Configuring an Oracle® Solaris 11.2 System as a Router or a Load Balancer

Part No: E37517-02
September 2014

ORACLE

Contents

Using This Documentation

- **Overview** – Describes how to configure Oracle Solaris 11.2 as an IPv4 or an IPv6 router. Provides an overview and configuration instructions for Virtual Router Redundancy Protocol (VRRP) and integrated load balancer (ILB).
- **Audience** – System administrators.
- **Required knowledge** – Basic and some advanced network administration skills.

Product Documentation Library

Late-breaking information and known issues for this product are included in the documentation library at http://www.oracle.com/pls/topic/lookup?ctx=E36784.

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.

Feedback

Provide feedback about this documentation at http://www.oracle.com/goto/docfeedback.

♦ ♦ ♦ CHAPTER 1

Introduction to Routers and Load Balancers

This chapter describes how routers and load balancers are used in Oracle Solaris to connect computer networks and to distribute workloads. Routers handle routing activity by using protocols, such as the Routing Information Protocol (RIP), the next generation RIP (RIPng), the Internet Control Message Protocol Router Discovery (RDISC), the Open Shortest Path First (OSPF), the Border Gateway Protocol (BGP), Intermediate System to Intermediate System (IS-IS), and the Virtual Router Redundancy Protocol (VRRP).

Load balancers distribute network traffic across a number of servers. The distribution of a network's workload helps to achieve optimal resource sharing and to increase throughput and availability.

This chapter contains the following topics:

Router Overview

A router is a device that is used in a computer network to connect computers and transfer packets of data among computers in the network. A router can have two or more connections from different networks. The router reads the address information from the incoming data packets to determine their destination. Then, packets are forwarded to the next network by using the information in the router's routing table. This traffic directing process of routers is repeated until the data packets reach the destination node.

Routing Protocols

Routing protocols handle routing activity on a system. Routers exchange routing information with other hosts to maintain known routes to remote networks. Both routers and hosts can run routing protocols. The routing protocols on the host communicate with routing daemons on

other routers and hosts. These protocols assist the host in determining where to forward packets. When network interfaces are enabled, the system automatically communicates with the routing daemons. These daemons monitor routers on the network and advertise the router's addresses to the hosts on the local network. Some routing protocols, although not all, also maintain statistics that you can use to measure routing performance. Similar to packet forwarding, you must explicitly configure routing on an Oracle Solaris system.

RIP and RDISC are standard TCP/IP protocols. The following table describes the supported routing protocols in Oracle Solaris.

TABLE 1-1 Oracle Solaris Routing Protocols

Protocol	Associated Daemon	Description	For Instructions
RIP	`in.routed`	Interior Gateway Protocol (IGP) that routes IPv4 packets and maintains a routing table	"Configuring an IPv4 Router" on page 17
RDISC	`in.routed`	Enables hosts to discover the presence of a router on the network	"Enabling Routing for Single-Interface Systems" in "Configuring and Administering Network Components in Oracle Solaris 11.2 "
RIPng	`in.ripngd`	IGP that routes IPv6 packets and maintains a routing table	"How to Configure an IPv6-Enabled Router" on page 23
Neighbor Discovery Protocol (NDP)	`in.ndpd`	Advertises the presence of an IPv6 router and discovers the presence of IPv6 hosts on a network	"How to Configure a System For IPv6" in "Configuring and Administering Network Components in Oracle Solaris 11.2 "

For more information about routing tables and types in Oracle Solaris, see "Routing Tables and Routing Types" in "Configuring and Administering Network Components in Oracle Solaris 11.2 ".

Routing Information Protocol

Routing Information Protocol (RIP) is a distance-vector routing protocol. RIP uses a hop counter as its routing metric. It is implemented by the routing daemon `in.routed`. The daemon automatically starts when the system is booted. When run on a router with the `-s` option specified, the `in.routed` daemon fills the kernel routing table with a route to every reachable network and advertises reachability through all network interfaces. When run on a host with the `-q` option specified, the `in.routed` daemon extracts routing information but does not advertise reachability.

On hosts, routing information can be extracted in the following two ways:

- By *not* specifying the flag (capital `S` or space-saving mode). The `in.routed` daemon builds a full routing table exactly as it does on a router.

- By specifying the flag. The `in.routed` daemon creates a minimal kernel table containing a single default route for each available router.

ICMP Router Discovery Protocol

Hosts use the Router Discovery (RDISC) protocol to obtain routing information from routers. When hosts run RDISC, routers must also run another protocol, such as RIP, to exchange router information.

RDISC is implemented by the daemon `in.routed`, which must run on both routers and hosts. On hosts, `in.routed` uses RDISC to discover default routes from routers that advertise the address through RDISC. On routers, `in.routed` uses RDISC to advertise default routes to hosts on directly-connected networks. See the `in.routed`(1M) man page and the `gateways`(4) man page for more information.

Quagga Routing Protocol Suite

Quagga is a routing software suite that enables the implementation of RIP, RIPng, Open Shortest Path First (OSPF), Intermediate System to Intermediate System (IS-IS), and Border Gateway Protocol (BGP) protocols for UNIX platforms including Oracle Solaris.

RIPng offers an extension of RIP for support of IPv6, including various enhancements for IPv6. The functions of RIPng are similar to those of RIP.

OSPF is a router protocol which is used to distribute routing information within a larger autonomous system network. The latest version of OSPF, OSPFv3, adds support for IPv6.

IS-IS is a link state dynamic routing protocol which is used to distribute routing information within a large service provider network.

BGP uses a prefixed set of IP networks to make routing decisions based on the path and rules among large autonomous system networks.

