate links that are provisioned in your networks such that when those links are automatically disabled by STP, the configured VLANs are not disconnected. This means that you should either run all VLANs everywhere on your bridged backbone or carefully examine all redundant links.

TRILL does not need to follow the complex STP rules. Instead, TRILL automatically encapsulates packets that have the VLAN tag intact, and passes them through the network. This means that TRILL binds together isolated VLANs where the same VLAN ID has been reused within a single bridged network.

This is an important difference from STP where you might reuse VLAN tags in isolated sections of the network to manage sets of VLANs that are larger than the 4094 limit. While you cannot use TRILL to manage networks in this way, you might be able to implement other solutions, such as provider-based VLANs.

In an STP network with VLANs, it might be difficult to configure the failover characteristics to prevent VLAN partitioning when STP disables the “wrong” link. The relatively small loss of functionality in isolated VLANs is more than made up for in the robustness of the TRILL model.

VLAN Behavior

The bridge performs forwarding by examining the allowed set of VLANs and the `default_tag` property for each link. The general process is as follows:

- **Input VLAN determination.** This task begins when a packet is received on a link. When a packet is received, it is checked for a VLAN tag. If that tag is not present or if the tag is priority-only (tag zero), the `default_tag` configured on that link (if not set to zero) is taken as the internal VLAN tag. If the tag is not present or zero and `default_tag` is zero, the packet is ignored. No untagged forwarding is performed. If the tag is present and is equal to `default_tag`, the packet is also ignored. Otherwise, the input tag is taken to be the input VLAN.
- **Link membership check.** If the input VLAN is not configured as an allowed VLAN on this link, the packet is ignored. Forwarding is then computed, and the same check is made for the output link.
- **Tag update.** If the VLAN (nonzero at this point) is equal to `default_tag` on the output link, the tag on the packet (if any) is removed, regardless of priority. If the VLAN is not equal to `default_tag` on the output link, a tag is added if not currently present, and the tag is set for the output packet, with the current priority copied into the packet.

Note – In the case where forwarding sends to multiple interfaces (for broadcast, multicast, and unknown destinations), the output link check and tag update must be done independently for each output link. Some transmissions might be tagged while others are untagged.

Bridge Configuration Examples

The following examples show how to view information about bridge configurations and bridging services.

- You can obtain information about bridges by running the following command:

```
# dladm show-bridge
BRIDGE      PROTECT ADDRESS            PRIORITY DESROOT
tonowhere   trill   32768/66:ca:b0:39:31:5d 32768 32768/66:ca:b0:39:31:5d
sanluisrey  stp     32768/ee:2:63:ed:41:94 32768 32768/ee:2:63:ed:41:94
pontoon     trill   32768/56:db:46:be:b9:62 32768 32768/56:db:46:be:b9:62
```

- You can obtain TRILL nickname information for a bridge by running the following command:

```
# dladm show-bridge -t tonowhere
NICK FLAGS LINK        NEXTHOP
38628 --   simblue2    56:db:46:be:b9:62
58753 L    --          --
```

Administering Bridges (Task Map)

Oracle Solaris uses the `dladm` command and the SMF feature to administer bridges. Use SMF commands to enable, disable, and monitor bridge instances by using the fault-managed resource identifier (FMRI) of the instance, `svc:/network/bridge`. Use the `dladm` command to create or destroy bridges, as well as to assign links to bridges or to remove links from them.

The following table points to the tasks that you can use to administer bridges.

Task	Description	For Instructions
View information about configured bridges.	Use the `dladm show-bridge` command to view information about configured bridges on the system. You can view information about configured bridges, links, statistics, and kernel forwarding entries.	"How to View Information About Configured Bridges" on page 218
View configuration information about links that are attached to a bridge.	Use the `dladm show-link` command to view information about configured links on the system. If the link is associated with a bridge, see the output in the `BRIDGE` field.	"How to View Configuration Information About Bridge Links" on page 220
Create a bridge.	Use the `dladm create-bridge` command to create a bridge and add optional links. By default, bridges are created by using STP. To use TRILL to create a bridge instead, add `-P trill` to the `dladm create-bridge` command line, or use the `dladm modify-bridge` command to enable TRILL.	"How to Create a Bridge" on page 220
Modify the protection type for a bridge.	Use the `dladm modify-bridge` command to modify the protection type for a bridge. By default, bridges are created by using STP. To use TRILL to create a bridge instead, use `-P trill` with the `dladm modify-bridge` command to enable TRILL.	"How to Modify the Protection Type for a Bridge" on page 221

Task	Description	For Instructions
Add a link to a bridge.	Use the `dladm add-bridge` command to add one or more links to an existing bridge.	"How to Add One or More Links to an Existing Bridge" on page 221
Remove links from a bridge.	Use the `dladm remove-bridge` command to remove links from a bridge. You cannot delete a bridge until all of its links are removed.	"How to Remove Links From a Bridge" on page 222
Delete a bridge from the system.	Use the `dladm delete-bridge` command to delete a bridge from the system.	"How to Delete a Bridge From the System" on page 223

▼ How to View Information About Configured Bridges

This procedure shows how to use the `dladm show-bridge` command with various options to show different kinds of information about configured bridges.

For more information about the `dladm show-bridge` command options, see the `dladm(1M)` man page.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 View information about a bridge or all configured bridges.

- View the list of bridges.

  ```
  # dladm show-bridge
  ```

- Show link-related status for the bridge.

  ```
  # dladm show-bridge -l bridge-name
  ```

- Show statistics for the bridge.

  ```
  # dladm show-bridge -s bridge-name
  ```

  Note – The names and definitions of the statistics reported are subject to change.

- Show link-related statistics for the bridge.

  ```
  # dladm show-bridge -ls bridge-name
  ```

- Show kernel forwarding entries for the bridge.

  ```
  # dladm show-bridge -f bridge-name
  ```

- Show TRILL information about the bridge.

  ```
  # dladm show-bridge -t bridge-name
  ```

Example 11–1 Viewing Bridge Information

The following are examples of using the `dladm show-bridge` command with various options.

- The following shows information about all bridges that are configured on the system:

  ```
  # dladm show-bridge
  BRIDGE       PROTECT ADDRESS           PRIORITY DESROOT
  goldengate   stp     32768/8:0:20:bf:f 32768    8192/0:d0:0:76:14:38
  baybridge    stp     32768/8:0:20:e5:8 32768    8192/0:d0:0:76:14:38
  ```

- The following `dladm show-bridge -l` command shows link-related status information for a single bridge instance, `tower`. To view configured parameters, use the `dladm show-linkprop` command instead.

  ```
  # dladm show-bridge -l tower
  LINK         STATE        UPTIME   DESROOT
  hme0         forwarding   117      8192/0:d0:0:76:14:38
  qfe1         forwarding   117      8192/0:d0:0:76:14:38
  ```

- The following `dladm show-bridge -s` command shows statistics for the specified bridge, `terabithia`:

  ```
  # dladm show-bridge -s terabithia
  BRIDGE       DROPS        FORWARDS
  terabithia   0            302
  ```

- The following `dladm show-bridge -ls` command shows statistics for all of the links on the specified bridge, `london`:

  ```
  # dladm show-bridge -ls london
  LINK         DROPS     RECV      XMIT
  hme0         0         360832    31797
  qfe1         0         322311    356852
  ```

- The following `dladm show-bridge -f` command shows kernel forwarding entries for the specified bridge, `avignon`:

  ```
  # dladm show-bridge -f avignon
  DEST              AGE     FLAGS  OUTPUT
  8:0:20:bc:a7:dc   10.860  --     hme0
  8:0:20:bf:f9:69   --      L      hme0
  8:0:20:c0:20:26   17.420  --     hme0
  8:0:20:e5:86:11   --      L      qfe1
  ```

- The following `dladm show-bridge -t` command shows TRILL information about the specified bridge, `key`:

  ```
  # dladm show-bridge -t key
  NICK FLAGS LINK        NEXTHOP
  38628 --   london      56:db:46:be:b9:62
  58753 L    --          --
  ```

How to View Configuration Information About Bridge Links

The `dladm show-link` output includes a `BRIDGE` field. If a link is a member of a bridge, this field identifies the name of the bridge of which it is a member. This field is shown by default. For links that are not part of a bridge, the field is blank if the `-p` option is used. Otherwise, the field shows `--`.

The bridge observability node also appears in the `dladm show-link` output as a separate link. For this node, the existing `OVER` field lists the links that are members of the bridge.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **View configuration information about any link that is a member of a bridge.**

   ```
   # dladm show-link [-p]
   ```

   The `-p` option produces output in a parseable format.

How to Create a Bridge

This procedure shows how to use STP to create a bridge, which is the default. For more information about bridge creation options, see the description of `dladm create-bridge` in the `dladm`(1M) man page.

Note – To use TRILL to create a bridge instead, add `-P trill` to the `dladm create-bridge` command line, or use the `dladm modify-bridge` command to enable TRILL.

The `dladm create-bridge` command creates a bridge instance and optionally assigns one or more network links to the new bridge. Because no bridge instances are present on the system by default, Oracle Solaris does not bridge between network links by default.

To bridge between links, you must create at least one bridge instance. Each bridge instance is separate. Bridges do not include a forwarding connection between them, and a link is a member of at most one bridge.

bridge-name is an arbitrary string that must be a legal SMF service instance name. This name is a FMRI component that has no escape sequences, which means that whitespace, ASCII control characters, and the following characters cannot be present:

```
; / ? : @ & = + $ , % < > # "
```

The name `default` is reserved, as are all names beginning with the `SUNW` string. Names that have trailing digits are reserved for the creation of "observability devices." Because of the use of the observability devices, the names of legal bridge instances are further constrained to be a legal `dlpi(7P)` name. The name must begin and end with an alphabetic character or an underscore character. The rest of the name can contain alphanumeric and underscore characters.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Create the bridge.**

```
# dladm create-bridge [-l link]... bridge-name
```

The `-l` *link* option adds a link to the bridge. Note that if any of the specified links cannot be added, the command fails and the bridge is not created.

The following example shows how to create the `brooklyn` bridge by connecting the `hme0` and `qfe1` links:

```
# dladm create-bridge -l hme0 -l qfe1 brooklyn
```

How to Modify the Protection Type for a Bridge

This procedure shows how to use the `dladm modify-bridge` command to modify the protection type from STP to TRILL or from TRILL to STP.

- **Modify the protection type for a bridge.**

```
# dladm modify-bridge -P protection-type bridge-name
```

The `-P` *protection-type* option specifies which protection type to use. By default, the protection type is STP (`-P stp`). To use the TRILL protection type instead, use the `-P trill` option.

The following example shows how to modify the protection type for the `brooklyn` bridge from the default STP to TRILL:

```
# dladm modify-bridge -P trill brooklyn
```

How to Add One or More Links to an Existing Bridge

This procedure shows how to add one or more links to a bridge instance.

A link can be a member of at most one bridge. So, if you want to move a link from one bridge instance to another, you must first remove the link from the current bridge before adding it to another one.

The links that are assigned to a bridge cannot be VLANs, VNICs, or tunnels. Only links that would be acceptable as part of an aggregation, or links that are aggregations themselves can be assigned to a bridge.

Links that are assigned to a bridge must all have the same MTU value. Note that Oracle Solaris allows you to change the MTU value on an existing link. In this case, the bridge instance goes into maintenance state until you remove or change the assigned links so that the MTU values match before you restart the bridge.

The links that are assigned to the bridge must be an Ethernet type, which includes 802.3 and 802.11 media.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Add a new link to the existing bridge.**

```
# dladm add-bridge -l new-link  bridge-name
```

The following example shows how to add the `qfe2` link to the existing bridge `rialto`:

```
# dladm add-bridge -l qfe2 rialto
```

▼ How to Remove Links From a Bridge

This procedure shows how to remove one or more links from a bridge instance. Use this procedure if you intend to delete a bridge. Before the bridge can be deleted, all of its links must first be removed.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Remove the links from the bridge.**

```
# dladm remove-bridge [-l link]... bridge-name
```

The following example shows how to remove the `hme0`, `qfe1`, and `qfe2` links from the bridge `charles`:

```
# dladm remove-bridge -l hme0 -l qfe1 -l qfe2 charles
```

How to Delete a Bridge From the System

This procedure shows how to delete a bridge instance. Before you can delete a bridge, you must first deactivate any attached links by running the `dladm remove-bridge` command. See "How to Remove Links From a Bridge" on page 222.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Delete the bridge from the system.

```
# dladm delete-bridge bridge-name
```

The following example shows how to first remove the `hme0`, `qfe1`, and `qfe2` links from the `coronado` bridge, and then remove the bridge itself from the system:

```
# dladm remove-bridge -l hme0 -l qfe1 -l qfe2 coronado
# dladm delete-bridge coronado
```

◆ ◆ ◆ CHAPTER 12

Administering Link Aggregations

This chapter describes procedures to configure and maintain link aggregations. The procedures include steps that avail of new features such as support for flexible link names.

Overview of Link Aggregations

Oracle Solaris supports the organization of network interfaces into link aggregations. A *link aggregation* consists of several interfaces on a system that are configured together as a single, logical unit. Link aggregation, also referred to as *trunking*, is defined in the IEEE 802.3ad Link Aggregation Standard (`http://www.ieee802.org/3/index.html`).

The IEEE 802.3ad Link Aggregation Standard provides a method to combine the capacity of multiple full-duplex Ethernet links into a single logical link. This link aggregation group is then treated as though it were, in fact, a single link.

The following are features of link aggregations:

- **Increased bandwidth** – The capacity of multiple links is combined into one logical link.
- **Automatic failover/failback** – Traffic from a failed link is failed over to working links in the aggregation.
- **Load balancing** – Both inbound and outbound traffic is distributed according to user selected load-balancing policies, such as source and destination MAC or IP addresses.
- **Support for redundancy** – Two systems can be configured with parallel aggregations.
- **Improved administration** – All interfaces are administered as a single unit.
- **Less drain on the network address pool** – The entire aggregation can be assigned one IP address.

Link Aggregation Basics

The basic link aggregation topology involves a single aggregation that contains a set of physical interfaces. You might use the basic link aggregation in the following situations:

- For systems that run an application with distributed heavy traffic, you can dedicate an aggregation to that application's traffic.
- For sites with limited IP address space that nevertheless require large amounts of bandwidth, you need only one IP address for a large aggregation of interfaces.
- For sites that need to hide the existence of internal interfaces, the IP address of the aggregation hides its interfaces from external applications.

Figure 12–1 shows an aggregation for a server that hosts a popular web site. The site requires increased bandwidth for query traffic between Internet customers and the site's database server. For security purposes, the existence of the individual interfaces on the server must be hidden from external applications. The solution is the aggregation `aggr1` with the IP address `192.168.50.32`. This aggregation consists of three interfaces,`bge0` through `bge2`. These interfaces are dedicated to sending out traffic in response to customer queries. The outgoing address on packet traffic from all the interfaces is the IP address of `aggr1`, `192.168.50.32`.

FIGURE 12–1 Basic Link Aggregation Topology

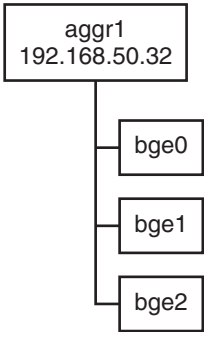
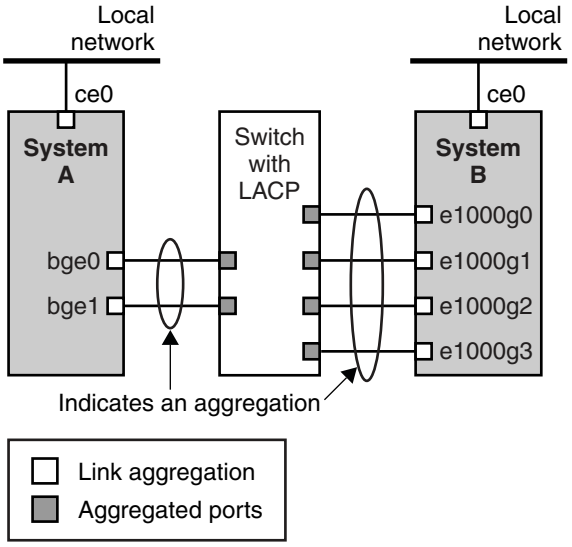


Figure 12–2 depicts a local network with two systems, and each system has an aggregation configured. The two systems are connected by a switch. If you need to run an aggregation through a switch, that switch must support aggregation technology. This type of configuration is particularly useful for high availability and redundant systems.

In the figure, System A has an aggregation that consists of two interfaces, `bge0` and `bge1`. These interfaces are connected to the switch through aggregated ports. System B has an aggregation of four interfaces, `e1000g0` through `e1000g3`. These interfaces are also connected to aggregated ports on the switch.

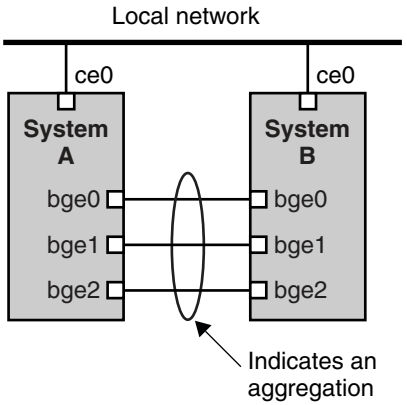
FIGURE 12–2 Link Aggregation Topology With a Switch



Back-to-Back Link Aggregations

The back-to-back link aggregation topology involves two separate systems that are cabled directly to each other, as shown in the following figure. The systems run parallel aggregations.

FIGURE 12–3 Basic Back-to-Back Aggregation Topology



In this figure, device `bge0` on System A is directly linked to `bge0` on System B, and so on. In this way, Systems A and B can support redundancy and high availability, as well as high-speed communications between both systems. Each system also has interface `ce0` configured for traffic flow within the local network.

The most common application for back-to-back link aggregations is mirrored database servers. Both servers need to be updated together and therefore require significant bandwidth, high-speed traffic flow, and reliability. The most common use of back-to-back link aggregations is in data centers.

Policies and Load Balancing

If you plan to use a link aggregation, consider defining a policy for outgoing traffic. This policy can specify how you want packets to be distributed across the available links of an aggregation, thus establishing load balancing. The following are the possible layer specifiers and their significance for the aggregation policy:

- **L2** – Determines the outgoing link by hashing the MAC (L2) header of each packet
- **L3** – Determines the outgoing link by hashing the IP (L3) header of each packet
- **L4** – Determines the outgoing link by hashing the TCP, UDP, or other ULP (L4) header of each packet

Any combination of these policies is also valid. The default policy is L4. For more information, refer to the `dladm`(1M) man page.

Aggregation Mode and Switches

If your aggregation topology involves connection through a switch, you must note whether the switch supports the *link aggregation control protocol (LACP)*. If the switch supports LACP, you must configure LACP for the switch and the aggregation. However, you can define one of the following *modes* in which LACP is to operate:

- **Off mode** – The default mode for aggregations. LACP packets, which are called *LACPDUs* are not generated.
- **Active mode** – The system generates LACPDUs at regular intervals, which you can specify.
- **Passive mode** – The system generates an LACPDU only when it receives an LACPDU from the switch. When both the aggregation and the switch are configured in passive mode, they cannot exchange LACPDUs.

See the `dladm`(1M) man page and the switch manufacturer's documentation for syntax information.

Requirements for Link Aggregations

Your link aggregation configuration is bound by the following requirements:

- You must use the `dladm` command to configure aggregations.
- An interface that has been created cannot become a member of an aggregation.
- All interfaces in the aggregation must run at the same speed and in full-duplex mode.
- You must set the value for MAC addresses to "true" in the EEPROM parameter `local-mac-address?` For instructions, refer to How to Ensure That the MAC Address of an Interface Is Unique.

Certain devices do not fulfill the requirement of the IEEE 802.3ad Link Aggregation Standard to support link state notification. This support must exist in order for a port to attach to an aggregation or to detach from an aggregation. Devices that do not support link state notification can be aggregated only by using the `-f` option of the `dladm create-aggr` command. For such devices, the link state is always reported as `UP`. For information about the use of the `-f` option, see "How to Create a Link Aggregation" on page 230.

Flexible Names for Link Aggregations

Flexible names can be assigned to link aggregations. Any meaningful name can be assigned to a link aggregation. For more information about flexible or customized names, see "Network Devices and Datalink Names" on page 26. Previous Oracle Solaris releases identify a link aggregation by the value of a *key* that you assign to the aggregation. For an explanation of this method, see Overview of Link Aggregations. Although that method continues to be valid, preferably, you should use customized names to identify link aggregations.

Similar to all other datalink configurations, link aggregations are administered with the `dladm` command.

Administering Link Aggregations (Task Map)

The following table links to procedures for administering link aggregations.

Tasks	Description	For Instructions
Create an aggregation.	Configure an aggregation consisting of multiple datalinks.	"How to Create a Link Aggregation" on page 230
Modify an aggregation.	Change an aggregations policy and mode.	"How to Modify an Aggregation" on page 232

Tasks	Description	For Instructions
Modify links that make up an aggregation.	Increase or decrease the number of datalinks that underly an aggregation.	"How to Add a Link to an Aggregation" on page 233 or "How to Remove a Link From an Aggregation" on page 234
Delete an aggregation.	Completely remove a link aggregation from your network configuration.	"How to Delete an Aggregation" on page 234

▼ How to Create a Link Aggregation

Before You Begin

Note – Link aggregation only works on full-duplex, point-to-point links that operate at identical speeds. Make sure that the interfaces in your aggregation conform to this requirement.

If you are using a switch in your aggregation topology, make sure that you have done the following on the switch:

- Configured the ports to be used as an aggregation
- If the switch supports LACP, configured LACP in either active mode or passive mode

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Display the network datalink information.

```
# dladm show-link
```

3 Make sure that the link over which you are creating the aggregation is not opened by any application.

For example, if the IP interface over the link is created, then remove the interface.

a. To determine whether a link is being used by any application, examine the output of either the `dladm show-link` syntax or the `ipadm show-if` syntax.

- If a datalink is in use, then `STATE` field in the output of the `dladm show-link` will indicate that the link is up. Thus:

```
# dladm show-link
LINK     CLASS     MTU     STATE     BRIDGE   OVER
qfe3     phys      1500    up        --       --
```

- If the datalink is in use, then the IP interface over that link will be included in the output of the `ipadm show-if` syntax. Thus:

```
# ipadm show-if
IFNAME       CLASS        STATE     ACTIVE     OVER
lo0          loopback     ok        yes        --
qfe3         ip           ok        no         --
```

Note – Even if the output displays an `offline` status, the datalink is still in use because an IP interface exists over the link.

b. To remove the IP interface, type the following command:

```
# ipadm delete-ip interface
```

where

interface Specifies the IP interface that is created over the link.

4 Create a link aggregation.

```
# dladm create-aggr [-f] -l link1 -l link2 [...] aggr
```

`-f` Forces the creation of the aggregation. Use this option when you are attempting to aggregate devices that do not support link state notification.

linkn Specifies the datalinks that you want to aggregate.

aggr Specifies the name that you want to assign to the aggregation.

5 Create an IP interface over the aggregation.

```
# ipadm create-ip interface
```

6 Configure the IP interface with a valid IP address.

```
# ipadm create-addr interface -T static -a IP-address addrobj
```

where *interface* should take the name of the aggregation and *addrobj* uses the naming convention *interface*/*user-defined-string*.

7 Check the status of the aggregation you just created.

The aggregation's state should be `UP`.

```
# dladm show-aggr
```

Example 12–1 Creating a Link Aggregation

This example shows the commands that are used to create a link aggregation with two datalinks, subvideo0 and subvideo1. The configuration is persistent across system reboots.

```
# dladm show-link
LINK          CLASS     MTU     STATE   BRIDGE     OVER
subvideo0     phys      1500    up      --         ----
subvideo1     phys      1500    up46    --         ----

# ipadm delete-ip subvideo0
# ipadm delete-ip subvideo1
# dladm create-aggr -l subvideo0 -l subvideo1 video0
# ipadm create-ip video0
# ipadm create-addr -T static -a 10.8.57.50/24 video/v4
# dladm show-aggr
LINK      POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER  FLAGS
video0    L4      auto            off            short      -----
```

When you display link information, the link aggregation is included in the list.

```
# dladm show-link
LINK          CLASS     MTU     STATE   BRIDGE     OVER
subvideo0     phys      1500    up      --         ----
subvideo1     phys      1500    up      --         ----
video0        aggr      1500    up      --         subvideo0, subvideo1
```

How to Modify an Aggregation

This procedure shows how to make the following changes to an aggregation definition:

- Modifying the policy for the aggregation
- Changing the mode for the aggregation

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Modify the policy of the aggregation.

 # dladm modify-aggr -P *policy-key aggr*

policy-key Represents one or more of the policies L2, L3, and L4, as explained in "Policies and Load Balancing" on page 228.

aggr Specifies the aggregation whose policy you want to modify.

3 Modify the LACP mode of the aggregation.

 # dladm modify-aggr -L *LACP-mode* **-T** *timer-value aggr*

`-L` *LACP-mode* Indicates the LACP mode in which the aggregation is to run. The values are `active`, `passive`, and `off`. If the switch runs LACP in passive mode, be sure to configure active mode for your aggregation.

`-T` *timer-value* Indicates the LACP timer value, either `short` or `long`.

Example 12–2 Modifying a Link Aggregation

This example shows how to modify the policy of aggregation `video0` to L2 and then turn on active LACP mode.

```
# dladm modify-aggr -P L2 video0
# dladm modify-aggr -L active -T short video0
# dladm show-aggr
LINK      POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER  FLAGS
video0    L2      auto            active         short      -----
```

▼ How to Add a Link to an Aggregation

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Ensure that the link you want to add has no IP interface that is plumbed over the link.**

   # ipadm delete-ip *interface*

3. **Add the link to the aggregation.**

   # dladm add-aggr -l *link* **[-l** *link***] [...]** *aggr*

   where *link* represents a datalink that you are adding to the aggregation.

4. **Perform other tasks to modify the entire link aggregation configuration after more datalinks are added.**

   For example, in the case of a configuration that is illustrated in Figure 12–3, you might need to add or modify cable connections and reconfigure switches to accommodate the additional datalinks. Refer to the switch documentation to perform any reconfiguration tasks on the switch.

Example 12–3 Adding a Link to an Aggregation

This example shows how to add a link to the aggregation `video0`.

```
# dladm show-link
LINK          CLASS     MTU     STATE     BRODGE   OVER
subvideo0     phys      1500    up        --       ----
subvideo1     phys      1500    up        --       ----
video0        aggr      1500    up        --       subvideo0, subvideo1
net3          phys      1500    unknown   --       ----

# ipadm delete-ip video0
# dladm add-aggr -l net3 video0
# dladm show-link
```

```
LINK          CLASS     MTU     STATE     BRIDGE   OVER
subvideo0     phys      1500    up        --       ----
subvideo1     phys      1500    up        --       ----
video0        aggr      1500    up        --       subvideo0, subvideo1, net3
net3          phys      1500    up        --       ----
```

How to Remove a Link From an Aggregation

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Remove a link from the aggregation.**

   ```
   # dladm remove-aggr -l link aggr-link
   ```

Example 12–4 Removing a Link From an Aggregation

This example shows how to remove a link from the aggregation `video0`.

```
dladm show-link
LINK          CLASS     MTU     STATE     OVER
subvideo0     phys      1500    up        --       ----
subvideo1     phys      1500    up        --       ----
video0        aggr      1500    up        --       subvideo0, subvideo1, net3
net3          phys      1500    up        --       ----

# dladm remove-aggr -l net3 video0
# dladm show-link
LINK          CLASS     MTU     STATE     BRIDGE   OVER
subvideo0     phys      1500    up        --       ----
subvideo1     phys      1500    up        --       ----
video0        aggr      1500    up        --       subvideo0, subvideo1
net3          phys      1500    unknown   --       ----
```

How to Delete an Aggregation

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Delete the IP interface that is configured over the aggregation.**

   ```
   # ipadm delete-ip IP-aggr
   ```

   where *IP-aggr* is the IP interface over the link aggregation.

3 Delete the link aggregation.

```
# dladm delete-aggr aggr
```

Example 12–5 Deleting an Aggregation

This example deletes the aggregation `video0`. The deletion is persistent.

```
# ipadm delete-ip video0
# dladm delete-aggr video0
```

◆ ◆ ◆ CHAPTER 13

Administering VLANs

This chapter describes procedures to configure and maintain virtual local area networks (VLANs). The procedures include steps that avail of features such as support for flexible link names.

Administering Virtual Local Area Networks

A *virtual local area network (VLAN)* is a subdivision of a local area network at the datalink layer of the TCP/IP protocol stack. You can create VLANs for local area networks that use switch technology. By assigning groups of users to VLANs, you can improve network administration and security for the entire local network. You can also assign interfaces on the same system to different VLANs.

Consider dividing your local network into VLANs if you need to do the following:

- Create a logical division of workgroups.

  For example, suppose all hosts on a floor of a building are connected on one switched-based local network. You could create a separate VLAN for each workgroup on the floor.
- Enforce differing security policies for the workgroups.

  For example, the security needs of a Finance department and an Information Technologies department are quite different. If systems for both departments share the same local network, you could create a separate VLAN for each department. Then, you could enforce the appropriate security policy on a per-VLAN basis.
- Split workgroups into manageable broadcast domains.

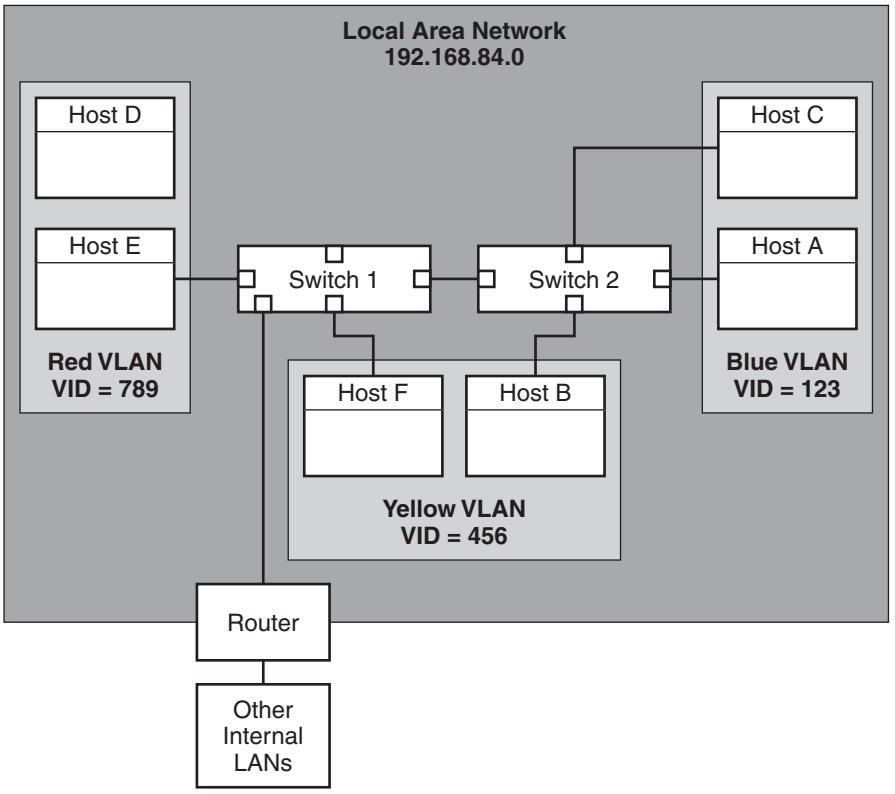
  The use of VLANs reduces the size of broadcast domains and improves network efficiency.

Overview of VLAN Topology

Switched LAN technology enables you to organize the systems on a local network into VLANs. Before you can divide a local network into VLANs, you must obtain switches that support VLAN technology. You can configure all ports on a switch to serve a single VLAN or multiple VLANs, depending on the VLAN topology design. Each switch manufacturer has different procedures for configuring the ports of a switch.

The following figure shows a local area network that has the subnet address `192.168.84.0`. This LAN is subdivided into three VLANs, Red, Yellow, and Blue.

FIGURE 13–1 Local Area Network With Three VLANs

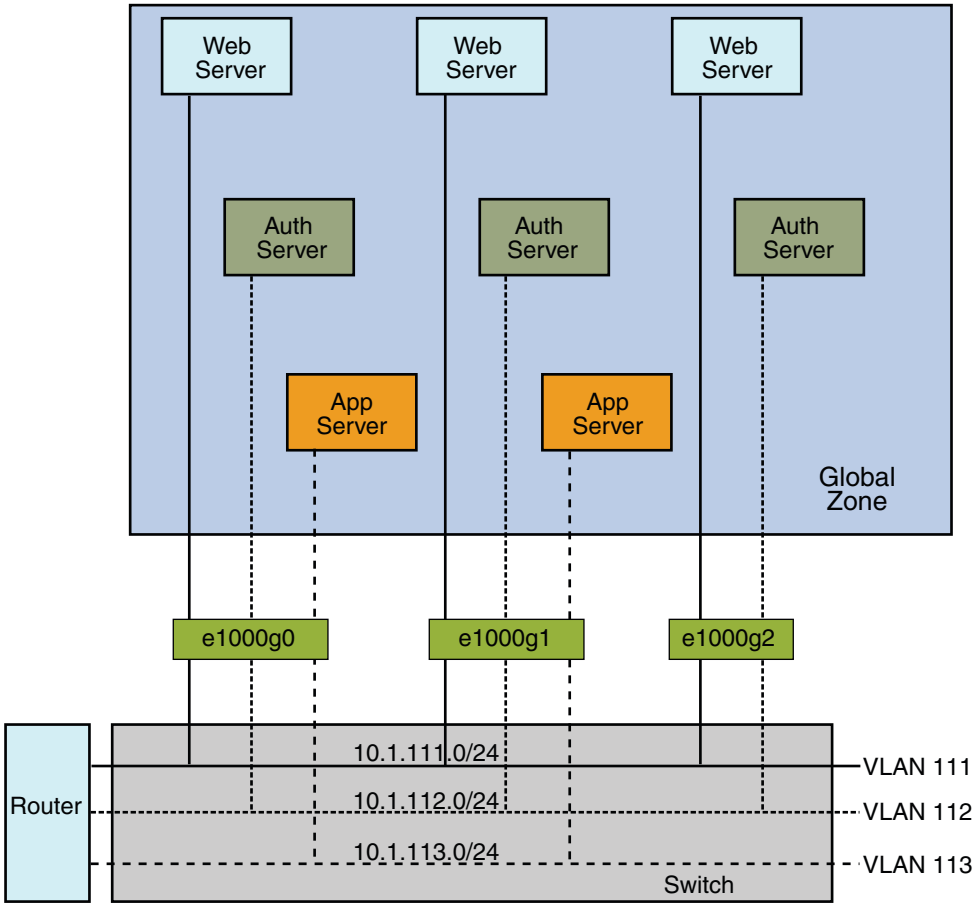


Connectivity on LAN `192.168.84.0` is handled by Switches 1 and 2. The Red VLAN contains systems in the Accounting workgroup. The Human Resources workgroup's systems are on the Yellow VLAN. Systems of the Information Technologies workgroup are assigned to the Blue VLAN.

Consolidating the Network by Using VLANs

VLANs on zones allow you to configure multiple virtual networks within a single network unit such as a switch. Consider the following illustration of a system with three physical NICs:

FIGURE 13–2 System With Multiple VLANs



Without VLANs, you would configure different systems to perform specific functions and connect these systems to separate networks. For example, web servers would be connected to one LAN, authentication servers to another, and application servers to a third network. With VLANs and zones, you can collapse all eight systems and configure them as zones in a single system. Then you use VLAN tags, or VLAN IDs (VIDs) to assign a VLAN to each set of zones that performs the same functions. The information provided in the figure can be tabulated as follows:

Function	Zone Name	VLAN Name	VID	IP Address	NIC
Web server	webzone1	web1	111	10.1.111.0	e1000g0
Authentication server	authzone1	auth1	112	10.1.112.0	e1000g0
Application server	appzone1	app1	113	10.1.113.0	e1000g0
Web server	webzone2	web2	111	10.1.111.0	e1000g1
Authentication server	authzone2	auth2	112	10.1.112.0	e1000g1
Application server	appzone2	app2	113	10.1.113.0	e1000g1
Web server	webzone3	web3	111	10.1.111.0	e1000g2
Authentication server	authzone3	auth3	112	10.1.112.0	e1000g2

To create the configuration shown in the figure, refer to Example 13–1.

Meaningful Names for VLANs

In Oracle Solaris, you can assign meaningful names to VLAN interfaces. VLAN names consist of a link name and the VLAN ID number (VID), such as `sales0` You should assign customized names when you create VLANs. For more information about customized names, see "Network Devices and Datalink Names" on page 26. For more information about valid customized names, see "Rules for Valid Link Names" on page 30.

VLAN Administration (Task Map)

The following table links you to different tasks to administer VLANs.

Task	Description	For Instructions
Plan a virtual local area network (VLAN).	Perform required planning tasks prior to creating a VLAN.	"How to Plan a VLAN Configuration" on page 241
Configure a VLAN	Create VLANs on your network.	"How to Configure a VLAN" on page 242
Configure a VLAN on an aggregation.	Deploy combined technologies that us both VLANs and link aggregations.	"How to Configure VLANs Over a Link Aggregation" on page 245

Task	Description	For Instructions
Display VLAN information.	Obtain information about a VLAN and its components.	"How to Display VLAN Information" on page 246
Remove a VLAN.	Select a VLAN to remove from multiple VLANs configured over a datalink.	"How to Remove a VLAN" on page 247

Planning for VLANs on a Network

Use the following procedure to plan for VLANs on your network.

▼ How to Plan a VLAN Configuration

1. **Examine the local network topology and determine where subdivision into VLANs is appropriate.**

   For a basic example of such a topology, refer to Figure 13–1.

2. **Create a numbering scheme for the VIDs, and assign a VID to each VLAN.**

   Note – A VLAN numbering scheme might already exist on the network. If so, you must create VIDs within the existing VLAN numbering scheme.

3. **On each system, determine which interfaces will be members of a particular VLAN.**

   a. **Determine which interfaces are configured on a system.**

      ```
      # dladm show-link
      ```

   b. **Identify which VID will be associated with each datalink on the system.**

   c. **Create the VLAN by using the `dladm create-vlan` command.**

4. **Check the connections of the interfaces to the network's switches.**

   Note the VID of each interface and the switch port where each interface is connected.

5. **Configure each port of the switch with the same VID as the interface to which it is connected.**

   Refer to the switch manufacturer's documentation for configuration instructions.

Configuring VLANs

The following procedure shows how to create and configure a VLAN. In Oracle Solaris, all Ethernet devices can support VLANs. However, some restrictions exist with certain devices. For these exceptions, refer to "VLANs on Legacy Devices" on page 246.

▼ How to Configure a VLAN

Before You Begin Data links must already be configured on your system before you can create VLANs. See "How to Configure an IP Interface" on page 171.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Determine the types of links that are in use in your system.

```
# dladm show-link
```

3 Create a VLAN link over a datalink.

```
# dladm create-vlan -l link -v VID vlan-link
```

link Specifies the link on which the VLAN interface is being created.

VID Indicates the VLAN ID number

vlan-link Specifies the name of the VLAN, which can also be an administratively-chosen name.

4 Verify the VLAN configuration.

```
# dladm show-vlan
```

5 Create an IP interface over the VLAN.

```
# ipadm create-ip interface
```

where *interface* uses the VLAN name.

6 Configure the IP interface with an IP address.

```
# ipadm create-addr -T static -a IP-address addrobj
```

where *addrobj* uses the naming convention *interface*/*user-defined-string*.

Example 13–1 Configuring a VLAN

This example creates the VLAN configuration that is illustrated in Figure 13–2. This example assumes that you have already configured the different zones in the system. For more

information about configuring zones, see Part II, "Oracle Solaris Zones," in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*.

```
global# dladm show-link
LINK        CLASS     MTU     STATE     BRIDGE     OVER
e1000g0     phys      1500    up        --         --
e1000g1     phys      1500    up        --         --
e1000g2     phys      1500    up        --         --

global# dladm create-vlan -l e1000g0 -v 111 web1
global# dladm create-vlan -l e1000g0 -v 112 auth1
global# dladm create-vlan -l e1000g0 -v 113 app1
global# dladm create-vlan -l e1000g1 -v 111 web2
global# dladm create-vlan -l e1000g1 -v 112 auth2
global# dladm create-vlan -l e1000g1 -v 113 app2
global# dladm create-vlan -l e1000g2 -v 111 web3
global# dladm create-vlan -l e1000g2 -v 112 auth3

global# dladm show-vlan
LINK      VID     OVER        FLAGS
web1      111     e1000g0     ----
auth1     112     e1000g0     ----
app1      113     e1000g0     ----
web2      111     e1000g1     ----
auth2     112     e1000g1     ----
app2      113     e1000g1     ----
web3      111     e1000g2     ----
auth3     113     e1000g2     ----
```

When link information is displayed, the VLANs are included in the list.

```
global# dladm show-link
LINK        CLASS    MTU      STATE     BRIDGE     OVER
e1000g0     phys     1500     up        --         --
e1000g1     phys     1500     up        --         --
e1000g2     phys     1500     up        --         --
web1        vlan     1500     up        --         e1000g0
auth1       vlan     1500     up        --         e1000g0
app1        vlan     1500     up        --         e1000g0
web2        vlan     1500     up        --         e1000g1
auth2       vlan     1500     up        --         e1000g1
app2        vlan     1500     up        --         e1000g1
web3        vlan     1500     up        --         e1000g2
auth3       vlan     1500     up        --         e1000g2
```

You assign the VLANs to their respective zones. For example, when you check for network information about individual zones, data similar to the following would be displayed for each zone:

```
global# zonecfg -z webzone1 info net
net:
     address not specified
     physical: web1

global# zonecfg -z authzone1 info net
```

```
net:
     address not specified
     physical: auth1

global# zonecfg -z appzone2 info net
net:
     address not specified
     physical: app2
```

The value of the property `physical` indicates the VLAN that is set for the given zone.

You log in to each non-global zone to configure the VLAN with an IP address.