The following table lists the Open Source Quagga routing protocols that are supported in Oracle Solaris.

TABLE 1-2 Quagga Routing Protocol Suite

Protocol	Associated Daemon	Description
RIP	`ripd`	IPv4 distance vectoring IGP that routes IPv4 packets and advertises its routing table to neighbors

Protocol	Associated Daemon	Description
RIPng	`ripngd`	IPv6 distance vectoring IGP that routes IPv6 packets and maintains a routing table
OSPF	`ospfd`	IPv4 link state IGP for packet routing and high availability networking
BGP	`bgpd`	IPv4 and IPv6 Exterior Gateway Protocol (EGP) for routing across administrative domains
IS-IS	`isisd`	IPv4 and IPv6 link state IGP for routing within an administrative domain or network

For more information about the Quagga protocols, go to the Quagga Routing Suite web site at http://www.nongnu.org/quagga/index.html.

Virtual Router Redundancy Protocol

VRRP provides high availability of IP addresses, such as those that used for routers and load balancers. VRRP is an Internet standard protocol specified in RFC 5798, Virtual Router Redundancy Protocol Version 3 for IPv4 and IPv6. Oracle Solaris provides an administrative tool that configures and manages the VRRP service.

In addition to the existing standard Layer 2 VRRP, Oracle Solaris 11.2 provides a proprietary Layer 3 VRRP to support VRRP over IPMP and InfiniBand interfaces and enhanced support for VRRP in zones.

For information about using VRRP and configuring VRRP routers, see Chapter 3, “Using Virtual Router Redundancy Protocol” and Chapter 4, “Configuring and Administering Virtual Router Redundancy Protocol”.

VRRP Router Overview

A Virtual Router Redundancy Protocol router is a single router image that is created by the operation of one or more routers that use the VRRP. VRRP runs on each VRRP router and manages the state of the router. A host can have multiple VRRP routers configured, where each VRRP router belongs to a different virtual router.

The Layer 2 VRRP router uses the standard VRRP protocol and requires a unique virtual router MAC address. The virtual IP addresses always resolves to the same virtual MAC address. You need to create a VRRP VNIC to get the unique virtual router MAC address. The proprietary Layer 3 VRRP feature in Oracle Solaris completely removes the need to configure unique VRRP virtual MAC addresses for VRRP routers, and thereby provides support for VRRP over IPMP and InfiniBand interfaces.

For information about using VRRP and configuring VRRP routers, see Chapter 3, “Using Virtual Router Redundancy Protocol” and Chapter 4, “Configuring and Administering Virtual Router Redundancy Protocol”.

Integrated Load Balancer Overview

In Oracle Solaris, Integrated Load Balancer (ILB) provides Layer 3 and Layer 4 load-balancing capabilities. ILB operates at the network (IP) and transport (TCP/UDP) layers for the Oracle Solaris operating system installed on SPARC-based and x86-based systems. ILB can be used to improve reliability and scalability, and to minimize the response time of network services.

ILB intercepts incoming requests from clients, decides which back-end server should handle the requests based on load-balancing rules, and then forwards the requests to the selected server. ILB can also be used as a router for the back-end server. ILB performs optional health checks and provides the data for the load-balancing algorithms to verify whether the selected server can handle the incoming requests.

Features of ILB

The key features of ILB include the following:

- Supports stateless Direct Server Return (DSR) and Network Address Translation (NAT) modes of operation for IPv4 and IPv6.

  For information about DSR and NAT modes of operation, see “ILB Operation Modes” on page 50.
- Assists traffic and load distribution and server selection by using a set of algorithms for the two modes of operation.
- Enables ILB administration through a command-line interface (CLI).

  For information about configuring ILB by using CLI, see “Configuring ILB by Using the Command-Line Interface” on page 58.
- Provides server monitoring capabilities through health checks.

  For information about server monitoring capabilities, see “Monitoring Health Checks in ILB” on page 64.

The following table lists and describes the features of ILB that are available for different modes of operation.

TABLE 1-3 ILB Features

Features	Description	Mode of Operation
Enables clients to ping virtual IP (VIP) addresses	ILB responds to ICMP echo requests from clients to VIP addresses.	Both DSR and NAT modes
Enables you to add and remove servers from a server group without interrupting service	ILB dynamically adds or removes servers from server group.	NAT mode
Enables you to configure session persistence ("stickiness")	ILB enables you to configure session persistence to your applications to send the connections or packets from a client to the same back-end server. ILB enables you to configure session persistence (that is, source address persistence) for a virtual service by using the `-p` option and specifying the `pmask` option in the `ilbadm create-rule` command. For more information, see "Creating an ILB Rule" on page 67.	Both DSR and NAT modes
Enables you to perform connection draining	ILB prevents new connections from being sent to a server that is disabled. This feature is useful for shutting down a server without disrupting the active connections or sessions. The existing connections to the server continue to function. After all the connections to that server are terminated, the server can be shut down for maintenance. After the server is ready to handle requests, it is enabled so that the load balancer can forward new connections to it.	NAT mode
Enables load-balancing of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) ports	ILB balances the load on all ports on a given IP address across different sets of servers without requiring you to set up explicit rules for each port.	Both DSR and NAT modes
Enables you to specify independent ports for virtual services within the same server group	ILB enables you to specify different destination ports for different servers in the same server group.	NAT mode
Enables you to load-balance a simple port range	ILB balances loads on a range of ports on the VIP to a given server group. For convenience, you can conserve IP addresses by load-balancing different port ranges on the same VIP to different sets of back-end servers. Also, when session persistence is enabled for NAT mode, ILB sends requests from the same client IP address for different ports in the range to the same back-end server.	Both DSR an NAT modes
Enables port range shifting and collapsing	Port range shifting and collapsing depend on the port range of a server in a load-balancing rule. If the port range of a server is different from the VIP port range, port shifting is automatically implemented. If the server port range is a single port, then port collapsing is implemented.	NAT mode

For information about the ILB components, operating modes, algorithms, and how ILB works, see Chapter 5, "Overview of an Integrated Load Balancer". For more information about

configuring and managing ILB, see Chapter 6, “Configuring and Managing the Integrated Load Balancer” and Chapter 7, “Configuring ILB for High Availability”.

Why Use VRRP Routers and Load Balancers?

When you set up a network such as a local area network (LAN), providing a high-availability service is very important. High-availability is a state where a redundant system takes over at the time of failure to ensure business continuity. High-availability is also relevant when an excessive workload is off-loaded to a redundant system. High-availability can become significant in situations such as planned or unplanned downtime, load-balancing, and disaster recovery.

Within a network domain, high availability can be implemented at various levels, such as link, IP, and routers. Load balancers and routers play a significant role in providing a high-availability service. In Oracle Solaris, VRRP routers and ILB are the network-level failover and load-sharing mechanisms that provide high availability.

For more information about working with and configuring VRRP routers, see Chapter 3, “Using Virtual Router Redundancy Protocol”. For more information about working with and configuring ILB, see Chapter 5, “Overview of an Integrated Load Balancer” and Chapter 6, “Configuring and Managing the Integrated Load Balancer”.

♦ ♦ ♦ CHAPTER 2

Configuring a System as a Router

A router provides the interface between two or more networks. You must assign a unique name and IP address to each of the router's physical network interfaces. Each router has a host name and an IP address that are associated with its primary network interface, in addition to a minimum of one more unique name and IP address for each additional network interface. This chapter describes how to configure your Oracle Solaris system as an IPv4 router or an IPv6 router.

This chapter contains the following topics:

For information about configuring routing for an Oracle Solaris host on a network, see "Enabling Routing for Single-Interface Systems" in "Configuring and Administering Network Components in Oracle Solaris 11.2 ".

For reference information about the routing protocols, see "Routing Protocols" on page 9 and "About IPv6 Routing" in "Configuring and Administering Network Components in Oracle Solaris 11.2 ".

Configuring an IPv4 Router

You can use the following procedure to configure a system with only one physical interface (by default, a host) as a router. You might configure a single-interface system as a router if the system serves as one endpoint on a PPP link, as explained in "Planning a Dial-up PPP Link" in "Managing Serial Networks Using UUCP and PPP in Oracle Solaris 11.2 ".

▼ How to Configure an IPv4 Router

The following procedure assumes that you are configuring interfaces for the router after installing the router.

Before You Begin After the router is physically installed on the network, configure the router to operate in local files mode. This configuration ensures that routers boot even if the network configuration server is down.

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in "Securing Users and Processes in Oracle Solaris 11.2 ".

2. **Configure the IP interfaces for the NICs on the system.**

   ```
   # ipadm create-ip IP-interface
   ```

3. **Configure the IP interface with a valid IP address by choosing one of the following commands:**

   - **To configure a static address, type the following command:**

     ```
     # ipadm create-addr -a address [interface | addr-obj]
     ```

   - **To configure a nonstatic address, type the following command:**

     ```
     # ipadm create-addr -T address-type [interface | addr-obj]
     ```

   For detailed instruction about how to configure IP interfaces, see Chapter 3, "Configuring and Administering IP Interfaces and Addresses in Oracle Solaris," in "Configuring and Administering Network Components in Oracle Solaris 11.2 ".

   Make sure that each IP interface is configured with the IP address of the network for which the system must route packets. Therefore, if the system serves the `192.168.5.0` and `10.0.5.0` networks, then one NIC must be configured for each network.

Caution - Make sure you are thoroughly knowledgeable about DHCP administration before configuring an IPv4 router to use DHCP.

4. **Add the host name and IP address of each interface to the `/etc/inet/hosts` file.**

   For example, assume that the names you assigned for the two interfaces of the router are `krakatoa` and `krakatoa-1`, respectively. The entries in the `/etc/inet/hosts` file are as follows:

   ```
   192.168.5.1     krakatoa        #interface for network 192.168.5.0
   10.0.5.1        krakatoa-1      #interface for network 10.0.5.0
   ```

5. **Perform the procedure "How to Configure a System for Local Files Mode" in "Configuring and Administering Network Components in Oracle Solaris 11.2 " to configure this router to run in local files mode.**

6. **If the router is connected to any subnetted network, add the network number and the netmask to the `/etc/inet/netmasks` file.**

   For example, for IPv4 address notation, such as `192.168.5.0`, type the following:

   ```
   192.168.5.0    255.255.255.0
   ```

7. **Enable IPv4 packet forwarding on the router.**

   ```
   # ipadm set-prop -p forwarding=on ipv4
   ```

8. **(Optional) Start a routing protocol.**

   Use one of the following commands:

   ```
   # routeadm -e ipv4-routing -u
   ```

   where -e option enables IPv4 routing and -u option applies the current configuration to the running system.