In `webzone1`:

```
webzone1# ipadm create-ip web1
webzone1# ipadm create-addr -T static -a 10.1.111.0/24 web1/v4
```

In `webzone2`:

```
webzone2# ipadm create-ip web2
webzone2# ipadm create-addr -T static -a 10.1.111.0/24 web2/v4
```

In `webzone3`:

```
webzone3# ipadm create-ip web3
webzone3# ipadm create-addr -T static -a 10.1.111.0/24 web3/v4
```

In `authzone1`:

```
authzone1# ipadm create-ip auth1
authzone1# ipadm create-addr -T static -a 10.1.112.0/24 auth1/v4
```

In `authzone2`:

```
authzone2# ipadm create-ip auth2
autzone2# ipadm create-addr -T static -a 10.1.112.0/24 auth2/v4
```

In `authzone3`:

```
authzone3# ipadm create-ip auth3
authzone3# ipadm create-addr -T static -a 10.1.112.0/24 auth3/v4
```

In `appzone1`:

```
appzone1# ipadm create-ip app1
appzone1# ipadm create-addr -T static -a 10.1.113.0/24 app1/v4
```

In `appzone2`:

```
appzone2# ipadm create-ip app2
appzone2# ipadm create-addr -T static -a 10.1.113.0/24 app2/v4
```

How to Configure VLANs Over a Link Aggregation

In the same manner as configuring VLANs over an interface, you can also create VLANs on a link aggregation. Link aggregations are described in Chapter 12, "Administering Link Aggregations." This section combines configuring VLANs and link aggregations.

Before You Begin Create the link aggregation first and configure it with a valid IP address. To create link aggregations, refer to "How to Create a Link Aggregation" on page 230.

1. **List the aggregations that are configured in the system.**

   ```
   # dladm show-link
   ```

2. **For every VLAN that you want to create over the aggregation, issue the following command.**

   ```
   # dladm create-vlan -l link -v VID vlan-link
   ```

   where

   link Specifies the link on which the VLAN interface is being created. In this specific case, the link refers to the link aggregation.

   VID Indicates the VLAN ID number

   vlan-link Specifies the name of the VLAN, which can also be an administratively-chosen name.

3. **Create IP interfaces over the VLANs.**

   ```
   # ipadm create-ip interface
   ```

   where *interface* uses the VLAN name.

4. **Configure IP interfaces over the VLANs with valid IP addresses.**

   ```
   # ipadm create-addr -T static -a IP-address addrobj
   ```

   where *addrobj* must follow the naming convention *vlan-int*/*user-defined-string*

Example 13–2 Configuring Multiple VLANs Over a Link Aggregation

In this example, two VLANs are configured on a link aggregation. The VLANs are assigned VIDs 193 and 194, respectively.

```
# dladm show-link
LINK          CLASS     MTU     STATE     BRIDGE     OVER
subvideo0     phys      1500    up        --         ----
subvideo1     phys      1500    up        --         ----
video0        aggr      1500    up        --         subvideo0, subvideo1

# dladm create-vlan -l video0 -v 193 salesregion1
# dladm create-vlan -l video0 -v 194 salesregion2
```

```
# ipadm create-ip salesregion1
# ipadm create-ip salesregion2

# ipadm create-addr -T static -a 192.168.10.5/24 salesregion1/v4static
# ipadm create-addr -T static -a 192.168.10.25/24 salesregion2/v4static
```

VLANs on Legacy Devices

Certain legacy devices handle only packets whose maximum frame size is 1514 bytes. Packets whose frame sizes exceed the maximum limit are dropped. For such cases, follow the same procedure listed in "How to Configure a VLAN" on page 242. However, when creating the VLAN, use the `-f` option to force the creation of the VLAN.

The general steps to perform are as follows:

1. Create the VLAN with the `-f` option.

   ```
   # dladm create-vlan -f -l link -v VID [vlan-link]
   ```

2. Set a lower size for the maximum transmission unit (`MTU`), such as 1496 bytes.

   ```
   # dladm set-linkprop -p default_mtu=1496 vlan-link
   ```

   The lower `MTU` value allows space for the link layer to insert the VLAN header prior to transmission.

3. Perform the same step to set the same lower value for the `MTU` size of each node in the VLAN.

   For more information about changing link property values, refer to "Configuration of Datalinks (Tasks)" on page 147.

Performing Other Administrative Tasks on VLANs

This section describes the usage of new `dladm` subcommands for other VLAN tasks. These `dladm` commands also work with link names.

▼ How to Display VLAN Information

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Display VLAN information.

```
# dladm show-vlan [vlan-link]
```

If you do not specify a VLAN link, the command displays information about all configured VLANs.

Example 13–3 Displaying VLAN Information

The following example is based on the system with multiple VLANs illustrated by Figure 13–2 and shows the available VLANs in the system.

```
# dladm show-vlan
LINK      VID     OVER        FLAGS
web1      111     e1000g0     ----
auth1     112     e1000g0     ----
app1      113     e1000g0     ----
web2      111     e1000g1     ----
auth2     112     e1000g1     ----
app2      113     e1000g1     ----
web3      111     e1000g2     ----
auth3     113     e1000g2     ----
```

Configured VLANs also appear when you issue the `dladm show-link` command. In the command output, the VLANs are appropriately identified in the `CLASS` column.

```
# dladm show-link
LINK        CLASS     MTU      STATE     BRIDGE     OVER
e1000g0     phys      1500     up        --         --
e1000g1     phys      1500     up        --         --
e1000g2     phys      1500     up        --         --
web1        vlan      1500     up        --         e1000g0
auth1       vlan      1500     up        --         e1000g0
app1        vlan      1500     up        --         e1000g0
web2        vlan      1500     up        --         e1000g1
auth2       vlan      1500     up        --         e1000g1
app2        vlan      1500     up        --         e1000g1
web3        vlan      1500     up        --         e1000g2
auth3       vlan      1500     up        --         e1000g2
```

▼ How to Remove a VLAN

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Determine which VLAN you want to remove.**

   ```
   # dladm show-vlan
   ```

3. **Unplumb the VLAN's IP interface.**

   ```
   # ipadm delete-ip vlan-interface
   ```

   where *vlan-interface* is the IP interface that is configured over the VLAN.

Note – You cannot remove a VLAN that is currently in use.

4 Remove the VLAN by performing one of the following steps:

- To delete the VLAN temporarily, use the `-t` option as follows:

  `# dladm delete-vlan -t` *vlan*
- To make the deletion persist, perform the following:

  a. Remove the VLAN.

  `# dladm delete-vlan` *vlan*

Example 13–4 Removing a VLAN

```
# dladm show-vlan
LINK      VID     OVER        FLAGS
web1      111     e1000g0     ----
auth1     112     e1000g0     ----
app1      113     e1000g0     ----
web2      111     e1000g1     ----
auth2     112     e1000g1     ----
app2      113     e1000g1     ----
web3      111     e1000g2     ----
auth3     113     e1000g2     ----

# ipadm delete-ip web1
# dladm delete-vlan web1
```

Combining Network Configuration Tasks While Using Customized Names

This section provides an example that combines all the procedures in the previous chapters about configuring links, link aggregations, and VLANs while using customized names. For a description of other networking scenarios that use customized names, see the article in `http://www.oracle.com/us/sun/index.htm`.

EXAMPLE 13–5 Configuring Links, VLANs, and Link Aggregations

In this example, a system that uses 4 NICs needs to be configured to be a router for 8 separate subnets. To attain this objective, 8 links will be configured, one for each subnet. First, a link aggregation is created on all 4 NICs. This untagged link becomes the default untagged subnet for the network to which the default route points.

Then VLAN interfaces are configured over the link aggregation for the other subnets. The subnets are named by basing on a color-coded scheme. Accordingly, the VLAN names are

EXAMPLE 13–5 Configuring Links, VLANs, and Link Aggregations *(Continued)*

likewise named to correspond to their respective subnets. The final configuration consists of 8 links for the eight subnets: 1 untagged link, and 7 tagged VLAN links.

To make the configurations persist across reboots, the same procedures apply as in previous Oracle Solaris releases. For example, IP addresses need to be added to configuration files like `/etc/inet/ndpd.conf`. Or, filter rules for the interfaces need to be included in a rules file. These final steps are not included in the example. For these steps, refer to the appropriate chapters in *Oracle Solaris Administration: IP Services*, particularly *TCP/IP Administration* and *DHCP*.

```
# dladm show-link
LINK        CLASS      MTU  STATE    BRIDGE     OVER
nge0        phys      1500  up       --         --
nge1        phys      1500  up       --         --
e1000g0     phys      1500  up       --         --
e1000g1     phys      1500  up       --         --

# dladm show-phys
LINK        MEDIA               STATE      SPEED  DUPLEX   DEVICE
nge0        Ethernet            up        1000Mb  full     nge0
nge1        Ethernet            up        1000Mb  full     nge1
e1000g0     Ethernet            up        1000Mb  full     e1000g0
e1000g1     Ethernet            up        1000Mb  full     e1000g1

# ipadm delete-ip nge0
# ipadm delete-ip nge1
# ipadm delete-ip e1000g0
# ipadm delete-ip e1000g1

# dladm rename-link nge0 net0
# dladm rename-link nge1 net1
# dladm rename-link e1000g0 net2
# dladm rename-link e1000g1 net3

# dladm show-link
LINK        CLASS      MTU  STATE    BRIDGE     OVER
net0        phys      1500  up       --         --
net1        phys      1500  up       --         --
net2        phys      1500  up       --         --
net3        phys      1500  up       --         --

# dladm show-phys
LINK        MEDIA               STATE      SPEED  DUPLEX   DEVICE
net0        Ethernet            up        1000Mb  full     nge0
net1        Ethernet            up        1000Mb  full     nge1
net2        Ethernet            up        1000Mb  full     e1000g0
net3        Ethernet            up        1000Mb  full     e1000g1

# dladm create-aggr -P L2,L3 -l net0 -l net1 -l net2 -l net3 default0

# dladm show-link
LINK        CLASS      MTU  STATE    BRIDGE     OVER
net0        phys      1500  up       --         --
```

EXAMPLE 13–5 Configuring Links, VLANs, and Link Aggregations *(Continued)*

```
net1        phys      1500  up       --         --
net2        phys      1500  up       --         --
net3        phys      1500  up       --         --
default0    aggr      1500  up       --         net0 net1 net2 net3

# dladm create-vlan -v 2 -l default0 orange0
# dladm create-vlan -v 3 -l default0 green0
# dladm create-vlan -v 4 -l default0 blue0
# dladm create-vlan -v 5 -l default0 white0
# dladm create-vlan -v 6 -l default0 yellow0
# dladm create-vlan -v 7 -l default0 red0
# dladm create-vlan -v 8 -l default0 cyan0

# dladm show-link
LINK        CLASS      MTU  STATE    BRIDGE     OVER
net0        phys      1500  up       --         --
net1        phys      1500  up       --         --
net2        phys      1500  up       --         --
net3        phys      1500  up       --         --
default0    aggr      1500  up       --         net0 net1 net2 net3
orange0     vlan      1500  up       --         default0
green0      vlan      1500  up       --         default0
blue0       vlan      1500  up       --         default0
white0      vlan      1500  up       --         default0
yellow0     vlan      1500  up       --         default0
red0        vlan      1500  up       --         default0
cyan0       vlan      1500  up       --         default0

# dladm show-vlan
LINK          VID   OVER        FLAGS
orange0         2   default0    -----
green0          3   default0    -----
blue0           4   default0    -----
white0          5   default0    -----
yellow0         6   default0    -----
red0            7   default0    -----
cyan0           8   default0    -----

# ipadm create-ip orange0
# ipadm create-ip green0
# ipadm create-ip blue0
# ipadm create-ip white0
# ipadm create-ip yellow0
# ipadm create-ip red0
# ipadm create-ip cyan0

# ipadm create-addr -T static -a IP-address orange0/v4
# ipadm create-addr -T static -a IP-address green0/v4
# ipadm create-addr -T static -a IP-address blue0/v4
# ipadm create-addr -T static -a IP-address white0/v4
# ipadm create-addr -T static -a IP-address yellow0/v4
# ipadm create-addr -T static -a IP-address red0/v4
# ipadm create-addr -T static -a IP-address cyan0/v4
```

◆ ◆ ◆ CHAPTER 14

Introducing IPMP

IP network multipathing (IPMP) provides physical interface failure detection, transparent network access failover, and packet load spreading for systems with multiple interfaces that are connected to a particular local area network or LAN.

This chapter contains the following information:

Note – Throughout the description of IPMP in this chapter and in Chapter 15, "Administering IPMP," all references to the term *interface* specifically mean *IP interface*. Unless a qualification explicitly indicates a different use of the term, such as a network interface card (NIC), the term always refers to the interface that is configured on the IP layer.

What's New With IPMP

The following features differentiate the current IPMP implementation from the previous implementation:

- An IPMP group is represented as an IPMP IP interface. This interface is treated just like any other interface on the IP layer of the networking stack. All IP administrative tasks, routing tables, Address Resolution Protocol (ARP) tables, firewall rules, and other IP-related procedures work with an IPMP group by referring to the IPMP interface.

- The system becomes responsible for the distribution of data addresses among underlying active interfaces. In the previous IPMP implementation, the administrator initially determines the binding of data addresses to corresponding interfaces when the IPMP group is created. In the current implementation, when the IPMP group is created, data addresses belong to the IPMP interface as an address pool. The kernel then automatically and randomly binds the data addresses to the underlying active interfaces of the group.
- The `ipmpstat` tool is introduced as the principal tool to obtain information about IPMP groups. This command provides information about all aspects of the IPMP configuration, such as the underlying IP interfaces of the group, test and data addresses, types of failure detection being used, and which interfaces have failed. The `ipmpstat` functions, the options you can use, and the output each option generates are all described in "Monitoring IPMP Information" on page 296.
- The IPMP interface can be assigned a customized name to identify the IPMP group more easily within your network setup. For the procedures to configure IPMP groups with customized names, see any procedure that describes the creation of an IPMP group in "Configuring IPMP Groups" on page 279.

Note – To use IPMP, make sure that the `DefaultFixed` profile is enabled on the system. For procedures, see "Profiles and Configuration Tools" on page 144. For more information about profile-managed network configuration, see Chapter 4, "NWAM Profile Configuration (Tasks)."

Deploying IPMP

This section describes various topics about the use of IPMP groups.

Why You Should Use IPMP

Different factors can cause an interface to become unusable. Commonly, an IP interface can fail. Or, an interface might be switched offline for hardware maintenance. In such cases, without an IPMP group, the system can no longer be contacted by using any of the IP addresses that are associated with that unusable interface. Additionally, existing connections that use those IP addresses are disrupted.

With IPMP, one or more IP interfaces can be configured into an *IPMP group*. The group functions like an IP interface with data addresses to send or receive network traffic. If an underlying interface in the group fails, the data addresses are redistributed among the remaining underlying active interfaces in the group. Thus, the group maintains network connectivity despite an interface failure. With IPMP, network connectivity is always available, provided that a minimum of one interface is usable for the group.

Additionally, IPMP improves overall network performance by automatically spreading out outbound network traffic across the set of interfaces in the IPMP group. This process is called outbound *load spreading*. The system also indirectly controls inbound load spreading by performing source address selection for packets whose IP source address was not specified by the application. However, if an application has explicitly chosen an IP source address, then the system does not vary that source address.

When You Must Use IPMP

The configuration of an IPMP group is determined by your system configurations. Observe the following rules:

- Multiple IP interfaces on the same local area network or LAN must be configured into an IPMP group. LAN broadly refers to a variety of local network configurations including VLANs and both wired and wireless local networks whose nodes belong to *the same link-layer broadcast domain*.

  Note – Multiple IPMP groups on the same link layer (L2) broadcast domain are unsupported. A L2 broadcast domain typically maps to a specific subnet. Therefore, you must configure only one IPMP group per subnet.

- Underlying IP interfaces of an IPMP group must not span different LANs.

For example, suppose that a system with three interfaces is connected to two separate LANs. Two IP interfaces link to one LAN while a single IP interface connects to the other. In this case, the two IP interfaces connecting to the first LAN must be configured as an IPMP group, as required by the first rule. In compliance with the second rule, the single IP interface that connects to the second LAN cannot become a member of that IPMP group. No IPMP configuration is required of the single IP interface. However, you can configure the single interface into an IPMP group to monitor the interface's availability. The single-interface IPMP configuration is discussed further in "Types of IPMP Interface Configurations" on page 262.

Consider another case where the link to the first LAN consists of three IP interfaces while the other link consists of two interfaces. This setup requires the configuration of two IPMP groups: a three-interface group that links to the first LAN, and a two-interface group to connect to the second.

Comparing IPMP and Link Aggregation

IPMP and link aggregation are different technologies to achieve improved network performance as well as maintain network availability. In general, you deploy link aggregation to obtain better network performance, while you use IPMP to ensure high availability.

The following table presents a general comparison between link aggregation and IPMP.

	IPMP	Link Aggregation
Network technology type	Layer 3 (IP layer)	Layer 2 (link layer)
Configuration tool	`ipadm`	`dladm`
Link-based failure detection	Supported.	Supported.
Probe-based failure detection	ICMP-based, targeting any defined system in the same IP subnet as test addresses, across multiple levels of intervening Layer 2 switches.	Based on Link Aggregation Control Protocol (LACP), targeting immediate peer host or switch.
Use of standby interfaces	Supported	Not supported
Span multiple switches	Supported	Generally not supported; some vendors provide proprietary and non-interoperable solutions to span multiple switches.
Hardware support	Not required	Required. For example, a link aggregation in the system that is running Oracle Solaris requires that corresponding ports on the switches be also aggregated.
Link layer requirements	Broadcast-capable	Ethernet-specific
Driver framework requirements	None	Must use GLDv3 framework
Load spreading support	Present, controlled by kernel. Inbound load spreading is indirectly affected by source address selection.	Finer grain control of the administrator over load spreading of outbound traffic by using `dladm` command. Inbound load spreading supported.

In link aggregations, incoming traffic is spread over the multiple links that comprise the aggregation. Thus, networking performance is enhanced as more NICs are installed to add links to the aggregation. IPMP's traffic uses the IPMP interface's data addresses as they are bound to the available active interfaces. If, for example, all the data traffic is flowing between only two IP addresses but not necessarily over the same connection, then adding more NICs will not improve performance with IPMP because only two IP addresses remain usable.

The two technologies complement each other and can be deployed together to provide the combined benefits of network performance and availability. For example, except where proprietary solutions are provided by certain vendors, link aggregations currently cannot span multiple switches. Thus, a switch becomes a single point of failure for a link aggregation

between the switch and a host. If the switch fails, the link aggregation is likewise lost, and network performance declines. IPMP groups do not face this switch limitation. Thus, in the scenario of a LAN using multiple switches, link aggregations that connect to their respective switches can be combined into an IPMP group on the host. With this configuration, both enhanced network performance as well as high availability are obtained. If a switch fails, the data addresses of the link aggregation to that failed switch are redistributed among the remaining link aggregations in the group.

For other information about link aggregations, see Chapter 12, "Administering Link Aggregations."

Using Flexible Link Names on IPMP Configuration

With support for customized link names, link configuration is no longer bound to the physical NIC to which the link is associated. Using customized link names allows you to have greater flexibility in administering IP interfaces. This flexibility extends to IPMP administration as well. If an underlying interface of an IPMP group fails and requires a replacement, the procedures to replace the interface is greatly facilitated. The replacement NIC, provided it is the same type as the failed NIC, can be renamed to inherit the configuration of the failed NIC. You do not have to create new configurations before you can add it a new interface to the IPMP group. After you assign the link name of the failed NIC to the new NIC, the new NIC is configured with the same settings as the failed interface. The multipathing daemon then deploys the interface according to the IPMP configuration of active and standby interfaces.

Therefore, to optimize your networking configuration and facilitate IPMP administration, you must employ flexible link names for your interfaces by assigning them generic names. In the following section "How IPMP Works" on page 255, all the examples use flexible link names for the IPMP group and its underlying interfaces. For details about the processes behind NIC replacements in a networking environment that uses customized link names, refer to "IPMP and Dynamic Reconfiguration" on page 268. For an overview of the networking stack and the use of customized link names, refer to "The Network Stack in Oracle Solaris" on page 22.

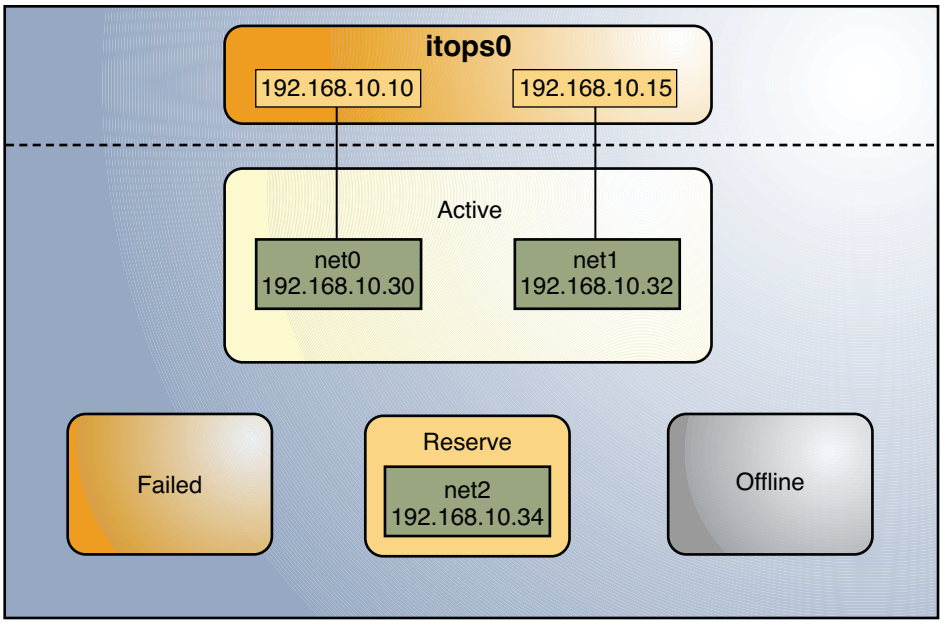
How IPMP Works

IPMP maintains network availability by attempting to preserve the original number of active and standby interfaces when the group was created.

IPMP failure detection can be link-based or probe-based or both to determine the availability of a specific underlying IP interface in the group. If IPMP determines that an underlying interface has failed, then that interface is flagged as failed and is no longer usable. The data IP address that was associated with the failed interface is then redistributed to another functioning interface in the group. If available, a standby interface is also deployed to maintain the original number of active interfaces.

Consider a three-interface IPMP group `itops0` with an active-standby configuration, as illustrated in Figure 14–1.

FIGURE 14–1 IPMP Active–Standby Configuration



The group `itops0` is configured as follows:

- Two data addresses are assigned to the group: `192.168.10.10` and `192.168.10.15`.
- Two underlying interfaces are configured as active interfaces and are assigned flexible link names: `net0` and `net1`.
- The group has one standby interface, also with a flexible link name: `net2`.
- Probe–based failure detection is used, and thus the active and standby interfaces are configured with test addresses, as follows:
  - `net0`: `192.168.10.30`
  - `net1`: `192.168.10.32`
  - `net2`: `192.168.10.34`

Note – The `Active`, `Offline`, `Reserve`, and `Failed` areas in the figures indicate only the status of underlying interfaces, and not physical locations. No physical movement of interfaces or addresses nor transfer of IP interfaces occur within this IPMP implementation. The areas only serve to show how an underlying interface changes status as a result of either failure or repair.

You can use the `ipmpstat` command with different options to display specific types of information about existing IPMP groups. For additional examples, see "Monitoring IPMP Information" on page 296.

The IPMP configuration in Figure 14–1 can be displayed by using the following `ipmpstat` command:

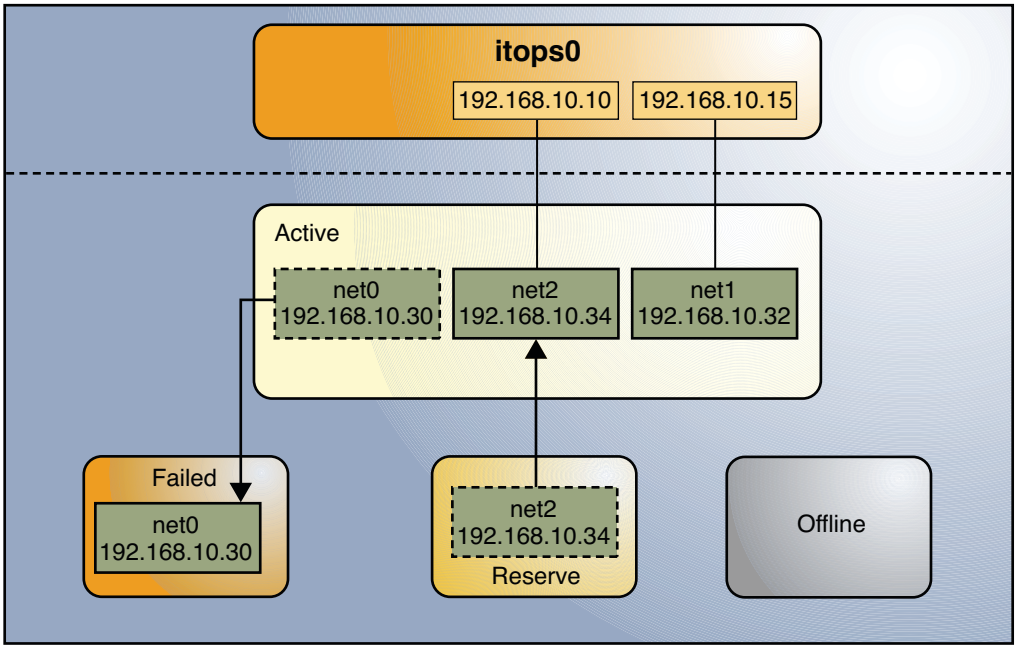
```
# ipmpstat -g
GROUP     GROUPNAME     STATE     FDT        INTERFACES
itops0    itops0        ok        10.00s     net1 net0 (net2)
```

To display information about the group's underlying interfaces, you would type the following:

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP     FLAGS      LINK        PROBE     STATE
net0        yes        itops0    -------    up          ok        ok
net1        yes        itops0    --mb---    up          ok        ok
net2        no         itops0    is-----    up          ok        ok
```

IPMP maintains network availability by managing the underlying interfaces to preserve the original number of active interfaces. Thus, if `net0` fails, then `net2` is deployed to ensure that the group continues to have two active interfaces. The activation of the `net2` is shown in Figure 14–2.

FIGURE 14–2 Interface Failure in IPMP



Note – The one–to–one mapping of data addresses to active interfaces in Figure 14–2 serves only to simplify the illustration. The IP kernel module can assign data addresses randomly without necessarily adhering to a one–to–one relationship between data addresses and interfaces.

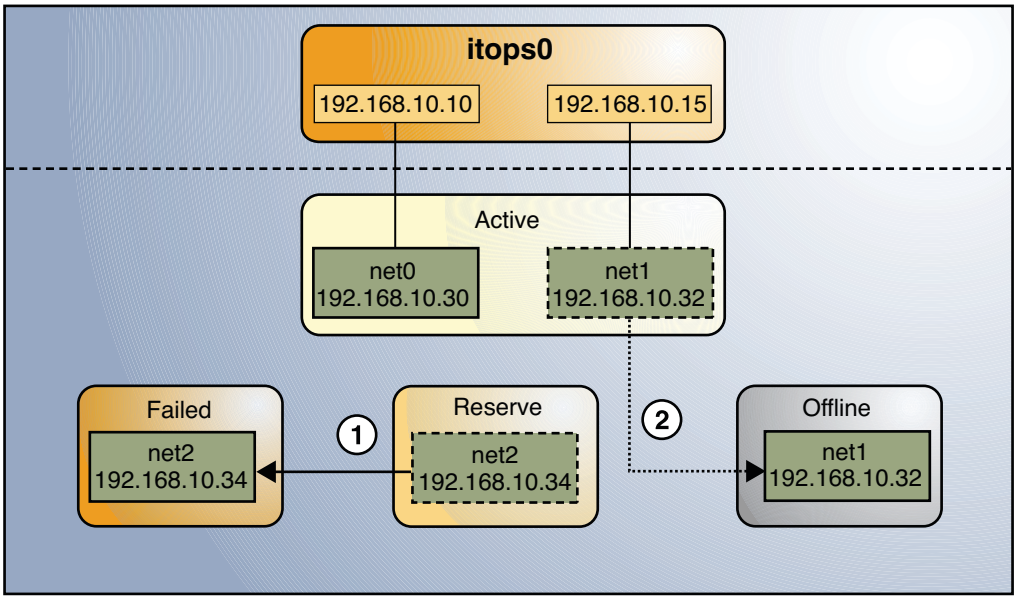
The `ipmpstat` utility displays the information in Figure 14–2 as follows:

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP     FLAGS      LINK        PROBE     STATE
net0        no         itops0    -------    up          failed    failed
net1        yes        itops0    --mb---    up          ok        ok
net2        yes        itops0    -s-----    up          ok        ok
```

After `net0` is repaired, then it reverts to its status as an active interface. In turn, `net2` is returned to its original standby status.

A different failure scenario is shown in Figure 14–3, where the standby interface `net` fails (1), and later, one active interface, `net1`, is switched offline by the administrator (2). The result is that the IPMP group is left with a single functioning interface, `net0`.

FIGURE 14–3 Standby Interface Failure in IPMP

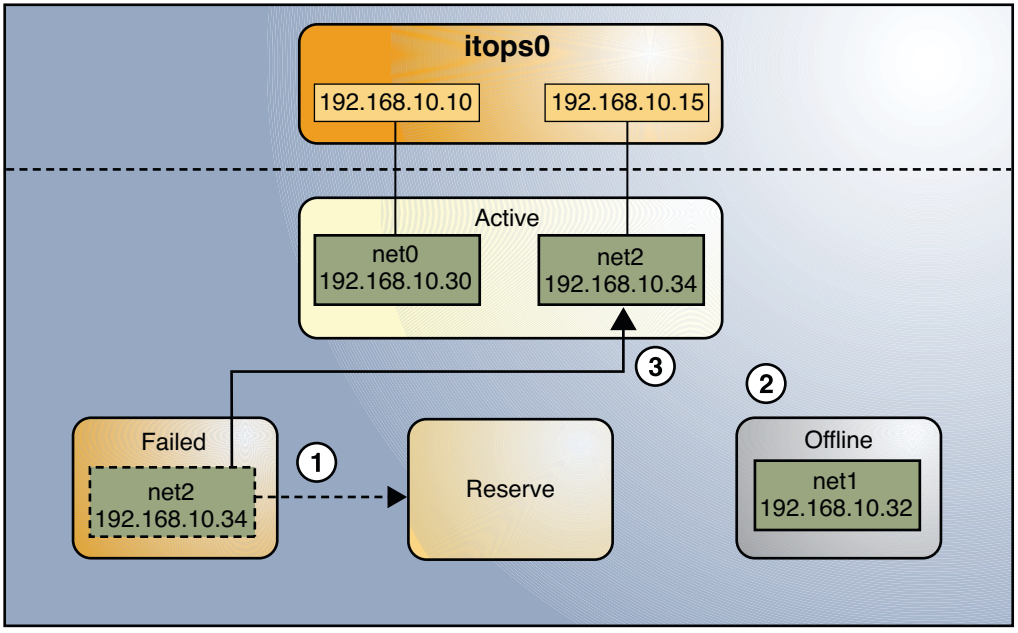


The `ipmpstat` utility would display the information illustrated by Figure 14–3 as follows:

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP     FLAGS       LINK        PROBE     STATE
net0        yes        itops0    -------     up          ok        ok
net1        no         itops0    --mb-d-     up          ok        offline
net2        no         itops0    is-----     up          failed    failed
```

For this particular failure, the recovery after an interface is repaired behaves differently. The restoration depends on the IPMP group's original number of active interfaces compared with the configuration after the repair. The recovery process is represented graphically in Figure 14–4.

FIGURE 14–4 IPMP Recovery Process



In Figure 14–4, when `net2` is repaired, it would normally revert to its original status as a standby interface (1). However, the IPMP group still would not reflect the original number of two active interfaces, because `net1` continues to remain offline (2). Thus, IPMP deploys `net2` as an active interface instead (3).

The `ipmpstat` utility would display the post-repair IPMP scenario as follows:

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP     FLAGS       LINK        PROBE     STATE
net0        yes        itops0    -------     up          ok        ok
net1        no         itops0    --mb-d-     up          ok        offline
net2        yes        itops0    -s-----     up          ok        ok
```

A similar restore sequence occurs if the failure involves an active interface that is also configured in `FAILBACK=no` mode, where a failed active interface does not automatically revert to active status upon repair. Suppose `net0` in Figure 14–2 is configured in `FAILBACK=no` mode. In that mode, a repaired `net0` is switched to a reserve status as a standby interface, even though it was originally an active interface. The interface `net2` would remain active to maintain the IPMP group's original number of two active interfaces. The `ipmpstat` utility would display the recovery information as follows:

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP     FLAGS       LINK        PROBE     STATE
net0        no         itops0    i------     up          ok        ok
net1        yes        itops0    --mb---     up          ok        ok
```

```
net2        yes        itops0    -s-----    up          ok        ok
```

For more information about this type of configuration, see "The FAILBACK=no Mode" on page 267.

IPMP Components in Oracle Solaris

Oracle Solaris IPMP involves the following software:

The *multipathing daemon* `in.mpathd` detects interface failures and repairs. The daemon performs both link-based failure detection and probe-based failure detection if test addresses are configured for the underlying interfaces. Depending on the type of failure detection method that is employed, the daemon sets or clears the appropriate flags on the interface to indicate whether the interface failed or has been repaired. As an option, the daemon can also be configured to monitor the availability of all interfaces, including those that are not configured to belong to an IPMP group. For a description of failure detection, see "Failure and Repair Detection in IPMP" on page 264.

The `in.mpathd` daemon also controls the designation of active interfaces in the IPMP group. The daemon attempts to maintain the same number of active interfaces that was originally configured when the IPMP group was created. Thus `in.mpathd` activates or deactivates underlying interfaces as needed to be consistent with the administrator's configured policy. For more information about the manner by which the `in.mpathd` daemon manages activation of underlying interfaces, refer to "How IPMP Works" on page 255. For more information about the daemon, refer to the `in.mpathd(1M)` man page.

The *IP kernel module* manages outbound load-spreading by distributing the set of available IP data addresses in the group across the set of available underlying IP interfaces in the group. The module also performs source address selection to manage inbound load-spreading. Both roles of the IP module improve network traffic performance.

The *IPMP configuration file* `/etc/default/mpathd` is used to configure the daemon's behavior. For example, you can specify how the daemon performs probe-based failure detection by setting the time duration to probe a target to detect failure, or which interfaces to probe. You can also specify what the status of a failed interface should be after that interface is repaired. You also set the parameters in this file to specify whether the daemon should monitor all IP interfaces in the system, not only those that are configured to belong to IPMP groups. For procedures to modify the configuration file, refer to "How to Configure the Behavior of the IPMP Daemon" on page 293.

The *ipmpstat utility* provides different types of information about the status of IPMP as a whole. The tool also displays other specific information about the underlying IP interfaces for each group, as well as data and test addresses that have been configured for the group. For more information about the use of this command, see "Monitoring IPMP Information" on page 296 and the `ipmpstat(1M)` man page.

Types of IPMP Interface Configurations

An IPMP configuration typically consists of two or more physical interfaces on the same system that are attached to the same LAN. These interfaces can belong to an IPMP group in either of the following configurations:

- active-active configuration – an IPMP group in which all underlying interfaces are active. An *active interface* is an IP interface that is currently available for use by the IPMP group. By default, an underlying interface becomes active when you configure the interface to become part of an IPMP group. For additional information about active interfaces and other IPMP terms, see also "IPMP Terminology and Concepts" on page 270.
- active-standby configuration – an IPMP group in which at least one interface is administratively configured as a reserve. The reserve interface is called the *standby interface*. Although idle, the standby IP interface is monitored by the multipathing daemon to track the interface's availability, depending on how the interface is configured. If link-failure notification is supported by the interface, link-based failure detection is used. If the interface is configured with a test address, probe-based failure detection is also used. If an active interface fails, the standby interface is automatically deployed as needed. You can configure as many standby interfaces as you want for an IPMP group.

A single interface can also be configured in its own IPMP group. The single interface IPMP group has the same behavior as an IPMP group with multiple interfaces. However, this IPMP configuration does not provide high availability for network traffic. If the underlying interface fails, then the system loses all capability to send or receive traffic. The purpose of configuring a single-interfaced IPMP group is to monitor the availability of the interface by using failure detection. By configuring a test address on the interface, you can set the daemon to track the interface by using probe-based failure detection. Typically, a single-interfaced IPMP group configuration is used in conjunction with other technologies that have broader failover capabilities, such as Oracle Solaris Cluster software. The system can continue to monitor the status of the underlying interface. But the Oracle Solaris Cluster software provides the functionalities to ensure availability of the network when failure occurs. For more information about the Oracle Solaris Cluster software, see *Sun Cluster Overview for Solaris OS*.

An IPMP group without underlying interfaces can also exist, such as a group whose underlying interfaces have been removed. The IPMP group is not destroyed, but the group cannot be used to send and receive traffic. As underlying IP interfaces are brought online for the group, then the data addresses of the IPMP interface are allocated to these interfaces and the system resumes hosting network traffic.

IPMP Addressing

You can configure IPMP failure detection on both IPv4 networks and dual-stack, IPv4 and IPv6 networks. Interfaces that are configured with IPMP support two types of addresses:

- *Data Addresses* are the conventional IPv4 and IPv6 addresses that are assigned to an IP interface dynamically at boot time by the DHCP server, or manually by using the `ipadm` command. Data addresses are assigned to the IPMP interface. The standard IPv4 packet traffic and, if applicable, IPv6 packet traffic are considered *data traffic*. Data traffic flow use the data addresses that are hosted on the IPMP interface and flow through the active interfaces of that group.
- *Test Addresses* are IPMP-specific addresses that are used by the `in.mpathd` daemon to perform probe-based failure and repair detection. Test addresses can also be assigned dynamically by the DHCP server, or manually by using the `ipadm` command. While data addresses are assigned to the IPMP interface, only test addresses are assigned to the underlying interfaces of the group. For an underlying interface on a dual-stack network, you can configure an IPv4 test address or an IPv6 test address or both. When an underlying interface fails, the interface's test address continues to be used by the `in.mpathd` daemon for probe-based failure detection to check for the interface's subsequent repair.

Note – You need to configure test addresses only if you specifically want to use probe-based failure detection. Otherwise, you can enable transitive probing to detect failure without using test addresses. For more information about probe-based failure detection with or without using test addresses, refer to "Probe-Based Failure Detection" on page 264.

In previous IPMP implementations, test addresses needed to be marked as `DEPRECATED` to avoid being used by applications especially during interface failures. In the current implementation, test addresses reside in the underlying interfaces. Thus, these addresses can no longer be accidentally used by applications that are unaware of IPMP. However, to ensure that these addresses will not be considered as a possible source for data packets, the system automatically marks any addresses with the `NOFAILOVER` flag as also `DEPRECATED`.

IPv4 Test Addresses

In general, you can use any IPv4 address on your subnet as a test address. IPv4 test addresses do not need to be routeable. Because IPv4 addresses are a limited resource for many sites, you might want to use non-routeable RFC 1918 private addresses as test addresses. Note that the `in.mpathd` daemon exchanges only ICMP probes with other hosts on the same subnet as the test address. If you do use RFC 1918-style test addresses, be sure to configure other systems, preferably routers, on the network with addresses on the appropriate RFC 1918 subnet. The `in.mpathd` daemon can then successfully exchange probes with target systems. For more information about RFC 1918 private addresses, refer to RFC 1918, Address Allocation for Private Internets (`http://www.ietf.org/rfc/rfc1918.txt?number=1918`).

IPv6 Test Addresses

The only valid IPv6 test address is the link-local address of a physical interface. You do not need a separate IPv6 address to serve as an IPMP test address. The IPv6 link-local address is based on the Media Access Control (MAC) address of the interface. Link-local addresses are automatically configured when the interface becomes IPv6-enabled at boot time or when the interface is manually configured through `ipadm`.

For more information on link-local addresses, refer to "Link-Local Unicast Address" in *System Administration Guide: IP Services*.

When an IPMP group has both IPv4 and IPv6 plumbed on all the group's interfaces, you do not need to configure separate IPv4 test addresses. The `in.mpathd` daemon can use the IPv6 link-local addresses as test addresses.

Failure and Repair Detection in IPMP

To ensure continuous availability of the network to send or receive traffic, IPMP performs failure detection on the IPMP group's underlying IP interfaces. Failed interfaces remain unusable until these are repaired. Remaining active interfaces continue to function while any existing standby interfaces are deployed as needed.

Types of Failure Detection in IPMP

The `in.mpathd` daemon handles the following types of failure detection:

- Probe-based failure detection, of two types:
  - No test addresses are configured (transitive probing).
  - Test addresses are configured.
- Link-based failure detection, if supported by the NIC driver

Probe-Based Failure Detection

Probe-based failure detection consists of using ICMP probes to check whether an interface has failed. The implementation of this failure detection method depends on whether test addresses are used or not.

Probe-Based Failure Detection Without Using Test Addresses

With no test addresses, this method is implemented by using two types of probes:

- ICMP probes

ICMP probes are sent by the active interfaces in the group to probe targets that are defined in the routing table. An *active* interface is the underlying interface that can receive inbound IP packets that are addressed to the interface's link layer (L2) address. The ICMP probe uses the data address as the probe's source address. If the ICMP probe reaches its target and gets a response from the target, then the active interface is operational.

- Transitive probes

  Transitive probes are sent by the alternate interfaces in the group to probe the active interface. An alternate interface is an underlying interface that does not actively receive any inbound IP packets.

  For example, consider an IPMP group that consists of four underlying interfaces. The group is configured with one data address but no test addresses. In this configuration, outbound packets can use all the underlying interfaces. However, inbound packets can only be received by the interface to which the data address is bound. The remaining three underlying interfaces that cannot receive inbound packets are the *alternate* interfaces.

  If an alternate interface can successfully send a probe to an active interface and receive a response, then the active interface is functional, and by inference, so is the alternate interface that sent the probe.

Note – You must enable transitive probing to use this failure detection method that does not require test addresses.

Probe-Based Failure Detection Using Test Addresses

This failure detection method involves sending and receiving ICMP probe messages that use test addresses. These messages, also called *probe traffic* or test traffic, go out over the interface to one or more target systems on the same local network. The daemon probes all the targets separately through all the interfaces that have been configured for probe-based failure detection. If no replies are made in response to five consecutive probes on a given interface, `in.mpathd` considers the interface to have failed. The probing rate depends on the *failure detection time* (FDT). The default value for failure detection time is 10 seconds. However, you can tune the failure detection time in the IPMP configuration file. For instructions, go to "How to Configure the Behavior of the IPMP Daemon" on page 293. To optimize probe-based failure detection, you must set multiple target systems to receive the probes from the multipathing daemon. By having multiple target systems, you can better determine the nature of a reported failure. For example, the absence of a response from the only defined target system can indicate a failure either in the target system or in one of the IPMP group's interfaces. By contrast, if only one system among several target systems does not respond to a probe, then the failure is likely in the target system rather than in the IPMP group itself.

The `in.mpathd` daemon determines which target systems to probe dynamically. First the daemon searches the routing table for target systems on the same subnet as the test addresses that are associated with the IPMP group's interfaces. If such targets are found, then the daemon uses them as targets for probing. If no target systems are found on the same subnet, then

`in.mpathd` sends multicast packets to probe neighbor hosts on the link. The multicast packet is sent to the all hosts multicast address, `224.0.0.1` in IPv4 and `ff02::1` in IPv6, to determine which hosts to use as target systems. The first five hosts that respond to the echo packets are chosen as targets for probing. If `in.mpathd` cannot find routers or hosts that responded to the multicast probes, then ICMP echo packets, `in.mpathd` cannot detect probe-based failures. In this case, the `ipmpstat -i` utility will report the probe state as `unknown`.

You can use host routes to explicitly configure a list of target systems to be used by `in.mpathd`. For instructions, refer to "Configuring for Probe-Based Failure Detection" on page 291.

Group Failure

A *group failure* occurs when all interfaces in an IPMP group appear to fail at the same time. In this case, no underlying interface is usable. Also, when all the target systems fail at the same time and probe-based failure detection is enabled, the `in.mpathd` daemon flushes all of its current target systems and probes for new target systems.

In an IPMP group that has no test addresses, a single interface that can probe the active interface will be designated as a prober. This designated interface will have both the FAILED flag and PROBER flag set. The data address is bound to this interface which allows the interface to continue probing the target to detect recovery.

Link-Based Failure Detection

Link-based failure detection is always enabled, provided that the interface supports this type of failure detection.

To determine whether a third-party interface supports link-based failure detection, use the `ipmpstat -i` command. If the output for a given interface includes an `unknown` status for its `LINK` column, then that interface does not support link-based failure detection. Refer to the manufacturer's documentation for more specific information about the device.

These network drivers that support link-based failure detection monitor the interface's link state and notify the networking subsystem when that link state changes. When notified of a change, the networking subsystem either sets or clears the `RUNNING` flag for that interface, as appropriate. If the `in.mpathd` daemon detects that the interface's `RUNNING` flag has been cleared, the daemon immediately fails the interface.

Failure Detection and the Anonymous Group Feature

IPMP supports failure detection in an anonymous group. By default, IPMP monitors the status only of interfaces that belong to IPMP groups. However, the IPMP daemon can be configured to also track the status of interfaces that do not belong to any IPMP group. Thus, these interfaces are considered to be part of an "anonymous group." When you issue the command `ipmpstat -g`, the anonymous group will be displayed as double-dashes (`--`). In anonymous groups, the interfaces would have their data addresses function also as test addresses. Because

these interfaces do not belong to a named IPMP group, then these addresses are visible to applications. To enable tracking of interfaces that are not part of an IPMP group, see "How to Configure the Behavior of the IPMP Daemon" on page 293.

Detecting Physical Interface Repairs

Repair detection time is twice the failure detection time. The default time for failure detection is 10 seconds. Accordingly, the default time for repair detection is 20 seconds. After a failed interface has been marked with the RUNNING flag again and the failure detection method has detected as repaired, `in.mpathd` clears the interface's `FAILED` flag. The repaired interface is redeployed depending on the number of active interfaces that the administrator has originally set.

When an underlying interface fails and probe-based failure detection is used, the `in.mpathd` daemon continues probing, either by means of the designated prober when no test addresses are configured, or by using the interface's test address. During an interface repair, the restoration proceeds depending on the original configuration of the failed interface:

- Failed interface was originally an active interface – the repaired interface reverts to its original active status. The standby interface that functioned as a replacement during the failure is switched back to standby status if enough interfaces are active for the group as defined by the system administrator.

  Note – An exception to this step are cases when the repaired active interface is also configured with the `FAILBACK=no` mode. For more information, see "The `FAILBACK=no` Mode" on page 267

- Failed interface was originally a standby interface – the repaired interface reverts to its original standby status, provided that the IPMP group reflects the original number of active interfaces. Otherwise, the standby interface is switched to become an active interface.

To see a graphical presentation of how IPMP behaves during interface failure and repair, see "How IPMP Works" on page 255.

The `FAILBACK=no` Mode

By default, active interfaces that have failed and then repaired automatically return to become active interfaces in the group. This behavior is controlled by the setting of the `FAILBACK` parameter in the daemon's configuration file. However, even the insignificant disruption that occurs as data addresses are remapped to repaired interfaces might not be acceptable to some administrators. The administrators might prefer to allow an activated standby interface to continue as an active interface. IPMP allows administrators to override the default behavior to

prevent an interface to automatically become active upon repair. These interfaces must be configured in the FAILBACK=no mode. For related procedures, see "How to Configure the Behavior of the IPMP Daemon" on page 293.

When an active interface in FAILBACK=no mode fails and is subsequently repaired, the IPMP daemon restores the IPMP configuration as follows:

- The daemon retains the interface's INACTIVE status, provided that the IPMP group reflects the original configuration of active interfaces.
- If the IPMP configuration at the moment of repair does not reflect the group's original configuration of active interfaces, then the repaired interface is redeployed as an active interface, notwithstanding the FAILBACK=no status.

Note – The FAILBACK=NO mode is set for the whole IPMP group. It is not a per-interface tunable parameter.

IPMP and Dynamic Reconfiguration

Dynamic reconfiguration (DR) feature allows you to reconfigure system hardware, such as interfaces, while the system is running. DR can be used only on systems that support this feature.

You typically use the cfgadm command to perform DR operations. However, some platforms provide other methods. Make sure to consult your platform's documentation for details to perform DR. For systems that use Oracle Solaris, you can find specific documentation about DR in the resources that are listed in Table 14–1. Current information about DR is also available at http://www.oracle.com/technetwork/indexes/documentation/index.html and can be obtained by searching for the topic "dynamic reconfiguration."

TABLE 14–1 Documentation Resources for Dynamic Reconfiguration

Description	For Information
Detailed information on the cfgadm command	cfgadm(1M) man page
Specific information about DR in the Oracle Solaris Cluster environment	*Oracle Solaris Cluster System Administration Guide*
Specific information about DR in the Sun Servers from Oracle	See documentation that came with your specific server
Introductory information about DR and the cfgadm command	Chapter 6, "Dynamically Configuring Devices (Tasks)," in *Oracle Solaris Administration: Devices and File Systems*

TABLE 14–1 Documentation Resources for Dynamic Reconfiguration *(Continued)*

Description	For Information
Tasks for administering IPMP groups on a system that supports DR	"Recovering an IPMP Configuration With Dynamic Reconfiguration" on page 294

The sections that follow explain how DR interoperates with IPMP.

On a system that supports DR of NICs, IPMP can be used to preserve connectivity and prevent disruption of existing connections. IPMP is integrated into the Reconfiguration Coordination Manager (RCM) framework. Thus, you can safely attach, detach, or reattach NICs and RCM manages the dynamic reconfiguration of system components.

Attaching New NICs

With DR support, you can attach, plumb, and then add new interfaces to existing IPMP groups. Or, if appropriate, you can configure the newly added interfaces into their own IPMP group. For procedures to configure IPMP groups, refer to "Configuring IPMP Groups" on page 279. After these interfaces have been configured, they are immediately available for use by IPMP. However, to benefit from the advantages of using customized link names, you must assign generic link names to replace the interface's hardware-based link names. Then you create corresponding configuration files by using the generic name that you just assigned. For procedures to configure a singe interface by using customized link names, refer to "How to Configure an IP Interface" on page 171. After you assign a generic link name to interface, make sure that you always refer to the generic name when you perform any additional configuration on the interface such as using the interface for IPMP.

Detaching NICs

All requests to detach system components that contain NICs are first checked to ensure that connectivity can be preserved. For instance, by default you cannot detach a NIC that is not in an IPMP group. You also cannot detach a NIC that contains the only functioning interfaces in an IPMP group. However, if you must remove the system component, you can override this behavior by using the `-f` option of `cfgadm`, as explained in the `cfgadm(1M)` man page.

If the checks are successful, the daemon sets the `OFFLINE` flag for the interface. All test addresses on the interfaces are unconfigured. Then, the NIC is unplumbed from the system. If any of these steps fail, or if the DR of other hardware on the same system component fails, then the previous configuration is restored to its original state. A status message about this event will be displayed. Otherwise, the detach request completes successfully. You can remove the component from the system. No existing connections are disrupted.

Replacing NICs

When an underlying interface of an IPMP group fails, a typical solution would be to replace the failed interface by attaching a new NIC. RCM records the configuration information associated with any NIC that is detached from a running system. If you replace a failed NIC with an *identical* NIC, then RCM automatically configures the interface according to the persistent configurations that had been previously defined by using the `ipadm` command.

For example, suppose you replace a failed `bge0` interface with another `bge0` interface. The failed `bge0`'s configuration settings that were defined by using the `ipadm` command are persistent settings. After you attach the replacement `bge` NIC, RCM plumbs and then configures the `bge0` interface according to these persistent settings. Thus the interface is properly configured with the test address and is added to the IPMP group.

You can replace a failed NIC with a different NIC, provided that both are the same type, such as Ethernet. In this case, RCM plumbs the new interface after it is attached. If you did not use customized link names when you first configured your interfaces, then you will have to configure the new NIC before you can add the interface to the IPMP group.

However, if you used customized link names, the additional configuration steps are unnecessary. By reassigning the failed interface's link name to the new interface, then the new interface acquires the configuration specified in the removed interface's persistent settings. RCM then configures the interface according to those settings. For procedures to recover your IPMP configuration by using DR when an interface fails, refer to "Recovering an IPMP Configuration With Dynamic Reconfiguration" on page 294.

IPMP Terminology and Concepts

This section introduces terms and concepts that are used throughout the IPMP chapters in this book.

active interface

Refers to an underlying interface that can be used by the system to send or receive data traffic. An interface is active if the following conditions are met:

- At least one IP address is `UP` in the interface. See `UP` address.
- The `FAILED`, `INACTIVE`, or `OFFLINE` flag is not set on the interface.
- The interface has not been flagged as having a duplicate hardware address.

Compare to unusable interface, `INACTIVE` interface.

data address

Refers to an IP address that can be used as the source or destination address for data. Data addresses are part of an

	IPMP group and can be used to send and receive traffic on any interface in the group. Moreover, the set of data addresses in an IPMP group can be used continuously, provided that one interface in the group is functioning. In previous IPMP implementations, data addresses were hosted on the underlying interfaces of an IPMP group. In the current implementation, data addresses are hosted on the IPMP interface.
`DEPRECATED` address	Refers to an IP address that cannot be used as the source address for data. Typically, IPMP test addresses, which have the `NOFAILOVER` flag, are also automatically marked as `DEPRECATED` by the system. However, any address can be marked `DEPRECATED` to prevent the address from being used as a source address.
dynamic reconfiguration	Refers to a feature that allows you to reconfigure a system while the system is running, with little or no impact on ongoing operations. Not all Sun platforms from Oracle support DR. Some platforms might only support DR of certain types of hardware. On platforms that support DR of NICs, IPMP can be used for uninterrupted network access to the system during DR. For more information about how IPMP supports DR, refer to "IPMP and Dynamic Reconfiguration" on page 268.
explicit IPMP interface creation	Applies only to the current IPMP implementation. The term refers to the method of creating an IPMP interface by using the `ipadm create-ipmp` command. Explicit IPMP interface creation is the preferred method for creating IPMP groups. This method allows the IPMP interface name and IPMP group name to be set by the administrator. Compare to implicit IPMP interface creation.
`FAILBACK=no` mode	Refers to a setting of an underlying interface that minimizes rebinding of incoming addresses to interfaces by avoiding redistribution during interface repair. Specifically, when an interface repair is detected, the interface's `FAILED` flag is cleared. However, if the mode of the repaired interface is `FAILBACK=no`, then the `INACTIVE` flag is also set to prevent use of the interface, provided that a second functioning interface also exists. If the second interface in the IPMP group fails, then the `INACTIVE`

	interface is eligible to take over. While the concept of failback no longer applies in the current IPMP implementation, the name of this mode is preserved for administrative compatibility.
`FAILED` interface	Indicates an interface that the `in.mpathd` daemon has determined to be malfunctioning. The determination is achieved by either link-based or probe-based failure detection. The `FAILED` flag is set on any failed interface.
failure detection	Refers to the process of detecting when a physical interface or the path from an interface to an Internet layer device no longer works. Two forms of failure detection are implemented: link-based failure detection, and probe-based failure detection.
implicit IPMP interface creation	Refers to the method of creating an IPMP interface by using the `ifconfig` command to place an underlying interface in a nonexistent IPMP group. Implicit IPMP interface creation is supported for backward compatibility with the IPMP implementation in previous Oracle Solaris releases. Thus, this method does not provide the ability to set the IPMP interface name or IPMP group name. Implicit IPMP interface creation is not supported by the `ipadm` command. Compare to explicit IPMP interface creation.
`INACTIVE` interface	Refers to an interface that is functioning but is not being used according to administrative policy. The `INACTIVE` flag is set on any `INACTIVE` interface. Compare to active interface, unusable interface.
IPMP anonymous group support	Indicates an IPMP feature in which the IPMP daemon tracks the status of all network interfaces in the system, regardless of whether they belong to an IPMP group. However, if the interfaces are not actually in an IPMP group, then the addresses on these interfaces are not available in case of interface failure.
IPMP group	Refers to a set of network interfaces that are treated as interchangeable by the system in order to improve network availability and utilization. Each IPMP group has a set of data addresses that the system can associate with any set of active interfaces in the group. Use of this set of data addresses maintains network availability and improves

	network utilization. The administrator can select which interfaces to place into an IPMP group. However, all interfaces in the same group must share a common set of properties, such as being attached to the same link and configured with the same set of protocols (for example, IPv4 and IPv6).
IPMP group interface	See IPMP interface.
IPMP group name	Refers to the name of an IPMP group, which can be assigned with the `ipadm set-ifprop` subcommand. All underlying interfaces with the same IPMP group name are defined as part of the same IPMP group. In the current implementation, IPMP group names are de-emphasized in favor of IPMP interface names. Administrators are encouraged to use the same name for both the IPMP interface and the group by using the `ipadm create-ipmp` subcommand to create the IPMP group.
IPMP interface	Applies only to the current IPMP implementation. The term refers to the IP interface that represents a given IPMP group, any or all of the interface's underlying interfaces, and all of the data addresses. In the current IPMP implementation, the IPMP interface is the core component for administering an IPMP group, and is used in routing tables, ARP tables, firewall rules, and so forth.
IPMP interface name	Indicates the name of an IPMP interface. This document uses the naming convention of `ipmp`*N*. The system also uses the same naming convention in implicit IPMP interface creation. However, the administrator can choose any name by using explicit IPMP interface creation.
IPMP singleton	Refers to an IPMP configuration that is used by Oracle Solaris Cluster software that allows a data address to also act as a test address. This configuration applies, for instance, when only one interface belongs to an IPMP group.
link-based failure detection	Specifies a passive form of failure detection, in which the link status of the network card is monitored to determine an interface's status. Link-based failure detection only tests whether the link is up. This type of failure detection is not supported by all network card drivers. Link-based failure detection requires no explicit configuration and provides instantaneous detection of link failures.

	Compare to probe-based failure detection.
load spreading	Refers to the process of distributing inbound or outbound traffic over a set of interfaces. Unlike load balancing, load spreading does not guarantee that the load is evenly distributed. With load spreading, higher throughput is achieved. Load spreading occurs only when the network traffic is flowing to multiple destinations that use multiple connections. Inbound load spreading indicates the process of distributing inbound traffic across the set of interfaces in an IPMP group. Inbound load spreading cannot be controlled directly with IPMP. The process is indirectly manipulated by the source address selection algorithm. Outbound load spreading refers to the process of distributing outbound traffic across the set of interfaces in an IPMP group. Outbound load spreading is performed on a per-destination basis by the IP module, and is adjusted as necessary depending on the status and members of the interfaces in the IPMP group.
`NOFAILOVER` address	Applies only to the previous IPMP implementation. Refers to an address that is associated with an underlying interface and thus remains unavailable if the underlying interface fails. All `NOFAILOVER` addresses have the `NOFAILOVER` flag set. IPMP test addresses must be designated as `NOFAILOVER`, while IPMP data addresses must never be designated as `NOFAILOVER`. The concept of failover does not exist in the IPMP implementation. However, the term `NOFAILOVER` remains for administrative compatibility.
`OFFLINE` interface	Indicates an interface that has been administratively disabled from system use, usually in preparation for being removed from the system. Such interfaces have the `OFFLINE` flag set. The `if_mpadm` command can be used to switch an interface to an offline status.
physical interface	See: underlying interface
probe	Refers to an ICMP packet, similar to the packets that are used by the `ping` command. This probe is used to test the send and receive paths of a given interface. Probe packets

	are sent by the `in.mpathd` daemon, if probe-based failure detection is enabled. A probe packet uses an IPMP test address as its source address.
probe-based failure detection	Indicates an active form of failure detection, in which probes are exchanged with probe targets to determine an interface's status. When enabled, probe-based failure detection tests the entire send and receive path of each interface. However, this type of detection requires the administrator to explicitly configure each interface with a test address. Compare to link-based failure detection.
probe target	Refers to a system on the same link as an interface in an IPMP group. The target is selected by the `in.mpathd` daemon to help check the status of a given interface by using probe-based failure detection. The probe target can be any host on the link that is capable of sending and receiving ICMP probes. Probe targets are usually routers. Several probe targets are usually used to insulate the failure detection logic from failures of the probe targets themselves.
source address selection	Refers to the process of selecting a data address in the IPMP group as the source address for a particular packet. Source address selection is performed by the system whenever an application has not specifically selected a source address to use. Because each data address is associated with only one hardware address, source address selection indirectly controls inbound load spreading.
`STANDBY` interface	Indicates an interface that has been administratively configured to be used only when another interface in the group has failed. All `STANDBY` interfaces will have the `STANDBY` flag set.
target systems	See probe target.
test address	Refers to an IP address that must be used as the source or destination address for probes, and must not be used as a source or destination address for data traffic. Test addresses are associated with an underlying interface. If an underlying interface is configured with an `UP` test address, the `in.mpathd` daemon monitors this address by using probe-based failure detection. All test addresses must be designated as `NOFAILOVER`. These addresses are also

	automatically marked DEPRECATED by the system to ensure that they will not be considered as a possible source address for data packets.
underlying interface	Specifies an IP interface that is part of an IPMP group and is directly associated with an actual network device. For example, if ce0 and ce1 are placed into IPMP group ipmp0, then ce0 and ce1 comprise the underlying interfaces of ipmp0. In the previous implementation, IPMP groups consist solely of underlying interfaces. However, in the current implementation, these interfaces underlie the IPMP interface (for example, ipmp0) that represents the group, hence the name.
undo-offline operation	Refers to the act of administratively enabling a previously offlined interface to be used by the system. The if_mpadm command can be used to perform an undo-offline operation.
unusable interface	Refers to an underlying interface that cannot be used to send or receive data traffic at all in its current configuration. An unusable interface differs from an INACTIVE interface, that is not currently being used but can be used if an active interface in the group becomes unusable. An interface is unusable if one of the following conditions exists: ■ The interface has no UP address. ■ The FAILED or OFFLINE flag has been set for the interface. ■ The interface has been flagged has having the same hardware address as another interface in the group.
UP address	Refers to an address that has been made administratively available to the system by setting the UP flag. An address that is not UP is treated as not belonging to the system, and thus is never considered during source address selection.

◆ ◆ ◆ CHAPTER 15

Administering IPMP

This chapter provides tasks for administering interface groups with IP network multipathing (IPMP). The following major topics are discussed:

IPMP Administration Task Maps

In Oracle Solaris, the `ipmpstat` command is the preferred tool to use to obtain information about IPMP group information. In this chapter, the `ipmpstat` command replaces certain functions of the `ifconfig` command that were used in previous Oracle Solaris releases to provide IPMP information.

For information about the different options for the `ipmpstat` command, see "Monitoring IPMP Information" on page 296.

This following sections provide links to the tasks in this chapter.

IPMP Group Creation and Configuration (Task Map)

Task	Description	For Instructions
Plan an IPMP group.	Lists all ancillary information and required tasks before you can configure an IPMP group.	"How to Plan an IPMP Group" on page 279

Task	Description	For Instructions
Configure an IPMP group by using DHCP.	Provides an alternative method to configure IPMP groups by using DHCP.	"How to Configure an IPMP Group by Using DHCP" on page 281
Configure an active-active IPMP group.	Configures an IPMP group in which all underlying interfaces are deployed to host network traffic.	"How to Manually Configure an Active-Active IPMP Group" on page 284
Configure an active-standby IPMP group.	Configures an IPMP group in which one underlying interface is kept inactive as a reserve.	"How to Manually Configure an Active-Standby IPMP Group" on page 285

IPMP Group Maintenance (Task Map)

Task	Description	For Instructions
Add an interface to an IPMP group.	Configures a new interface as a member of an existing IPMP group.	"How to Add an Interface to an IPMP Group" on page 287
Remove an interface from an IPMP group.	Removes an interface from an IPMP group.	"How to Remove an Interface From an IPMP Group" on page 287
Add IP addresses to or remove IP addresses from an IPMP group.	Adds or removes addresses for an IPMP group.	"How to Add or Remove IP Addresses" on page 288
Change an interface's IPMP membership.	Moves interfaces among IPMP groups.	"How to Move an Interface From One IPMP Group to Another Group" on page 289
Delete an IPMP group.	Deletes an IPMP group that is no longer needed.	"How to Delete an IPMP Group" on page 290
Replace cards that failed.	Removes or replaces failed NICs of an IPMP group.	"How to Replace a Physical Card That Has Failed" on page 294

Probe-Based Failure Detection Configuration (Task Map)

Task	Description	For Instructions
Manually specify target systems	Identifies and adds systems to be targeted for probe-based failure detection.	"How to Manually Specify Target Systems for Probe-Based Failure Detection" on page 292

Task	Description	For Instructions
Configure the behavior of probe-based failure detection.	Modifies parameters to determine the behavior of probe-based failure detection.	"How to Configure the Behavior of the IPMP Daemon" on page 293

IPMP Group Monitoring (Task Map)

Task	Description	For Instructions
Obtain group information.	Displays information about an IPMP group.	"How to Obtain IPMP Group Information" on page 296
Obtain data address information.	Displays information about the data addresses that are used by an IPMP group.	"How to Obtain IPMP Data Address Information" on page 297
Obtain IPMP interface information.	Displays information about the underlying interfaces of IPMP interfaces or groups.	"How to Obtain Information About Underlying IP Interfaces of a Group" on page 298
Obtain probe target information.	Displays information about targets of probe-based failure detection.	"How to Obtain IPMP Probe Target Information" on page 299
Obtain probe information.	Displays real-time information about ongoing probes in the system.	"How to Observe IPMP Probes" on page 301
Customize the information display for monitoring IPMP groups.	Determines the IPMP information that is displayed.	"How to Customize the Output of the `ipmpstat` Command in a Script" on page 302

Configuring IPMP Groups

This section provides procedures that are used to plan and configure IPMP groups. The overview in Chapter 14, "Introducing IPMP," describes the implementation of the IPMP group as an interface. Thus, in this chapter, the terms *IPMP group* and *IPMP interface* are used interchangeably.

▼ How to Plan an IPMP Group

The following procedure includes the required planning tasks and information to be gathered prior to configuring an IPMP group. The tasks do not have to be performed in sequence.

Note – You must configure only one IPMP group for each subnet or L2 broadcast domain. For more information, see "When You Must Use IPMP" on page 253.

1 Determine the general IPMP configuration that would suit your needs.

Your IPMP configuration depends on what your network needs to handle the type of traffic that is hosted on your system. IPMP spreads outbound network packets across the IPMP group's interfaces, and thus improves network throughput. However, for a given TCP connection, inbound traffic normally follows only one physical path to minimize the risk of processing out-of-order packets.

Thus, if your network handles a huge volume of outbound traffic, configuring a big number of interfaces into an IPMP group can improve network performance. If instead, the system hosts heavy inbound traffic, then the number of interfaces in the group does not necessarily improve performance by load spreading traffic. However, having more underlying interfaces helps to guarantee network availability during interface failure.

2 For SPARC based systems, verify that each interface in the group has a unique MAC address.

To configure a unique MAC address for each interface in the system, see "SPARC: How to Ensure That the MAC Address of an Interface Is Unique" on page 169.

3 Ensure that the same set of STREAMS modules is pushed and configured on all interfaces in the IPMP group.

All interfaces in the same group must have the same STREAMS modules configured in the same order.

a. Check the order of STREAMS modules on all interfaces in the prospective IPMP group.

You can print a list of STREAMS modules by using the `ifconfig` *interface* `modlist` command. For example, here is the `ifconfig` output for an `net0` interface:

```
# ifconfig net0 modlist
    0 arp
    1 ip
    2 e1000g
```

As the output shows, interfaces normally exist as network drivers directly below the IP module. These interfaces should not require additional configuration.

However, certain technologies insert themselves as a STREAMS module between the IP module and the network driver. If a STREAMS module is stateful, then unexpected behavior can occur on failover, even if you push the same module onto all of the interfaces in a group. However, you can use stateless STREAMS modules, provided that you push them in the same order on all interfaces in the IPMP group.

b. Push the modules of an interface in the standard order for the IPMP group.

ifconfig *interface* **modinsert** *module-name*@*position*

```
ifconfig net0 modinsert vpnmod@3
```

4 Use the same IP addressing format on all interfaces of the IPMP group.

If one interface is configured for IPv4, then all interfaces of the group must be configured for IPv4. For example, if you add IPv6 addressing to one interface, then all interfaces in the IPMP group must be configured for IPv6 support.

5 Determine the type of failure detection that you want to implement.

For example, if you want to implement probe-based failure detection, then you must configure test addresses on the underlying interfaces. For related information, see"Types of Failure Detection in IPMP" on page 264.

6 Ensure that all interfaces in the IPMP group are connected to the same local network.

For example, you can configure Ethernet switches on the same IP subnet into an IPMP group. You can configure any number of interfaces into an IPMP group.

Note – You can also configure a single interface IPMP group, for example, if your system has only one physical interface. For related information, see "Types of IPMP Interface Configurations" on page 262.

---

7 Ensure that the IPMP group does not contain interfaces with different network media types.

The interfaces that are grouped together should be of the same interface type, as defined in `/usr/include/net/if_types.h`. For example, you cannot combine Ethernet and Token ring interfaces in an IPMP group. As another example, you cannot combine a Token bus interface with asynchronous transfer mode (ATM) interfaces in the same IPMP group.

8 For IPMP with ATM interfaces, configure the ATM interfaces in LAN emulation mode.

IPMP is not supported for interfaces using Classical IP over ATM.

▼ How to Configure an IPMP Group by Using DHCP

In the current IPMP implementation, IPMP groups can be configured with Dynamic Host Configuration Protocol (DHCP) support.

A multiple-interfaced IPMP group can be configured with active-active interfaces or active-standby interfaces. For related information, see "Types of IPMP Interface Configurations" on page 262. The following procedure describes steps to configure an active-standby IPMP group by using DHCP.

Before You Begin Make sure that IP interfaces that will be in the IPMP group have been correctly configured over the system's network datalinks. You can create an IPMP interface even if underlying IP interfaces do not yet exist. However, subsequent configurations on this IPMP interface will fail.

For procedures to configure links and IP interfaces, see "IP Interface Configuration (Tasks)" on page 168. For information about configuring IPv6 interfaces, see "Configuring an IPv6 Interface" in *Oracle Solaris Administration: IP Services*.

Additionally, if you are using a SPARC system, configure a unique MAC address for each interface. For procedures, see "SPARC: How to Ensure That the MAC Address of an Interface Is Unique" on page 169.

Finally, if you are using DHCP, make sure that the underlying interfaces have infinite leases. Otherwise, in case of a group failure, the test addresses will expire and the IPMP daemon will then disable probe-based failure detection and link-based failure detection will be used. If link-based failure detection discovers that the interface is functioning, the daemon might erroneously report that the interface has been repaired. For more information about configuring DHCP, refer to Chapter 13, "Planning for DHCP Service (Tasks)," in *System Administration Guide: IP Services*.

Note – You cannot use IPMP if the active network profile on the system is a reactive profile. Before configuring IPMP groups, if necessary enable the `DefaultFixed` profile to switch to a fixed network configuration profile. For procedures, see "Profiles and Configuration Tools" on page 144.

---

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Create an IPMP interface.

```
# ipadm create-ipmp ipmp-interface
```

where

ipmp-interface specifies the name of the IPMP interface. You can assign any meaningful name to the IPMP interface. As with any IP interface, the name consists of a string and a number, such as `ipmp0`.

3 Create the underlying IP interfaces if these do not exist yet.

```
# ipadm create-ip under-interface
```

where *under-interface* refers to the IP interface that you will add to the IPMP group.

4 Add underlying IP interfaces that will contain test addresses to the IPMP group.

```
# ipadm add-ipmp -i under-interface1 [-i under-interface2 ...] ipmp-interface
```

You can create as many IP interfaces to the IPMP group as are available in the system.

5 Have DHCP configure and manage the data addresses on the IPMP interface.

```
# ipadm create-addr -T dhcp addrobj
```

The *addrobj* represents an address object and uses the format *interface*/*string*. The *interface* in this step is the IPMP interface. The string can be any user-defined string. Thus, if you have multiple data addresses on the IPMP interface, the corresponding address objects would be *ipmp-interface*/*string1*, *ipmp-interface*/*string2*, *ipmp-interface*/*string3*, and so on.

6 Have DHCP manage the test addresses in the underlying interfaces.

You need to issue the following command for each underlying interface of the IPMP group.

```
# ipadm create-addr -T dhcp addrobj
```

The *addrobj* represents an address object and uses the format *interface*/*string*. The *interface* in this step is the underlying interface. The string can be any user-defined string. Thus, if you have multiple underlying interfaces for the IPMP group, the corresponding address objects would be *under-interface1*/*string*, *ipmp-interface2*/*string*, *ipmp-interface3*/*string*, and so on.

Example 15–1 Configuring an IPMP Group With DHCP

This example shows how to configure an active-standby IPMP group with DHCP and is based on the following scenario:

- Three underlying interfaces for the IPMP group will be configured over their respective datalinks `net0`, `net1`, and `net2` are designated members of the IPMP group.
- The IPMP interface `itops0` shares the same name with the IPMP group.
- `net2` is the designated standby interface.
- To use probe-based failure detection, all the underlying interfaces are assigned test addresses.

```
# ipadm create-ipmp itops0

# ipadm create-ip net0
# ipadm create-ip net1
# ipadm create-ip net2

# ipadm add-ipmp -i net0 -i net1 -i net2 itops0

# ipadm create-addr -T dhcp itops0/dhcp0
# ipadm create-addr -T dhcp itops0/dhcp1

# ipadm create-addr -T dhcp net0/test
# ipadm create-addr -T dhcp net2/test
# ipadm create-addr -T dhcp net3/test

# ipadm set-ifprop -p standby=on net2
```

How to Manually Configure an Active-Active IPMP Group

The following procedure describes steps to manually configure an active-active IPMP group.

Before You Begin Make sure that IP interfaces that will be in the prospective IPMP group have been correctly configured over the system's network datalinks. For procedures to configure links and IP interfaces, see "IP Interface Configuration (Tasks)" on page 168. For information about configuring IPv6 interfaces, see "Configuring an IPv6 Interface" in *Oracle Solaris Administration: IP Services*. You can create an IPMP interface even if underlying IP interfaces do not yet exist. However, subsequent configurations on this IPMP interface will fail.

Additionally, if you are using a SPARC system, configure a unique MAC address for each interface. For procedures, see "SPARC: How to Ensure That the MAC Address of an Interface Is Unique" on page 169.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Create an IPMP interface.

```
# ipadm create-ipmp ipmp-interface
```

where

ipmp-interface specifies the name of the IPMP interface. You can assign any meaningful name to the IPMP interface. As with any IP interface, the name consists of a string and a number, such as `ipmp0`.

3 Add underlying IP interfaces to the group.

```
# ipadm add-ipmp -i under-interface1 [-i underinterface2 ...] ipmp-interface
```

where *under-interface* refers to the underlying interface of the IPMP group. You can add as many IP interfaces as are available in the system.

Note – In a dual-stack environment, placing the IPv4 instance of an interface under a particular group automatically places the IPv6 instance under the same group as well.

---

4 Add data addresses to the IPMP interface.

```
# ipadm create-addr -T static IP-address addrobj
```

The *IP-address* can be in CIDR notation.

The *addrobj* must use the naming convention *ipmp-interface*/*any-string*. Thus, if the name of the IPMP interface is `ipmp0`, then the *addrobj* can be `ipmp0/dataaddr`.

5 **Add test addresses on the underlying interfaces.**

```
# ipadm create-addr -T static IP-address addrobj
```

The *IP-address* can be in CIDR notation.

The *addrobj* must use the naming convention *under-interface*/*any-string*. Thus, if the name of an underlying interface is `net0`, then the *addrobj* can be `net0/testaddr`.

Note – You need to configure a test address only if you want to use probe-based failure detection on a particular interface.

All test IP addresses in an IPMP group must use the same network prefix. The test IP addresses must belong to a single IP subnet.

How to Manually Configure an Active-Standby IPMP Group

For more information about standby interfaces, see "Types of IPMP Interface Configurations" on page 262. The following procedure configures an IPMP group where one interface is kept as a reserve. This interface is deployed only when an active interface in the group fails.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Create an IPMP interface.**

```
# ipadm create-ipmp ipmp-interface
```

where

ipmp-interface specifies the name of the IPMP interface. You can assign any meaningful name to the IPMP interface. As with any IP interface, the name consists of a string and a number, such as `ipmp0`.

3 **Add underlying IP interfaces to the group.**

```
# ipadm add-ipmp -i under-interface1 [-i underinterface2 ...] ipmp-interface
```

where *under-interface* refers to the underlying interface of the IPMP group. You can add as many IP interfaces as are available in the system.

Note – In a dual-stack environment, placing the IPv4 instance of an interface under a particular group automatically places the IPv6 instance under the same group as well.

---

4 **Add data addresses to the IPMP interface.**

```
# ipadm create-addr -T static IP-address  addrobj
```

The *IP-address* can be in CIDR notation.

The *addrobj* must use the naming convention *ipmp-interface*/*any-string*. Thus, if the name of the IPMP interface is `ipmp0`, then the *addrobj* can be `ipmp0/dataaddr`.

5 **Add test addresses on the underlying interfaces.**

```
# ipadm create-addr -T static IP-address  addrobj
```

The *IP-address* can be in CIDR notation.

The *addrobj* must use the naming convention *under-interface*/*any-string*. Thus, if the name of an underlying interface is `net0`, then the *addrobj* can be `net0/testaddr`.

Note – You need to configure a test address only if you want to use probe-based failure detection on a particular interface.

All test IP addresses in an IPMP group must use the same network prefix. The test IP addresses must belong to a single IP subnet.

---

6 **Configure one of the underlying interfaces as a standby interface.**

```
# ipadm set-ifprop -p standby=yes under-interface
```

Example 15–2 Configuring an Active-Standby IPMP Group

This example shows how to manually create an active-standby IPMP configuration. The example begins with creating the underlying interfaces.

```
# ipadm create-ip net0
# ipadm create-ip net1
# ipadm create-ip net2

# ipadm create-ipmp itops0

# ipadm add-ipmp -i net0 -i net1 -i net2 itops0
# ipadm create-addr -T static -a 192.168.10.10/24 itops0/v4add1
# ipadm create-addr -T static -a 192.168.10.15/24 itops0/v4add2

# ipadm create-addr -T static -a 192.168.85.30/24 net0/test
# ipadm create-addr -T static -a 192.168.85.32/24 net1/test
# ipadm create-addr -T static -a 192.168.85.34/24 net2/test

# ipadm set-ifprop -p standby=yes net2

# ipmpstat -g
GROUP     GROUPNAME   STATE      FDT        INTERFACES
itops0    itops0      ok         10.00s     net0 net1 (net2)
```

```
# ipmpstat -t
INTERFACE  MODE     TESTADDR         TARGETS
net0       routes   192.168.10.30    192.168.10.1
net1       routes   192.168.10.32    192.168.10.1
net2       routes   192.168.10.34    192.168.10.5
```

Maintaining IPMP Groups

This section contains tasks for maintaining existing IPMP groups and the interfaces within those groups. The tasks presume that you have already configured an IPMP group, as explained in "Configuring IPMP Groups" on page 279.

▼ How to Add an Interface to an IPMP Group

Before You Begin Make sure that the interface that you add to the group matches all the constraints to be in the group. For a list of the requirements of an IPMP group, see "How to Plan an IPMP Group" on page 279.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **If the underlying IP interface does not yet exist, create the interface.**

   # ipadm create-ip *interface*

3. **Add the IP interface to the IPMP group.**

   # ipadm add-ipmp -i *under-interface ipmp-interface*

Example 15–3 Adding an Interface to an IPMP Group

To add the interface `net4` to the IPMP group `itops0`, you would type the following commands:

```
# ipadm create-ip net4
# ipadm add-ipmp -i net4 itops0
# ipmpstat -g
GROUP   GROUPNAME   STATE      FDT       INTERFACES
itops0  itops0      ok         10.00s    net0 net1 net4
```

▼ How to Remove an Interface From an IPMP Group

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Remove the interface from the IPMP group.