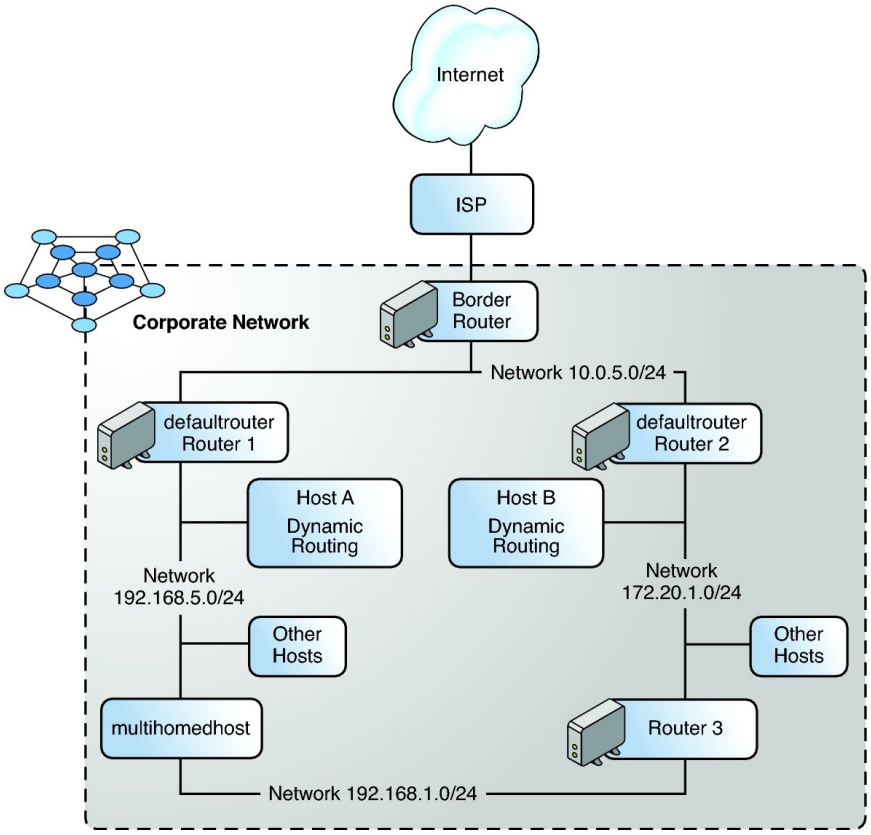
   ```
   # svcadm enable route:default
   ```

   When you start a routing protocol, the routing daemon `/usr/sbin/in.routed` automatically updates the routing table, a process that is known as *dynamic routing*. For more information about the types of routing, see "Routing Tables and Routing Types" in "Configuring and Administering Network Components in Oracle Solaris 11.2 ". For information about the `routeadm` command, see the `routeadm`(1M) man page and for more information about the `ipadm` command, see the `ipadm`(1M) man page.

   The Service Management Facility (SMF) Fault Management Resource Identifier (FMRI) associated with the `in.routed` daemon is `svc:/network/routing/route`.

Example 2-1 Configuring a System as a Router

This example is based on the following figure.



Router 2 contains two wired network connections, one connection to network `172.20.1.0` and one to network `10.0.5.0`. The example shows how to configure a system as a router (Router 2) of the `172.20.1.0` network. The example also assumes that Router 2 has been configured to operate in the local files mode as described in "How to Configure a System for Local Files Mode" in "Configuring and Administering Network Components in Oracle Solaris 11.2 ".

1. Determine the status of the system's interfaces.

```
# dladm show-link
LINK      CLASS     MTU     STATE   BRIDGE   OVER
net0      phys      1500    up      --       --
net1      phys      1500    up      --       --
net2      phys      1500    up      --       --
# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           10.0.0.1/8
net0/v4           static   ok           172.20.1.10/24
```

2. Only `net0` has been configured with an IP address. To make Router 2 the default router, physically connect the `net1` interface to the `10.0.5.0` network.

```
# ipadm create-ip net1
# ipadm create-addr -a 10.0.5.10/24 net1
# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           192.168.0.1/8
net0/v4           static   ok           172.20.1.10/24
net1/v4           static   ok           10.0.5.10/24
```

3. Update the following network databases with information about the newly configured interface and the network to which it is connected.

```
# pfedit /etc/inet/hosts
192.168.0.1       localhost
172.20.1.10        router2        #interface for network 172.20.1
10.0.5.10           router2-out    #interface for network 10.0.5
# pfedit /etc/inet/netmasks
172.20.1.0    255.255.255.0
10.0.5.0      255.255.255.0
```

4. Enable packet forwarding as well as the `in.routed` routing daemon.

```
# ipadm set-prop -p forwarding=on ipv4
# svcadm enable route:default
```

   Now, IPv4 packet forwarding and dynamic routing through RIP are enabled on Router 2. However, to complete the default router configuration for the network `172.20.1.0` you must do the following:

   - Modify each host on the `172.20.1.0` network so that the host gets its routing information from the new default router. For more information, refer to "Creating Persistent (Static) Routes" in "Configuring and Administering Network Components in Oracle Solaris 11.2 ".
   - Define a static route to the border router in the routing table of Router 2. For more details, refer to "Routing Tables and Routing Types" in "Configuring and Administering Network Components in Oracle Solaris 11.2 ". For more information about the `ipadm` command, see the `ipadm`(1M) man page.

Configuring an IPv6 Router

This section describes the how to configure an IPv6 router.

`in.ripngd` Daemon, for IPv6 Routing

The `in.ripngd` daemon implements the Routing Information Protocol next-generation for IPv6 routers (RIPng). RIPng defines the IPv6 equivalent of RIP. When you configure an IPv6 router with the `routeadm` command and turn on IPv6 routing, the `in.ripngd` daemon implements RIPng on the router. For information on the supported options of RIPng, see `in.ripngd`(1M).

Router Advertisement, Prefixes, and Messages

On multicast-capable links and point-to-point links, each router periodically sends to the multicast group a router advertisement packet that announces its availability. A host receives router advertisements from all routers, building a list of default routers. Routers generate router advertisements frequently enough so that hosts learn of their presence within a few minutes. However, routers do not advertise frequently enough to rely on an absence of advertisements to detect router failure. A separate detection algorithm that determines neighbor unreachability provides failure detection.