```
# ipadm remove-ipmp -i under-interface[, -i under-interface, ...] ipmp-interface
```

You can remove as many underlying interfaces in a single command as required. Removing all underlying interfaces does not delete the IPMP interface. Rather exists as an empty IPMP interface or group.

Example 15–4 Removing an Interface From a Group

To remove the interface `net4` from the IPMP group `itops0`, you would type the following command:

```
# ipadm remove-ipmp net4 itops0
# ipmpstat -g
GROUP   GROUPNAME   STATE      FDT       INTERFACES
itops0  itops0      ok         10.00s    net0 net1
```

How to Add or Remove IP Addresses

You use the `ipadm create-addr` subcommand to add addresses or the `ipadm delete-addr` subcommand to remove addresses from interfaces. In the current IPMP implementation, test addresses are hosted on the underlying IP interface, while data addresses are assigned to the IPMP interface. The following procedures describes how to add or remove IP addresses that are either test addresses or data addresses.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Add or remove data addresses.

- To add data addresses to the IPMP group, type the following command:

  ```
  # ipadm create-addr -T static -a ip-address addrobj
  ```

  The *addrobj* uses the naming convention *ipmp-interface*/*user-string*.
- To remove an address from the IPMP group, type the following command:

  ```
  # ipadm delete-addr addrobj
  ```

  The *addrobj* uses the naming convention *inder-interface*/*user-string*.

3 Add or remove test addresses.

- To assign a test address to an underlying interface of the IPMP group, type the following command:

  ```
  # ipadm create-addr -T static ip-address adrobj
  ```

- To remove a test address from an underlying interface of the IPMP group, type the following command:

  # ipadm delete-addr *addrobj*

Example 15–5 Removing a Test Address From an Interface

The following example uses the configuration of `itops0` in Example 15–2. The step removes the test address from the interface `net1`. In this example, assume that the test address is named `net1/test1`

```
# ipmpstat -t
INTERFACE      MODE     TESTADDR        TARGETS
net1      routes   192.168.10.30   192.168.10.1

# ipadm show-addr
ADDROBJ           TYPE     STATE     ADDR
lo0               static   ok        127.0.0.1/8
...
net1/test1        static   ok        192.168.10.30

# ipadm delete-addr net1/test1
```

How to Move an Interface From One IPMP Group to Another Group

You can place an interface in a new IPMP group when the interface belongs to an existing IPMP group. You do not need to remove the interface from the current IPMP group. When you place the interface in a new group, the interface is automatically removed from any existing IPMP group.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Move the interface to a new IPMP group.**

   # ipadm add-ipmp -i *under-interface ipmp-interface*

   where *under-interface* refers to the underlying interface that you want to move and *ipmp-interface* refers to the IPMP interface or group to which you want to move the underlying interface.

   Placing the interface in a new group automatically removes the interface from any existing group.

Example 15–6 Moving an Interface to a Different IPMP Group

This example assumes that the underlying interfaces of your group are `net0`, `net11`, and `net2`. To move `net0` to the IPMP group `cs-link1`, you would type the following:

```
# ipadm add-ipmp -i net0 ca-link1
```

This command removes the `net0` interface from IPMP group `itops0` and then puts `net0` to `cs-link1`.

How to Delete an IPMP Group

Use this procedure if you no longer need a specific IPMP group.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Identify the IPMP group and the underlying IP interfaces.**

   ```
   # ipmpstat -g
   ```

3. **Delete all IP interfaces that currently belong to the IPMP group.**

   # ipadm remove-ipmp -i *under-interface*[**, -i** *under-interface*, **...]** *ipmp-interface*

   Note – To successfully delete an IPMP interface, no IP interface must exist as part of the IPMP group.

4. **Delete the IPMP interface.**

   # ipadm delete-ipmp *ipmp-interface*

   After you delete the IPMP interface, any IP address that is associated with the interface is deleted from the system.

Example 15–7 Deleting an IPMP Interface

To delete the interface `itops0` that has the underlying IP interface `net0` and `net1`, you would type the following commands:

```
# ipmpstat -g
GROUP   GROUPNAME   STATE      FDT        INTERFACES
itops0  itops0      ok         10.00s     net0 net1

# ipadm remove-ipmp -i net0 -i net1 itops0

# ipadm delete-ipmp itops0
```

Configuring for Probe-Based Failure Detection

Probe-based failure detection involves the use of target systems, as explained in "Probe-Based Failure Detection" on page 264. In identifying targets for probe-based failure detection, the `in.mpathd` daemon operates in two modes: router target mode or multicast target mode. In the router target mode, the multipathing daemon probes targets that are defined in the routing table. If no targets are defined, then the daemon operates in multicast target mode, where multicast packets are sent out to probe neighbor hosts on the LAN.

Preferably, you should set up host targets for the `in.mpathd` daemon to probe. For some IPMP groups, the default router is sufficient as a target. However, for some IPMP groups, you might want to configure specific targets for probe-based failure detection. To specify the targets, set up host routes in the routing table as probe targets. Any host routes that are configured in the routing table are listed before the default router. IPMP uses the explicitly defined host routes for target selection. Thus, you should set up host routes to configure specific probe targets rather than use the default router.

To set up host routes in the routing table, you use the `route` command. You can use the `-p` option with this command to add persistent routes. For example, `route -p add` adds a route which will remain in the routing table even after you reboot the system. The `-p` option thus allows you to add persistent routes without needing any special scripts to recreate these routes every system startup. To optimally use probe-based failure detection, make sure that you set up multiple targets to receive probes.

The sample procedure that follows shows the exact syntax to add persistent routes to targets for probe-based failure detection. For more information about the options for the `route` command, refer to the `route(1M)` man page.

Consider the following criteria when evaluating which hosts on your network might make good targets.

- Make sure that the prospective targets are available and running. Make a list of their IP addresses.
- Ensure that the target interfaces are on the same network as the IPMP group that you are configuring.
- The netmask and broadcast address of the target systems must be the same as the addresses in the IPMP group.
- The target host must be able to answer ICMP requests from the interface that is using probe-based failure detection.

How to Manually Specify Target Systems for Probe-Based Failure Detection

1. **Log in with your user account to the system where you are configuring probe-based failure detection.**

2. **Add a route to a particular host to be used as a target in probe-based failure detection.**

   ```
   $ route -p add -host destination-IP gateway-IP -static
   ```

   where *destination-IP* and *gateway-IP* are IPv4 addresses of the host to be used as a target. For example, you would type the following to specify the target system `192.168.10.137`, which is on the same subnet as the interfaces in IPMP group `itops0`:

   ```
   $ route -p add -host 192.168.10.137 192.168.10.137 -static
   ```

   This new route will be automatically configured every time the system is restarted. If you want to define only a temporary route to a target system for probe-based failure detection, then do not use the `-p` option.

3. **Add routes to additional hosts on the network to be used as target systems.**

How to Select Which Failure Detection Method to Use

By default, probe-based failure detection can only be performed by using test addresses. If the NIC driver supports it, link-based failure detection is also enabled automatically.

You cannot disable link-based failure detection if this method is supported by the NIC driver. However, you can select which type of probe-based failure detection to implement.

1. **To use only transitive probing, perform the following steps:**

   a. **Use the appropriate SMF commands to switch on the IPMP property `transitive-probing`.**

      ```
      # svccfg -s svc:/network/ipmp setprop config/transitive-probing=true
      # svcadm refresh  svc:/network/ipmp:default
      ```

      For more information about setting this property, see the `in.mpathd(1M)` man page.

   b. **Remove any existing test addresses that have been configured for the IPMP group.**

2. **To use only test addresses to probe for failure, perform the following steps:**

   a. **If necessary, turn off transitive probing.**

      ```
      # svccfg -s svc:/network/ipmp setprop config/transitive-probing=false
      # svcadm refresh  svc:/network/ipmp:default
      ```

b. Assign test addresses to the underlying interfaces of the IPMP group.

How to Configure the Behavior of the IPMP Daemon

Use the IPMP configuration file `/etc/default/mpathd` to configure the following system-wide parameters for IPMP groups.

- `FAILURE_DETECTION_TIME`
- `TRACK_INTERFACES_ONLY_WITH_GROUPS`
- `FAILBACK`

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Edit the `/etc/default/mpathd` file.

Change the default value of one or more of the three parameters.

a. Type the new value for the `FAILURE_DETECTION_TIME` parameter.

`FAILURE_DETECTION_TIME=`*n*

where *n* is the amount of time in seconds for ICMP probes to detect whether an interface failure has occurred. The default is 10 seconds.

b. Type the new value for the `FAILBACK` parameter.

`FAILBACK=[yes | no]`

- *yes*– The *yes* value is the default for the failback behavior of IPMP. When the repair of a failed interface is detected, network access fails back to the repaired interface, as described in "Detecting Physical Interface Repairs" on page 267.
- *no* – The *no* value indicates that data traffic does not move back to a repaired interface. When a failed interfaces is detected as repaired, the `INACTIVE` flag is set for that interface. This flag indicates that the interface is currently not to be used for data traffic. The interface can still be used for probe traffic.

  For example, the IPMP group `ipmp0` consists of two interfaces, `net0` and `net1`. In the `/etc/default/mpathd` file, the *FAILBACK=no* parameter is set. If `net0` fails, then it is flagged as `FAILED` and becomes unusable. After repair, the interface is flagged as `INACTIVE` and remains unusable because of the *FAILBACK=no* setting.

  If `net1` fails and only `net0` is in the `INACTIVE` state, then `net0`'s `INACTIVE` flag is cleared and the interface becomes usable. If the IPMP group has other interfaces that are also in the `INACTIVE` state, then any one of these `INACTIVE` interfaces, and not necessarily `net0`, can be cleared and become usable when `net1` fails.

c. Type the new value for the `TRACK_INTERFACES_ONLY_WITH_GROUPS` parameter.

```
TRACK_INTERFACES_ONLY_WITH_GROUPS=[yes | no]
```

Note – For information about this parameter and the anonymous group feature, see "Failure Detection and the Anonymous Group Feature" on page 266.

- *yes*– The *yes* value is the default for the behavior of IPMP. This parameter causes IPMP to ignore network interfaces that are not configured into an IPMP group.
- *no* – The *no* value sets failure and repair detection for *all* network interfaces, regardless of whether they are configured into an IPMP group. However, when a failure or repair is detected on an interface that is not configured into an IPMP group, no action is triggered in IPMP to maintain the networking functions of that interface. Therefore, the*no* value is only useful for reporting failures and does not directly improve network availability.

3 Restart the `in.mpathd` daemon.

```
# pkill -HUP in.mpathd
```

Recovering an IPMP Configuration With Dynamic Reconfiguration

This section contains procedures that relate to administering systems that support dynamic reconfiguration (DR).

▼ How to Replace a Physical Card That Has Failed

This procedure explains how to replace a physical card on a system that supports DR. The procedure assumes the following conditions:

- Your system's active NCP is `DefaultFixed`. Refer to the section *Dynamic Reconfiguration and Network Configuration Profiles* in "How NWAM Works With Other Oracle Solaris Networking Technologies" on page 40 for information about using DR if your system's active NCP is not `DefaultFixed`.
- The system's IP interfaces are `net0` and `net1`.
- Both interfaces belong to the IPMP group, `itops0`.
- The underlying interface `net0` contains a test address.
- The underlying interface `net0` has failed, and you need to remove `net0`'s card, `bge`.
- You are replacing the `bge` card with a `e1000g` card.

Before You Begin The procedures for performing DR vary with the type of system. Therefore, make sure that you complete the following:

- Ensure that your system supports DR.
- Consult the appropriate manual that describes DR procedures on your system. For Sun hardware from Oracle, all systems that support DR are servers. To locate current DR documentation on Sun systems, search for "dynamic reconfiguration" on `http://www.oracle.com/technetwork/indexes/documentation/index.html`.

Note – The steps in the following procedure refer only to aspects of DR that are specifically related to IPMP and the use of link names. The procedure does not contain the complete steps to perform DR. For example, some layers beyond the IP layer require manual configuration steps, such as for ATM and other services, if the configuration is not automated. Follow the appropriate DR documentation for your system.

For the detailed procedure to replace NICs, refer to "How to Replace a Network Interface Card With Dynamic Reconfiguration" on page 161.

---

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Perform the appropriate DR steps to remove the failed NIC from the system.

For example, you would remove the `bge` card.

3 Attach the replacement NIC to the system.

For example, you would install the `e1000g` card on the same location that the `bge` card used to occupy. The `e1000g`'s datalink assumes the name `net0` and inherits that datalink's configuration.

4 Complete the DR process by enabling the new NIC's resources to become available for use.

For example, you use the `cfgadm` command to perform this step. For more information, see the `cfgadm`(1M) man page.

After this step, the new interface is configured with the test address, added as an underlying interface of the IPMP group, and deployed either as an active or a standby interface, all depending on the persistent configurations of `net0`. The kernel can then allocate data addresses to this new interface according to the persistent configurations of the IPMP interface, `itops0`.

Monitoring IPMP Information

The following procedures use the `ipmpstat` command, enabling you to monitor different aspects of IPMP groups on the system. You can observe the status of the IPMP group as a whole or its underlying IP interfaces. You can also verify the configuration of data and test addresses for the group. Information about failure detection is also obtained by using the `ipmpstat` command. For more details about the `ipmpstat` command and its options, see the `ipmpstat(1M)` man page.

By default, host names are displayed on the output instead of the numeric IP addresses, provided that the host names exist. To list the numeric IP addresses in the output, use the `-n` option together with other options to display specific IPMP group information.

Note – In the following procedures, use of the `ipmpstat` command does not require system administrator privileges, unless stated otherwise.

▼ How to Obtain IPMP Group Information

Use this procedure to list the status of the various IPMP groups on the system, including the status of their underlying interfaces. If probe-based failure detection is enabled for a specific group, the command also includes the failure detection time for that group.

- **Display the IPMP group information.**

```
$ ipmpstat -g
GROUP   GROUPNAME   STATE      FDT        INTERFACES
itops0  itops0      ok         10.00s     net0 net1
acctg1  acctg1      failed     --         [net3 net4]
field2  field2      degraded   20.00s     net2 net5 (net7) [net6]
```

`GROUP` Specifies the IPMP interface name. In the case of an anonymous group, this field will be empty. For more information about anonymous groups, see the `in.mpathd(1M)` man page.

`GROUPNAME` Specifies the name of the IPMP group. In the case of an anonymous group, this field will be empty.

`STATE` Indicates a group's current status, which can be one of the following:

- `ok` indicates that all underlying interfaces of the IPMP group are usable.
- `degraded` indicates that some of the underlying interfaces in the group are unusable.
- `failed` indicates that all of the group's interfaces are unusable.

`FDT` Specifies the failure detection time, if failure detection is enabled. If failure detection is disabled, this field will be empty.

INTERFACES Specifies the underlying interfaces that belong to the group. In this field, active interfaces are listed first, then inactive interfaces, and finally unusable interfaces. The status of the interface is indicated by the manner in which it is listed:

- *interface* (without parentheses or brackets) indicates an active interface. Active interfaces are those interfaces that being used by the system to send or receive data traffic.
- *(interface)* (with parentheses) indicates a functioning but inactive interface. The interface is not in use as defined by administrative policy.
- *[interface]* (with brackets) indicates that the interface is unusable because it has either failed or been taken offline.

How to Obtain IPMP Data Address Information

Use this procedure to display data addresses and the group to which each address belongs. The displayed information also includes which address is available for use, depending on whether the address has been toggled by the `ipadm [up-addr/down-addr]` command. You can also determine on which inbound or outbound interface an address can be used.

- **Display the IPMP address information.**

```
$ ipmpstat -an
ADDRESS         STATE    GROUP      INBOUND     OUTBOUND
192.168.10.10   up       itops0     net0   net0 net1
192.168.10.15   up       itops0     net1   net0 net1
192.0.0.100     up       acctg1     --          --
192.0.0.101     up       acctg1     --          --
128.0.0.100     up       field2     net2       net2 net7
128.0.0.101     up       field2     net7       net2 net7
128.0.0.102     down     field2     --          --
```

ADDRESS Specifies the hostname or the data address, if the `-n` option is used in conjunction with the `-a` option.

STATE Indicates whether the address on the IPMP interface is `up`, and therefore usable, or `down`, and therefore unusable.

GROUP Specifies the IPMP IP interface that hosts a specific data address.

INBOUND Identifies the interface that receives packets for a given address. The field information might change depending on external events. For example, if a data address is down, or if no active IP interfaces remain in the IPMP group, this field will be empty. The empty field indicates that the system is not accepting IP packets that are destined for the given address.

OUTBOUND Identifies the interface that sends packets that are using a given address as a source address. As with the `INBOUND` field, the `OUTBOUND` field information might

also change depending on external events. An empty field indicates that the system is not sending out packets with the given source address. The field might be empty either because the address is down, or because no active IP interfaces remain in the group.

▼ How to Obtain Information About Underlying IP Interfaces of a Group

Use this procedure to display information about an IPMP group's underlying IP interfaces. For a description of the corresponding relationship between the NIC, datalink, and IP interface, see "The Network Stack in Oracle Solaris" on page 22.

- **Display the IPMP interface information.**

```
$ ipmpstat -i
INTERFACE   ACTIVE   GROUP      FLAGS      LINK       PROBE      STATE
net0        yes      itops0     --mb---    up         ok         ok
net1        yes      itops0     -------    up         disabled   ok
net3        no       acctg1     -------    unknown    disabled   offline
net4        no       acctg1     is-----    down       unknown    failed
net2        yes      field2     --mb---    unknown    ok         ok
net6        no       field2     -i-----    up         ok         ok
net5        no       filed2     -------    up         failed     failed
net7        yes      field2     --mb---    up         ok         ok
```

`INTERFACE` Specifies each underlying interface of each IPMP group.

`ACTIVE` Indicates whether the interface is functioning and is in use (`yes`) or not (`no`).

`GROUP` Specifies the IPMP interface name. In the case of anonymous groups, this field will be empty. For more information about anonymous groups, see the `in.mpathd(1M)` man page.

`FLAGS` Indicates the status of the underlying interface, which can be one or any combination of the following:

- `i` indicates that the `INACTIVE` flag is set for the interface and therefore the interface is not used to send or receive data traffic.
- `s` indicates that the interface is configured to be a standby interface.
- `m` indicates that the interface is designated by the system to send and receive IPv4 multicast traffic for the IPMP group.
- `b` indicates that the interface is designated by the system to receive broadcast traffic for the IPMP group.
- `M` indicates that the interface is designated by the system to send and receive IPv6 multicast traffic for the IPMP group.
- `d` indicates that the interface is down and therefore unusable.

- `h` indicates that the interface shares a duplicate physical hardware address with another interface and has been taken offline. The `h` flag indicates that the interface is unusable.

`LINK` Indicates the state of link-based failure detection, which is one of the following states:

- `up` or `down` indicates the availability or unavailability of a link.
- `unknown` indicates that the driver does not support notification of whether a link is `up` or `down` and therefore does not detect link state changes.

`PROBE` Specifies the state of the probe–based failure detection for interfaces that have been configured with a test address, as follows:

- `ok` indicates that the probe is functional and active.
- `failed` indicates that probe-based failure detection has detected that the interface is not working.
- `unknown` indicates that no suitable probe targets could be found, and therefore probes cannot be sent.
- `disabled` indicates that no IPMP test address is configured on the interface. Therefore probe-based failure detection is disabled.

`STATE` Specifies the overall state of the interface, as follows:

- `ok` indicates that the interface is online and working normally based on the configuration of failure detection methods.
- `failed` indicates that the interface is not working because either the interface's link is down, or the probe detection has determined that the interface cannot send or receive traffic.
- `offline` indicates that the interface is not available for use. Typically, the interface is switched offline under the following circumstances:
  - The interface is being tested.
  - Dynamic reconfiguration is being performed.
  - The interface shares a duplicate hardware address with another interface.
- `unknown` indicates the IPMP interface's status cannot be determined because no probe targets can be found for probe-based failure detection.

▼ How to Obtain IPMP Probe Target Information

Use this procedure to monitor the probe targets that are associated with each IP interface in an IPMP group.

- **Display the IPMP probe targets.**

```
$ ipmpstat -nt
INTERFACE   MODE          TESTADDR         TARGETS
net0        routes        192.168.85.30    192.168.85.1 192.168.85.3
net1        disabled      --               --
net3        disabled      --               --
net4        routes        192.1.2.200      192.1.2.1
net2        multicast     128.9.0.200      128.0.0.1 128.0.0.2
net6        multicast     128.9.0.201      128.0.0.2 128.0.0.1
net5        multicast     128.9.0.202      128.0.0.1 128.0.0.2
net7        multicast     128.9.0.203      128.0.0.1 128.0.0.2

$ ipmpstat -nt
INTERFACE   MODE         TESTADDR         TARGETS
net3        transitive   <net1>           <net1> <net2> <net3>
net2        transitive   <net1>           <net1> <net2> <net3>
net1        routes       172.16.30.100    172.16.30.1
```

`INTERFACE` Specifies the underlying interfaces of the IPMP group.

`MODE` Specifies the method for obtaining the probe targets.

- `routes` indicates that the system routing table is used to find probe targets.
- `mcast` indicates that multicast ICMP probes are used to find targets.
- `disabled` indicates that probe-based failure detection has been disabled for the interface.
- `transitive` indicates that transitive probing is used to for failure detection, as shown in the second example. Note that you cannot implement probe-based failure detection by simultaneously using transitive probes and test addresses. If you do not want to use test addresses, then you must switch on transitive probing. If you do not want to use transitive probing, then you must configure test addresses. For an overview, see "Probe-Based Failure Detection" on page 264.

`TESTADDR` Specifies the hostname or, if the `-n` option is used in conjunction with the `-t` option, the IP address that is assigned to the interface to send and receive probes.

If transitive probing is used, then the interface names refer to the underlying IP interfaces that are not actively used to receive data. The names also indicate that the transitive test probes are being sent out with the source address of these specified interfaces. For active underlying IP interfaces that receive data, an IP address that is displayed indicates the source address of outgoing ICMP probes.

Note – If an IP interface is configured with both IPv4 and IPv6 test addresses, the probe target information is displayed separately for each test address.

TARGETS — Lists the current probe targets in a space-separated list. The probe targets are displayed either as hostnames or IP addresses, if the `-n` is used in conjunction with the `-t` option.

How to Observe IPMP Probes

Use this procedure to observe ongoing probes. When you issue the command to observe probes, information about probe activity on the system is continuously displayed until you terminate the command with `Ctrl-C`. You must have Primary Administrator privileges to run this command.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Display the information about ongoing probes.

```
# ipmpstat -pn
TIME    INTERFACE   PROBE     NETRTT   RTT      RTTAVG     TARGET
0.11s   net0        589       0.51ms   0.76ms   0.76ms     192.168.85.1
0.17s   net4        612       --       --       --         192.1.2.1
0.25s   net2        602       0.61ms   1.10ms   1.10ms     128.0.0.1
0.26s   net6        602       --       --       --         128.0.0.2
0.25s   net5        601       0.62ms   1.20ms   1.00ms     128.0.0.1
0.26s   net7        603       0.79ms   1.11ms   1.10ms     128.0.0.1
1.66s   net4        613       --       --       --         192.1.2.1
1.70s   net0        603       0.63ms   1.10ms   1.10ms     192.168.85.3
^C

# ipmpstat -pn
TIME    INTERFACE   PROBE      NETRTT   RTT      RTTAVG     TARGET
1.39S   net4        t28        1.05ms   1.06ms   1.15ms     <net1>
1.39s   net1        i29        1.00ms   1.42ms   1.48ms     172.16.30.1
```

TIME — Specifies the time a probe was sent relative to when the `ipmpstat` command was issued. If a probe was initiated prior to `ipmpstat` being started, then the time is displayed with a negative value, relative to when the command was issued.

INTERFACE — Specifies the interface on which the probe is sent.

PROBE — Specifies the identifier that represents the probe. If transitive probing is used for failure detection, the identifier is prefixed with either `t` for transitive probes or `i` for ICMP probes.

NETRTT — Specifies the total network round-trip time of the probe and is measured in milliseconds. `NETRTT` covers the time between the moment when the IP module sends the probe and the moment the IP module receives the `ack` packets from the target. If the `in.mpathd` daemon has determined that the probe is lost, then the field will be empty.

RTT	Specifies the total round-trip time for the probe and is measured in milliseconds. RTT covers the time between the moment the daemon executes the code to send the probe and the moment the daemon completes processing the ack packets from the target. If the in.mpathd daemon has determined that the probe is lost, then the field will be empty. Spikes that occur in the RTT which are not present in the NETRTT might indicate that the local system is overloaded.
RTTAVG	Specifies the probe's average round-trip time over the interface between local system and target. The average round-trip time helps identify slow targets. If data is insufficient to calculate the average, this field will be empty.
TARGET	Specifies the hostname or, if the -n option is used in conjunction with -p, the target address to which the probe is sent.

How to Customize the Output of the ipmpstat Command in a Script

When you use the ipmpstat, by default, the most meaningful fields that fit in 80 columns are displayed. In the output, all the fields that are specific to the option that you use with the ipmpstat command are displayed, except in the case of the ipmpstat -p syntax. If you want to specify the fields to be displayed, then you use the -o option in conjunction with other options that determine the output mode of the command. This option is particularly useful when you issue the command from a script or by using a command alias

- **To customize the output, issue one of the following commands:**
  - To display selected fields of the ipmpstat command, use the -o option in combination with the specific output option. For example, to display only the GROUPNAME and the STATE fields of the group output mode, you would type the following:

    ```
    $ ipmpstat -g -o groupname,state

    GROUPNAME  STATE
    itops0     ok
    accgt1     failed
    field2     degraded
    ```

  - To display all the fields of a given ipmpstat command, use the following syntax:

    ```
    # ipmpstat -o all
    ```

How to Generate Machine Parseable Output of the `ipmpstat` Command

You can generate machine parseable information by using the `ipmpstat -P` syntax. The `-P` option is intended to be used particularly in scripts. Machine-parseable output differs from the normal output in the following ways:

- Headers are omitted.
- Fields are separated by colons (`:`).
- Fields with empty values are empty rather than being filled with the double dash (`--`).
- In the case of multiple fields being requested, if a field contains a literal colon (`:`) or back slash (`\`), these can be escaped or excluded by prefixing these characters with a back slash (`\`) .

To correctly use the `ipmpstat -P` syntax, observe the following rules:

- Use the `-o` *option fields* together with the `-P` option.
- Never use `-o all` with the `-P` option.

Ignoring either one of these rules will cause `ipmpstat -P` to fail.

- **To display in machine parseable format the group name, the failure detection time, and the underlying interfaces, you would type the following:**

```
$ ipmpstat -P -o -g groupname,fdt,interfaces
itops0:10.00s:net0 net1
acctg1::[net3 net4]
field2:20.00s:net2 net7 (net5) [net6]
```

The group name, failure detection time, and underlying interfaces are group information fields. Thus, you use the `-o -g` options together with the `-P` option.

Example 15–8 Using `ipmpstat -P` in a Script

This sample script displays the failure detection time of a particular IPMP group.

```
getfdt() {
         ipmpstat -gP -o group,fdt | while IFS=: read group fdt; do
             [[ "$group" = "$1" ]] && { echo "$fdt"; return; }
         done
     }
```

◆ ◆ ◆ CHAPTER 16

Exchanging Network Connectivity Information With LLDP

This chapter describes how to enable systems to exchange system and network connectivity information throughout the local network by using the Link Layer Discovery Protocol (LLDP).

Overview of LLDP in Oracle Solaris

LLDP is used to advertise information throughout a local network for purposes of topology discovery. With this protocol, a system can advertise connectivity and management information to other systems on the network. This information can include system capabilities, management addresses, and other relevant information. This protocol also enables that same system to receive similar information about other the systems that are on the same local network.

In Oracle Solaris, support for LLDP also includes Data Center Bridging (DCB) for exchanging configuration information about DCB features such as priority-based flow control (PFC) and the Application TLV.

With LLDP, the system administrator can easily detect faulty system configurations particularly in complex networks that include virtual local area networks (VLANs), link aggregations, and other link types.

Components of an LLDP Implementation

LLDP is implemented with the following components:

- The LLDP package must be installed to enable the LLDP feature. This package delivers the LLDP daemon, command-line utilities, service manifest and scripts, and other components that are required for LLDP to operate.
- The `lldp` service is enabled by the `svcadm` command. This service manages the LLDP daemon and is responsible for starting, stopping, restarting, or refreshing the daemon. The service is disabled by default. Therefore, to use LLDP, the service must first be enabled

globally for the system. After the `lldp` service is enabled and the daemon is started, then the LLDP functionality can be enabled on individual links as determined by the system administrator.

- The `lldpadm` command administers LLDP on individual links and is used, for example, to configure the operating mode of LLDP, to specify Time-Length-Value (TLV) units that will be transmitted, and to configure DCB application information. Specifically, the command is used to set per-agent LLDP properties as well as global LLDP properties. The general subcommands of the `lldpadm` command parallel those of the `dladm` and `ipadm` commands.
  - `lldpadm set-*` specifies the action to be performed in which one or more values are set for a given LLDP property.
  - `lldpadm show-*` displays the values that are set for a specified LLDP property.
  - `lldpadm reset-*` returns the configuration of a specified LLDP property to its default values.

  Use of these subcommands is illustrated in subsequent sections. For more information about the `lldpadm` command, refer to the `lldpadm(1M)` man page.
- The LLDP library (`liblldp.so`) provides APIs that can be used to retrieve LLDP information on a link, to parse LLDP packets, and to perform other functions.
- LLDP agents are LLDP instances that are associated with the physical NICs where LLDP is enabled. An LLDP agent controls LLDP behavior on the associated NIC. LLDP agents can be configured only on physical NICs.
- The LLDP daemon (`lldpd`) functions as a manager of the LLDP agents on the system. It also interacts with `snmpd`, the daemon for the Simple Network Management Protocol (SNMP), to retrieve LLDP information that is received on the system through SNMP. In addition, the daemon posts `sysevents` information as well as responds to queries from the LLDP library.

The following section describes the LLDP agent in more detail.

Functions of the LLDP Agent

The LLDP agent transmits as well as receives LLDP packets, which are also called *protocol data units (PDUs)*. The agent manages and stores the information contained in these packets in two types of data stores:

- Local management information base, or local MIB. This data store contains network information that pertains to the specific link on which the LLDP agent is enabled. A local MIB contains both common and unique information. For example, the chassis ID is common information that is shared among all the LLDP agents on the system. However, port numbers are different for the system's datalinks. Thus, each agent manages its own local MIB.
- Remote MIB. Information in this data store pertains to other systems on the local network.

Configuring How the LLDP Agent Operates

The LLDP agent can be configured to operate in the following modes:

- In transmit only (`txonly`) mode, the agent does not process incoming LLDP packets. Therefore, the remote MIB is empty.
- In receive only (`rxonly`) mode, the agent processes only incoming LLDP packets and stores the information in remote MIBs. However, no information from the local MIB is being transmitted.
- In both transmit and receive (`both`) mode, the agent transmits as well as receives LLDP packets. Both types of MIBs are actively in use. This mode also automatically enables DCB features that are supported by the underlying link.
- In disabled (`disable`) mode, the agent does not exist.

How to Enable LLDP

This procedure enables LLDP on your system for the first time.

1. **Install the LLDP package.**

   ```
   # pkg install lldp
   ```

   Note – For an overview about Oracle Solaris packages and how to install them, see Chapter 12, "Managing Software Packages (Tasks)," in *Oracle Solaris Administration: Common Tasks*.

2. **Start the LLDP service on the system.**

   ```
   # svcadm enable svc:/network/lldp:default
   ```

3. **Identify the datalink on which you want to enable LLDP.**

4. **Set the mode of operation for that datalink's LLDP agent.**

   ```
   # lldpadm set-agentprop -p mode=value  agent
   ```

   where *value* can be one of the modes of operation, and *agent* uses the name of the datalink on which LLDP is enabled.

   Note – The subcommands of the `lldpadm` command can be typed in abbreviated form to facilitate the command's use. For example, `lldpadm set-agentprop` can instead be typed as `lldpadm set-ap`. Refer to the `lldpadm(1M)` man page for the subcommands and their abbreviated forms.

5. **To confirm the LLDP agent's mode of operation, type the following command:**

   ```
   # lldpadm show-agentprop -p mode agent
   ```

6. **To disable an LLDP agent, use either of the following commands:**

   - **`lldpadm set-agentprop -p mode=disable`** *agent*
   - **`lldpadm reset-agentprop -p mode`** *agent*

7. **To turn off LLDP in the entire system, type the following:**

```
# svcadm disable svc:/network/lldp:default
```

Example 16–1 Enabling LLDP on Multiple Datalinks

In this example, a system has two datalinks, `net0` and `net1`, and LLDP is enabled in different modes for each LLDP agent. One agent operates by both transmitting and receiving LLDP packets while the other agent only transmits LLDP packets.

```
# svcadm enable svc:/network/lldp:default
# lldpadm set-agentprop -p mode=both net0
# lldpadm set-agentprop -p mode=txonly net1
```

Configuring What Information To Advertise

The LLDP agent transmits system and connectivity information in LLDP packets or LLDPDUs. Such packets would contain information units that are individually formatted in Type-Length-Value (TLV) format. Thus, the information units are also called TLV units. Certain TLV units are mandatory and are included in LLDP packets by default when LLDP is enabled. The mandatory TLV units are as follows:

- Chassis ID
- Port ID
- TTL (time to live)
- End of PDU

The Chassis ID is the information that is generated by the `hostid` command while the Port ID is the MAC address of the physical NIC. Multiple LLDP agents can be enabled in a single system depending on the number of links. The combined Chassis ID and Port ID uniquely identifies an agent and distinguishes it from other agents on the system.

You cannot use the `lldpadm` command to exclude any of the mandatory TLV units from LLDP packets.

Optional TLV units can be added to an LLDP packet. These optional TLV units are means for vendors to insert vendor-specific TLV units to be advertised. The TLV units are identified by individual organization unique identifiers (OUIs) and are typed according to whether these OUIs are IEEE 802.1 specifications or IEEE 802.3 specifications. LLDP agent properties that correspond to each TLV type are created so that you can set the values for each type.

The following table lists the TLV types or groups, their corresponding property names, the TLV units for each property, and their descriptions.

TABLE 16–1 TLV Units That Can Be Enabled for an LLDP Agent

TLV Type	Property Name	TLVs	Description
Basic management	`basic-tlv`	`sysname`, `portdesc`, `syscapab`, `sysdesc`, `mgmtaddr`	Specifies the system name, port description, system capability, system description, and management address to be advertised
802.1 OUI	`dot1-tlv`	`vlanname`, `pvid`, `linkaggr`, `pfc`, `appln`	Specifies the VLAN name, port VLAN ID, link aggregation, port description, and application TLV to be advertised
802.3 OUI	`dot3-tlv`	`max-framesize`	Specifies the maximum frame size to be advertised
Oracle-specific OUI (which is defined as `0x0003BA`)	`virt-tlv`	`vnic`	Specifies the VNIC to be advertised if a virtual network is configured

You configure any one of these properties to specify the TLV units to be included in the packets when LLDP is enabled.

How to Specify TLV Units for LLDP Packets

This procedure shows how to add a TLV unit to be advertised in the LLDP packet. To set TLV units for LLDP packets, you use the `lldpadm set-agentprop` subcommand.

1 If necessary, identify the LLDP agent property that can contain the TLV unit that you want to add.

This subcommand also displays the TLV units that are already set for each property.

```
# lldpadm show-agentprop agent
```

Without specifying the property, this subcommand displays all the LLDP agent properties and their TLV values.

2 Add the TLV unit to the property.

```
# lldpadm set-agentprop -p property[+|-]=value[,...] agent
```

The +|- qualifiers are used for properties that accept multiple values. These qualifiers enable you to add (+) or remove (-) values from the list. If you do not use the qualifiers, then the value that you set replaces all the values that were previously defined for the property.

3 (Optional) Display the new values for the property.

```
# lldpadm show-agentprop -p property agent
```

Example 16–2 Adding Optional TLV Units to the LLDP Packet

In this example, the LLDP agent `net0` is already configured to advertise VLAN information in the packet. You want to include system capabilities, link aggregation, and network virtualization information to be advertised as well. However, you want to remove the VLAN description from the packet.

```
# lldpadm show-agentprop net0
# lldpadm set-agentprop -p dot1-tlv+=linkaggr net0
AGENT   PROPERTY   PERM   VALUE          DEFAULT   POSSIBLE
net0    mode       rw     both           disable   txonly,rxonly,both,
                                                   disable
net0    basic-tlv  rw     sysname,        none     none,portdesc,
                          sysdesc                  sysname,sysdesc,
                                                   syscapab,mgmtaddr,
                                                   all
net0    dot1-tlv   rw     vlanname,       none     none,vlanname,pvid,
                          pvid,pfc                 linkaggr,pfc,appln,
                                                   all
net0    dot3-tlv   rw     max-framesize   none     none, max-framesize,
                                                   all
net0    virt-tlv   rw     none            none     none,vnic,all


# lldpadm set-agentprop -p basic-tlv+=syscapab,dot1-tlv+=linkaggr,virt-tlv=vnic net0
# lldpadm set-agentprop -p dot1-tlv-=pfc net0
# lldpadm show-agentprop -p net0
AGENT   PROPERTY   PERM   VALUE          DEFAULT   POSSIBLE
net0    mode       rw     both           disable   txonly,rxonly,both,
                                                   disable
net0    basic-tlv  rw     sysname,        none     none,portdesc,
                          sysdesc,                 sysname,sysdesc,
                          syscapab                 syscapab,mgmtaddr,
                                                   all
net0    dot1-tlv   rw     vlanname,       none     none,vlanname,pvid,
                          pvid,                    linkaggr,pfc,appln,
                          linkaggr                 all
net0    dot3-tlv   rw     max-framesize   none     none, max-framesize,
                                                   all
net0    virt-tlv   rw     vnic            none     none,vnic,all
```

Managing TLV Units

Each TLV unit has properties that you can further configure with specific values. When that TLV unit is enabled as an LLDP agent's property, then that TLV unit is advertised in the network only with the specified values. Consider, for example, the TLV value `syscapab` which advertises a system's capabilities. These capabilities can potentially include support for routers, bridges, repeaters, telephones, and other devices. However, you can set `syscapab` so that only those capabilities that are actually supported in your specific system, such as routers and bridges, are advertised.

The procedure for managing TLVs depends on whether you are configuring global TLVs or per-agent TLVs.

Global TLVs apply to all the LLDP agents on the system. The following table displays the global TLV values and their corresponding possible configurations.

TABLE 16–2 Global TLVs and Their Properties

TLV Name	TLV Property Name	Possible Property Values	Value Description
`syscapab`	`supported`	`other`, `repeater`, `bridge`, `wlan-ap`, `router`, `telephone`, `docsis-cd`, `station`, `cvlan`, `sylvan`, `tpmr`	Represent the primary supported functions of the system. Default values are `router`, `station`, and `bridge`.
	`enabled`	Subset of the values listed for `supported`	Represents the enabled functions of the system.
`mgmtaddr`	`ipaddr`	`ipv4` or `ipv6`	Specifies the type of IP addresses that will be associated with the local LLDP agent. The addresses will be used to reach higher layer entities and will assist in discovery by network management. Only one type can be specified.

TLV units that cannot have global values are managed at the LLDP agent level. With *per-agent TLV units*, the values that you provide are used when the TLV unit is enabled for transmission by a specific LLDP agent.

The following table displays the TLV values and their corresponding possible configurations for an LLDP agent.

TABLE 16–3 Per-Agent TLV Units and Their Properties

TLV Name	TLV Property Name	Possible Property Values	Value Description
`pfc`	`willing`	`on`, `off`	Sets an LLDP agent to accept or reject configuration information from a remote machine.
`appln`	`apt`	Values are taken from the information that is defined in the Application Priority Table.	Configures the Application Priority Table. This table contains the list of application TLV units and their corresponding priorities. The application is identified by the `id/selector` pair. The contents of the table use the following format: `id/selector/priority`

The following procedure shows how to define global TLV values. For a discussion about how to define per-agent TLV units, see "Data Center Bridging" on page 313.

How to Define Global TLV Values

This procedure shows how to provide global values for specific TLV units. To set global TLV values, you use the `llpadm set-tlvprop` subcommand.

1. **Configure the appropriate TLV property to contain the values that you want to advertise.**

   For reference, see Table 16–2.

   ```
   # lldpadm set-tlvprop -p tlv-property=value[,value,value,...] tlv
   ```

2. **(Optional) Display the values of the property that you have just configured.**

   ```
   # lldpadm show-tlvprop
   ```

Example 16–3 Specifying the System's Capabilities and the Management IP Address

This example accomplishes two objectives:

- Provides specific information about the system's capabilities to be advertised in the LLDP packet. To achieve this objective, both `supported` and `enabled` properties of the `syscapab` TLV unit must be configured.
- Provides the management IP address that is used in the advertisement.

```
# llpdadm set-tlvprop -p supported=bridge,router,repeater syscapab
# llpdadm set-tlvprop -p enabled=router syscapab
# llpdadm set-tlvprop -p ipaddr=192.168.1.2 mgmtaddr
# llpdadm show-tlvprop
TLVNAME    PROPERTY   PERM  VALUE        DEFAULT         POSSIBLE
syscapab   supported  rw    bridge,      bridge,router,  other,router,
                            router,      station         repeater,bridge,
                            repeater                     wlan-ap,telephone,
                                                         docis-cd,station,
                                                         cvlan,svlan,tpmr
syscapab   enabled    rw    router       none            bridge,router,
                                                         repeater
mgmtaddr   ipaddr     rw    192.162.1.2  none            --
```

Data Center Bridging

To support Fibre Channel over Ethernet (FCoE) traffic, the LLDP implementation in Oracle Solaris includes data center bridging (DCB) support.

In networks that use traditional Ethernet for traffic exchange, an ongoing risk is that packets might be dropped when the network is busy. A key requirement for FCoE traffic is that no packet drops can occur during transmission. With support for Data Center Bridging Exchange (DCBx), the priority—based flow control (PFC) TLV, and the Application TLV, dropped packets are avoided.

PFC extends the standard PAUSE frame to include the priority information for packets. Typically, a PAUSE frame is sent on a link when traffic is heavy to enable the receiving end to process packets it has already received. With PFC, instead of transmitting a PAUSE frame to stop all traffic on the link, traffic is paused according to priorities defined for the packets. A PFC frame can be sent for the priority for which traffic needs to be paused. The sender stops traffic for that specific priority, while traffic for other priorities are unaffected. After a specified time, another PFC frame is sent to signal that the paused traffic can resume.

PFC configuration information is exchanged between peer stations by means of DCBx. If peers in a traffic exchange have matching PFC configurations, then PFC can pause or resume traffic transmission as needed. To enable different packets to be assigned different priorities, the Application TLV is used to define priority information. If peers have mismatching PFC configurations, the PFC TLV can be customized to accept the other peer's configuration, as shown in the procedure that follows.

Data Center Bridging is a specific case to illustrate how to configure per-agent TLV units as explained in "Managing TLV Units" on page 311.

▼ How to Set Per-Agent TLV Values

This procedure shows how to set TLV values at the LLDP agent level by using the `llpadm set-agenttlvprop` subcommand.

1. **Configure the appropriate TLV property to contain the values that you want to advertise by a given LLDP agent.**

   For reference, see Table 16–3.

   ```
   # lldpadm set-agenttlvprop -p tlv-property[+|-]=value[,value,value,...] -a agent tlv-name
   ```

2. **(Optional) Display the values of the property that you have just configured.**

   ```
   # lldpadm show-agenttlvprop
   ```

Example 16–4 Enabling the LLDP Agent to Accept Information and Specifying TLV Application Priorities

This example shows how the `pfc` as well as the `appln` TLV values are customized. The TLV units in this example specify how DCB operates for FCoE traffic. The system is configured to accept the peer's PFC configuration in case the local configuration does not match the peer's configuration. The example also shows how the priority is set for the LLDP agent's application TLV.

```
# lldpadm set-agenttlvprop -p willing=on -a net0 pfc
# lldpadm set-agenttlvprop -p apt=8906/1/4 -a net0 appln
# lldpadm show-agenttlvprop
AGENT   TLVNAME   PROPERTY   PERM   VALUE      DEFAULT   POSSIBLE
net0    pfc       willing    rw     on         off       on,off
net0    appln     apt        rw     8906/1/4   --        --
```

Monitoring LLDP Agents

The `lldpadm show-agent` subcommand displays the complete information that is advertised by an LLDP agent. Relative to a given system, the advertisement can be information about the local system that is transmitted to the rest of the network. Or, the advertisement can be information that is received by the system from other systems on the same network.

▼ How to Display Advertisements

This procedure shows how to display the information that is being advertised by an LLDP agent. The information can be either local or remote. *Local* information comes from the local system. *Remote* information comes from other systems on the network, which is received by the local system.

- **Use the `lldpadm show-agent` subcommand with the appropriate option to display the information what you want.**

  - **To display local information that is advertised by the LLDP agent, type the following command:**

    ```
    # lldpadm show-agent -l agent
    ```

- **To display remote information that is received by the LLDP agent, type the following command:**

  ```
  # lldpadm show-agent -r agent
  ```

- **To display either the local or the remote information in detail, type the following command:**

  ```
  # lldpadm show-agent -[l|r]v agent
  ```

Example 16–5 Obtaining LLDP Agent Information That Is Advertised

The following example shows how to display the information that is being advertised locally or remotely by an LLDP agent. By default, the information is displayed in short form. By using the `-v` option, you can obtain verbose or detailed information.

```
# lldpadm show-agent -l net0
AGENT   CHASSISID   PORTID
net0    004bb87f    00:14:4f:01:77:5d

# lldpadm show-agent -lv net0
                      Agent: net0
         Chassis ID Subtype: Local(7)
            Port ID Subtype: MacAddress(3)
                    Port ID: 00:14:4f:01:77:5d
           Port Description: net0
               Time to Live: 81 (seconds)
                System Name: hosta.example.com
         System Description: SunOS 5.11 dcb-clone-x-01-19-11 i86pc
     Supported Capabilities: bridge,router
       Enabled Capabilities: router
         Management Address: 192.168.1.2
         Maximum Frame Size: 3000
               Port VLAN ID: --
               VLAN Name/ID: vlan25/25
        VNIC PortID/VLAN ID: 02:08:20:72:71:31
    Aggregation Information: Capable, Not Aggregated
                 PFC Willing: --
                     PFC Cap: --
                     PFC MBC: --
                  PFC Enable: --
Application(s) (ID/Sel/Pri): --
    Information Valid Until: 117 (seconds)

# lldpdadm show-agent -r net0
AGENT   SYSNAME   CHASSISID   PORTID
net0    hostb     0083b390    00:14:4f:01:59:ab

# llpdadm show-agent -rv net0
                      Agent: net0
         Chassis ID Subtype: Local(7)
            Port ID Subtype: MacAddress(3)
                    Port ID: 00:14:4f:01:59:ab
           Port Description: net0
               Time to Live: 121 (seconds)
                System Name: hostb.example.com
         System Description: SunOS 5.11 dcb-clone-x-01-19-11 i86pc
```

```
     Supported Capabilities: bridge,router
       Enabled Capabilities: router
         Management Address: 192.168.1.3
         Maximum Frame Size: 3000
               Port VLAN ID: --
               VLAN Name/ID: vlan25/25
        VNIC PortID/VLAN ID: 02:08:20:72:71:31
    Aggregation Information: Capable, Not Aggregated
                PFC Willing: --
                    PFC Cap: --
                    PFC MBC: --
                 PFC Enable: --
Application(s) (ID/Sel/Pri): --
    Information Valid Until: 117 (seconds)
```

How to Display LLDP Statistics

You can display LLDP statistics to obtain information about LLDP packets that are being advertised by the local system or by remote systems. The statistics refer to significant events that involve LLDP packet transmission and reception.

1 **To display all the statistics about LLDP packet transmission and reception, use the following command:**

```
# lldpadm show-agent -s agent
```

2 **To display selected statistics information, use the `-o` option.**

```
# lldpadm show-agent -s -o field[,field,...]agent
```

where *field* refers to any field name in the output of the `show-agent -s` command.

Example 16–6 Displaying LLDP Packet Statistics

This example shows how to display information about LLDP packet advertisement.

```
# lldpadm show-agent -s net0
AGENT IFRAMES IEER IDISCARD OFRAMES OLENERR TLVDISCARD TLVUNRECOG AGEOUT
net0        9    0        0      14       0          4          5      0
```

The command output provides the following information:

- `AGENT` specifies the name of the LLDP agent, which is identical to the datalink on which the LLDP agent is enabled.
- `IFRAMES`, `IEER`, and `IDISCARD` display information about packets being received, incoming packets with errors, and incoming packets that are dropped.
- `OFRAMES` and `OLENERR` refer to outgoing packets as well as packets that have length errors.
- `TLVDISCARD` and `TLVUNRECOG` display information about TLV units that are discarded as well as TLV units that are not recognized.
- `AGEOUT` refers to packets that have timed out.

The example indicates that out of 9 frames received into the system, 5 TLVs are unrecognized, possibly because of noncompliance with standards. The example also shows that 14 frames were transmitted by the local system to the network.

PART III

Network Virtualization and Resource Management

◆ ◆ ◆ CHAPTER 17

Introducing Network Virtualization and Resource Control (Overview)

This chapter explains the basic concepts involved in network virtualization and resource control. The following topics are covered:

- Network virtualization
- Types of virtual networks
- Virtual machines and zones
- Resource control, including flow management
- Enhanced network observability

These features help you to manage flow control, improve system performance, and configure the network utilization needed to achieve OS virtualization, utility computing, and server consolidation.

For specific tasks, refer to the following chapters:

- Chapter 19, "Configuring Virtual Networks (Tasks)"
- Chapter 22, "Monitoring Network Traffic and Resource Usage"
- Chapter 20, "Using Link Protection in Virtualized Environments"
- Chapter 21, "Managing Network Resources"

Network Virtualization and Virtual Networks

Network virtualization is the process of combining hardware network resources and software network resources into a single administrative unit. The goal of network virtualization is to provide systems and users with efficient, controlled, and secure sharing of the networking resources.

The end product of network virtualization is the *virtual network*. Virtual networks are classified into two broad types, external and internal. *External virtual networks* consist of several local networks that are administered by software as a single entity. The building blocks of classic external virtual networks are switch hardware and VLAN software technology. Examples of external virtual networks include large corporate networks and data centers.

An *internal virtual network* consists of one system using virtual machines or zones that are configured over at least one pseudo-network interface. These containers can communicate with each other as though on the same local network, providing a virtual network on a single host. The building blocks of the virtual network are *virtual network interface cards or virtual NICs (VNICs)* and virtual switches. Oracle Solaris network virtualization provides the internal virtual network solution.

You can combine networking resources to configure both internal and external virtual networks. For example, you can configure individual systems with internal virtual networks onto LANs that are part of a large, external virtual network. The network configurations that are described in this part include examples of combined internal and external virtual networks.

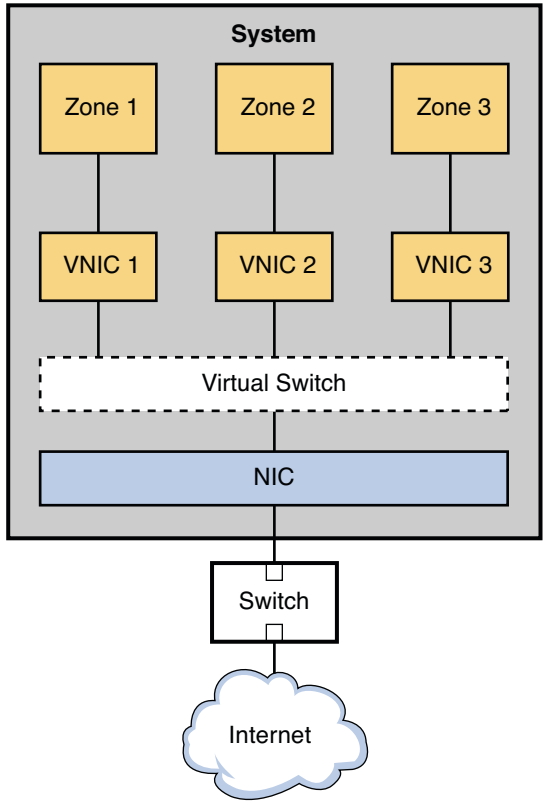
Parts of the Internal Virtual Network

An internal virtual network built on Oracle Solaris contains the following parts:

- At least one network interface card, or NIC.
- A virtual NIC, or VNIC, which is configured on top of the network interface
- A virtual switch, which is configured at the same time as the first VNIC on the interface.
- A container, such as a zone or virtual machine , which is configured on top of the VNIC.

The next figure shows these parts and how they fit together on a single system.

FIGURE 17–1 VNIC Configuration for a Single Interface



The figure shows a single system with one NIC. The NIC is configured with three VNICs. Each VNIC supports a single zone. Therefore, Zone 1, Zone 2, and Zone 3 are configured over VNIC 1, VNIC 2, and VNIC 3, respectfully. The three VNICs are virtually connected to one virtual switch. This switch provides the connection between the VNICs and the physical NIC upon which the VNICs are built. The physical interface provides the system with its external network connection.

Alternatively, you can create a virtual network based on the etherstub. Etherstubs are purely software and do not require a network interface as the basis for the virtual network.

A *VNIC* is a virtual network device with the same datalink interface as a physical interface. You configure VNICs on top of a physical interface. For the current list of physical interfaces that support VNICs, refer to the Network Virtualization and Resource Control FAQ (`http://hub.opensolaris.org/bin/view/Project+crossbow/faq`). You can configure up to 900 VNICs on a single physical interface. When VNICs are configured, they behave like physical NICs. In addition, the system's resources treat VNICs as if they were physical NICs.

Each VNIC is implicitly connected to a *virtual switch* that corresponds to the physical interface. The virtual switch provides the same connectivity between VNICs on a virtual network that switch hardware provides for the systems connected to a switch's ports.

In accordance with Ethernet design, if a switch port receives an outgoing packet from the host connected to that port, that packet cannot go to a destination on the same port. This design is a drawback for systems that are configured with zones or virtual machines. Without network virtualization, outgoing packets from a virtual machine or a zone with an exclusive stack cannot be passed to another virtual machine or zone on the same system. The outgoing packets go through a switch port out onto the external network. The incoming packets cannot reach their destination zone or virtual machine because the packets cannot return through the same port as they were sent. Therefore, when virtual machines and zones on the same system need to communicate, a data path between the containers must open on the local machine. Virtual switches provide these containers with the method to pass packets.

How Data Travels Through a Virtual Network

Figure 17–1 illustrates a simple VNIC configuration for a virtual network on a single system.

When the virtual network is configured, a zone sends traffic to an external host in the same fashion as a system without a virtual network. Traffic flows from the zone, through the VNIC to the virtual switch, and then to the physical interface, which sends the data out onto the network.

But what happens if one zone on a virtual network wants to send packets to another zone on the virtual network, given the previously mentioned Ethernet restrictions? As shown in Figure 17–1, suppose Zone 1 needs to send traffic to Zone 3? In this case packets pass from Zone 1 through its dedicated VNIC 1. The traffic then flows through the virtual switch to VNIC 3. VNIC 3 then passes the traffic to Zone 3. The traffic never leaves the system, and therefore never violates the Ethernet restrictions.

Who Should Implement Virtual Networks?

If you need to consolidate resources on Oracle's Sun servers, consider implementing VNICs and virtual networks. Consolidators at ISPs, telecommunications companies, and large financial institutions can use the following network virtualization features to improve the performance of their servers and networks.

- NIC hardware, including the powerful new interfaces that support hardware rings
- Multiple MAC addresses for the VNICs
- The large amount of bandwidth provided by newer interfaces

You can replace many systems with a single system that implements running multiple zones or virtual machines, without significantly losing separation, security, and flexibility.

What Is Resource Control?

Resource control is the process of allocating a system's resources in a controlled fashion. Oracle Solaris resource control features enable bandwidth to be shared among the VNICs on a system's virtual network. You can also use resource control features to allocate and manage bandwidth on a physical interface without VNICs and virtual machines. This section introduces the major features of resource control and briefly explains how these features work.

How Bandwidth Management and Flow Control Works

Searchnetworking.com (`http://searchnetworking.techtarget.com`) defines bandwidth as "the amount of data that can be carried from one point to another in a given time period (usually a second)." *Bandwidth management* enables you to assign a portion of the available bandwidth of a physical NIC to a consumer, such as an application or customer. You can control bandwidth on a per- application, per-port, per-protocol, and per-address basis. Bandwidth management assures efficient use of the large amount of bandwidth available from the new GLDv3 network interfaces.

Resource control features enable you implement a series of controls on an interface's available bandwidth. For example, you can set a *guarantee* of an interface's bandwidth to a particular consumer. That guarantee is the minimum amount of assured bandwidth allocated to the application or enterprise The allocated portion of bandwidth is known as a *share*. By setting up guarantees, you can allocate enough bandwidth for applications that cannot function properly without a certain amount of bandwidth. For example, streaming media and Voice over IP consume a great deal of bandwidth. You can use the resource control features to guarantee that these two applications have enough bandwidth to successfully run.

You can also set a *limit* on the share. The limit is the maximum allocation of bandwidth the share can consume. Using limits, you can contain non-critical services from taking away bandwidth from critical services.

Finally, you can prioritize among the various shares allotted to consumers. You can give highest priority to critical traffic, such as heartbeat packets for a cluster, and lower priority for less critical applications.

For example, application service providers (ASPs) can offer customers fee-based levels of service that are based on the bandwidth share that the customer purchases. As part of the service level agreement (SLA), each share is then guaranteed an amount of bandwidth, to not exceed the purchased limit. (For more information on service level agreements, see "Implementing Service-Level Agreements" in *Oracle Solaris Administration: IP Services*. Priority controls might be based on different tiers of the SLA, or different prices paid by the SLA customer.

Bandwidth usage is controlled through management of flows. A *flow* is a stream of packets that all have certain characteristics, such as the port number or destination address. These flows are

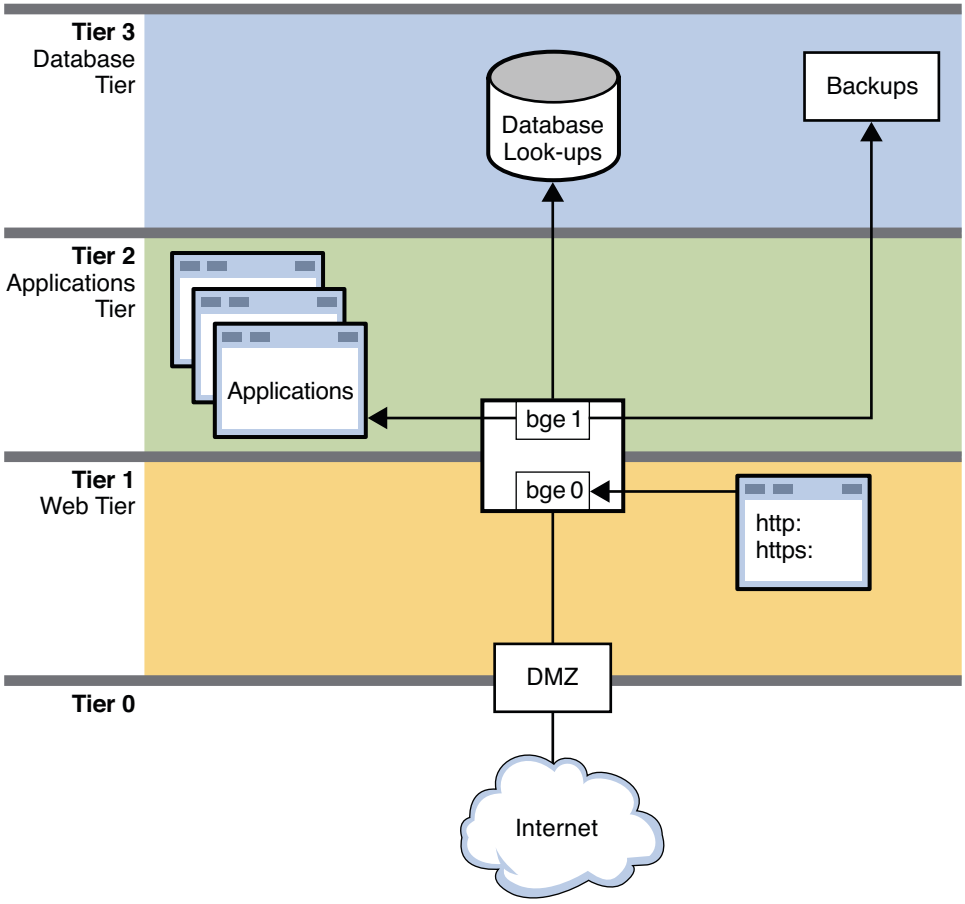
managed by transport, service, or virtual machine, including zones. Flows cannot exceed the amount of bandwidth that is guaranteed to the application or to the customer's purchased share.

When a VNIC or flow is assigned a guarantee, the VNIC is assured its designated bandwidth even if other flows or VNICs also use the interface. However, assigned guarantees are workable only if they do not exceed the maximum bandwidth of the physical interface.

Allocating Resource Control and Bandwidth Management on a Network

The following figure shows a corporate network topology that uses resource control to manage various applications.

FIGURE 17–2 Network With Resource Controls in Place



This figure shows a typical network topology that uses resource controls to improve network efficiency and performance. The network does not implement VNICs and containers, such as exclusive zones and virtual machines. However, VNICs and containers could be used on this network for consolidation and other purposes.

The network is divided into four tiers:

- **Tier 0** is the demilitarized zone (DMZ). This is a small local network that controls access to and from the outside world. Resource control is not used on the systems of the DMZ.
- **Tier 1** is the web tier and includes two systems. The first system is a proxy server that does filtering. This server has two interfaces, `bge0` and `bge1`. The `bge0` link connects the proxy server to the DMZ on Tier 0. The `bge0` link also connects the proxy server to the second system, the web server. The `http` and `https` services share the bandwidth of the web server

with other standard applications. Due to the size and critical nature of web servers, shares of `http` and `https` require guarantees and prioritization.

- **Tier 2** is the applications tier and also includes two systems. The second interface of the proxy server, `bge1`, provides the connection between the web tier and the applications tier. Through a switch, an applications server connects to `bge1` on the proxy server. The applications server requires resource control to manage the shares of bandwidth given to the various applications that are run. Critical applications that need a lot of bandwidth must be given higher guarantees and priorities than smaller, or less critical applications.
- **Tier 3** is the database tier. The two systems on this tier connect through a switch to the proxy server's `bge1` interface. The first system, a database server, needs to issue guarantees and to prioritize the various processes involved in database lookups. The second system is a backup server for the network. This system must consume a great deal of bandwidth during backups. However, backup activities are typically carried out overnight. Using resource controls, you can control when the backup processes have the highest bandwidth guarantees and highest priorities.

Who Should Implement Resource Control Features

Any system administrator who wants to improve a system's efficiency and performance should consider implementing the resource control features. Consolidators can delegate bandwidth shares in combination with VNICs to help balance the load of large servers. Server administrators can use share allocation features to implement SLA's, such as those offered by ASPs. Traditional system administrators can use the bandwidth management features to isolate and prioritize certain applications. Finally, share allocation makes it easy for you to observe bandwidth usage by individual consumers.

Observability Features for Network Virtualization and Resource Control

Network virtualization and resource control includes observability features to help you view resource usage before setting up controls such as VNICs and flows. In tandem with Oracle Solaris extended accounting, the resource control observability features allow you to accumulate systems statistics into logs. The observability features of network virtualization and resource control include:

- Ability to monitor a running system.
- Ability to log and report statistics.
- Extended accounting features to log historical data

The new `flowadm` command and extensions to the `dladm` and `netstat` commands implement the network virtualization observability features. You can use these commands to monitor current system usage and to gather statistical data into logs.

By analyzing the historical logs, you can determine the following:

- Where network resources can be consolidated from many systems to a single system, possibly with greater bandwidth through the new generation of network interfaces. Do this prior to setting up VNICs and virtual machines or exclusive zones.
- Which applications consume the most bandwidth. This information can help you to set up bandwidth management, so that critical applications are guaranteed the most bandwidth within a particular time slot. For example, you can guarantee a video stream the greatest amount of an interface's bandwidth for 20 hours a day. For a designated four hours a day, you can give highest priority to the system's backup program. Do this as part of bandwidth management implementation.
- How to much bill customers for bandwidth used. Application service providers and other businesses that rent out system space can use the Resource control observability features to determine usage by paying customers. Some businesses offer customers Service Level Agreements, wherein the customer buys a guaranteed percentage of bandwidth from the provider. The observability features let you view how much bandwidth each customer uses and bill for possible overages. Other businesses offer customers bandwidth on a per use basis. Here the observability features directly help in billing. Do this after you have implemented resource control and, possibly, VNICs and virtual machines on a system.

The next chapter, Chapter 18, “Planning for Network Virtualization and Resource Control,” contains scenarios that show where the observability features are used for planning consolidation and resource control.

◆ ◆ ◆ CHAPTER 18

Planning for Network Virtualization and Resource Control

This chapter contains information and example scenarios to help you evaluate and then design network virtualization and resource control solutions for your site. The chapter discusses the following scenarios:

- "Basic Virtual Network on a Single System" on page 332
- "Private Virtual Network on a Single System" on page 334
- "Interface-based Resource Control for a Traditional Network" on page 338

Each scenario contains "best usage" suggestions that explain the types of networks that best benefit from the particular scenario.

Network Virtualization and Resource Control Task Map

The following table describes tasks for configuring a virtual network and implementing resource controls on the network.

Task	Description	For Instructions
Design and plan a virtual network on a single host	Consolidate network services and applications offered by the local network onto a single host. This scenario is especially useful for consolidators and service providers.	"Planning and Designing a Virtual Network" on page 332
Design and plan for a private virtual network on a single host	Run a virtual network that does not allow public access. This scenario is recommended for system administrators who need to run a development environment.	"Private Virtual Network on a Single System" on page 334

Task	Description	For Instructions
Provide bandwidth management and resource control for systems on a per-interface basis.	Isolate, prioritize, and assign a specific amount of interface bandwidth for packet traffic. This scenario is useful for systems that handle heavy traffic for particular services, such as a web service or a database server.	"Interface-based Resource Control for a Traditional Network" on page 338

Planning and Designing a Virtual Network

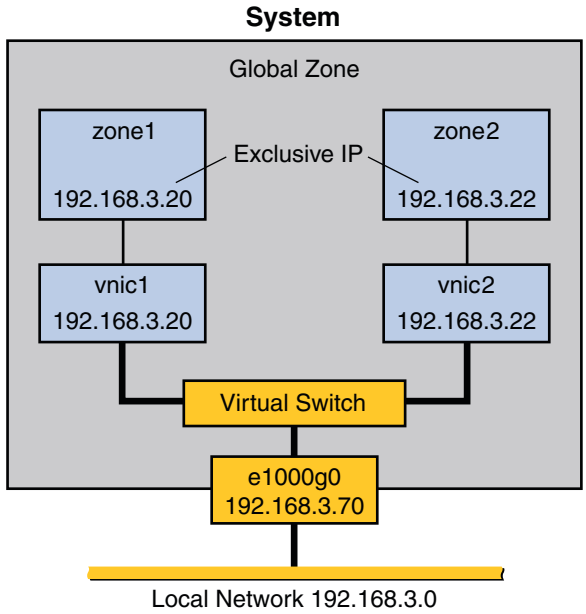
This section describes two different scenarios for configuring a virtual network. Look over the scenarios to help determine which most closely fits the needs of your site. Then use that scenario as the basis for designing your specific virtualization solution. The scenarios include:

- Basic virtual network of two zones, especially useful for consolidating network services from the local network onto a single host.
- Private virtual network, useful for a development environment where you isolate applications and services from the public network.

Basic Virtual Network on a Single System

Figure 18–1 shows the basic virtual network, or "network in a box" that is used in examples throughout the section "Configuring Components of Network Virtualization in Oracle Solaris" on page 342.

FIGURE 18–1 Virtual Network on a Single Host



This virtual network consists of the following:

- A single GLDv3 network interface `e1000g0`. This interface connects to the public network `192.168.3.0/24`. Interface `e1000g0` has the IP address `192.168.3.70`.
- A virtual switch, which is automatically configured when you create the first VNIC.
- Two VNICs. `vnic1` has the IP address `192.168.3.20`, and `vnic2` has the IP address `192.168.3.22`.
- Two exclusive IP zones to which the VNICs are assigned. `vnic1` is assigned to `zone1`, and `vnic2` is assigned to `zone2`.

The VNICs and zones in this configuration allow access to the public. Therefore, the zones can pass traffic beyond the `e1000g0` interface. Likewise, users on external networks can reach applications and services offered by the zones.

Best Uses for the Basic Virtual Network

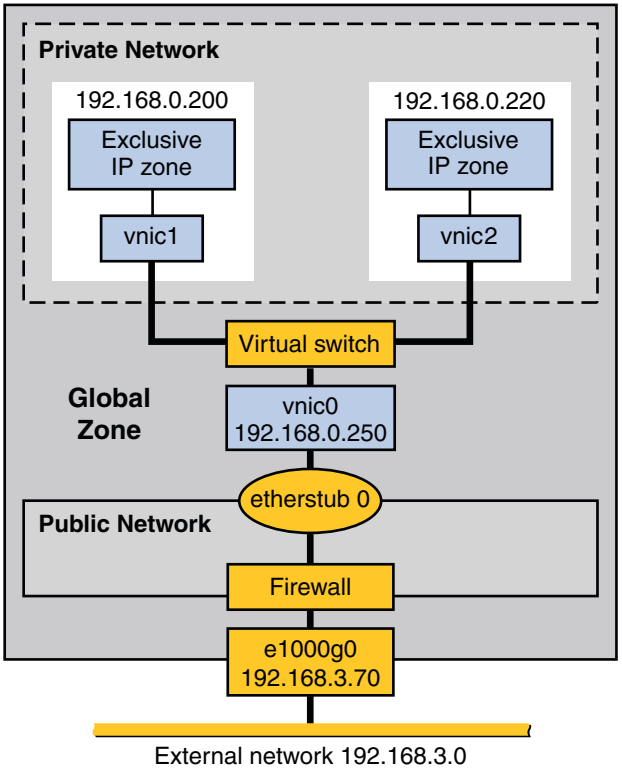
The network in a box scenario enables you to isolate processes and applications into individual virtual machines or zones on a single host. Furthermore, this scenario is expandable to include many containers, each of which could run a completely isolated set of applications. The scenario improves a system's efficiency and, by extension, the efficiency of the local network. Therefore, this scenario is ideal for the following users:

- Network consolidators and others who want to consolidate the services of a LAN onto a single system.
- Any site that rents out services to customers. You can rent out individual zones or virtual machines, observe traffic, and take statistics for performance measuring or for billing purposes on each zone in the virtual network.
- Any administrator who wants to isolate processes and applications to separate containers to improve system efficiency .

Private Virtual Network on a Single System

Figure 18–2 shows a single system with a private network behind packet filtering software that performs network address translation (NAT). This figure illustrates the scenario that is built in Example 19–5.

FIGURE 18–2 Private Virtual Network on a Single Host



The topology features a single system with a public network, including a firewall, and a private network built on an etherstub pseudo-interface. The public network runs in the global zone and consists of the following elements:

- GLDv3 network interface `e1000g0` with the IP address `192.168.3.70`.
- A firewall implemented in the IP Filter software. For an introduction to IP Filter, refer to "Introduction to IP Filter" in *Oracle Solaris Administration: IP Services*.
- `etherstub0`, a pseudo-interface upon which the virtual network topology is built. *Etherstubs* provide the ability to create a virtual network on a host. That network is totally isolated from the external network.

The private network consists of the following elements:

- A virtual switch which provides packet forwarding among the VNICs of the private network.
- `vnic0`, which is the VNIC for the global zone, and has the IP address `192.168.0.250`.
- `vnic1` with the IP address `192.168.0.200` and `vnic2` with the IP address `192.168.0.220`. All three VNICs are configured over `etherstub0`.
- `vnic1` is assigned to `zone1`, and `vnic2` is assigned to `zone2`.

Best Uses for a Private Virtual Network

Consider creating a private virtual network for a host that is used in a development environment. By using the etherstub framework, you can totally isolate software or features under development to the containers of the private network. Moreover, you can use firewall software for network address translation of outgoing packets that originate from the containers of the private network. The private network is a smaller version of the eventual deployment environment.

For More Information

- For procedures that configure a virtual network and implement the scenarios described in this chapter, go to "Creating a Private Virtual Network" on page 356.
- For conceptual information about VNICs and virtual networks, go to "Network Virtualization and Virtual Networks" on page 321.
- For conceptual information about zones, go to Chapter 15, "Introduction to Oracle Solaris Zones," in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*.
- For information about IP Filter, go to "Introduction to IP Filter" in *Oracle Solaris Administration: IP Services*.

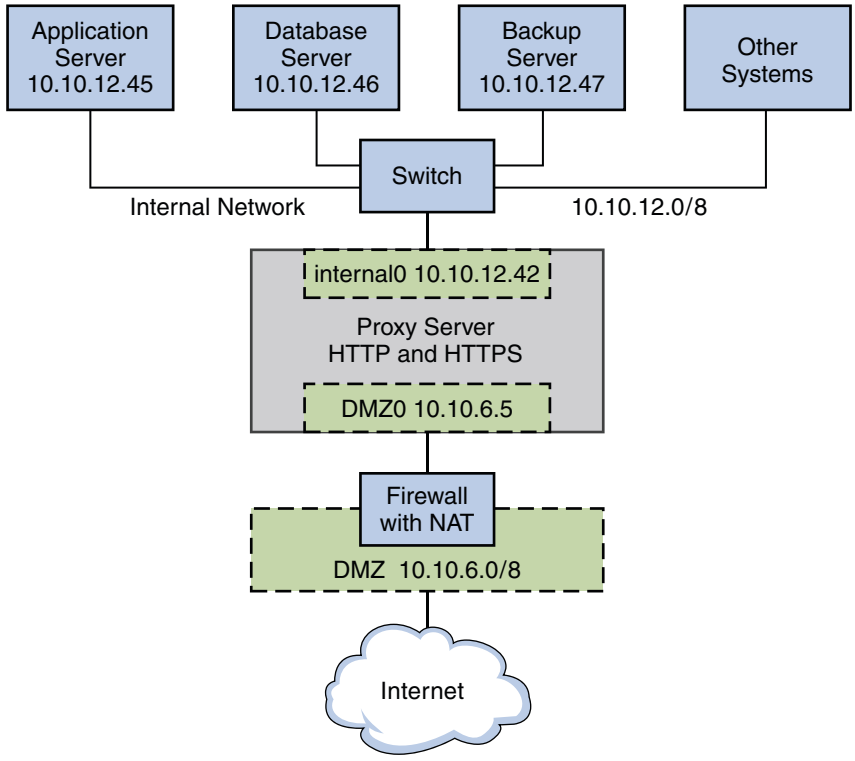
Implementing Controls on Network Resources

Network virtualization enables you to implement your network setup more efficiently at lower cost by constructing a network-in-a-box. To increase efficiency, you can also implement controls to determine how resources are being used by the networking processes. Link properties that are specifically related to network resources, such as rings, CPUs, and so on, can be customized to process network packets. In addition, you can also create flows to manage network usage. Network resource control is discussed in detail in Chapter 21, "Managing Network Resources."

Figure 18–3 shows the network topology for a small business that needs to manage the bandwidth on its proxy server. The proxy server offers a public web site as well as a proxy for internal clients that require services from various servers on the site's internal network.

Note – This scenario does not show how to configure flow control for a virtual network, and consequentially does not include VNICs. For flow control on a virtual network, refer to Flow Control for a Virtual Network.

FIGURE 18–3 Resource Control for a Proxy Server on a Traditional Network



The figure shows that the company has a public network, `10.10.6.0/8`, that also serves as a demilitarized zone (DMZ). A system on the DMZ provides name-to-address translation (NAT) through an IP Filter firewall. The company has a large system that functions as the proxy server. The system has two wired interfaces and 16 processor sets with IDs 0–16. This system is connected to the public network through the interface `nge0`, with IP address `l0 10.6.5`. The link name for the interface is `DMZ0`. Through `DMZ0`, the proxy server offers HTTP and HTTPS service through the company's public web site.

The figure also illustrates the company's internal network, `10.10.12.0/24`. The proxy server connects to the internal `10.10.12.0/8` network through interface `nge1`, with the IP address `10.10.12.42`. The link name for this interface is `internal0`. Through the `internal0` datalink, the proxy server operates on behalf of internal clients that request the services of an application server, `10.10.12.45`, database server, `10.10.12.46`, and backup server, `10.10.12.47`.

Interface-based Resource Control for a Traditional Network

Best Use of Interface–based Resource Control on a Traditional Network

Consider establishing flow control for heavily used systems, especially those with newer GLDv3 interfaces with large amounts of available bandwidth. Interface-based flow control improves the efficiency of the interface, the system, and potentially the network. You can apply flow control to any system on any type of network. Furthermore, if your goal is to improve network efficiency, you can separate various services into individual flows. This action assigns separate hardware and software resources to the individual flows, thus isolating them from other services on a particular system. After you establish flows, you can observe traffic for each flow and gather statistics. Thereafter, you can assign bandwidth amount and priorities to control usage on the interfaces.

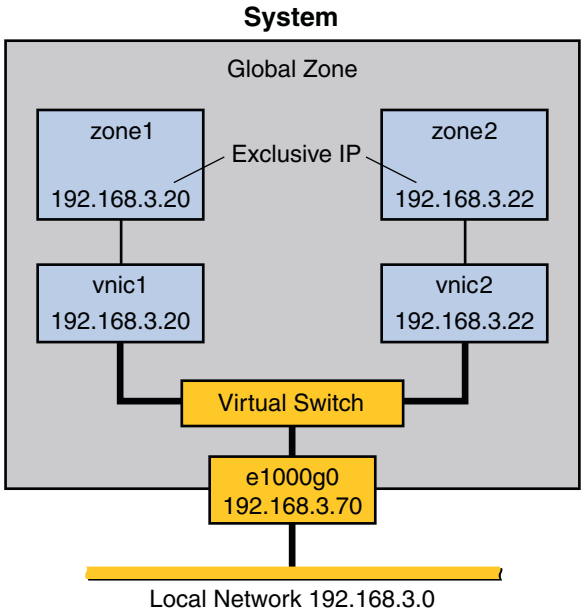
For More Information

- For tasks for implementing flow control, refer to Chapter 21, "Managing Network Resources"
- For conceptual information about bandwidth management and resource control, refer to "What Is Resource Control?" on page 325
- For detailed technical information, refer to the `dladm`(1M) and `flowadm`(1M)man pages.

Flow Control for the Virtual Network

This scenario shows how flow control is used within a virtual network, such as the basic virtual network that is introduced in "Basic Virtual Network on a Single System" on page 332.

FIGURE 18–4 Basic Virtual Network With Flow Controls



The topology is described in "Basic Virtual Network on a Single System" on page 332. Here a host has one network interface, `e1000g0`, with two VNICs, `vnic1` and `vnic2`. `zone1` is configured over `vnic1`, and `zone2` is configured over `vnic2`. Resource management for the virtual network involves creating flows on a per-VNIC basis. These flows define and isolate packets with similar characteristics, such as port number or IP address of the sending host. You assign bandwidth based on the usage policy for the system.

Another very common usage for flow controls on VNIC traffic is by companies that rent out zones. You create different service level agreements for customers, and rent out zones with a guaranteed amount of bandwidth. When you create flows on a per-zone basis, you can isolate and observe each customer's traffic and monitor bandwidth usage. If your service level agreement is based strictly on usage, you can use statistics and accounting features to bill customers.

Flow controls are effective for any network that requires bandwidth management for traffic over zones. Larger organizations, such as application service providers (ASPs) or Internet service providers (ISP), can take advantage of resource control for VNICs for data centers and for multiprocessor systems. The individual zones can be rented out to customers for different levels of service. Therefore, you could rent out `zone1` at the standard price and offer a standard bandwidth. Then, you could rent out `zone2` at a premium price and give that customer a high level of bandwidth.

How to Create a Usage Policy for Applications on a Virtual Network

1. **List the applications that you want to run on the host.**
2. **Determine which applications have historically used the most bandwidth or require the most bandwidth.**

   For example, the `telnet` application might not consume huge amounts of bandwidth on your system, but it could be heavily used. Conversely, database applications consume a huge amount of bandwidth, but might only be used on a sporadic basis. Consider monitoring traffic for these applications prior to assigning them to zones. You can use the statistical option of the `dladm show-link` command to gather statistics, as described in "Gathering Statistics About Network Traffic on Links" on page 399.
3. **Assign these applications to separate zones.**
4. **Create flows for any application running in `zone1` whose traffic you want to isolate and control.**
5. **Assign bandwidth to flows based on usage policies in place for your site.**

How to Create a Service Level Agreement for the Virtual Network

1. **Design a policy that offers different levels of services at different prices.**

   For example, you might create a basic, superior, and high levels of service, and price each level accordingly.
2. **Decide whether you want to charge customers on a monthly, per service level basis, or charge customers on an actual bandwidth consumed basis.**

   If you choose the latter pricing structure, you need to gather statistics on each customer's usage.
3. **Create a virtual network on a host, with containers for each customer.**

   A very common implementation is to give each customer their own zone running over a VNIC.
4. **Create flows that isolate traffic for each zone.**

   To isolate all traffic for the zone, you use the IP address that is assigned to the zone's VNIC.
5. **Assign bandwidth to each VNIC based on the service level purchased by the customer assigned to that VNIC's zone.**

◆ ◆ ◆ CHAPTER 19

Configuring Virtual Networks (Tasks)

This chapter contains tasks for configuring internal virtual networks, or "networks in a box." The topics that are covered include:

Virtual Networks Task Map

This table lists the tasks for configuring a virtual network, including links to the specific tasks. Note that not all tasks will apply to your virtual network scenario.

Task	Description	For Instructions
Create VNICs in the system.	Create one or more virtual network interfaces (VNICs). VNICs are the pseudo-interfaces upon which you build the virtual network	"How to Create a Virtual Network Interface" on page 343
Create etherstubs in the system.	Create one or more etherstubs. Etherstubs are virtual switches that allow you to create a private virtual network that is isolated from the larger network.	"How to Create Etherstubs" on page 345
Create zones to use VNICs.	Creates VNICs and new zones and configure these to create a basic virtual network.	"Creating New Zones for Use With VNICs" on page 347

Task	Description	For Instructions
Modify zones to use VNICs.	Changes an existing zone to become a virtual network.	"Modifying the Configuration of Existing Zones to Use VNICs" on page 352
Create a private virtual network.	Configures a private network that is isolated from the larger network by using etherstubs and VNICs.	"Creating a Private Virtual Network" on page 356
Remove VNICs.	Remove VNICs that were assigned to a zone without deleting the zone itself.	"How to Remove the Virtual Network Without Removing the Zones" on page 358

Configuring Components of Network Virtualization in Oracle Solaris

This section contains tasks for configuring the building blocks of network virtualization in Oracle Solaris. The following comprise the basic components:

- Virtual network interface cards (VNICs)
- Etherstubs

VNICs are pseudo interfaces that you create on top of datalinks. A VNIC has an automatically generated MAC address. Depending on the network interface in use, you can explicitly assign to a VNIC a MAC address other than the default address, as described in the `dladm(1M)` man page. You can create as many VNICs over a datalink as you require.

Etherstubs are pseudo Ethernet NICs which are managed by the system administrator. You can create VNICs over etherstubs instead of over physical links. VNICs over an etherstub become independent of the physical NICs in the system. With etherstubs, you can construct a private virtual network that is isolated both from the other virtual networks in the system and from the external network. For example, you want to create a network environment whose access is limited only to your company developers than to the network at large. Etherstubs can be used to create such an environment.

Etherstubs and VNICs are only a part of the virtualization features of Oracle Solaris. You typically use these components together with Oracle Solaris containers or zones. By assigning VNICs or etherstubs for use by zones, you can create a network within a single system.

How to Create a Virtual Network Interface

This procedure shows how to create a virtual network interface card (VNIC).

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **(Optional) To view information about the system's available physical interfaces, type the following command:**

```
# dladm show-phys
```

This command displays the physical NICs on the system and their corresponding datalink names. Unless you create customized names for your datalinks, the datalink has the same name as the network interface device name. For example, the device `e1000g0` uses the data link name `e1000g0` until you replace the link name with another name. For more information about customized datalink names, see "Network Devices and Datalink Names" on page 26.

3 **(Optional) To view information about the system's datalinks, type the following command:**

```
# dladm show-link
```

This command lists the datalinks and their current status. Make sure that a datalink's `STATE` field indicates that the datalink is `up`. You can configure VNICs only over datalinks whose status is `up`.

4 **(Optional) To view IP address information on configured interfaces, type the following command:**

```
# ipadm show-addr
```

This command lists configured interfaces on your system including their corresponding IP addresses.

5 **Create a VNIC over a datalink.**

```
# dladm create-vnic -l link vnic
```

- *link* is the name of the datalink over which the VNIC is configured.
- *vnic* is the VNIC which you can label with a customized name as well.

6 **Create a VNIC IP interface over the link.**

```
# ipadm create-ip vnic
```

7 **Configure the VNIC with a valid IP address.**

If you are assigning a static IP address, use the following syntax:

```
# ipadm create-addr -T static -a address addrobj
```

where *addrobj* uses the naming format *interface*/*user-defined-string*, such as `e1000g0/v4globalz`. For other options when using this command, refer to the `ipadm(1M)` man page.

8 If you are using static IP addresses, add the address information in the `/etc/hosts` file.

9 (Optional) To display the VNIC's address configuration, type the following:

```
# ipadm show-addr
```

10 (Optional) To display VNIC information, type the following:

```
# dladm show-vnic
```

Example 19–1 Creating Virtual Network Interfaces

This example contains the commands to create VNICs. You must log in to the system as superuser or the equivalent role to run the commands.

```
# dladm show-phys
LINK      MEDIA               STATE      SPEED DUPLEX   DEVICE
net0      Ethernet            up         1000 full      e1000g0
net1      Ethernet            unknown    0    half      e1000g1

# dladm show-link
LINK     CLASS    MTU    STATE    BRIDGE     OVER
net0     phys     1500   up       --         --
net1     phys     1500   unknown  --         --

# ipadm show-if
IFNAME   CLASS        STATE     ACTIVE     OVER
lo0      loopback     ok        yes        --
net0     ip           ok        yes        --

# ipadm show-addr
ADDROBJ        TYPE       STATE     ADDR
lo0/?          static     ok        127.0.0.1/8
net0/v4addr    static     ok        192.168.3.70/24

# dladm create-vnic -l net0 vnic0
# dladm create-vnic -l net0 vnic1

# dladm show-vnic
LINK        OVER          SPEED  MACADDRESS         MACADDRTYPE
vnic0       net0      1000 Mbps  2:8:20:c2:39:38    random
vnic1       net0      1000 Mbps  2:8:20:5f:84:ff    random
#
# ipadm create-ip vnic0
# ipadm create-ip vnic1

# ipadm create-addr -T static -a 192.168.3.80/24 vnic0/v4address
# ipadm create-addr -T static -a 192.168.3.85/24 vnic1/v4address
# ipadm show-addr
ADDROBJ            TYPE       STATE     ADDR
lo0/?              static     ok        127.0.0.1/8
```

```
net0/v4addr        static     ok        192.168.3.70/24
vnic0/v4address    static     ok        192.168.3.80/24
vnic1/v4address    static     ok        192.168.3.85/24
```

The system's `/etc/hosts` file would contain information similar to the following:

```
# cat /etc/hosts
#
::1             localhost
127.0.0.1       localhost
192.168.3.70    loghost   #For e1000g0
192.168.3.80    vnic1
192.168.3.85    vnic2
```

How to Create Etherstubs

You use etherstubs to isolate the virtual network from the rest of the virtual networks in the system as well as the external network to which the system is connected. You cannot use an etherstub just by itself. Instead, you use VNICs with an etherstub to create the private or isolated virtual networks. You can create as many etherstubs as you require. You can also create as many VNICs over each etherstub as required.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Create an etherstub**

   ```
   # dladm create-etherstub etherstub
   ```

3. **Create a VNIC over the etherstub.**

   ```
   # dladm create-vnic -l etherstub vnic
   ```

4. **Configure the VNIC with a private address.**

   Note – To isolate the network for which you are configuring the VNIC over an etherstub, make sure to use a private IP address that cannot be forwarded by the default router of the external network. For example, suppose the physical interface has an address `192.168.3.0/24` that indicates that the system is on a `192.168.3.x` network. You therefore assign another address that is not known to the default router, for example, `192.168.0.x`.

5. **(Optional) To display information about VNICs, type the following command.**

   ```
   # dladm show-vnic
   ```

   This command lists all the VNICs in the system and the datalinks or etherstubs over which the VNICs are created.

6 **(Optional) To display information about all the physical and virtual links on the system, type the following command.**

```
# dladm show-link
```

Example 19–2 Creating an Etherstub

The following example shows how to create an etherstub and then configure a VNIC over the etherstub. This example develops the previous example by adding a third VNIC that is configured over the etherstub.

You must log in to the system as superuser or equivalent role to run the next commands.