Router advertisements contain a list of subnet prefixes that is used to determine if a host is on the same link (on-link) as the router. The list of prefixes is also used for autonomous address configuration. Flags that are associated with the prefixes specify the intended uses of a particular prefix. Hosts use the advertised on-link prefixes to build and maintain a list that is used to decide when a packet's destination is on-link or beyond a router. A destination can be on-link even though the destination is not covered by any advertised on-link prefix. In such instances, a router can send a redirect. The redirect informs the sender that the destination is a neighbor.

Router advertisements, and per-prefix flags, enable routers to inform hosts how to perform stateless address autoconfiguration.

Router advertisement messages also contain Internet parameters, such as the hop limit, that hosts should use in outgoing packets. Optionally, router advertisement messages also contain link parameters, such as the link MTU. This feature enables the centralized administration of critical parameters. The parameters can be set on routers and automatically propagated to all hosts that are attached.

Nodes accomplish address resolution by sending to the multicast group a neighbor solicitation that asks the target node to return its link-layer address. Multicast neighbor solicitation messages are sent to the solicited-node multicast address of the target address. The target returns its link-layer address in a unicast neighbor advertisement message. A single request-response pair of packets is sufficient for both the initiator and the target to resolve each other's link-layer addresses. The initiator includes its link-layer address in the neighbor solicitation.

How to Configure an IPv6-Enabled Router

The following procedure assumes that you have already configured the system for IPv6. For the procedures, refer to Chapter 3, “Configuring and Administering IP Interfaces and Addresses in Oracle Solaris,” in “Configuring and Administering Network Components in Oracle Solaris 11.2 ”.

1. **Become an administrator.**

   For more information, see “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2 ”.

2. **Configure IPv6 packet forwarding on all interfaces of the router.**

   ```
   # ipadm set-prop -p forwarding=on ipv6
   ```

3. **Start the routing daemon.**

   The `in.ripngd` daemon handles IPv6 routing. Enable IPv6 routing by using either of the following commands:

   - Use the `routeadm` command:

     ```
     # routeadm -e ipv6-routing -u
     ```

     where `-e` option enables IPv4 routing and `-u` option applies the current configuration to the running system.
   - Use the appropriate SMF command:

     ```
     # svcadm enable ripng:default
     ```

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

4. **Create the `/etc/inet/ndpd.conf` file.**

   Specify the site prefix to be advertised by the router and other configuration information in the `/etc/inet/ndpd.conf` file. This file is read by the `in.ndpd` daemon, which implements the IPv6 Neighbor Discovery protocol.

   For a list of variables and allowable values, refer to the `ndpd.conf`(4) man page.

5. **Type the following text into the `/etc/inet/ndpd.conf` file:**

   ```
   ifdefault AdvSendAdvertisements true
   prefixdefault AdvOnLinkFlag on AdvAutonomousFlag on
   ```

   This text tells the `in.ndpd` daemon to send out router advertisements over all interfaces of the router that are configured for IPv6.

6. **To configure the site prefix on the various interfaces of the router, add additional text to the `/etc/inet/ndpd.conf` file.**

   The text should be added in the following format:

   `prefix` *global-routing-prefix*`:`*subnet ID*`/64` *interface*

   In the following example, `/etc/inet/ndpd.conf` file configures the router to advertise the site prefix `2001:0db8:3c4d::/48` over the interfaces `net0` and `net1`.

```
ifdefault AdvSendAdvertisements true
prefixdefault AdvOnLinkFlag on AdvAutonomousFlag on

if net0 AdvSendAdvertisements 1
prefix 2001:0db8:3c4d:15::0/64 net0

if net1 AdvSendAdvertisements 1
prefix 2001:0db8:3c4d:16::0/64 net1
```

7. **Reboot the system.**

   The IPv6 router begins advertising on the local link any site prefix that is in the `ndpd.conf` file.

8. **Display the interface configured for IPv6.**

```
# ipadm show-addr
ADDROBJ      TYPE       STATE   ADDR
lo0/v4       static     ok      192.68.0.1/8
net0/v4      static     ok      172.16.15.232/24
net1/v4      static     ok      172.16.16.220/24
net0/v6      addrconf   ok      fe80::203:baff:fe11:b115/10
lo0/v6       static     ok      ::1/128
net0/v6a     static     ok      2001:db8:3c4d:15:203:baff:fe11:b115/64
net1/v6      addrconf   ok      fe80::203:baff:fe11:b116/10
net1/v6a     static     ok      2001:db8:3c4d:16:203:baff:fe11:b116/64
```

   In the output, each interface that was configured for IPv6 now has two addresses. The entry with the address object name such as *interface*`/v6` shows the link-local address for that interface. The entry with the address object name such as *interface*`/v6a` shows a global IPv6 address. In addition to the interface ID, this address includes the site prefix that you configured in the `/etc/ndpd.conf` file. Note that the designation `v6a` is a randomly defined string. You can define other strings to constitute the second part of the address object name provided that the *interface* reflects the interface over which you are creating the IPv6 addresses, for example, `net0/mystring`, `net0/ipv6addr`.

See Also

- To find out how to configure any tunnels from the routers that you have identified in your IPv6 network topology, refer to "Administering IP Tunnels" in "Administering TCP/IP Networks, IPMP, and IP Tunnels in Oracle Solaris 11.2 ".
- For information about configuring switches and hubs on your network, refer to the manufacturer's documentation.

- To find out how to improve IPv6 support on servers, refer to "Configuring IPv6-Enabled Interfaces on Servers" in "Configuring and Administering Network Components in Oracle Solaris 11.2 ".

♦ ♦ ♦ CHAPTER 3

Using Virtual Router Redundancy Protocol

One way to increase the reliability of the network is to provide backups of the critical components in the network. Oracle Solaris provides an administrative tool that configures and manages the use of the Virtual Router Redundancy Protocol (VRRP) to provide high availability. VRRP is an Internet standard protocol specified in RFC 5798 (http://www.rfc-editor.org/rfc/rfc5798.txt).

Oracle Solaris 11.2 provides the proprietary Layer 3 VRRP to support the creation of VRRP routers over IPMP and InfiniBand interfaces and to enhance the existing support for VRRP in zones.

Note - Throughout this chapter, all references to the term Layer 2 VRRP (L2 VRRP) specifically refer to the Internet standard VRRP and all references to the term Layer 3 VRRP (L3 VRRP) refer to the proprietary Oracle Solaris Layer 3 VRRP.

This chapter provides an overview of Layer 2 VRRP and proprietary Layer 3 VRRP in Oracle Solaris.

This chapter contains the following topics:

About VRRP

VRRP provides high availability of IP addresses, such as those that are used for routers or load balancers. Services that use VRRP are also referred to as VRRP routers even though the services provide functionality other than routing, such as load balancing. For information about

how VRRP is used with load balancer to ensure high availability, see Chapter 7, “Configuring ILB for High Availability”.

A VRRP router is a router that is running the VRRP. VRRP runs on each VRRP router and manages the state of the router. A host can have multiple routers on which VRRP is configured and each router belongs to a different virtual router. You can introduce virtual routers in a local area network (LAN) by using VRRP to provide failure recovery for a router.

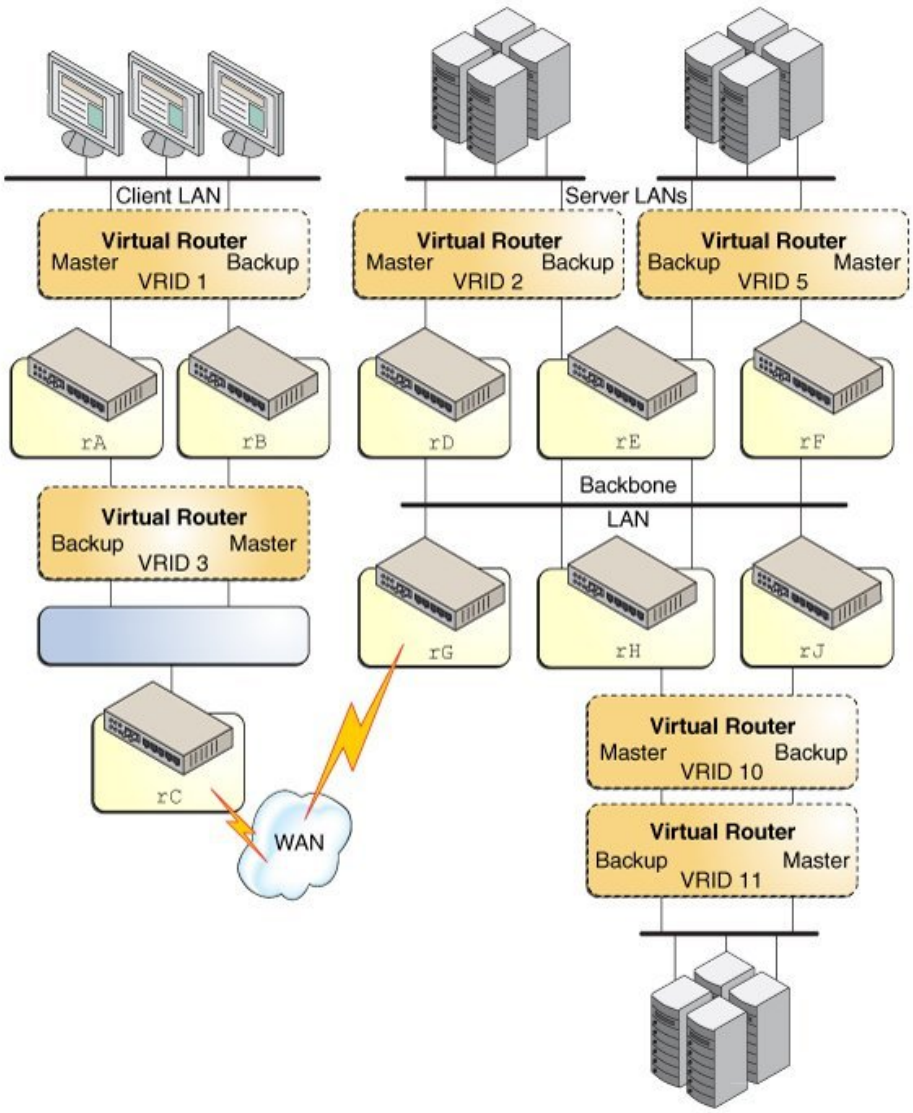
How VRRP Works

Note the following VRRP router terms:

- Router name – A system-wide unique identifier.
- Virtual Router ID (VRID) – A unique number used to identify a virtual router on a given network segment. VRIDs identify the virtual router within a LAN.
- Primary IP address – The source IP address of the VRRP advertisement.
- Virtual IP addresses (VRIP) – An IP address associated with a VRID from which other hosts can obtain network service. The VRIP is managed by the VRRP instances belonging to a VRID.
- Master router – A VRRP instance that performs the routing function for the virtual router at a given time. Only one *master router* is active at a time for a given VRID. The master router controls the IPv4 or IPv6 address or addresses that are associated with the virtual router. The virtual router forwards the packets that are sent to the IP address of the master router.
- Backup router – A VRRP instance for a VRID that is active but not in the master state is called a *backup router*. Any number of backup routers can exist for a VRID. A backup router assumes the role of a master router if the current master router fails.
- VRRP parameters – Includes priority, advertise interval, pre-empt mode, and accept mode.
- VRRP state information and statistics.