```
# dladm create-etherstub stub0
#
dladm show-vnic
LINK        OVER          SPEED  MACADDRESS         MACADDRTYPE
vnic1       net9      1000 Mbps  2:8:20:c2:39:38    random
vnic2       net0      1000 Mbps  2:8:20:5f:84:ff    random
#
# dladm create-vnic -l stub0 vnic3
# ipadm create-vnic vnic3
# ipadm create-addr -T static -a 192.168.0.10/24 vnic3/privaddr
#
# dladm show-vnic
LINK        OVER          SPEED  MACADDRESS         MACADDRTYPE
vnic1       net0      1000 Mbps  2:8:20:c2:39:38    random
vnic2       net0      1000 Mbps  2:8:20:5f:84:ff    random
vnic3       stub0        1000 Mbps  2:8:20:54:f4:74    random
#
# ipadm show-addr
ADDROBJ            TYPE       STATE     ADDR
lo0/?              static     ok        127.0.0.1/8
net0/v4addr        static     ok        192.168.3.70/24
vnic1/v4address    static     ok        192.168.3.80/24
vnic2/v4address    static     ok        192.168.3.85/24
vnic3/privaddr     static     ok        192.168.0.10/24
```

The system's `/etc/hosts` file would contain information similar to the following:

```
# cat /etc/hosts
#
::1             localhost
127.0.0.1       localhost
192.168.3.70    loghost   #For e1000g0
192.168.3.80    vnic1
192.168.3.85    vnic2
192.168.0.10    vnic3
```

Working With VNICs and Zones

This section shows you how you deploy the network virtualization components by configuring these components to be used by zones. This section provides two approaches when working with zones to use VNICs:

- Creating entirely new zones and configuring VNICs over these zones
- Modifying existing zone configurations to use VNICs.

When you first log in to a system, you are automatically in its *global zone*. You create VNICs on the global zone. Then you further configure these VNICs depending on whether they are to be used by the global zone or non-global exclusive type zones. For an introduction to zones, refer to "Zones Overview" in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*.

Creating New Zones for Use With VNICs

Use this approach if no configured zones exist in the system, or if you want to create new zones to use VNICs.

To use VNICs, a zone must be configured as an exclusive IP zone. The steps that follow configure `zone1` with `vnic1`. You must perform the same steps to configure `zone2`. For clarity, the prompts indicate in which zone a specific command is issued. However, the actual path that the prompts display might vary depending on the prompt settings of your specific system.

▼ How to Create and Configure the Exclusive IP Zone

When creating zones, you can set several parameters. The zone procedures throughout this chapter focus only on those parameters that are relevant to make the zone operate with VNICs. For more detailed information about zone configuration, refer to Part II, "Oracle Solaris Zones," in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*.

Before You Begin Make sure you have accomplished the following:

- Created the VNICs for the zones, as explained in "How to Create a Virtual Network Interface" on page 343.
- Defined the zone names.
- Determined zone home directories.
- Determined the specific VNIC to be associated with a specific zone.
- Determined the IP addresses for the VNICs.
- Obtained other network information such as router address to supply to the zone.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **For every zone that you create, perform the following steps.**

   a. **Start the zone configuration utility and create the zone.**

   ```
   global# zonecfg -z zone
   zonecfg:zone> create
   ```

   b. **Set the home directory by defining the parameter `zonepath`.**

   ```
   zonecfg:zone> set zonepath=/home/export/zone
   ```

   c. **Enable automatic booting.**

   ```
   zonecfg:zone> set autoboot=true
   ```

   d. **Configure the zone to be an exclusive IP zone.**

   ```
   zonecfg:zone> set ip-type=exclusive
   ```

   e. **Set the zone's interface to be a designated VNIC.**

   ```
   zonecfg:zone> add net
   zonecfg:zone:net> set physical=vnic
   zonecfg:zone:net> end
   zonecfg:zone>
   ```

   f. **Verify and commit the settings, then exit the zone configuration utility.**

   ```
   zonecfg:zone>verify
   zonecfg:zone> commit
   zonecfg:zone> exit
   global#
   ```

   g. **(Optional) To verify that the information for the zone is correct, type the following:**

   ```
   global# zonecfg -z zone info
   ```

   Note – You can display the same information while running the zone configuration utility by typing the following:

   ```
   zonecfg:zone> info
   ```

3. **Install the zone.**

   ```
   global# zoneadm -z zone install
   ```

   Note – The installation process can take a while.

4 **(Optional) After the zone is completely installed, check the status of the zone.**

```
zoneadm list -iv
```

Note – The `-iv` option lists all configured zones regardless of whether they are running or not. At this stage, the status of the zone you just created will be "`installed`" rather than "`running`." If you use the `-v` option, only zones that are running are listed, and the zone you just created will be excluded.

5 **Start the zone.**

```
global# zoneadm -z zone boot
```

6 **(Optional) Verify that the zone is now running.**

```
global# zoneadm list -v
```

7 **After the zone completely boots up, connect to the zone's console.**

```
# zlogin -C zone
```

8 **Supply the information as you are prompted.**

Some of the information are terminal type, region, language, and so on. Most of the information is supplied by selecting from a list of choices. Typically, the default options suffice unless your system configuration requires otherwise.

The following information are relevant to the current procedure which you need to supply or verify:

- Host name of the zone, for example `zone1`.
- IP address of the zone which is based on the IP address of the zone's VNIC.
- Whether IPv6 should be enabled.
- Whether the system with the virtual network is part of a subnet.
- Netmask of the IP address.
- Default route, which can be the IP address of the physical interface on which the virtual network is built.

After you have supplied the required information for the zone, the zone is restarted.

Example 19–3 Configuring a Basic Virtual Network by Creating Zones and VNICs

This example consolidates all the steps that were previously provided to creating zones and VNICs to configure the virtual network. The example uses `zone1` as the sample zone

The example is based on the following assumptions:

- VNICs: `vnic1`

- Zone names: zone1
- Zone home directories: /home/export/*zone-name*.
- VNIC zone assignments: vnic1 for zone1
- IP addresses: vnic1 uses 192.168.3.80
- Physical interface IP address: 192.168.3.70
- Router address: 192.168.3.25

```
global# dladm show-phys
LINK   MEDIA      STATE      SPEED  DUPLEX    DEVICE
net0   Ethernet   up         1000   full      e1000g0
net1   Ethernet   unknown    1000   full      bge0

global# dladm show-lnk
LINK      CLASS     MTU    STATE     BRIDGE   OVER
net0   phys      1500   up        --       --
net1      phys      1500   unknown   --       --

global# ipadm show-if
IFNAME   CLASS        STATE     ACTIVE     OVER
lo0      loopback     ok        yes        --
net0     ip           ok        yes        --

global # ipadm show-addr
ADDROBJ        TYPE       STATE     ADDR
lo0/?          static     ok        127.0.0.1/8
net0/v4addr    static     ok        192.168.3.70/24

global # dladm create-vnic -l net0 vnic1

global # dladm show-vnic
LINK      OVER         SPEED        MACADDRESS      MACADDRTYPE
vnic1     net0     1000 Mbps    2:8:20:5f:84:ff     random

global # ipadm create-ip vnic1
global # ipadm create-addr -T static -a 192.168.3.80/24 vnic1/v4address
global # ipadm show-addr
ADDROBJ           TYPE       STATE     ADDR
lo0/?             static     ok        127.0.0.1/8
net0/v4addr       static     ok        192.168.3.70/24
vnic1/v4address   static     ok        192.168.3.80/24

global # cat /etc/hosts
::1             localhost
127.0.0.1       localhost
192.168.3.70    loghost   #For net0
192.168.3.80    zone1  #using vnic1

global # zonecfg -z zone1
zonecfg:zone1> create
zonecfg:zone1> set zonepath=/export/home/zone1
zonecfg:zone1> seet autoboot=true
zonecfg:zone1> set ip-type=exclusive
zonecfg:zone1> add net
zonecfg:zone1:net> set physical=vnic1
zonecfg:zone1:net> end
zonecfg:zone1> verify
```

```
zonecfg:zone1> info
zonename: zone1
zonepath: /export/home/zone1
brand:    native
autoboot: true
net:
          address not specified
          physical: vnic1

zonecfg:zone1> commit
zonecfg:zone1> exit
global#
global# zoneadm -z zone1 verify
WARNING: /export/home/zone1 does not exist, so it could not be verified.
When 'zoneadm install' is run, 'install' will try to create
/export/home/zone1, and 'verify' will be tried again,
but the 'verify' may fail if:
the parent directory of /export/home/zone1 is group- or other-writable
or
/export/home/zone1 overlaps with any other installed zones.

global# zoneadm -z zone1 install
Preparing to install zone <zone1>
Creating list of files to copy from the global zone.
.
.
Zone <zone1> is initialized.

global# zoneadm list -iv
ID NAME     STATUS      PATH                 BRAND     IP
0  global   running     /                    native   shared
-  zone1    installed   /export/home/zone1   native   excl

global# zoneadm -z zone1 boot
global# zoneadm list -v
ID NAME     STATUS    PATH                 BRAND     IP
0  global   running   /                    native   shared
1  zone1    running   /export/home/zone1   native   excl

zlogin -C zone1
What type of terminal are you using?
.
.
.
8) Sun Workstation
9) Televideo 910
10) Televideo 925
11) Wyse Model 50
12) X Terminal Emulator (xterms)
13) CDE Terminal Emulator (dtterm)
14) Other
Type the number of your choice and press Return: 13
.
(More prompts)
..
```

Provide the information as prompted. For network information, supply the following:

```
Hostname: zone1
IP address: 192.168.3.80
System part of a subnet: Yes
Netmask: 255.255.255.0
Enable IPv6: No
Default route: 192.168.3.70
Router IP address: 192.168.3.25
```

Next Steps You can use various tools to observe network traffic and take statistics on zone usage.

- To verify that your network is properly configured, refer to Chapter 5, "Administering a TCP/IP Network," in *Oracle Solaris Administration: IP Services*.
- To observe traffic over the network, refer to "Monitoring Packet Transfers With the snoop Command" in *Oracle Solaris Administration: IP Services*.
- To manage how the network uses system resources, refer to Chapter 21, "Managing Network Resources."
- To obtain statistics for accounting purposes, refer to Chapter 22, "Monitoring Network Traffic and Resource Usage."

If you need to disassemble the virtual network, refer to "How to Remove the Virtual Network Without Removing the Zones" on page 358.

Modifying the Configuration of Existing Zones to Use VNICs

Use this approach if you want existing zones to use VNICs. In this case, the zones already have zone names and their home directories or `zonepaths` are already defined.

▼ How to Reconfigure a Zone to Use a VNIC

Before You Begin Make sure you have accomplished the following:

- Created the VNICs for the zones, as explained in "How to Create a Virtual Network Interface" on page 343.
- Determined the specific VNIC to be associated with a specific zone.
- Determined the IP addresses for the VNICs.
- Obtained other network information such as router address to supply to the zone.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Verify that zones are properly configured and running on the system.

```
global# zoneadm list -v
```

Note – The `-v` option lists only zones that are running. To list all configured zones including those that have not been started, use the `-iv` option.

3 For every zone that you want to configure with VNICs, perform the following steps:

a. Verify the information about the zone.

```
global# zonecfg -z zone info
```

Check the information about IP type and network interface. The network interface is designated by the parameter *physical*. For a zone to be configured with a VNIC, the zone must be an exclusive IP zone and the network interface must specify the VNIC.

b. If necessary, change the shared zone to an exclusive IP zone.

```
global# zonecfg -z zone
zonecfg:zone1> set ip-type=exclusive
zonecfg:zone1>
```

c. Change the zone's interface to use a VNIC.

```
zonecfg:zone1> remove net physical=non-vnic-interface
zonecfg:zone1> add net
zonecfg:zone1:net> set physical=vnic
zonecfg:zone1:net> end
zonecfg:zone1>
```

d. Change other parameter values as appropriate.

e. Verify and commit the changes you have implemented and then exit the zone.

```
zonecfg:zone1 verify
zonecfg:zone1> commit
zonecfg:zone1> exit
global#
```

f. Reboot the zone.

```
global# zoneadm -z zone reboot
```

g. After the zone reboots, verify that the zone information about `ip-type` and `physical` are correct.

```
global# zonecfg -z zone info ip-type
global# zonecfg -z zone info net
```

The information must show that the zone's IP type is exclusive and that it uses the designated VNIC.

4 Log in to the zone.

```
global# zlogin zone
```

5 Configure the VNIC with a valid IP address.

If you are assigning a static address to the VNIC, you would type the following:

zone# **ipadm create-addr -T static -a** *address addrobj*

where *address* can use CIDR notation while *addrobj* follows the naming convention *interface*/*user-defined-string*.

6 (Optional) Verify the interface configuration within the zone.

zone# **ipadm show-if**

or

zone# **ipadm show-addr**

Example 19–4 Configuring a Basic Virtual Network by Modifying Zone Configuration to Use VNICs

This example uses the same system and operates on the same assumptions as the previous example. Suppose that in this system, zone2 already exists as a shared zone. You want to modify zone2 to use vnic2.

```
global# dladm show-link
LINK   CLASS     MTU    STATE     BRIDGE   OVER
net0   phys      1500   up        --       --
net1   phys      1500   unknown   --       --
vnic1  vnic      1500   up        --       e1000g0

global# ipadm show-if
IFNAME   CLASS        STATE     ACTIVE     OVER
lo0      loopback     ok        yes        --
net0     ip           ok        yes        --
vnic1    ip           ok        yes        --

global # ipadm show-addr
ADDROBJ           TYPE       STATE     ADDR
lo0/?             static     ok        127.0.0.1/8
net0/v4addr       static     ok        192.168.3.70/24
vnic1/v4address   static     ok        192.168.3.80/24

global # dladm create-vnic -l net0 vnic2
global # dladm show-vnic
LINK      OVER        SPEED        MACADDRESS          MACADDRTYPE
vnic1     net0     1000 Mbps    2:8:20:5f:84:ff     random
vnic2     net0     1000 Mbps    2:8:20:54:f4:74     random

global# zoneadm list -v
ID NAME     STATUS    PATH                BRAND     IP
0  global   running   /                   native   shared
1  zone1    running   /export/home/zone1  native   excl
2  zone2    running   /export/home/zone2  native   shared

global# zonecfg -z zone2 info
zonename: zone2
zonepath: /export/home/zone2
```

```
brand: native
autoboot: true
bootargs:
pool: z2-pool
limitpriv:
scheduling-class:
ip-type: shared
hostid:
inherit-pkg-dir:
        dir: /lib
inherit-pkg-dir:
        dir: /platform
inherit-pkg-dir:
        dir: /sbin
inherit-pkg-dir:
        dir: /usr
inherit-pkg-dir:
        dir: /etc/crypto
net:
        address not specified
        physical: e1000g0
        defrouter not specified
global#


global# zonecfg -z zone2
zonecfg:zone1> set ip-type=exclusive
zonecfg:zone1> remove net physical=net0
zonecfg:zone1> add net
zonecfg:zone1:net> set physical=vnic2
zonecfg:zone1:net> end
zonecfg:zone1> verify
zonecfg:zone1> commit
zonecfg:zone1> exit
global#

global# zonecfg -z zone2 info ip-type
ip-type: exclusive
global#

global# zonecfg -z zone2 info net
net:
        address ot specified
        physical: vnic2
        defrouter not specified
global#

global# zlogin zone2
zone2# ipadm create-ip vnic2
zone2# ipadm create-addr -T static -a 192.168.3.85/24 vnic2/v4address

zone2# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
vnic2/v4address   static   ok           192.168.3.85/24

zone1# exit
global#
```

```
global# vi /etc/hosts
#
::1             localhost
127.0.0.1       localhost
192.168.3.70    loghost   #For e1000g0
192.168.3.80    zone1   #using vnic1
192.168.3.85    zone2   #using vnic2
```

Next Steps You can either configure the network setup further to customize use of system resources, or use various tools to observe network traffic and take statistics on resource usage.

- To verify that your network is properly configured, refer to
- To observe traffic over the network, refer to
- To manage how the network uses system resources, refer to
- To obtain statistics for accounting purposes, refer to

If you need to disassemble the virtual network, refer to "How to Remove the Virtual Network Without Removing the Zones" on page 358

Creating a Private Virtual Network

The example in this section shows how to configure a *private virtual network* on a single system. Private virtual networks are different from virtual private networks (VPNs). VPN software creates a secure point-to-point link between two endpoint systems. The private network configured by the tasks in this section is a virtual network on a box that cannot be accessed by external systems.

To allow the zones of the private network to send packets beyond the host, configure a network address translation (NAT) device. NAT translates the VNIC's private IP addresses to routeable IP addresses of the physical network interface, but without exposing the private IP addresses to the external network. Routing configuration is also included in the following example.

EXAMPLE 19–5 Creating a Private Virtual Network Configuration

The following example uses the same system and proceeds on the same assumptions as the previous examples. Specifically, `zone1` and `zone2` are now configured as virtual networks. Suppose that `zone3` already exists in the system. You will modify `zone3` to become a private network isolated from the rest of the network. Then you will configure NAT and IP forwarding to allow the virtual private network to send packets outside the host but still concealing its private address from the external network.

```
global# dladm create-etherstub stub0

global# dladm create-vnic -l etherstub0 vnic3
global# dladm show-vnic
LINK      OVER        SPEED        MACADDRESS          MACADDRTYPE
vnic1     net0     1000 Mbps    2:8:20:5f:84:ff     random
vnic2     net0     1000 Mbps    2:8:20:54:f4:74     random
```

EXAMPLE 19–5 Creating a Private Virtual Network Configuration *(Continued)*

```
vnic3     stub0          0 Mbps    2:8:20:6b:8:ab      random

global# vi /etc/hosts
#
::1             localhost
127.0.0.1       localhost
192.168.3.70    loghost   #For e1000g0
192.168.3.80    zone1   #using vnic1
192.168.3.85    zone2   #using vnic2
```

At this stage, you modify `zone3` to become an exclusive IP zone over `vnic3`.

```
global# zonecfg -z zone3
zonecfg:zone3> set ip-type=exclusive
zonecfg:zone3> remove net physical=e1000g0
zonecfg:zone3> add net
zonecfg:zone3:net> set physical=vnic3
zonecfg:zone3:net> end
zonecfg:zone3> verify
zonecfg:zone3> commit
zonecfg:zone3> exit
global#

global# zonecfg -z zone3 info ip-type
ip-type: exclusive
global#

global# zonecfg -z zone3 info net
net:
        address ot specified
        physical: vnic3
        defrouter not specified
global#

global# zlogin zone3
zone3# ipadm create-ip vnic3
zone3# ipadm create-addr -T static -a 192.168.0.10/24 vnic3/privaddr

zone3# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
vnic3/privaddr    static   ok           192.168.0.10/24
zone3# exit

global# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
net0/v4addr       static   ok           192.168.3.70/24
vnic1/v4address   static   ok           192.168.3.80/24
vnic2/v4address   static   ok           192.168.3.85/24
vnic3/privaddr    static   ok           192.168.0.10/24

global# vi /etc/hosts
::1             localhost
127.0.0.1       localhost
```

EXAMPLE 19–5 Creating a Private Virtual Network Configuration *(Continued)*

```
192.168.3.70    loghost   #For e1000g0
192.168.3.80    zone1   #using vnic1
192.168.3.85    zone2   #using vnic2
192.168.0.10    zone3   #using vnic3

global# routeadm
              Configuration   Current             Current
                     Option   Configuration       System State
---------------------------------------------------------------
                IPv4 routing   enabled             enabled
                IPv6 routing   disabled            disabled
              IPv4 forwarding  disabled            disabled
              IPv6 forwarding  disabled            disabled

            Routing services   "route:default ripng:default"

global# ipadm set-ifprop -p forwarding=on -m ipv4 e1000g0

global# vi /etc/ipf/ipnat.conf
map e1000g0 192.168.0.0/24 -> 0/32  portmap tcp/udp auto
map e1000g0 192.168.0.0/24 -> 0/32

global# svcadm enable network/ipfilter

global# zoneadm -z zone1 boot
global# zoneadm -z zone2 boot
global# zoneadm -z zone3 boot
```

▼ How to Remove the Virtual Network Without Removing the Zones

The following procedure shows how to disable a zone's virtual network but maintain the zone intact.

Use this procedure if you must do any of the following:

- Use the existing zones in a different configuration. For example, you might need to configure the zones as part of a private network that would require the zone to be created by using an etherstub.
- Migrate the zones to another network.
- Move the zones to a different zone path.
- Clone the zones, as explained in "Cloning a Non-Global Zone on the Same System" in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*.

Before You Begin This task assumes that you have a running virtual network that consists of exclusive IP zones.

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Verify the state of the currently configured zones.**

   ```
   # zoneadm list -v
   ```

   Information similar to the following is displayed:

   ```
   ID  NAME     STATUS       PATH                          BRAND           IP
    0  global   running      /                             native          shared
    1  zone1    running      /export/home/zone1            native          excl
    2  zone2    running      /export/home/zone2            native          excl
    3  zone3    running      /export/home/zone3            native          excl
   ```

3. **Halt the exclusive IP zones of the virtual network.**

   Issue the following command separately for each zone to be halted.

   ```
   global# zoneadm -z zone-name halt
   ```

   When you halt the zone, you remove the zone's application environment and terminate a number of system activities, as explained in "Halting a Zone" in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*.

4. **Verify that the zones have been halted.**

   ```
   # zoneadm list -iv
   ID NAME             STATUS     PATH                           BRAND    IP
      0 global           running    /                              native   shared
      - zone1            installed  /export/home/zone1             native   excl
      - zone2            installed  /export/home/zone2             native   excl
      - zone3            installed  /export/home/zone3             native   excl
   ```

   Note that the zones are no longer running, although they remain installed. To reboot a halted zone, refer to "How to Boot a Zone" in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*.

5. **List the VNICs that were configured for the halted zones.**

   ```
   # dladm show-vnic
   LINK        OVER              SPEED  MACADDRESS         MACADDRTYPE
   vnic1       net0      1000 Mbps  2:8:20:5f:84:ff    random
   vnic2       net1     1000 Mbps  2:8:20:54:f4:74    random
   vnic3       stub0    1000 MBps  2:8:20:c2:39:38    random
   ```

   The resulting output shows that the VNICs are still configured as datalinks in the global zone. However, their corresponding IP interfaces were created and enabled on the zones with which these VNICs are associated, and not on the global zone. These non–global zones are now halted.

6. **Delete the VNICs.**

   ```
   # dladm delete-vnic vnic
   ```

For example, you would type the following to delete the VNICs in the zones in Figure 18–1.

```
# dladm delete-vnic vnic1
# dladm delete-vnic vnic2
```

◆ ◆ ◆ CHAPTER 20

Using Link Protection in Virtualized Environments

This chapter describes link protection and how to configure it on Oracle Solaris systems. The chapter covers the following topics:

Overview of Link Protection

With the increasing adoption of virtualization in system configurations, guest virtual machines (VMs) can be given exclusive access to a physical or virtual link by the host administrator. This configuration improves network performance by allowing the virtual environment's network traffic to be isolated from the wider traffic that is received or sent by the host system. At the same time, this configuration can expose the system and the entire network to the risk of harmful packets that a guest environment might generate.

Link protection aims to prevent the damage that can be caused by potentially malicious guest VMs to the network. The feature offers protection from the following basic threats:

- IP and MAC spoofing
- L2 frame spoofing such as Bridge Protocol Data Unit (BPDU) attacks

---

Note – Link protection should not replace the deployment of a firewall, particularly for configurations with more complex filtering requirements.

Link Protection Types

The link protection mechanism is disabled by default. To enable link protection, specify one or more of the following protection types as values of the `protection` link property:

`mac-nospoof` Enables protection against MAC spoofing. An outbound packet's source MAC address must match the datalink's configured MAC address. Otherwise, the packet is dropped. If the link belongs to a zone, enabling `mac-nospoof` prevents the zone's owner from modifying that link's MAC address.

`ip-nospoof` Enables protection against IP spoofing. Any outgoing IP, ARP, or NDP packet must have an address field that matches either a DHCP-configured IP address or one of the addresses listed in the `allowed-ips` link property. Otherwise, the packet is dropped.

The `allowed-ips` link property works with the `ip-nospoof` protection type. By default, the list specified by this property is empty. If the property is empty or unconfigured, the following IP addresses are implicitly included in the property. These IP addresses are matched with the IP address of the outgoing packets to determine if the packets are allowed to pass or are dropped.

- DHCP-configured IPv4 or IPv6 addresses that are dynamically learned
- Link local IPv6 addresses that conform to RFC 2464 and which are derived from the link's MAC address

The following list indicates a protocol and the corresponding outbound packet's associated address field that must match an address in the `allowed-ips` property. If this property is empty, then the packet's address must match a DHCP-configured IP address.

- IP (IPv4 or IPv6) – The packet's source address
- ARP – The packet's sender protocol address.

`restricted` Restricts outgoing packets to only those packets of the IPv4, IPv6, and ARP protocol types. Other packets that are not of the listed types are dropped. Using this protection type prevents the link from generating potentially harmful L2 control frames.

Note – Packets that are dropped because of link protection are tracked by the following kernel statistics: `mac_spoofed`, `ip_spoofed`, and `restricted`. These statistics correspond to the three protection types. Use the `kstat` command to retrieve these per-link statistics. For more details about retrieving these statistics, see the `kstat(1M)` man page.

Configuring Link Protection (Task Map)

To use link protection, you use one of the options of the `dladm` command to set the link properties. If the type of protection works with other configuration files, for example, `ip-nospoof` with `allowed-ips`, then you perform two general actions. First, you enable link protection. Then, you customize the configuration file to determine how the link protection operates.

Note – You must configure link protection in the global zone.

The following points to the tasks that you can use to configure link protection on a Oracle Solaris server.

Task	Description	For Instructions
Enable link protection mechanism.	Use the `dladm set-linkprop` command to enable link protection types for a link.	"How to Enable the Link Protection Mechanism" on page 363
Disable link protection mechanism.	Use the `dladm reset-linkprop` command to disable link protection.	"How to Disable Link Protection" on page 364
Customize the IP link protection type.	Use the `dladm set-linkprop` command to configure or modify the values in the `allowed-ips` property.	"How to Specify IP Addresses for Protection Against IP Spoofing" on page 364
View the link protection configuration.	Use the `dladm show-linkprop` command to view the link protection configuration by specifying the `protection` and `allowed-ips` property names.	"How to View the Link Protection Configuration" on page 365

▼ How to Enable the Link Protection Mechanism

This procedure enables one or more of the following link protection types: `mac-nospoof`, `ip-nospoof`, and `restricted`.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Enable link protection by specifying one or more protection types.**

```
# dladm set-linkprop -p protection=value[,value,...] link
```

In the following example, all three link protection types on the `vnic0` link are enabled:

```
# dladm set-linkprop -p protection=mac-nospoof,ip-nospoof,restricted vnic0
```

How to Disable Link Protection

This procedure resets link protection to the default values, which disables link protection.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Disable link protection by resetting the `protection` property to its default value.**

```
# dladm reset-linkprop -p protection link
```

How to Specify IP Addresses for Protection Against IP Spoofing

Note that the `allowed-ips` property is used only if the `protection` property enables the `ip-nospoof` protection type.

1 **Become an administrator.**

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 **Make sure that you have enabled protection from IP spoofing.**

If you have not yet enabled this type of link protection, then issue the following command:

```
# dladm set-linkprop -p protection=ip-nospoof
```

3 **Specify a list of IP addresses as values for the `allowed-ips` link property.**

```
# dladm set-linkprop -p allowed-ips=IP-addr[,IP-addr,...] link
```

The following example shows how to specify the `10.0.0.1` and `10.0.0.2` IP addresses as values for the `allowed-ips` property for the `vnic0` link:

```
# dladm set-linkprop -p allowed-ips=10.0.0.1,10.0.0.2 vnic0
```

How to View the Link Protection Configuration

The values of the `protection` and `allowed-ips` properties indicate how link protection is configured. Note that the `allowed-ips` property is used only if the `protection` property specifies the `ip-nospoof` protection type.

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 View the link protection property values.

```
# dladm show-linkprop -p protection,allowed-ips link
```

The following example shows the values for the `protection` and `allowed-ips` properties for the `vnic0` link:

```
# dladm show-linkprop -p protection,allowed-ips vnic0
LINK      PROPERTY        PERM     VALUE           DEFAULT     POSSIBLE
vnic0     protection      rw       ip-nospoof      --          --
                                   mac-nospoof
                                   restricted
vnic0     allowed-ips     rw       10.0.0.1,       --          --
                                   10.0.0.2
```

◆ ◆ ◆ CHAPTER 21

Managing Network Resources

This chapter explains how to manage resources on datalinks, including virtual links such as VNICs. Network resource management implements quality of service to enhance performance especially in the virtual network.

The chapter covers the following topics:

Overview of Network Resource Management

This section explains network resource management by introducing network lanes. It also describes how you implement network resource management by setting datalink properties. Flows are also defined as another way of further setting resource controls to process network traffic.

Datalink Properties for Resource Control

In previous Oracle Solaris releases, implementing quality of service is a complicated process. The process consists of defining queuing disciplines, classes, and filter rules and indicating the relationships among all of these components. For more information, see Part V, "IP Quality of Service (IPQoS)," in *Oracle Solaris Administration: IP Services*.

In this release, quality of service is obtained more easily and dynamically by managing network resources. Network resource management consists of setting datalink properties that pertain to network resources. By setting these properties, you determine how much of a given resource can be used for networking processes. For example, a link can be associated with a specific number of CPUs that are reserved exclusively for networking processes. Or, a link can be

allotted a given bandwidth to process a specific type of network traffic. After a resource property is defined, the new setting takes effect immediately. This method makes managing resources flexible. You can set resource properties when you create the link. Alternatively, you can set these properties later, for example, after studying resource usage over time and determining how to better allocate the resource. The procedures for allocating resources apply to both the virtual network environment as well as the traditional physical network.

Network resource management is comparable to creating dedicated lanes for traffic. When you combine different resources to cater to specific types of network packets, those resources form a *network lane* for those packets. Resources can be assigned differently for each network lane. For example, you can allocate more resources to a lane where network traffic is heaviest. By configuring network lanes where resources are distributed according to actual need, you increase the system's efficiency to process packets. For more information about network lanes, see "Overview of Network Traffic Flow" on page 395.

Network resource management is helpful for the following tasks:

- Network provisioning.
- Establishing service level agreements.
- Billing clients.
- Diagnosing security problems.

You can isolate, prioritize, track, and control data traffic on an individual system without the complex QoS rule definitions in previous releases.

Network Resource Management by Using Flows

A *flow* is a customized way of categorizing packets to further control how resources are used to process these packets. Network packets can be categorized according to an *attribute*. Packets that share an attribute constitute a flow and are labeled with a specific flow name. The flow can then be assigned specific resources.

The attributes that serve as the basis for creating flows are derived from the information in a packet's header. You can organize packet traffic into flows according to one of the following attributes:

- IP address
- Transport protocol name (UDP, TCP, or SCTP)
- Application port number, for example, port 21 for FTP
- DS field attribute, which is used for quality of service in IPv6 packets only. For more information about the DS field, refer to "DS Codepoint" in *Oracle Solaris Administration: IP Services*.

A flow can be based on only one of the attributes in the list. For example, you can create a flow according to the port that is being used, such as port 21 for FTP, or according to IP addresses, such as packets from a specific source IP address. However, you cannot create a flow for packets

from a specified IP address that are received on port number 21 (FTP). Likewise, you cannot create a flow for all traffic from IP address `192.168.1.10`, and then create a flow for transport layer traffic on `192.168.1.10`. Thus, you can configure multiple flows on a system, with each flow based on a different attribute.

Commands for Network Resource Management

The command for allocating network resources depends on whether you are directly working on datalinks or on flows.

- For datalinks, you use the appropriate `dladm` subcommand depending on whether you are setting the property while creating the link or setting the property of an existing link. To simultaneously create a link and allocate resources to it, use the following syntax:

  ```
  # dladm create-vnic -l link -p property=value[,property=value] vnic
  ```

  where *link* can be either a physical link or a virtual link.

  To set the property of an existing link, use the following syntax:

  ```
  # dladm set-linkprop -p property=value[,property=value] link
  ```

  For more details about the `dladm` command and the properties that this command manages, refer to the `dladm(1M)` man page.

  The following are link properties that you can set for resource allocation:

  - Bandwidth – You can limit a hardware's bandwidth for a certain link's use.
  - NIC rings – If a NIC supports ring allocation, its transmit and receive rings can be assigned for dedicated use by datalinks. NIC rings are discussed in "Transmit and Receive Rings" on page 370
  - CPU pools – Pools of CPUs are generally created and associated with specific zones. These pools can be assigned to datalinks to reserve the sets of CPUs to manage the network processes of their associated zones. CPUs and pools are discussed in "Pools and CPUs" on page 384.
  - CPUs – In a system with multiple CPUs, you can dedicate a given number of CPUs for specific network processing.
- For flows, you use `flowadm` subcommands. First you create the flow by using the `flowadm add-flow` subcommand. Then you assign resources to the flow by using the `flowadm set-flowprop` subcommand. The set of defined attributes that characterizes the flows together constitutes the system's *flow control policy*.

Note – The properties for resource allocation that can be assigned to a flow are the same as the properties that are assigned directly to a link. Currently however, only the bandwidth properties can be associated with flows. Although the commands to set properties are different for datalinks and for flows, the syntax is similar. To configure the bandwidth properties, see the examples in "How to Configure a Flow" on page 389

For more information, refer to the `flowadm(1M)` man page.

Network Resource Management (Task Map)

The following table lists different methods of establishing resource controls and determining how these resources are allocated for network processing.

Task	Description	For Instructions
Allocate rings to MAC clients.	Configure MAC clients on a datalink to use rings.	"Properties for Ring Allocation" on page 372
Assign a pool of CPUs to a datalink.	Use the `pool` property to allocate a set of CPUs to manage a zone's network processes.	"How to Configure a CPU Pool for a Datalink " on page 386
Assign a set of CPUs to a defined datalink	On a system that has multiple CPUs, reserve a set of CPUs for networking purposes.	"How to Allocate CPUs to Links" on page 388
Implement network resource management by using flows on a physical network.	Isolate network traffic into individual flows. Then assign the flows a set amount of interface bandwidth among other flows.	"How to Configure a Flow" on page 389

Managing Resources on Datalinks

This section describes selected link properties that you can set to improve network performance for either a physical network or a virtual network.

Transmit and Receive Rings

On NICs, receive (Rx) rings and transmit (Tx) rings are hardware resources through which the system receives and sends network packets, respectively. The following sections provide an

overview of rings followed by procedures that are used to allocate rings for networking processes. Examples are also provided to show the mechanism works when you issue commands to allocate rings.

MAC Clients and Ring Allocation

MAC clients such as VNICs and other datalinks are configured over the NIC to enable communication between a system and other network nodes. After a client is configured, it uses both Rx and Tx rings to receive or transmit network packets respectively. A MAC client can either be hardware-based or software-based. A hardware-based client fulfills any one of the following conditions:

- It has dedicated use of one or more Rx rings.
- It has dedicated use of one or more Tx rings.
- It has dedicated use of one or more Rx rings and one or more Tx rings.

Clients that do not fulfill any of these conditions are called software-based MAC clients.

Hardware-based clients can be assigned rings for exclusive use depending on the NIC. NICs such as `nxge` support *dynamic ring allocation*. On such NICs, you can configure not only hardware-based clients. You also have the flexibility to determine the number of rings to allocate to such clients, assuming that rings remain available for allocation. Use of rings is always optimized for the primary interface, for example, `nxge0`. The primary interface is also known as the *primary client*. Any available rings that have not been assigned for exclusive use by other clients are automatically assigned to the primary interface.

Other NICs such as `ixge` only support *static ring allocation*. On these NICs, you can only create hardware-based clients. The clients are automatically configured with a fixed set of rings per client. The fixed set is determined during the NIC driver's initial configuration. For more information about a driver's initial configuration for static ring allocation, refer to the *Oracle Solaris Tunable Parameters Reference Manual*.

Ring Allocation in VLANs

With VLANs, the assignment of rings proceeds differently depending on how the VLAN is created. VLANs are created in one of two ways:

- By using the `dladm create-vlan` subcommand:

  ```
  # dladm create-vlan -l link -v VID vlan
  ```

- By using the `dladm create-vnic` subcommand:

  ```
  # dladm create-vnic -l link -v VID vnic
  ```

A VLAN that is created by the `dladm create-vlan` subcommand shares the same MAC address as the underlying interface. Consequently, that VLAN also shares the Rx and Tx rings of the underlying interface. A VLAN that is created as a VNIC by using the `dladm create-vnic` command has a different MAC address from its underlying interface. The allocation of rings for

such a VLAN is independent of the allocation for the underlying link. Thus, that VLAN can be assigned its own dedicated rings, assuming that the NIC supports hardware-based clients.

Properties for Ring Allocation

To administer rings, two ring properties can be set by using the `dladm` command:

- `rxrings` refers to the number of assigned Rx rings to a specified link.
- `txrings` refers to the number of assigned Tx rings to a specified link.

You can set each property to one of three possible values:

- `sw` indicates that you are configuring a software-based client. The client does not have exclusive use of rings. Rather, the client shares rings with any other existing clients that are similarly configured.
- *n > 0* (number greater than zero) applies to the configuration of a hardware-based client only. The number refers to the quantity of rings that you allocate to the client for its exclusive use. You can specify a number only if the underlying NIC supports dynamic ring allocation.
- `hw` also applies to the configuration of a hardware-based client. However, for such a client, you cannot specify the actual number of dedicated rings. Rather, the fixed number of rings per client is already set according to the NIC driver's initial configuration. You set the `*rings` properties to `hw` if the underlying NIC supports static ring allocation only.

To provide information about current ring assignments and use, the following additional read-only ring properties are available:

- `rxrings-available` and `txrings-available` indicate the number of Rx and Tx rings that are available for allocation.
- `rxhwclnt-available` and `txhwclnt-available` indicate the number of Rx and Tx hardware-based clients that can be configured over a NIC.

Preparations for Configuring Hardware-Based Clients

Before you configure hardware-based clients, you must know the ring allocation capabilities of the NIC on your system. To obtain the required information, use the following command:

```
# dladm show-linkprop link
```

where *link* refers to the datalink of your specific NIC.

To display specific properties, use the following command:

```
# dladm show-linkprop -p property[,property,...] link
```

To properly configure hardware-based clients, you must determine the following:

- Whether the NIC supports hardware-based clients

The `rxrings` and `txrings` properties in the command output indicate whether a NIC supports hardware-based clients. From the same data, you can also determine the type of ring allocation that is supported by the NIC.

- The availability of rings to allocate to hardware-based clients

  The `rxrings-available` and `txrings-available` properties in the command output indicate the available Rx rings and Tx rings that you can allocate to a hardware-based client.

- The availability of hardware-based clients that you can configure on the link

  Rings are allocated as sets. No one-to-one correspondence exists between the number of available rings and the number of clients that can use dedicated rings. Consequently, to allocate rings, you must check not only the availability of rings but also the number of additional hardware-based clients that you can still configure to use dedicated rings. You can allocate rings only if both rings and hardware-based clients are available.

  The `rxhwclnt-available` and `txhwclnt-available` properties in the command output indicate how many hardware-based clients you can configure that can use dedicated Rx and Tx rings.

If the NIC supports ring allocation, and rings and hardware-based clients are available, then you can configure this type of client on the system, as explained in "How to Configure a Hardware-Based Client" on page 375. Alternatively, you can configure a software-based client instead, as explained in "How to Create a Software-Based Client" on page 376.

The following examples show different information that is displayed for ring-related link properties of an `nxge` NIC, an `ixgbe` NIC, and an `e1000g` NIC.

EXAMPLE 21–1 `nxge` NIC Ring Information

The following example shows ring information for an `nxge` NIC.

```
# dladm show-linkprop nxge0
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
nxge0   rxrings             rw    --     --       sw,<1-7>
...
nxge0   txrings             rw    --     --       sw,<1-7>
...
nxge0   rxrings-available   r-    5      --       --
nxge0   txrings-available   r-    5      --       --
nxge0   rxhwclnt-available  r-    2      --       --
nxge0   txhwclnt-available  r-    2      --       --
...
```

The `POSSIBLE` field lists `sw` and `<1-7>` as acceptable values for the `rxrings` and `txrings` properties. These values indicate that `nxge` supports hardware-based clients as well as software-based clients. The range `<1-7>` indicates that the number of Rx rings or Tx rings you set must be within the specified range. You can also infer from the range that the NIC supports dynamic ring allocation for both the receive and transmit sides.

EXAMPLE 21–1 nxge NIC Ring Information *(Continued)*

In addition, the `*rings-available` properties indicate that five Rx rings and five Tx rings are available to allocate to hardware-based clients.

However, based on the `*clnt-available` properties, you can configure only two clients that can have exclusive use of available Rx rings. Likewise, you can configure only two clients that can have exclusive use of available Tx rings.

EXAMPLE 21–2 ixgbe NIC Ring Information

The following example shows ring information for an `ixgbe` NIC.