The following VRRP load-sharing configuration figure shows that the multiple VRIDs can exist on a single router interface. The accompanying text explains the VRRP components that are used in the figure. This VRRP load-sharing configuration illustrates that multiple VRIDs can exist on a single router interface.

FIGURE 3-1 Load-Sharing Configuration of VRRP in a LAN



- Router `rA` is the master router for virtual router VRID 1 and the backup router for VRID 3. Router `rA` handles the routing of packets that are addressed to the virtual IP (VIP) address for VRID 1 and is ready to assume the routing role for VRID 3.

- Router `rB` is the master router for virtual router VRID 3 and the backup router for VRID 1. Router `rB` handles the routing of packets that are addressed to the VIP for VRID 3 and is ready to assume the routing role for VRID 1.
- Router `rC` does not have VRRP functions, but it uses the VIP for VRID 3 to reach the client LAN subnet.
- Router `rD` is the master router for VRID 2. Router `rF` is the master router for VRID 5. Router `rE` is the backup router for both of these VRIDs. If `rD` or `rF` fails, `rE` becomes the master router for that VRID. Both `rD` and `rF` could fail at the same time. A VRRP router can be the master router for one or more VRIDs.
- Router `rG` is the wide area network (WAN) gateway for the backbone LAN. All of the routers attached to the backbone are sharing routing information with the routers on the WAN by using a dynamic routing protocol such as OSPF. VRRP is not involved in this aspect, although router `rC` advertises that the path to the client LAN subnet is through the VIP of VRID 3.
- Router `rH` is the master router for VRID 10 and the backup router for VRID 11. Likewise, router `rJ` is the master router for VRID 11 and the backup router for VRID 10.

About the Layer 3 VRRP Feature

The proprietary Layer 3 Virtual Router Redundancy Protocol (L3 VRRP) feature in Oracle Solaris removes the need to configure unique VRRP virtual MAC addresses for VRRP routers, and thereby provides better support for VRRP over IPMP and InfiniBand interfaces, and in zones. The L3 VRRP protocol does not conform to the standard VRRP specification. Instead of using a unique virtual MAC address among VRRP routers in the same virtual router, the L3 VRRP implementation uses the gratuitous Address Resolution Protocol (ARP) messages and Neighbor Discovery Protocol (NDP) messages to refresh the mapping between the virtual IP addresses and the MAC address of the current master VRRP router.

The layer 3 VRRP provides the benefit of support for VRRP over IPMP and InfiniBand interfaces, and better support in zones and also eliminates the need to create the VRRP VNIC.

Comparing Layer 2 and Layer 3 VRRP

The following table provides a comparison of Layer 2 and Layer 3 VRRP.

TABLE 3-1 Comparison of Layer 2 and Layer 3 VRRP

Feature	Layer 2 VRRP	Layer 3 VRRP
Creation of a VRRP VNIC	You need to create a VRRP VNIC.	You do not need to create a VRRP VNIC because the virtual VRRP MAC address that is provided by the VRRP VNIC is not needed.
Support for IPMP	Not supported.	Supported. When a Layer 3 VRRP router is created over an IPMP group interface, each virtual IP address on the master router is associated with a MAC address of the active IPMP underlying interface according to the existing IPMP policy. If the failover occurs in the IPMP group, the L2 or L3 mappings are advertised by using the gratuitous ARP or NDP messages.
Zones support	There are issues running multiple VRRP routers that belong to the same virtual router in different zones. On a system with two or more VRRP routers that share the same VRRP virtual MAC address, the built-in virtual switch disrupts the normal flow of the VRRP advertisement packets to the VRRP router. For more information, see "Limitations of Layer 2 and Layer 3 VRRP" on page 32.	Supported.
InfiniBand support	Not supported.	Supported.
Unique virtual router MAC address	Requires a unique virtual router MAC address. The virtual IP addresses always resolve to the same virtual MAC address.	Not required. Uses the MAC address on which the VRRP router is created. The MAC address is different among all the VRRP routers that are in the same virtual router. The same MAC address is associated with the virtual IP addresses that are protected by this L3 VRRP router.
Configuring VRRP virtual IP addresses	Need to configure.	Need to configure.
Internet Control Message Protocol (ICMP) Redirects	Might be used when the L2 VRRP is running between group of routers. When an L2 VRRP router needs to use the ICMP redirects, it checks the destination MAC address (VRRP virtual MAC address) of the packets that need to be redirected. By using the destination MAC address, the L2 VRRP router determines the virtual router to which the packet was initially sent. Hence, the L2 VRRP router is able to select the source address and send the ICMP redirect message to the source.	Need to disable ICMP redirects. When multiple VRRP routers are created over the same interface, they share the same MAC address. Therefore, the L3 VRRP cannot determine the destination MAC address.
Election of master router	The election of the master router is transparent to the host. When the master router changes, the switch that exists between the host and the router identifies the new port to send the traffic by using its MAC learning capability.	The election of the master router changes the Layer 2 mapping of the virtual IP addresses and the new mapping must be advertised by the gratuitous ARP or NDP messages.