```
# dladm show-linkprop ixgbe0
LINK     PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
ixgbe0   rxrings             rw     --     --       sw,hw
...
ixgbe0   txrings             rw     --     --       sw,hw,<1-7>
...
ixgbe0   rxrings-available   r-     0      --       --
ixgbe0   txrings-available   r-     5      --       --
ixgbe0   rxhwclnt-available  r-     0      --       --
ixgbe0   txhwclnt-available  r-     7      --       --
...
```

The `POSSIBLE` field for both the `rxrings` and `txrings` properties indicates that both hardware-based clients and software-based clients can be configured on `ixgbe0`. Only static ring allocation is supported for Rx rings, where the hardware assigns a fixed set of Rx rings to each hardware-based client. However, you can allocate Tx rings dynamically, meaning that you can determine the number of Tx rings to assign to a hardware-based client, in this example, up to seven rings.

In addition, the `*rings-available` properties indicate that five Tx rings are available to allocate to hardware-based clients, but no Rx rings can be assigned.

Finally, based on the `*hwclnt-available` properties, you can configure seven hardware-based Tx clients to use Tx rings exclusively. However, dynamic Rx ring allocation is not supported in `ixgbe` cards. Therefore, you cannot create a hardware-based client with a specified set of dedicated Rx rings.

A zero (`0`) under the `VALUE` field for either of the `*rings-available` properties can mean one of the following:

- No more rings are available to allocate to clients.
- Dynamic ring allocation is not supported.

You can verify the meaning of the zero by comparing the `POSSIBLE` field for `rxrings` and `txrings` and the `VALUE` field for `rxrings-available` and `txrings-available`.

For example, suppose that `txrings-available` is `0`, as follows:

EXAMPLE 21–2 `ixgbe` NIC Ring Information *(Continued)*

```
# dladm show-linkprop ixgbe0
LINK     PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
ixgbe0   rxrings             rw    --     --       sw,hw
ixgbe0   txrings             rw    --     --       sw,hw,<1-7>
ixgbe0   rxrings-available   r-    0      --       --
ixgbe0   txrings-available   r-    0      --       --
...
```

In this output, the `VALUE` field for `rxrings-available` is `0` while the `POSSIBLE` field for `rxrings` is `sw,hw`. The combined information means that no Rx rings are available because the NIC does not support dynamic ring allocation. On the transmit side, the `VALUE` field for `txrings-available` is `0` while the `POSSIBLE` field for `txrings` is `sw,hw,<1-7>`. The combined information indicates that, no Tx rings are available because all the Tx rings have already been allocated. However, as the `POSSIBLE` field for `txrings` indicates, dynamic ring allocation is supported. Thus, you can allocate Tx rings as these rings become available.

EXAMPLE 21–3 `e1000g` NIC Ring Information

The following example shows ring information for an `e1000g` NIC.

```
# dladm show-linkprop e1000g0
LINK     PROPERTY             PERM  VALUE  DEFAULT  POSSIBLE
...
e1000g0   rxrings             rw     --      --     --
...
e1000g0   txrings             rw     --      --     --
...
e1000g0   rxrings-available   r-     0       --     --
e1000g0   txrings-available   r-     0       --     --
e1000g0   rxhwclnt-available  r-     0       --     --
e1000g0   txhwclnt-available  r-     0       --     --
...
```

The output indicates that neither rings nor hardware-based clients can be configured because ring allocation is not supported in `e1000g` NICs.

▼ How to Configure a Hardware-Based Client

This procedure shows how to configure a hardware-based client either on a NIC that supports dynamic ring allocation or on a NIC that supports static ring allocation.

Before You Begin Make sure that you have obtained the following information about the NIC on your system:

- Whether the NIC supports hardware-based clients
- The type of ring allocation that the NIC supports
- The availability of rings to allocate to hardware-based clients
- The availability of hardware-based clients that you can configure on the link

To obtain the information, refer to "Preparations for Configuring Hardware-Based Clients" on page 372.

1 Perform one of the following steps depending on the type of ring allocation that your NIC supports:

- If the NIC supports dynamic ring allocation, use the following syntax:

  ```
  # dladm create-vnic -p rxrings=number[,txrings=number] -l link vnic
  ```

  number Refers to the number of Rx rings and Tx rings that you allocate to the client. The number must be within the range of the number of available rings for allocation.

  Note – Some NICs support dynamic allocation on either Rx rings or Tx rings, but not on both types. You specify *number* on the ring type for which dynamic ring allocation is supported.

  link Refers to the datalink over which you are creating the client.

  vnic Refers to the client that you are configuring.

- If the NIC supports static ring allocation, use the following syntax:

  ```
  # dladm create-vnic -p rxrings=hw[,txrings=hw] -l link vnic
  ```

  Note – Some NICs support static allocation on either Rx rings or Tx rings, but not on both types. You specify `hw` on the ring type for which static ring allocation is supported.

2 (Optional) Check the newly created client's ring information.

```
# dladm show-linkprop vnic
```

How to Create a Software-Based Client

A software-based client does not have exclusive use of rings. Rather, the client shares the use of rings with the primary client or interface with other existing software-based clients. The ring count for software-based clients depends on the number of existing hardware-based clients.

Perform one of the following steps:

- To create a new software-based client, type the following command:

  ```
  # dladm create-vnic -p rxrings=sw[,txrings=sw] -l link vnic
  ```

  link Refers to the datalink over which you are creating the client.

  vnic Refers to the client that you are configuring.

- To configure an existing client to share rings with other clients, type the following command:

  ```
  # dladm set-linkprop -p rxrings=sw[,txrings=sw] vnic
  ```

Example 21–4 Configuring Hardware-Based Clients and Software-Based Clients

This example shows how to configure both hardware-based clients and software-based clients on a system with an `ixgbe` NIC. To show how ring allocation is implemented, the example is divided into parts. Ring-related information is displayed and explained at each step of the configuration process. The configuration proceeds as follows:

1. Display the links and ring usage on the system prior to the configuration of clients.
2. Configure the primary client.
3. Configure a software-based client.
4. Configure another client without any dedicated rings.
5. Statically allocate rings to the newly configured client.
6. Configure a third client with dedicated rings that are dynamically allocated.

First, display the links, ring usage, and ring-related properties.

```
# dladm show-link
LINK     CLASS   MTU    STATE   BRIDGE   OVER
ixgbe0   phys    1500   down    --       --

# dladm show-phys -H ixgbe0
LINK     RINGTYPE   RINGS   CLIENTS
ixgbe0   RX         0-1     <default,mcast>
ixgbe0   TX         0-7     <default>
ixgbe0   RX         2-3     --
ixgbe0   RX         4-5     --
ixgbe0   RX         6-7     --

# dladm show-linkprop ixgbe0
LINK    PROPERTY              PERM  VALUE  DEFAULT  POSSIBLE
...
ixgbe0   rxrings              rw    --     --       sw,hw
ixgbe0   rxrings-effective    r     --     --       --
ixgbe0   txrings              rw    --     --       sw,hw,<1-7>
ixgbe0   txrings-effective    r     --     --       --
ixgbe0   txrings-available    r-    7      --       --
ixgbe0   rxrings-available    r-    0      --       --
ixgbe0   rxhwclnt-available   r-    3      --       --
ixgbe0   txhwclnt-available   r-    7      --       --
...
```

The command output shows a single link `ixgbe0` on the system, but no existing clients. In addition, the following information is also gleaned from this output:

- The NIC has eight Rx rings and eight Tx rings (rings `0` to `7`).

- For hardware-based clients, only static ring allocation is supported for Rx rings, while both static and dynamic ring allocations are supported for Tx rings.
- Software-based clients can be configured for both Rx rings and Tx rings.
- Seven Tx rings, `1` to `7`, are available to be dynamically allocated to other clients (ring `0` is typically reserved for the primary client). No Rx rings are available because dynamic ring allocation is not supported for Rx rings.
- Three hardware-based clients can be configured to use Rx rings, while seven hardware-based clients can be configured to use Tx rings.

For an explanation of the `*rings-effective` properties, see "How to Identify Ring Assignments in Static Ring Allocation" on page 382.

Next, configure the primary client.

```
# ipadm create-ip ixgbe0
# ipadm create-addr -T static -a 192.168.10.10/24 ixgbe0/v4
# dladm show-phys -H ixgbe0
LINK     RINGTYPE   RINGS   CLIENTS
ixgbe0   RX         0-1     <default,mcast>
ixgbe0   TX         0-7     <default>ixgbe0
ixgbe0   RX         2-3     ixgbe0
ixgbe0   RX         4-5     --
ixgbe0   RX         6-7     --

# dladm show-linkprop ixgbe0
LINK    PROPERTY              PERM  VALUE  DEFAULT  POSSIBLE
...
ixgbe0   rxrings              rw    --     --       sw,hw
ixgbe0   rxrings-effective    r     2      --       --
ixgbe0   txrings              rw    --     --       sw,hw,<1-7>
ixgbe0   txrings-effective    r     8      --       --
ixgbe0   txrings-available    r-    7      --       --
ixgbe0   rxrings-available    r-    0      --       --
ixgbe0   rxhwclnt-available   r-    3      --       --
ixgbe0   txhwclnt-available   r-    7      --       --
...
```

The output provides the following information:

- `ixgbe0`, the primary client, automatically receives two Rx rings (rings `2` and `3`) for dedicated use. However, `ixgbe0` uses all Tx rings. By default, all unused rings are automatically assigned to the primary client.
- The number of available Tx rings that can be allocated to other clients remains at seven.
- The number of available hardware-based clients that can be configured with Rx rings remains at three. The number of available hardware-based clients that can be dynamically configured with Tx rings remains at seven.

Next, create a VNIC as a software-based client.

```
# dladm create-vnic -l ixgbe0 -p rxrings=sw,txrings=sw vnic0
# dladm show-phys -H ixgbe0
LINK     RINGTYPE   RINGS   CLIENTS
ixgbe0   RX         0-1     <default,mcast>,vnic0
ixgbe0   TX         0-7     <default>vnic0,ixgbe0
ixgbe0   RX         2-3     ixgbe0
ixgbe0   RX         4-5     --
ixgbe0   RX         6-7     --

# dladm show-linkprop vnic0
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
vnic0   rxrings             rw    sw     --       sw,hw
...
vnic0   txrings             rw    sw     --       sw,hw,<1-7>
...
# dladm show-linkprop ixgbe0
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
ixgbe0   rxrings             rw    --     --       --
ixgbe0   rxrings-effective   r     2      --       --
ixgbe0   txrings             rw    --     --        sw,hw,<1-7>
ixgbe0   txrings-effective   r     --     --        --
ixgbe0   txrings-available   r-    7      --        --
ixgbe0   rxrings-available   r-    0      --        --
ixgbe0   rxhwclnt-available  r-    3      --        --
ixgbe0   txhwclnt-available  r-    7      --        --
...
```

The output provides the following information:

- As a software based-client, `vnic0` is automatically assigned to use Rx rings `0` and `1`. Other software-based clients with Rx rings that are subsequently created will be assigned to use this pair by default. By default, `vnic0` is also assigned the use of all eight Tx rings (rings `0` to `7`). Other software-based clients with Tx rings that are subsequently created will be assigned to use this set of rings by default.
- As a software-based client, `vnic0`'s `rxrings` and `txrings` properties are accordingly set to `sw`.
- No Tx rings are assigned. Therefore, the number of available Tx rings that can be allocated to other clients remains at seven.
- The number of available hardware-based clients that can be configured with Rx rings remains at three. The number of available hardware-based clients that can be configured with Tx rings remains at seven.

Next, configure another client without any ring allocation.

```
# dladm create-vnic -l ixgbe0 vnic1
# dladm show-phys -H ixgbe0
LINK     RINGTYPE   RINGS     CLIENTS
ixgbe0   RX         0-1       <default,mcast>,vnic0
ixgbe0   TX         0,2-7     <default>vnic0,ixgbe0
ixgbe0   RX         2-3       ixgbe0
```

```
ixgbe0   RX         4-5       vnic1
ixgbe0   RX         6-7       --
ixgbe0   TX         1         vnic1

# dladm show-linkprop vnic1
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
vnic1   rxrings             rw    --     --       sw,hw
vnic1   rxrings-effective   r-    2      --       --
vnic1   txrings             rw    --     --       sw,hw,<1-7>
vnic1   txrings-effective   r-    --     --       --
...

# dladm show-linkprop ixgbe0
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
ixgbe0  rxrings             rw    --     --       sw,hw
ixgbe0  rxrings-effective   r-    2      --       --
ixgbe0  txrings             rw    --     --       sw,hw,<1-7>
ixgbe0  txrings-effective   r-    --     --       --
ixgbe0  txrings-available   r-    7      --       --
ixgbe0  rxrings-available   r-    0      --       --
ixgbe0  rxhwclnt-available  r-    3      --       --
ixgbe0  txhwclnt-available  r-    7      --       --
...
```

The output provides the following information:

- When ring allocation is supported, a client that is configured is considered a hardware-based client, even though the `rxrings` and `txrings` properties are not set. Thus, `vnic1` automatically receives two dedicated Rx rings (rings `4` and `5`) for its use. Likewise, `vnic1` also receives a dedicated Tx ring (ring `1`).
- Of the eight Tx rings, `ixgbe0` and `vnic0` now share seven rings (ring `0` and rings `2` through `7`). Ring `1` has become a dedicated Tx ring for `vnic1`.
- No Tx rings are assigned. Therefore, the number of available Tx rings that can be allocated to other clients remains at seven.
- The number of available hardware-based clients that can be configured with Rx rings remains at three. The number of available hardware-based clients that can be configured with Tx rings remains at seven.

Next, statically allocate rings to the newly configured client, `vnic1`.

```
# dladm set-linkprop -p rxrings=hw,txrings=hw vnic1
# dladm show-phys -H ixgbe0
LINK     RINGTYPE   RINGS     CLIENTS
ixgbe0   RX         0-1       <default,mcast>,vnic0
ixgbe0   TX         0,2-7     <default>vnic0,ixgbe0
ixgbe0   RX         2-3       ixgbe0
ixgbe0   RX         4-5       vnic1
ixgbe0   RX         6-7       --
ixgbe0   TX         1         vnic1
```

```
# dladm show-linkprop vnic1
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
vnic1   rxrings             rw    hw     --       sw,hw
vnic1   rxrings-effective   r-    2      --       --
vnic1   txrings             rw    hw     --       sw,hw,<1-7>
vnic1   txrings-effective   r-    --     --       --
...
# dladm show-linkprop ixgbe0
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
ixgbe0  rxrings             rw     --    --       sw,hw
ixgbe0  rxrings-effective   r-     2     --       --
ixgbe0  txrings             rw     --    --       sw,hw,<1-7>
ixgbe0  txrings-effective   r-     --    --       --
ixgbe0  txrings-available   r-     6     --       --
ixgbe0  rxrings-available   r-     0     --       --
ixgbe0  rxhwclnt-available  r-     3     --       --
ixgbe0  txhwclnt-available  r-     6     --       --
...
```

The output provides the following information:

- The distribution of Rx and Tx rings for `vnic1` remains the same as when `vnic1` was created without ring allocation.
- Similarly, ring information remains the same as when `vnic1` was created without ring allocation.
- The `rxrings` and `txrings` properties of `vnic1` have been explicitly set to `hw`. Consequently, the number of available Tx rings for dynamic allocation has been reduced to six. Likewise, the number of available hardware-based clients that can be configured has been reduced to six.

Next, configure a hardware-based client with Tx rings that are dynamically allocated.

```
# dladm create-vnic -l ixgbe0 -p txrings=2 vnic2
# dladm show-phys -H ixgbe0
LINK     RINGTYPE   RINGS     CLIENTS
ixgbe0   RX         0-1       <default,mcast>,vnic0
ixgbe0   TX         0,4-7     <default>vnic0,ixgbe0
ixgbe0   RX         2-3       ixgbe0
ixgbe0   RX         4-5       vnic1
ixgbe0   RX         6-7       vnic2
ixgbe0   TX         1         vnic1
ixgbe0   TX         2-3       vnic2

# dladm show-linkprop vnic2
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
vnic2   rxrings             rw    --     --       sw,hw
vnic2   rxrings-effective   r-    2      --       --
vnic2   txrings             rw    2      --       sw,hw,<1-7>
vnic2   txrings-effective   r-    2      --       --
...
# dladm show-linkprop ixgbe0
```

```
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
ixgbe0  rxrings             rw    --     --       sw,hw
ixgbe0  rxrings-effective   r-    2      --       --
ixgbe0  txrings             rw    --     --       sw,hw,<1-7>
ixgbe0  txrings-effective   r-    --     --       --
ixgbe0  txrings-available   r-    4      --       --
ixgbe0  rxrings-available   r-    0      --       --
ixgbe0  rxhwclnt-available  r-    3      --       --
ixgbe0  txhwclnt-available  r-    5      --       --
...
```

The output provides the following information:

- The hardware automatically assigned a pair of Rx rings (rings 6 and 7) to `vnic2` for exclusive use. However, `vnic2`'s two dedicated Tx rings (rings 2 and 3) were assigned by the administrator.
- With two Tx rings administratively assigned to `vnic2`, the number of available Tx rings that can be allocated to other clients has been reduced to four.
- With `vnic2` configured as a hardware-based client with two Tx rings, the number of available clients that can be configured has been reduced to five.

How to Identify Ring Assignments in Static Ring Allocation

When you configure a hardware-based client with static ring allocation, the hardware determines the number of rings to assign. However, the `rxrings` and `txrings` properties are set to `hw` and do not indicate the number of rings that are actually allocated. Instead, the number can be obtained by checking the `rxrings-effective` and `txrings-effective` properties.

1 Configure a hardware-based client with static ring allocation by performing one of the following steps:

- To create the client with static ring allocation, type the following command:

  ```
  # dladm create-vnic -l link -p rxrings=hw[,txrings=hw] vnic
  ```

  link Refers to the datalink over which you are creating the client.

  vnic Refers to the client that you are configuring.

- To statically allocate rings to an existing client, type the following command:

  ```
  # dladm set-linkprop -p rxrings=hw[,txrings=hw] vnic
  ```

2 To identify the number of rings that have been allocated, perform the following substeps:

a. Display the client's properties.

```
# dladm show-linkprop link
```

where *link* refers to the hardware-based client or VNIC.

b. Check the value of the `*rings-effective` property that corresponds to the ring type that you allocated statically.

For example, if you statically allocated Rx rings, check the `rxrings-effective` property. If you statically allocated Tx rings, check the `txrings-effective` property. The number indicates how many rings were allocated by the hardware.

3 To verify which rings have been statically allocated, perform the following substeps:

a. Display the NIC's ring usage.

```
# dladm show-phys -H link
```

where *link* refers to the primary client.

b. From the command output, check which Rx rings or Tx rings have been assigned to the hardware-based client that you configured in the first step.

Example 21–5 Identifying Rings That Are Statically Allocated

This example shows how Rx rings have been statically allocated to a client that is configured over an `ixgbe` NIC. On such a NIC, only static allocation is supported for Rx rings. The example proceeds as follows:

1. Display the links on the system. In this example, the system has only one link, which is `ixgbe0`.
2. Create `vnic1` as a hardware-based client with Rx rings that are statically allocated.
3. Display ring information to know the number of rings allocated by the hardware.
4. Display ring usage to identify which rings have been allocated.

```
# dladm show-link
LINK     CLASS   MTU    STATE   BRIDGE   OVER
ixgbe0   phys    1500   down    --       --

# dladm create-vnic -l ixgbe0 -p rxrings=hw vnic1
# dladm show-linkprop vnic1
LINK    PROPERTY            PERM  VALUE  DEFAULT  POSSIBLE
...
vnic1   rxrings             rw    hw     --       sw,hw
vnic1   rxrings-effective   r-    2      --       --
vnic1   txrings             rw    --     --       sw,hw,<1-7>
vnic1   txrings-effective   r-    --     --       --

# dladm show-phys -H ixgbe0
LINK     RINGTYPE   RINGS     CLIENTS
ixgbe0   RX         0-1       <default,mcast>
ixgbe0   TX         0,2-7     <default>
ixgbe0   RX         2-3       vnic1
ixgbe0   RX         4-5       --
ixgbe0   RX         6-7       --
```

```
ixgbe0   TX         1         vnic1
...
```

The output indicates that after `vnic1` was configured with Rx rings, the hardware allocated two dedicated Rx rings, as reflected by the `rxrings-effective` property. Based on the output of the **`dladm show-phys -H`** command, Rx rings `2` and `3` were dedicated for `vnic1`'s use.

As a result of being configured as a client, `vnic1` also automatically received Tx ring `1` for its dedicated use. However, the `txrings-effective` property displays no value because the `txrings` property is not explicitly set.

Pools and CPUs

The *pool* is a link property that enables you to bind network processing to a pool of CPUs. With this property, you can better integrate network resource management with CPU binding and administration in zones. In Oracle Solaris, zone administration includes the binding of non-networking processes to a pool of CPU resources by using the `zonecfg` or `poolcfg` command. To bind that same pool of resources to also manage network processes, you use the `dladm set-linkprop` command to configure a link's `pool` property. Then you assign that link to the zone.

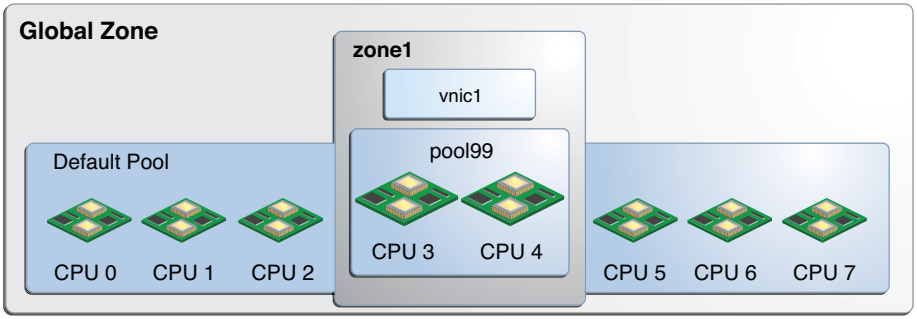
By setting the `pool` property for a link and assigning the link as the zone's network interface, then that link becomes bound to a zone's pool as well. If that zone is set to become an exclusive zone, then CPU resources in the pool can no longer be used by other datalinks that are not assigned to that zone.

Note – A separate property, `cpu`, can be set to assign specific CPUs to a datalink. The two properties, `cpu` and `pool`, are mutually exclusive. You cannot set both properties for a given datalink. To assign CPU resources to a datalink by using the `cpu` property, see "How to Allocate CPUs to Links" on page 388.

For more information about pools within a zone, see Chapter 13, "Creating and Administering Resource Pools (Tasks)," in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*. For more information about creating pools and assigning CPU sets to the pools, refer to the `poolcfg`(1M) man page.

The following figure show how pools work when the `pool` property is assigned to a datalink.

FIGURE 21–1 `pool` Property of a VNIC Assigned to a Zone

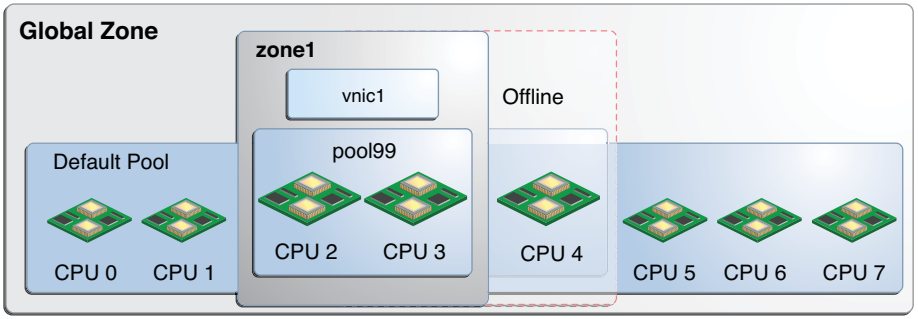


In the figure, the system has eight CPUs. When no pools are configured on the system, all the CPUs belong to the *default pool* and are used by the global zone. However, in this example, the `pool99` pool has been created and consists of `CPU 3` and `CPU 4`. This pool is associated with `zone1`, which is an exclusive zone. If `pool99` is set as a property of `vnic1`, then `pool99` becomes dedicated to also manage `vnic1`'s networking processes. After `vnic1` is assigned to be `zone1`'s network interface , then the CPUs in `pool99` become reserved to manage both networking and non–networking processes of `zone1`.

The `pool` property is dynamic in nature. Zone pools can be configured with a range of CPUs, and the kernel determines which CPUs are assigned to the pool's CPU set. Changes to the pool are automatically implemented for the datalink, which simplifies pool administration for that link. By contrast, assigning specific CPUs to the link by using the `cpu` property requires you to specify the CPU to be assigned. You have to set the `cpu` property every time you want to change the CPU components of the pool.

For example, suppose that in the system in Figure 21–1, `CPU 4` is taken offline. Because the `pool` property is dynamic, the software automatically associates an additional CPU with the pool. Thus, the pool's original configuration of two CPUs is preserved. For `vnic1`, the change is transparent. The adjusted configuration is shown in the following figure.

FIGURE 21–2 Automatic Reconfiguration of the pool Property



Additional pool related properties display information about a datalink's use of CPUs or a pool of CPUs. These properties are read-only and cannot be set by the administrator.

- `pool-effective` displays the pool that is being used for network processes.
- `cpus-effective` displays the list of CPUs that are being used for network processes.

To manage CPU resources of a zone, setting a datalink's `pool` property is not normally performed as an initial step. More frequently, commands such as `zonecfg` and `poolcfg` are used to configure a zone to use a pool of resources. The `cpu` and `pool` link properties themselves are not set. In such cases, the `pool-effective` as well as the `cpus-effective` properties of these datalinks are set automatically according to those zone configurations when the zone is booted. The default pool is displayed under `pool-effective`, while the value of `cpus-effective` is selected by the system. Thus, if you use the `dladm show-linkprop` command, the `pool` and `cpu` properties will be empty, while the `pool-effective` and `cpus-effective` properties will contain values.

Directly setting the `pool` and `cpu` properties of a datalink is an alternative step that you can use to bind a zone's CPU pool for networking processes. After you configure these properties, their values are reflected in the `pool-effective` and `cpus-effective`properties as well. Note, however, that this alternative step is less typically used to manage a zone's network resources.

How to Configure a CPU Pool for a Datalink

As with other link properties, the `pool` property can be set for a datalink either at the moment when the link is created or later when the link requires further configuration. For example:

```
# dladm create-vnic -p pool=pool-name -l link vnic
```

sets the `pool` property while you create the VNIC. To set the `pool` property of an existing VNIC, you use the following syntax:

```
# dladm setlinkprop -p pool=pool-name vnic
```

The following procedure provides the steps to configure a CPU pool for a VNIC.

Before You Begin You must have completed the following:

- Created a processor set with its assigned number of CPUs.
- Created a pool with which the processor set will be associated.
- Associated the pool with the processor set.

Note – For the instructions to complete these prerequisites, see "How to Modify a Configuration" in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*

---

1 Set the link's `pool` property to the pool of CPUs that you created for the zone. Perform one of the following steps, depending on whether the VNIC exists.

- If the VNIC has not yet been created, use the following syntax:

  ```
  # dladm create-vnic -l link -p pool=pool vnic
  ```

  where *pool* refers to the name of the pool that was created for the zone.

- If the VNIC exists, use the following syntax:

  ```
  # dladm setlinkprop -p pool=pool vnic
  ```

2 Set a zone to use the VNIC.

```
zonecfg>zoneid:net> set physical=vnic
```

Note – For the complete steps that explain how to assign a networking interface to a zone, refer to the "Configuring, Verifying, and Committing a Zone" in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*

Example 21–6 Assigning a Link's CPU Pool to a Zone With an Exclusive IP-Type

This example shows how a pool is assigned to a zone's datalink. The scenario is based on the configuration in Figure 21–1. The example assumes that a pool of CPUs named `pool99` has already been configured for the zone. The pool is then assigned to a VNIC. Finally, the non–global zone `zone1` is set to use the VNIC as the networking interface.

```
# dladm create-vnic -l e1000g0 -p pool99 vnic0

# zonecfg -c zone1
zonecfg:zone1> set ip-type=exclusive
zonecfg:zone1> add net
zonecfg:zone1>net> set physical=vnic0
```

```
zonecfg:zone1>net> end
zonecfg:zone1> exit
```

How to Allocate CPUs to Links

The following procedure explains how to assign specific CPUs to process traffic traversing a datalink by configuring the `cpu` property.

1 Check CPU assignments for the interface.

```
# dladm show-linkprop -p cpus link
```

By default, no CPUs are assigned to any specific interface. Thus, the parameter `VALUE` in the command output will not contain any entry.

2 List the interrupts and the CPUs with which the interrupts are associated.

```
# echo ::interrupts | mdb -k
```

The output lists parameters for each link in the system, including the CPU number.

3 Assign CPUs to the link.

The CPUs can include those with which the link's interrupts are associated.

```
# dladm set-linkprop -p cpus=cpu1,cpu2,... link
```

where *cpu1* is the CPU number to be assigned to the link. You can dedicate multiple CPUs to the link.

4 Check the link interrupt to verify the new CPU assignments.

```
# echo ::interrupts | mdb -k
```

5 (Optional) Display the CPUs that are associated with the link.

```
# dladm show-linkprop -p cpus link
```

Example 21–7 Allocating CPUs to the Interface

This example shows how to dedicate specific CPUs to the `internal0` interface in Figure 18–3.

Note the following information in the output that is generated by the different commands. For clarity, the significant information is emphasized in the output.

- By default `internal0` has no dedicated CPU. Thus `VALUE` is `--`.
- The interrupt of `internal0` is associated with CPU `18`.
- After CPUs are allocated, `internal0` displays a new CPU list under `VALUE`.

```
# dladm show-linkprop -p cpus internal0
LINK          PROPERTY     PERM     VALUE     DEFAULT     POSSIBLE
internal0     cpus         rw       --        --          --
```

```
# echo ::interrupts | mdb -k
    Device   Shared   Type   MSG #   State   INO    Mondo   Pil   CPU
external#0   no       MSI    3       enbl    0x1b   0x1b    6     0
internal#0   no       MSI    2       enbl    0x1a   0x1a    6     18

# dladm set-linkprop -p cpus=14,18,19,20 internal0

# dladm show-linkprop -p cpus internal0
LINK        PROPERTY   PERM   VALUE        DEFAULT   POSSIBLE
internal0   cpus       rw     14,18,19,20  --        --
```

All the supporting threads including the interrupt are now confined to the newly assigned set of CPUs.

Managing Resources on Flows

Flows consist of network packets that are organized according to an attribute. Flows enable you to further allocate network resources. For an overview of flows, see "Network Resource Management by Using Flows" on page 368.

To use flows for managing resources, you perform the following general steps:

1. Create the flow by basing it on a specific attribute as listed in "Network Resource Management by Using Flows" on page 368.
2. Customize the flow's use of resources by setting properties that pertain to network resources. Currently, only the bandwidth for processing packets can be set.

Configuring Flows on the Network

Flows can be created on the physical network as well as the virtual network. To configure flows, you use the `flowadm` command. For detailed technical information, refer to the `flowadm(1M)` man page.

▼ How to Configure a Flow

1 (Optional) Determine the link on which you will configure flows.

```
# dladm show-link
```

2 Verify that IP interfaces over the selected link are properly configured with IP addresses.

```
# ipadm show-addr
```

3 Create flows according to the attribute you have determined for each flow.

```
# flowadm add-flow -l link -a attribute=value[,attribute=value] flow
```

attribute Refers to one of the following classifications by which you can organize network packets into a flow:

- IP address
- Transport protocol (UDP, TCP, or SCTP)
- Port number for an application (for example port 21 for FTP)
- DS field attribute, which is used for quality of service in IPv6 packets only. For more information about the DS field, refer to "DS Codepoint" in *Oracle Solaris Administration: IP Services*.

flow Refers to the name that you assign to the particular flow.

For more details about flows and flow attributes, see the `flowadm(1M)` man page.

4 Implement resource controls on the flows by setting the appropriate flow properties.

```
# flowadm set-flowprop -p property=value[,property=value,...] flow
```

You can specify the following flow properties that control resources:

`maxbw` The maximum amount of the link's bandwidth that packets identified with this flow can use. The value you set must be within the allowed range of values for the link's bandwidth. To display the possible range of values for a link's bandwidth, check the `POSSIBLE` field in the output that is generated by the following command:

```
# dladm show-linkprop -p maxbw link
```

Note – Currently, only a flow's bandwidth can be customized.

5 (Optional) Display the flows that you have created over the link.

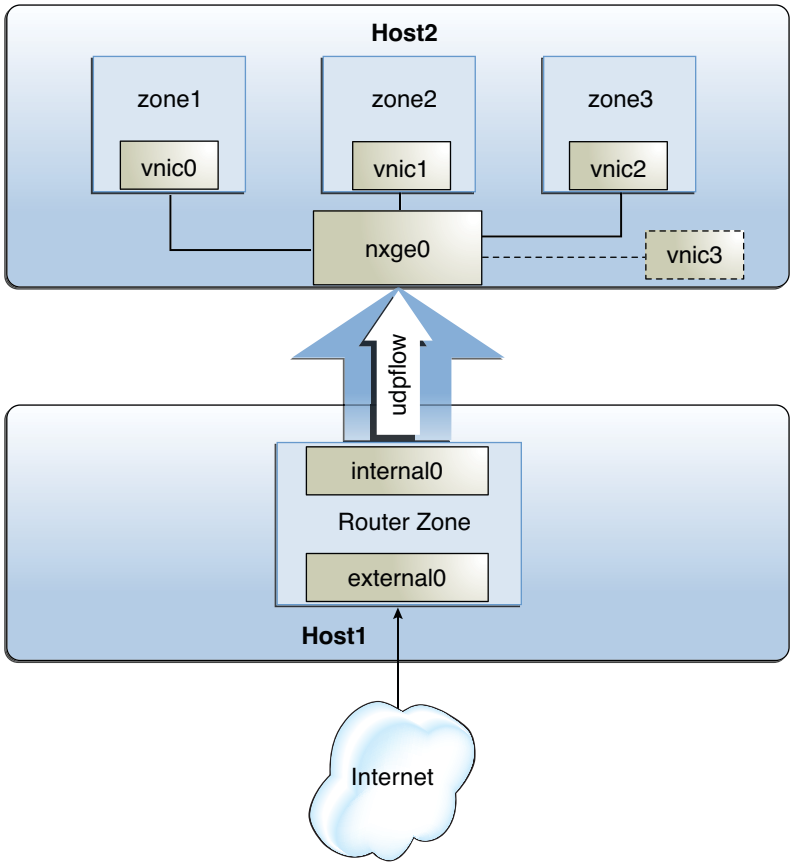
```
# flowadm show-flow -l link
```

6 (Optional) Display the property settings for a specified flow.

```
# flowadm show-flowprop flow
```

Example 21–8 Managing Resources by Setting Link and Flow Properties

This example combines the steps for allocating network resources to both datalinks and flows. The example is based on the configuration shown in the following figure.



The figure shows two physical hosts that are connected to each other.

- `Host1` has the following configuration:
  - It has one non-global zone that functions as a router zone. Two interfaces are assigned to the zone: `external0` connects to the Internet while `internal0` connects to the internal network including the second host.
  - The IP interfaces have been renamed to use customized names. Although not required, using customized names on links and interfaces provides advantages when you administer the network. See "Network Devices and Datalink Names" on page 26.
  - A flow is configured over `internal0` to isolate UDP traffic and implement control over how UDP packets use resources. For information about configuring flows, see "Managing Resources on Flows" on page 389.
- `Host2` has the following configuration:

- It has three non-global zones and their respective VNICs. The VNICs are configured over an nxge card that supports dynamic ring allocation. For more information about ring allocation, see "Transmit and Receive Rings" on page 370.
- Each zone's network processing load is different. For the purposes of this example, the load for `zone1` is heavy, the load for `zone2` is medium, and the load for `zone3` is light. Resources are assigned to these zones according to their loads.
- A separate VNIC is configured as a software-based client. For an overview of MAC clients, see "MAC Clients and Ring Allocation" on page 371.

The tasks in this example involve the following:

- Creating a flow and configuring flow controls – A flow is created over `internal0` to create separate resource controls over UDP packets that are received by `Host2`.
- Configuring network resource properties for the VNICs on `Host2` – Based on the processing load on each zone, each zone's VNIC is configured with a set of dedicated rings. A separate VNIC is also configured without dedicated rings as an example of a software-based client.

Note that the example does not include any procedure for zone configuration. To configure zones, refer to Chapter 17, "Planning and Configuring Non-Global Zones (Tasks)," in *Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management*.

First, view information about links and IP interfaces on `Host1`.

```
# dladm show-phys
LINK         MEDIA      STATE      SPEED DUPLEX   DEVICE
internal0    Ethernet   up         1000 full      nge1
e1000g0      n          unknown    0    half      e1000g0
e1000g1      n          unknown    0    half      e1000g1
external0    Ethernet   up         1000 full      nge0

# dladm show-link
LINK        CLASS    MTU    STATE    BRIDGE  OVER
internal0   phys     1500   up       --      nge1
e1000g0     phys     1500   unknown  --      --
e1000g1     phys     1500   unknown  --      --
external0   phys     1500   up       --      nge0

# ipadm show-addr
ADDROBJ     TYPE     STATE     ADDR
lo0/4       static   ok        127.0.0.1/8
external0   static   ok        10.10.6.5/24
internal0   static   ok        10.10.12.42/24
```

Next, create a flow over `internal0` to isolate UDP traffic to `Host2`. Then, implement resource controls on the flow.

```
# flowadm add-flow -l external0 -a transport=udp udpflow
# flowadm set-flowprop -p maxbw=80 udpflow
```

Then, check the information about the created flow.

```
flowadm show-flow
FLOW        LINK        IPADDR   PROTO   PORT   DFSLD
udpflow     internal0   --       udp     --     --

# flowadm show-flowprop
SECURE OUTPUT FOR THIS
```

On `Host2`, configure VNICs over `nxge0` for each zone. Implement resource controls on each VNIC. Then, assign the VNICs to their respective zones.

```
# dladm create-vnic -l nxge0 vnic0
# dladm create-vnic -l nxge0 vnic1
# dladm create-vnic -l nxge0 vnic2

# dladm set-prop -p rxrings=4,txrings=4 vnic0
# dladm set-prop -p rxrings=2,txrings=2 vnic1
# dladm set-prop -p rxrings=1,txrings=1 vnic2

# zone1>zonecfg>net> set physical=vnic0
# zone2>zonecfg>net> set physical=vnic1
# zone3>zonecfg>net> set physical=vnic2
```

Suppose that `pool1`, a set of CPUs in `Host2`, was previously configured for use by `zone1`. Bind that pool of CPUs to also manage network processes for `zone1` as follows:

```
# dladm set-prop -p pool=pool01 vnic0
```

Finally, create a software-based client that shares rings with `nxge0`, the primary interface.

```
dladm create-vnic -p rxrings=sw,txrings=sw -l nxge0 vnic3
```

◆ ◆ ◆ CHAPTER 22

Monitoring Network Traffic and Resource Usage

This chapter describes tasks for monitoring and gathering statistics about the use of network resources in a physical as well as a virtual network environment. The information can help you analyze resource allocation for provisioning, consolidation, and billing purposes. This chapter introduces the two commands that you use to display statistics: `dlstat` and `flowstat`.

The following subjects are discussed:

Overview of Network Traffic Flow

Packets traverse a path when they flow into or out of a system. On a granular level, packets are received and transmitted through receive (Rx) rings and transmit (Tx) rings of a NIC. From these rings, received packets are passed up the network stack for further processing while outbound packets are sent to the network.

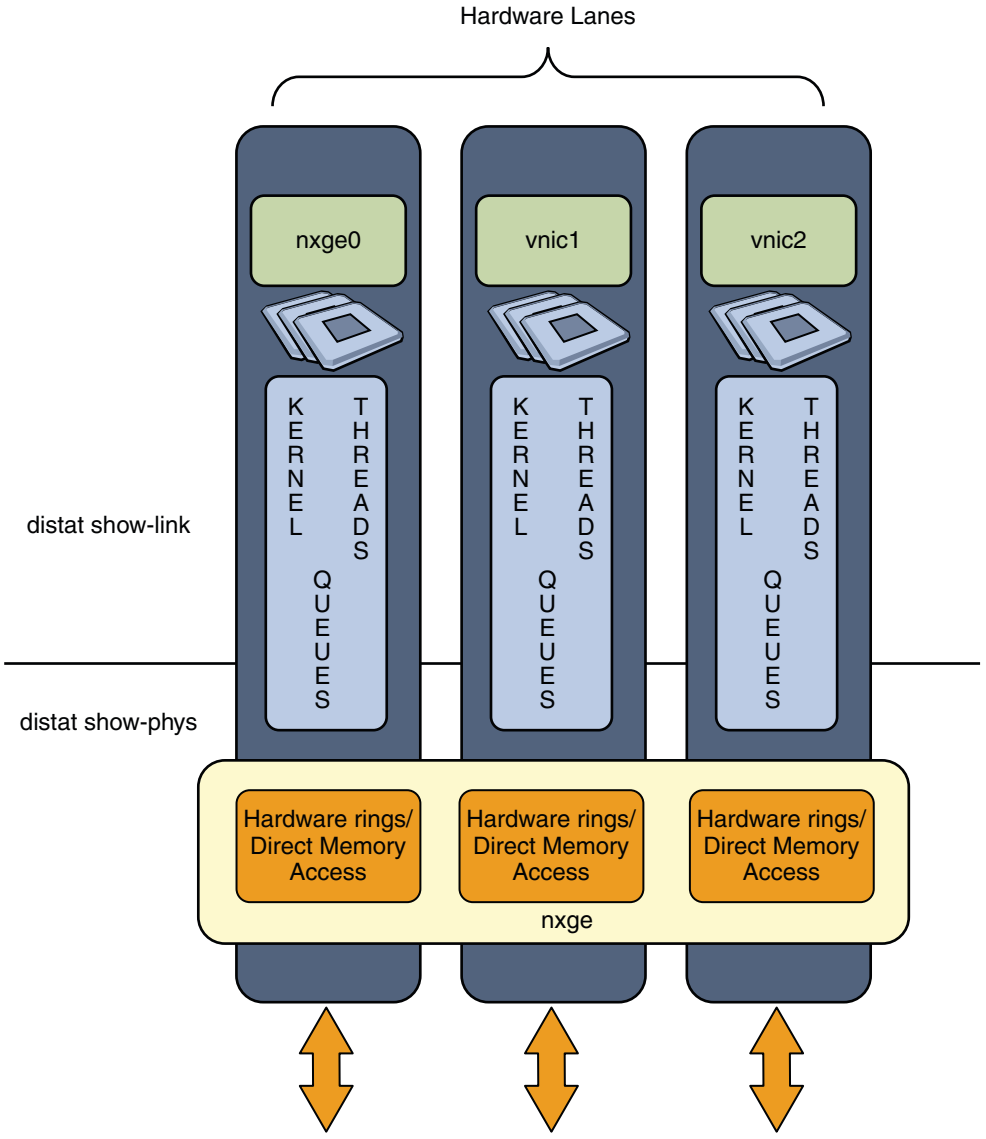
Chapter 21, "Managing Network Resources," introduces the concept of network lanes. A combination of system resources that are allocated to manage network traffic constitutes a network lane. Thus, *network lanes* are customized paths for specific types of network traffic. Each lane can be either a *hardware* lane or a *software* lane. In addition, each lane type can be either a *receive* lane or a *transmit* lane. The distinction between hardware and software lanes is based on a NIC's ability to support ring allocation. For more information about ring allocation, see "Transmit and Receive Rings" on page 370. This chapter focuses primarily on incoming traffic that is received through receive lanes.

On hardware lanes, rings are dedicated to the packets that use those lanes. In contrast, rings on software lanes are shared among datalinks. Datalinks are configured to share rings for the following reasons:

- Administrative intent. The datalink might not be performing intensive processes to require dedicated rings.
- The NIC does not support ring allocation.
- Despite support for ring allocation, rings are no longer available to be assigned for exclusive use.

Consider the following figure that shows different hardware lanes:

FIGURE 22–1 Hardware Lanes



The figure shows the following configuration:

- The system has a single NIC, `nxge`.
- Links are configured over the physical device: `nxge0`, `vnic1`, and `vnic2`. Note that as a datalink, `nxge0` can be assigned a customized name. However, in the figure, the link retains its default device name.

- The system has multiple CPUs.
- The NIC supports dynamic ring allocation. Thus, a set of hardware rings can be assigned to each link to constitute a hardware lane. In addition, a set of CPUs is also allocated to each lane.

Monitoring Traffic and Use of Resources (Task Map)

You can obtain information about how packets use network resources by observing packet flow on network lanes. The `dlstat` command provides this information about datalinks. The `flowstat` command performs similar functions for existing flows.

The following table lists different methods you can use to obtain statistics about network traffic and the use of resources in the system.

Task	Description	For Instructions
Obtain statistical information about network traffic.	View incoming and outgoing traffic on a system's network interfaces.	"How to Obtain Basic Statistics About Network Traffic" on page 399
Obtain statistical information about ring use.	View how incoming and outgoing traffic is distributed among a NIC's rings.	"How to Obtain Statistics About Ring Usage" on page 401
Obtain statistical information about network traffic on specific lanes.	View detailed information about incoming and outgoing traffic as packets traverse network lanes that are configured on a system's network interfaces.	"How to Obtain Statistics About Network Traffic on Lanes" on page 402
Obtain statistical information about traffic on flows.	View information about incoming and outgoing traffic traversing user–defined flows.	"How to Obtain Statistics on Flows" on page 405
Configure accounting of network traffic.	Configure network accounting to capture traffic information for accounting purposes.	"How to Configure Extended Network Accounting" on page 407
Obtain historical statistics on network traffic.	Extract information from the log file of extended network accounting to obtain historical statistics of network traffic on lanes as well as flows.	"How to Obtain Historical Statistics on Network Traffic" on page 408

For a description of the steps to configure flows, see "Managing Resources on Flows" on page 389. For more information about these two commands, see the `dlstat(1M)` and the `flowstat(1M)` man pages.

Gathering Statistics About Network Traffic on Links

The `dlstat` and `flowstat` commands are tools for monitoring and obtaining statistics on network traffic on datalinks and flows, respectively. These commands parallel the `dladm` and `flowadm` commands. The following table shows the parallelism between the pair of `*adm` commands and the pair of `*stat` commands and their respective functions:

Administrative Commands		Monitoring Commands	
Command	**Function**	**Command**	**Function**
`dladm` command options	User interface and tool for configuring and administering datalinks.	`dlstat` command options	User interface and tool for obtaining statistics on traffic on datalinks.
`flowadm` command options	User interface and tool for configuring and administering flows.	`flowstat` command options	User interface and tool for obtaining statistics on traffic on flows.

The following variants of the `dlstat` command can be used to gather network traffic information:

- `dlstat` – Displays general information about packets that are being received or transmitted by a system.
- `dlstat show-phys` – Displays information about the use of receive and transmit rings. This command corresponds to the `dladm show-phys` command, which displays non-traffic information about a network physical device. For an illustration of the level of the network lane to which this command applies, refer to Figure 22–1.
- `dlstat show-link` – Displays detailed information about traffic flow on a given lane. The lane is identified by its datalink. This command corresponds to the `dladm show-link` and `dladm show-vnic` commands, which display non-traffic information about datalinks. For an illustration of the level of the network lane to which the `dlstat show-link` command applies, refer to Figure 22–1.
- `dlstat show-aggr` – Displays information about the use of ports in a link aggregation. This command corresponds to the `dladm show-aggr` command, which displays non-traffic information about a link aggregation.

▼ How to Obtain Basic Statistics About Network Traffic

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Observe basic traffic flow over all the datalinks.

```
# dlstat [-r|-t] [-i interval] [link]
```

`[-r|-t]` Displays either receive-side statistics only (`-r` option) or transmit-side statistics only (`-t` option). If you do not use these options, then statistics for both the receive-side and the transmit-side are displayed.

`-i` *interval* Specifies the time in seconds at which you want the displayed statistics to be refreshed. If you do not use this option, then static output is displayed.

link Indicates that you want to monitor the statistics of the specified datalink only. If you do not use this option, then information about all datalinks is displayed.

Used by itself, the `dlstat` command displays information about incoming and outgoing packets on all configured datalinks.

The following information is displayed by most of the options that you use with the `dlstat` command:

- Links in the system that have been configured with IP interfaces and that can receive or transmit traffic
- Packet and byte sizes
- Interrupts and MAC polling statistics
- Packet chain lengths

Example 22–1 Displaying Basic Receive-Side and Transmit-Side Statistics

This example shows information about network traffic that is being received and sent on all configured datalinks on the system.

```
# dlstat
   LINK    IPKTS   RBYTES    OPKTS   OBYTES
e1000g0  101.88K   32.86M   40.16K    4.37M
  nxge1    4.50M    6.78G    1.38M   90.90M
  vnic1        8      336        0        0
```

Example 22–2 Displaying Receive-Side Statistics at One-Second Intervals

This example shows information about traffic that is being received on all datalinks. The information is refreshed every second. To stop the display from refreshing, press Control-C.

```
# dlstat -r -i 1
   LINK   IPKTS  RBYTES   INTRS   POLLS   CH<10 CH10-50   CH>50
e1000g0 101.91K  32.86M  87.56K  14.35K   3.70K     205       5
  nxge1   9.61M  14.47G   5.79M   3.82M 379.98K  85.66K   1.64K
  vnic1       8     336       0       0       0       0       0
e1000g0       0       0       0       0       0       0       0
  nxge1  82.13K 123.69M  50.00K  32.13K   3.17K     724      24
```

```
  vnic1       0       0       0       0       0       0       0
...
^C
```

In this output, the statistics for interrupt (`INTRS`) are significant. Low interrupt numbers indicate greater efficiency in performance. If the interrupt numbers are high, then you might need to add more resources to the specific link.

Example 22–3 Displaying Transmit-Side Statistics at Five-Second Intervals

This example displays information about traffic that is being sent on all datalinks. The information is refreshed every 5 seconds.

```
# dlstat -t -i 5
   LINK   OPKTS  OBYTES  BLKCNT UBLKCNT
e1000g0  40.24K   4.37M       0       0
  nxge1   9.76M 644.14M       0       0
  vnic1       0       0       0       0
e1000g0       0       0       0       0
  nxge1  26.82K   1.77M       0       0
  vnic1       0       0       0       0
...
^C
```

How to Obtain Statistics About Ring Usage

1. **Become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **Display ring statistics.**

   ```
   # dlstat show-phys [-r|-t] [-i interval] [link]
   ```

   `[-r|-t]` Displays either receive-side statistics only (`-r` option) or transmit-side statistics only (`-t` option). If you do not use these options, then statistics for both the receive-side and the transmit-side are displayed.

   `-i` *interval* Specifies the time in seconds at which you want the displayed statistics to be refreshed. If you do not use this option, then static output is displayed.

   link Indicates that you want to monitor the statistics of the specified datalink only. If you do not use this option, then information about all datalinks is displayed.

   Used by itself, the `dlstat show-phys` command displays information about incoming and outgoing packets on all configured datalinks.

Example 22–4 Displaying Receive-Ring Statistics for a Datalink

This example shows the usage of the receive rings for the datalink.

```
# dlstat show-phys -r nxge1
     LINK TYPE INDEX   IPKTS  RBYTES
nxge1   rx     0      21   1.79K
nxge1   rx     1       0       0
nxge1   rx     2   1.39M   2.10G
nxge1   rx     3       0       0
nxge1   rx     4   6.81M  10.26G
nxge1   rx     5   4.63M   6.97G
nxge1   rx     6   3.97M   5.98G
nxge1   rx     7       0       0
```

The nxge device has eight receive rings, which are identified under the `INDEX` field. An even distribution of packets per ring is an ideal configuration that indicates that the rings are properly allocated to links according to the links' load. An uneven distribution might indicate a disproportionate distribution of rings per link. The resolution depends on whether the NIC supports dynamic ring allocation, which allows you to redistribute rings per link. For more information about dynamic ring allocation, see "Transmit and Receive Rings" on page 370.

Example 22–5 Displaying Transmit Ring Statistics of a Datalink

This example shows the usage of the transmit rings for the datalink.

```
# dlstat show-phys -t nxge1
 LINK TYPE INDEX   OPKTS   OBYTES
nxge1   tx     0      44    3.96K
nxge1   tx     1       0        0
nxge1   tx     2   1.48M  121.68M
nxge1   tx     3   2.45M  201.11M
nxge1   tx     4   1.47M  120.82M
nxge1   tx     5       0        0
nxge1   tx     6   1.97M  161.57M
nxge1   tx     7   4.59M  376.21M
nxge1   tx     8   2.43M  199.24M
nxge1   tx     9       0        0
nxge1   tx     10  3.23M  264.69M
nxge1   tx     11  1.88M  153.96M
```

▼ How to Obtain Statistics About Network Traffic on Lanes

1 Become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 Display statistics about network lanes.

```
# dlstat show-link [-r [F]|-t] [-i interval] [link]
```

`[-r|-t]` Displays either receive-side statistics only (`-r` option) or transmit-side statistics only (`-t` option). If you do not use these options, then statistics for both the receive-side and the transmit-side are displayed.

`-i` *interval* Specifies the time in seconds at which you want the displayed statistics to be refreshed. If you do not use this option, then static output is displayed.

link Indicates that you want to monitor the statistics of the specified datalink only. If you do not use this option, then information about all datalinks is displayed.

If ring grouping is supported and dedicated rings were configured, then hardware lane statistics are displayed. If no dedicated rings are configured, then software lane statistics are displayed.

Example 22–6 Displaying Receive-Side Statistics for a Lane

This example shows the following information:

- How packets are received on a hardware lane
- How packets are received on a software lane
- How packets are received on a software lane and fanned out to assigned CPUs

The following command shows receive-side statistics for the specific link. The information indicates ring usage. However, the data might also reflect the implementation of other resource allocations, such as bandwidth limits and priority processing.

```
# dlstat show-link -r nxge1
 LINK TYPE    ID INDEX  IPKTS   RBYTES  INTRS   POLLS   CH<10 CH10-50  CH>50
nxge1   rx local    --      0        0      0       0       0       0      0
nxge1   rx    hw     1      0        0      0       0       0       0      0
nxge1   rx    hw     2  1.73M    2.61G  1.33M 400.22K  67.03K   7.49K     38
nxge1   rx    hw     3      0        0      0       0       0       0      0
nxge1   rx    hw     4  8.44M   12.71G  4.35M   4.09M 383.28K  91.24K  2.09K
nxge1   rx    hw     5  5.68M    8.56G  3.72M   1.97M 203.68K  43.94K    854
nxge1   rx    hw     6  4.90M    7.38G  3.11M   1.80M 168.59K  42.34K    620
nxge1   rx    hw     7      0        0      0       0       0       0      0
```

The following command shows receive-side statistics for the specific link. In the output, the `ID` field indicates whether hardware rings are exclusively assigned or shared among clients. In the `ixgbe` card, Rx rings are shared if other clients such as VNICs are configured over the link as well. Thus, for this specific example, Rx rings are shared, as indicated by the `sw` value under the `ID` field.

```
# dlstat show-link -r ixgbe0
  LINK TYPE    ID INDEX   IPKTS  RBYTES    INTRS  POLLS  CH<10 CH10-50  CH>50
ixgbe0   rx local    --       0       0        0      0      0       0      0
```

```
ixgbe0   rx    sw    -- 794.28K   1.19G  794.28K      0      0       0      0
```

The following command shows usage of receive-side statistics for the specific link. In addition, with the use of the `-F` option in the command, the output also provides fanout information. Specifically, the fanout count is two (`0` and `1`). Network traffic that is received on the hardware lane that uses ring `0` is split and passed on across the two fanouts. Likewise, network traffic that is received on the hardware lane that uses ring `1` is also split and divided across the two fanouts.

```
# dlstat show-link -r -F nxge1
 LINK     ID   INDEX   FOUT   IPKTS
nxge1  local      --      0       0
nxge1     hw       0      0  382.47K
nxge1     hw       0      1        0
nxge1     hw       1      0  367.50K
nxge1     hw       1      1  433.24K
```

Example 22–7 Displaying Transmit-Side Statistics for a Lane

The following example shows statistics about outbound packets on a specific lane.

```
# dlstat show-link -t nxge1
 LINK   TYPE   ID  INDEX    OPKTS    OBYTES   BLKCNT   UBLKCNT
nxge1     tx   hw      0       32     1.44K        0         0
nxge1     tx   hw      1        0         0        0         0
nxge1     tx   hw      2    1.48M    97.95M        0         0
nxge1     tx   hw      3    2.45M   161.87M        0         0
nxge1     tx   hw      4    1.47M    97.25M        0         0
nxge1     tx   hw      5        0       276        0         0
nxge1     tx   hw      6    1.97M   130.25M        0         0
nxge1     tx   hw      7    4.59M   302.80M        0         0
nxge1     tx   hw      8    2.43M   302.80M        0         0
nxge1     tx   hw      9        0         0        0         0
nxge1     tx   hw      10   3.23M   213.05M        0         0
nxge1     tx   hw      11   1.88M   123.93M        0         0
```

Gathering Statistics About Network Traffic on Flows

Flow statistics help you evaluate packet traffic on any defined flows on the system. To obtain flow information, you use the `flowstat` command. For more information about this command, refer to the `flowstat(1M)` man page.

The most commonly used syntax of the `flowstat` command follows:

```
# flowstat [-r|-t] [-i interval] [-l link flow]
```

`[-r|-t]` Displays either receive-side statistics only (`-r` option) or transmit-side statistics only (`-t` option). If you do not use these options, then statistics for both the receive-side and the transmit-side are displayed.

`-i` *interval* — Specifies the time in seconds at which you want the displayed statistics to be refreshed. If you do not use this option, then static output is displayed.

link — Indicates that you want to monitor the statistics for all the flows on the specified datalink. If you do not use this option, then information about all the flows on all the datalinks is displayed.

flow — Indicates that you want to monitor the statistics of a specified flow only. If you do not use this option, then depending on whether you specified a link, all flow statistics are displayed.

How to Obtain Statistics on Flows

Before You Begin You can use the `flowstat` command only if flows exist in your network configuration. To configure flows, see Chapter 21, "Managing Network Resources."

1. **On the system where you previously configured flow control, become an administrator in the global zone.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **For a sampling of how to observe network traffic on flows, perform any of the following commands:**

   - Display statistics about incoming and outgoing packets on all flows.

     ```
     # flowstat
     ```

     This command provides a static display of traffic information on all configured flows.

   - Display basic network traffic statistics on all flows at a specified interval

     ```
     # flowstat -i interval
     ```

     The display of statistics is refreshed at the specified interval until you stop the output generation by pressing Control–C.

   - Display statistics about incoming packets on all flows that are configured over a specified datalink.

     ```
     # flowstat -r -l link
     ```

   - Display statistics about outgoing packets on a specified flow at a specified interval.

     ```
     # flowstat -t -i interval flow
     ```

Example 22–8 Displaying Traffic Statistics for All Flows at One-Second Intervals

This example shows information every second about incoming and outgoing traffic on all configured flows on the system.

```
# flowstat -i 1
FLOW     IPKTS   RBYTES  IERRS    OPKTS  OBYTES  OERRS
flow1  528.45K  787.39M      0  179.39K  11.85M      0
flow2  742.81K    1.10G      0        0       0      0
flow3        0        0      0        0       0      0
flow1   67.73K  101.02M      0   21.04K    1.39M     0
flow2        0        0      0        0       0      0
flow3        0        0      0        0       0      0
...
^C
```

Example 22–9 Displaying Transmit–Side Statistics for All Flows

```
# flowstat -t
 FLOW    OPKTS   OBYTES   OERRS
flow1   24.37M    1.61G       0
flow2        0        0       0
flow1        4      216       0
```

Example 22–10 Displaying Receive-Side Statistics for All Flows on a Specified Link

This example shows incoming traffic in hardware lanes in all the flows that were created over the `net0`, the datalink.

```
# flowstat -r -i 2 -l net0
    FLOW    IPKTS   RBYTES    IERRS
tcp-flow  183.11K  270.24M        0
udp-flow        0        0        0
tcp-flow  373.83K  551.52M        0
udp-flow        0        0        0
tcp-flow  372.35K  549.04M        0
udp-flow        0        0        0
tcp-flow  372.87K  549.61M        0
udp-flow        0        0        0
tcp-flow  371.57K  547.89M        0
udp-flow        0        0        0
tcp-flow  191.92K  282.95M        0
udp-flow  206.51K  310.70M        0
tcp-flow        0        0        0
udp-flow  222.75K  335.15M        0
tcp-flow        0        0        0
udp-flow  223.00K  335.52M        0
tcp-flow        0        0        0
udp-flow  160.22K  241.07M        0
tcp-flow        0        0        0
udp-flow  167.89K  252.61M        0
tcp-flow        0        0        0
```

```
udp-flow    9.52K   14.32M        0
^C
```

Setting Up Network Accounting

You can use the extended accounting facility to capture statistics about network traffic in a log file. In this manner, you can maintain records of traffic for tracking, provisioning, consolidation, and billing purposes. Later, you can refer to the log file to obtain historical information about network use over a period of time.

To configure the extended accounting facility, you use the `acctadm` command.

▼ How to Configure Extended Network Accounting

1 On the system with the interfaces whose network usage you want to track, become an administrator.

For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2 View the status of extended network accounting in the system.

```
# acctadm net
```

Four types of extended accounting can be enabled by the `acctadm` command:

- Process accounting
- Task accounting
- Flow accounting for IP Quality of Service (IPQoS)
- Network accounting for links and flows

Specifying `net` displays the status of network accounting. If `net` is not used, then the status of all four accounting types is displayed.

Note – Network accounting also applies to flows that are managed by the `flowadm` and `flowstat` commands as discussed in "Managing Resources on Flows" on page 389. Therefore, to set up accounting for these flows, use the `net` option with the `acctadm` command. Do *not* use the `flow` option that enables flow accounting and which applies to IPQoS configurations.

---

3 Enable extended accounting for network traffic.

```
# acctadm -e extended -f filename net
```

where *filename* includes the full path of the log file that will capture network traffic statistics. The log file can be created in any directory that you specify.

4 **Verify that extended network accounting has been activated.**

```
# acctadm net
```

Example 22–11 Configuring Extended Accounting for Network Traffic

This example shows how to capture and display historical information about network traffic on datalinks and any configured flows on the system.

First, view the status of all accounting types as follows:

```
# acctadm
            Task accounting: inactive
       Task accounting file: none
     Tracked task resources: none
   Untracked task resources: extended
          Process accounting: inactive
    Process accounting file: none
  Tracked process resources: none
Untracked process resources: extended,host
            Flow accounting: inactive
       Flow accounting file: none
     Tracked flow resources: none
   Untracked flow resources: extended
            Network accounting: inactive
       Network accounting file: none
     Tracked Network resources: none
   Untracked Network resources: extended
```

The output shows that network accounting is not active.

Next, enable extended network accounting.

```
# acctadm -e extended -f /var/log/net.log net
# acctadm net
            Net accounting: active
       Net accounting file: /var/log/net.log
     Tracked net resources: extended
   Untracked net resources: none
```

After you have enabled network accounting, you can use the `dlstat` and `flowstat` commands to extract information from the log file. The following procedure explains the steps.

▼ How to Obtain Historical Statistics on Network Traffic

Before You Begin You must enable extended accounting for the network before you can display historical data about the network. Further, to display historical data about traffic on flows, you must first configure flows in the system as explained in "Managing Resources on Flows" on page 389.

1. **On the system with the interfaces whose network usage you want to track, become an administrator.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

2. **To extract and display historical information about resource usage on datalinks, use the following command:**

```
# dlstat show-link -h [-a] -f filename [-d date] [-F format] [-s start-time] [-e end-time] [link]
```

`-h`	Displays a summary of historical information about resource usage by incoming and outgoing packets on datalinks.
`-a`	Displays resource usage on all datalinks, including those that have already been deleted after the data capture.
`-f` *filename*	Specifies the log file that was defined when network accounting was enabled with the `acctadm` command.
`-d`	Displays logged information for dates when information is available.
`-F` *format*	Displays the data in a specific format. Currently, `gnuplot` is the only supported format.
`-s` *start-time*, `-e` *end-time*	Display available logged information for a specified date and time range. Use the `MM/DD/YYY,hh:mm:ss` format. The `hour` (`hh`) must use the 24–hour clock notation. If you do not include the date, then data for the current date's time range is displayed.
link	Displays historical data for a specified datalink. If you do not use this option, then historical network data for all configured datalinks is displayed.

3. **To extract and display historical information about network traffic on configured flows, use the following command:**

```
# flowstat -h [-a] -f filename [-d date] [-F format] [-s start-time] [-e end-time] [flow]
```

`-h`	Displays a summary of historical information about resource usage by incoming and outgoing packets on datalinks.
`-a`	Displays resource usage on all datalinks, including those that have already been deleted after the data capture.
`-f` *filename*	Specifies the log file that was defined when network accounting was enabled with the `acctadm` command.
`-d`	Displays logged information for dates when information is available.
`-F` *format*	Displays the data in a specific format. Currently, `gnuplot` is the only supported format.

`-s` *start-time*,
`-e` *end-time*
: Display available logged information for a specified date and time range. Use the `MM/DD/YYY,hh:mm:ss` format. The `hour` (`hh`) must use the 24–hour clock notation. If you do not include the date, then data for the current date's time range is displayed.

link
: Displays historical data for a specified datalink. If you do not use this option, then historical network data for all configured datalinks is displayed.

flow
: Displays historical data for a specified flow. If you do not use this option, then historical network data for all configured flows is displayed.

Example 22–12 Displaying Historical Information About Resource Usage on Datalinks

The following example shows historical statistics about network traffic and its use of resources on a specified datalink.

```
# dlstat show-link -h -f /var/log/net.log
LINK      DURATION  IPACKETS RBYTES      OPACKETS OBYTES      BANDWIDTH
e1000g0   80        1031     546908      0        0           2.44 Kbps
```

Example 22–13 Displaying Historical Information About Resource Usage on Flows

The following examples show different ways of displaying historical statistics about network traffic on a flow and its use of resources.

Display historical statistics of resource usage by traffic on a flow:

```
# flowstat -h -f /var/log/net.log
FLOW      DURATION  IPACKETS RBYTES      OPACKETS OBYTES      BANDWIDTH
flowtcp   100       1031     546908      0        0            43.76Kbps
flowudp   0         0        0           0        0            0.00Mbps
```

Display historical statistics of resource usage by traffic on a flow over a given date and time range.

```
# flowstat -h -s 02/19/2008,10:39:06 -e 02/19/2008,10:40:06 \
-f /var/log/net.log flowtcp

FLOW      START       END         RBYTES   OBYTES     BANDWIDTH
flowtcp   10:39:06    10:39:26    1546     6539        3.23 Kbps
flowtcp   10:39:26    10:39:46    3586     9922        5.40 Kbps
flowtcp   10:39:46    10:40:06    240      216       182.40 bps
flowtcp   10:40:06    10:40:26    0        0           0.00 bps
```

Display historical statistics of resource usage by traffic on a flow over a given date and time range. Display the information by using the gnuplot format.

```
# flowstat -h -s 02/19/2008,10:39:06 -e 02/19/2008,10:40:06 \
-F gnuplot -f /var/log/net.log flowtcp
# Time tcp-flow
10:39:06 3.23
10:39:26 5.40
10:39:46 0.18
10:40:06 0.00
```

Glossary

3DES See Triple-DES.

AES Advanced Encryption Standard. A symmetric 128-bit block data encryption technique. The U.S. government adopted the Rijndael variant of the algorithm as its encryption standard in October 2000. AES replaces DES encryption as the government standard.

anycast address An IPv6 address that is assigned to a group of interfaces (typically belonging to different nodes). A packet that is sent to an anycast address is routed to the *nearest* interface having that address. The packet's route is in compliance with the routing protocol's measure of distance.

anycast group A group of interfaces with the same anycast IPv6 address. The Oracle Solaris implementation of IPv6 does not support the creation of anycast addresses and groups. However, Oracle Solaris IPv6 nodes can send traffic to anycast groups.

asymmetric key cryptography An encryption system in which the sender and receiver of a message use different keys to encrypt and decrypt the message. Asymmetric keys are used to establish a secure channel for symmetric key encryption. The Diffie-Hellman protocol is an example of an asymmetric key protocol. Contrast with symmetric key cryptography.

authentication header An extension header that provides authentication and integrity, without confidentiality, to IP datagrams.

autoconfiguration The process where a host automatically configures its IPv6 address from the site prefix and the local MAC address.

bidirectional tunnel A tunnel that can transmit datagrams in both directions.

Blowfish A symmetric block cipher algorithm that takes a variable-length key from 32 bits to 448 bits. Its author, Bruce Schneier, claims that Blowfish is optimized for applications where the key does not change often.

broadcast address IPv4 network addresses with the host portion of the address having all zeroes (10.50.0.0) or all one bits (10.50.255.255). A packet that is sent to a broadcast address from a machine on the local network is delivered to all machines on that network.

CA See certificate authority (CA).

certificate authority (CA) A trusted third-party organization or company that issues digital certificates used to create digital signatures and public-private key pairs. The CA guarantees the identity of the individual who is granted the unique certificate.

certificate revocation list (CRL) A list of public key certificates that have been revoked by a CA. CRLs are stored in the CRL database that is maintained through IKE.

class In IPQoS, a group of network flows that share similar characteristics. You define classes in the IPQoS configuration file.

classless inter-domain routing (CIDR) address An IPv4 address format that is not based on network classes (Class A, B, and C). CIDR addresses are 32 bits in length. They use the standard IPv4 dotted decimal notation format, with the addition of a network prefix. This prefix defines the network number and the network mask.

data address An IP address which can be used as the source or destination address for data. Data addresses are part of an IPMP group and can be used to send and receive traffic on any interface in the group. Moreover, the set of data addresses in an IPMP group can be used continuously provided that one interface in the group is functioning.

datagram See IP datagram.

`DEPRECATED` address An IP address that cannot be used as the source address for data in an IPMP group. Typically, IPMP test addresses are `DEPRECATED`. However, any address can be marked `DEPRECATED` to prevent the address from being used as a source address.

DES Data Encryption Standard. A symmetric-key encryption method developed in 1975 and standardized by ANSI in 1981 as ANSI X.3.92. DES uses a 56-bit key.

Diffie-Hellman protocol Also known as public key cryptography. An asymmetric cryptographic key agreement protocol that was developed by Diffie and Hellman in 1976. The protocol enables two users to exchange a secret key over an insecure medium without any prior secrets. Diffie-Hellman is used by the IKE protocol.

diffserv model Internet Engineering Task Force architectural standard for implementing differentiated services on IP networks. The major modules are classifier, meter, marker, scheduler, and dropper. IPQoS implements the classifier, meter, and marker modules. The diffserv model is described in RFC 2475, *An Architecture for Differentiated Services*.

digital signature A digital code that is attached to an electronically transmitted message that uniquely identifies the sender.

domain of interpretation (DOI) A DOI defines data formats, network traffic exchange types, and conventions for naming security-relevant information. Security policies, cryptographic algorithms, and cryptographic modes are examples of security-relevant information.

DS codepoint (DSCP) A 6-bit value that, when included in the DS field of an IP header, indicates how a packet must be forwarded.

DSA Digital Signature Algorithm. A public key algorithm with a variable key size from 512 to 4096 bits. The U.S. Government standard, DSS, goes up to 1024 bits. DSA relies on SHA-1 for input.

dual stack A TCP/IP protocol stack with both IPv4 and IPv6 at the network layer, with the rest of the stack being identical. When you enable IPv6 during an Oracle Solaris installation, the host receives the dual-stack version of TCP/IP.

dynamic packet filter See stateful packet filter.

dynamic reconfiguration (DR) A feature that allows you to reconfigure a system while the system is running, with little or no impact on ongoing operations. Not all Sun platforms from Oracle support DR. Some platforms might only support DR of certain types of hardware such as NICs.

encapsulating security payload (ESP) An extension header that provides integrity and confidentiality to datagrams. ESP is one of the five components of the IP Security Architecture (IPsec).

encapsulation The process of a header and payload being placed in the first packet, which is subsequently placed in the second packet's payload.

failure detection The process of detecting when an interface or the path from an interface to an Internet layer device no longer works. IP network multipathing (IPMP) includes two types of failure detection: link based (default) and probe based (optional).

filter A set of rules that define the characteristics of a class in the IPQoS configuration file. The IPQoS system selects for processing any traffic flows that conform to the filters in its IPQoS configuration file. See packet filter.

firewall Any device or software that isolates an organization's private network or intranet from the Internet, thus protecting it from external intrusions. A firewall can include packet filtering, proxy servers, and NAT (network address translation).

flow accounting In IPQoS, the process of accumulating and recording information about traffic flows. You establish flow accounting by defining parameters for the `flowacct` module in the IPQoS configuration file.

hash value A number that is generated from a string of text. Hash functions are used to ensure that transmitted messages have not been tampered with. MD5 and SHA-1 are examples of one-way hash functions.

header See IP header.

HMAC Keyed hashing method for message authentication. HMAC is a secret key authentication algorithm. HMAC is used with an iterative cryptographic hash function, such as MD5 or SHA-1, in combination with a secret shared key. The cryptographic strength of HMAC depends on the properties of the underlying hash function.

hop A measure that is used to identify the number of routers that separate two hosts. If three routers separate a source and destination, the hosts are four hops away from each other.

host A system that does not perform packet forwarding. Upon installation of Oracle Solaris, a system becomes a host by default, that is, the system cannot forward packets. A host typically has one physical interface, although it can have multiple interfaces.

ICMP Internet Control Message Protocol. Used to handle errors and exchange control messages.

ICMP echo request packet A packet sent to a machine on the Internet to solicit a response. Such packets are commonly known as “ping” packets.

IKE Internet Key Exchange. IKE automates the provision of authenticated keying material for IPsec security associations (SAs).

Internet Protocol (IP) The method or protocol by which data is sent from one computer to another on the Internet.

IP See Internet Protocol (IP), IPv4, IPv6.

IP datagram A packet of information that is carried over IP. An IP datagram contains a header and data. The header includes the addresses of the source and the destination of the datagram. Other fields in the header help identify and recombine the data with accompanying datagrams at the destination.

IP header Twenty bytes of data that uniquely identify an Internet packet. The header includes source and destination addresses for the packet. An option exists within the header to allow further bytes to be added.

IP in IP encapsulation The mechanism for tunneling IP packets within IP packets.

IP link A communication facility or medium over which nodes can communicate at the link layer. The link layer is the layer immediately below IPv4/IPv6. Examples include Ethernets (simple or bridged) or ATM networks. One or more IPv4 subnet numbers or prefixes are assigned to an IP link. A subnet number or prefix cannot be assigned to more than one IP link. In ATM LANE, an IP link is a single emulated LAN. When you use ARP, the scope of the ARP protocol is a single IP link.

IP stack TCP/IP is frequently referred to as a "stack." This refers to the layers (TCP, IP, and sometimes others) through which all data passes at both client and server ends of a data exchange.

IPMP group IP multipathing group, composed of a set of network interfaces with a set of data addresses that are treated as interchangeable by the system to improve network availability and utilization. The IPMP group, including all its underlying IP interfaces and data addresses, is represented by an IPMP interface.

IPQoS A software feature that provides an implementation of the diffserv model standard, plus flow accounting and 802.1 D marking for virtual LANs. Using IPQoS, you can provide different levels of network services to customers and applications, as defined in the IPQoS configuration file.

IPsec IP security. The security architecture that provides protection for IP datagrams.

IPv4 Internet Protocol, version 4. IPv4 is sometimes referred to as IP. This version supports a 32-bit address space.

IPv6 Internet Protocol, version 6. IPv6 supports a 128-bit address space.

key management The way in which you manage security associations (SAs).

keystore name The name that an administrator gives to the storage area, or keystore, on a network interface card (NIC). The keystore name is also called the token or the token ID.

link layer The layer immediately below IPv4/IPv6.

link-local address In IPv6, a designation that is used for addressing on a single link for purposes such as automatic address configuration. By default, the link-local address is created from the system's MAC address.

load spreading The process of distributing inbound or outbound traffic over a set of interfaces. With load spreading, higher throughput is achieved. Load spreading occurs only when the network traffic is flowing to multiple destinations that use multiple connections. Two types of load spreading exists: inbound load spreading for inbound traffic and outbound load spreading for outbound traffic.

local-use address A unicast address that has only local routeability scope (within the subnet or within a subscriber network). This address also can have a local or global uniqueness scope.

marker 1. A module in the diffserv architecture and IPQoS that marks the DS field of an IP packet with a value that indicates how the packet is to be forwarded. In the IPQoS implementation, the marker module is `dscpmk`.

2. A module in the IPQoS implementation that marks the virtual LAN tag of an Ethernet datagram with a user priority value. The user priority value indicates how datagrams are to be forwarded on a network with VLAN devices. This module is called `dlcosmk`.

MD5 An iterative cryptographic hash function that is used for message authentication, including digital signatures. The function was developed in 1991 by Rivest.

message authentication code (MAC) MAC provides assurance of data integrity and authenticates data origin. MAC does not protect against eavesdropping.

meter A module in the diffserv architecture that measures the rate of traffic flow for a particular class. The IPQoS implementation includes two meters, `tokenmt` and `tswtclmt`.

minimal encapsulation An optional form of IPv4 in IPv4 tunneling that can be supported by home agents, foreign agents, and mobile nodes. Minimal encapsulation has 8 or 12 bytes less of overhead than does IP in IP encapsulation.

MTU Maximum Transmission Unit. The size, given in octets, that can be transmitted over a link. For example, the MTU of an Ethernet is 1500 octets.

multicast address An IPv6 address that identifies a group of interfaces in a particular way. A packet that is sent to a multicast address is delivered to all of the interfaces in the group. The IPv6 multicast address has similar functionality to the IPv4 broadcast address.

multihomed host A system that has more than one physical interface and that does not perform packet forwarding. A multihomed host can run routing protocols.

NAT See network address translation.

neighbor advertisement A response to a neighbor solicitation message or the process of a node sending unsolicited neighbor advertisements to announce a link-layer address change.

neighbor discovery An IP mechanism that enables hosts to locate other hosts that reside on an attached link.

neighbor solicitation A solicitation that is sent by a node to determine the link-layer address of a neighbor. A neighbor solicitation also verifies that a neighbor is still reachable by a cached link-layer address.

network address translation NAT. The translation of an IP address used within one network to a different IP address known within another network. Used to limit the number of global IP addresses that are needed.

network interface card (NIC) Network adapter card that is an interface to a network. Some NICs can have multiple physical interfaces, such as the `igb` card.

node In IPv6, any system that is IPv6-enabled, whether a host or a router.

outcome The action to take as a result of metering traffic. The IPQoS meters have three outcomes, red, yellow, and green, which you define in the IPQoS configuration file.

packet A group of information that is transmitted as a unit over communications lines. Contains an IP header plus a payload.

packet filter A firewall function that can be configured to allow or disallow specified packets through a firewall.

packet header See IP header.

payload The data that is carried in a packet. The payload does not include the header information that is required to get the packet to its destination.

per-hop behavior (PHB) A priority that is assigned to a traffic class. The PHB indicates the precedence which flows of that class have in relation to other traffic classes.

perfect forward secrecy (PFS) In PFS, the key that is used to protect transmission of data is not used to derive additional keys. Also, the source of the key that is used to protect data transmission is never used to derive additional keys.

PFS applies to authenticated key exchange only. See also Diffie-Hellman protocol.

physical interface A system's attachment to a link. This attachment is often implemented as a device driver plus a network interface card (NIC). Some NICs can have multiple points of attachment, for example, `igb`.

PKI Public Key Infrastructure. A system of digital certificates, Certificate Authorities, and other registration authorities that verify and authenticate the validity of each party involved in an Internet transaction.

private address An IP address that is not routeable through the Internet. Private addresses can used by internal networks on hosts that do not require Internet connectivity. These addresses are defined in Address Allocation for Private Internets (`http://www.ietf.org/rfc/rfc1918.txt?number=1918`) and often referred to as "1918" addresses.

protocol stack See IP stack.

proxy server A server that sits between a client application, such as a Web browser, and another server. Used to filter requests – to prevent access to certain web sites, for instance.

public key cryptography A cryptographic system that uses two different keys. The public key is known to everyone. The private key is known only to the recipient of the message. IKE provides public keys for IPsec.

redirect In a router, to inform a host of a better first-hop node to reach a particular destination.

repair detection The process of detecting when a NIC or the path from the NIC to some Layer 3 device starts operating correctly after a failure.

replay attack In IPsec, an attack in which a packet is captured by an intruder. The stored packet then replaces or repeats the original at a later time. To protect against such attacks, a packet can contain a field that increments during the lifetime of the secret key that is protecting the packet.

reverse tunnel A tunnel that starts at the mobile node's care-of address and terminates at the home agent.

router A system that usually has more than one interface, runs routing protocols, and forwards packets. You can configure a system with only one interface as a router if the system is the endpoint of a PPP link.

router advertisement The process of routers advertising their presence together with various link and Internet parameters, either periodically or in response to a router solicitation message.

router discovery The process of hosts locating routers that reside on an attached link.

router solicitation The process of hosts requesting routers to generate router advertisements immediately, rather than at their next scheduled time.

RSA A method for obtaining digital signatures and public key cryptosystems. The method was first described in 1978 by its developers, Rivest, Shamir, and Adleman.

SA See security association (SA).

SADB Security Associations Database. A table that specifies cryptographic keys and cryptographic algorithms. The keys and algorithms are used in the secure transmission of data.

SCTP See streams control transport protocol.

security association (SA) An association that specifies security properties from one host to a second host.

security parameter index (SPI) An integer that specifies the row in the security associations database (SADB) that a receiver should use to decrypt a received packet.

security policy database (SPD) Database that specifies the level of protection to apply to a packet. The SPD filters IP traffic to determine whether a packet should be discarded, should be passed in the clear, or should be protected with IPsec.

selector The element that specifically defines the criteria to be applied to packets of a particular class in order to select that traffic from the network stream. You define selectors in the filter clause of the IPQoS configuration file.

SHA-1 Secure Hashing Algorithm. The algorithm operates on any input length less than 2^{64} to produce a message digest. The SHA-1 algorithm is input to DSA.

site-local-use address A designation that is used for addressing on a single site.

smurf attack To use ICMP echo request packets directed to an IP broadcast address or multiple broadcast addresses from remote locations to create severe network congestion or outages.

sniff To eavesdrop on computer networks – frequently used as part of automated programs to sift information, such as clear-text passwords, off the wire.

SPD See security policy database (SPD).

SPI See security parameter index (SPI).

spoof To gain unauthorized access to a computer by sending a message to it with an IP address indicating that the message is coming from a trusted host. To engage in IP spoofing, a hacker must first use a variety of techniques to find an IP address of a trusted host and then modify the packet headers so that it appears that the packets are coming from that host.

stack See IP stack.

standby A physical interface that is not used to carry data traffic unless some other physical interface has failed.

stateful packet filter A packet filter that can monitor the state of active connections and use the information obtained to determine which network packets to allow through the firewall. By tracking and matching requests and replies, a stateful packet filter can screen for a reply that doesn't match a request.

stateless autoconfiguration The process of a host generating its own IPv6 addresses by combining its MAC address and an IPv6 prefix that is advertised by a local IPv6 router.

stream control transport protocol A transport layer protocol that provides connection-oriented communications in a manner similar to TCP. Additionally, SCTP supports multihoming, in which one of the endpoints of the connection can have more than one IP address.

symmetric key cryptography An encryption system in which the sender and receiver of a message share a single, common key. This common key is used to encrypt and decrypt the message. Symmetric keys are used to encrypt the bulk of data transmission in IPsec. DES is one example of a symmetric key system.

TCP/IP TCP/IP (Transmission Control Protocol/Internet Protocol) is the basic communication language or protocol of the Internet. It can also be used as a communications protocol in a private network (either an intranet or an extranet).

test address An IP address in an IPMP group which must be used as the source or destination address for probes, and must not be used as a source or destination address for data traffic.

Triple-DES Triple-Data Encryption Standard. A symmetric-key encryption method. Triple-DES requires a key length of 168 bits. Triple-DES is also written as 3DES.

tunnel The path that is followed by a datagram while it is encapsulated. See encapsulation.

unicast address An IPv6 address that identifies a single interface of an IPv6-enabled node. The parts of the unicast address are site prefix, subnet ID, and interface ID.

user-priority A 3-bit value that implements class-of-service marks, which define how Ethernet datagrams are forwarded on a network of VLAN devices.

virtual LAN (VLAN) device Network interfaces that provide traffic forwarding at the Ethernet (datalink) level of the IP protocol stack.

virtual network A combination of software and hardware network resources and functionality that are administered together as a single software entity. An *internal* virtual network consolidates network resources onto a single system, sometimes referred to as a "network in a box."

virtual network interface (VNIC) A pseudo-interface that provides virtual network connectivity whether or not it is configured on a physical network interface. Containers such as exclusive IP zones or xVM domains are configured above VNICs to form a virtual network.

virtual private network (VPN) A single, secure, logical network that uses tunnels across a public network such as the Internet.

Index

N

P

R

S

T