Feature	Layer 2 VRRP	Layer 3 VRRP
Failover time	Normal.	Might be longer because of the additional requirement of gratuitous ARP or the NDP messages when election of the Master router changes.

Limitations of Layer 2 and Layer 3 VRRP

Both Layer 2 and Layer 3 VRRP have a common limitation that you must configure the Layer 2 and Layer 3 VRRP virtual IP addresses statically. You cannot auto-configure the VRRP virtual IP addresses by using the two existing auto-configuration tools for IP addresses: `in.ndpd` for IPv6 auto-configuration and `dhcpagent` for Dynamic Host Configuration Protocol (DHCP) configuration. In addition, Layer 2 and Layer 3 VRRP have specific limitations.

The Layer 2 VRRP feature has the following limitations:

- **Exclusive-IP Zone Support**

  When any VRRP router is created in an exclusive-IP zone, the VRRP service `svc:/network/vrrp/default` is enabled automatically. The VRRP service manages the VRRP router for that specific zone. However, support for an exclusive-IP zone is limited as follows:

  - Because a Virtual Network Interface Card (VNIC) cannot be created inside a non-global zone, you must create the VRRP VNIC in the global zone first. Then assign the VNIC to the non-global zone where the VRRP router resides. You can then create the VRRP router in the non-global zone by using the `vrrpadm` command.
  - On a single Oracle Solaris system, you cannot create two VRRP routers in different zones to participate with the same virtual router. Oracle Solaris does not allow you to create two VNICs with the same media access control (MAC) address.

- **Interoperations With Other Networking Features**
  - The L2 VRRP service cannot work on an IP network multipathing (IPMP) interface. VRRP requires specific VRRP MAC addresses but IPMP works completely in the IP layer. For information about IPMP, see Chapter 2, “About IPMP Administration,” in “Administering TCP/IP Networks, IPMP, and IP Tunnels in Oracle Solaris 11.2 ”.

    VRRP can be used on link aggregations in trunk or DLMP aggregation modes. For more information about aggregations, see Chapter 2, “Configuring High Availability by Using Link Aggregations,” in “Managing Network Datalinks in Oracle Solaris 11.2 ”.
  - The L2 VRRP service cannot work on an IP over Infiniband (IPoIB) interface.
- **Ethernet Over InfiniBand Support**

  L2 VRRP does not support the Ethernet over InfiniBand (EoIB) interface. Because every L2 VRRP router is associated with a unique virtual MAC address, the VRRP routers

participating with the same virtual router need to use the same virtual MAC address simultaneously, which is not supported by the EoIB interface. L3 VRRP overcomes this limitation as it uses a different MAC address among all the VRRP routers that exist on the same virtual router.

The Layer 3 VRRP feature has the following limitations:

- Using gratuitous ARP or NDP messages might result in a longer failover time during the election of the master router.

  L3 VRRP uses gratuitous ARP or NDP messages to advertise the new L2 or L3 mapping when the election of the master router changes. This additional requirement of using gratuitous ARP or NDP messages might result in a longer failover time. In some cases, if all the advertised gratuitous ARP or NDP messages are lost, it might take more time for a host to receive the refreshed ARP or NDP entry. Therefore, sending of packets to the new master router might be delayed.
- Unable to determine the destination MAC address when using ICMP redirects because the same destination MAC address is shared by multiple routers.

  You can use ICMP redirects when you are using VRRP among a group of routers in a network topology that is not symmetric. The IPv4 or IPv6 source address of an ICMPv4 redirect or ICMPv6 redirect must be the address used by the end host when making the next-hop routing decision.

  When an L3 VRRP router needs to use ICMP redirects, the L3 VRRP router checks the destination MAC address (VRRP virtual MAC address) of the packets that need to be redirected. Because the same destination MAC address is shared by multiple routers created over the same interface, the L3 VRRP router cannot determine the destination MAC address. Therefore, it might be useful to disable ICMP redirects when you use L3 VRRP routers. You can disable ICMP redirects by using the `send_redirects` public IPv4 and IPv6 protocol properties as follows:

  ```
  # ipadm set-prop -m ipv4 -p send_redirects=off
  ```
- VRRP virtual IP addresses cannot be configured automatically either by `in.ndpd` or DHCP.

♦ ♦ ♦ CHAPTER 4

Configuring and Administering Virtual Router Redundancy Protocol

This chapter describes the tasks for configuring Layer 2 and Layer 3 VRRP.

This chapter contains the following topics:

Planning a VRRP Configuration

Planning a Layer 2 or Layer 3 VRRP configuration involves the following steps:

1. Determining whether to configure an L2 VRRP or an L3 VRRP router.
2. (For L2 VRRP router only) Creating a VRRP VNIC. For more information, see "Creating a VRRP VNIC for Layer 2 VRRP" on page 37.

   You can automatically create a VRRP VNIC by using the `-f` option of the `vrrpadm` command while you create the L2 VRRP router.
3. Creating a VRRP router. For more information, see "Creating a VRRP Router" on page 37.
4. Configuring the virtual IP address for the VRRP router. For more information, see "Configuring the Virtual IP Address for Layer 2 and Layer 3 VRRP Routers" on page 40.

   You can configure the virtual IP addresses by using the `-a` option of the `vrrpadm` command. For more information, see "Creating a VRRP Router" on page 37.

Installing VRRP

You must install VRRP to use VRRP on your system.

▼ How to Install VRRP

1. **Become an administrator.**

   For more information, see “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2 ”.

2. **Verify whether the VRRP package is installed.**

   ```
   # pkg info vrrp
   ```

3. **Install VRRP package if it is not installed.**

   ```
   # pkg install vrrp
   ```

Configuring VRRP

You can use the `vrrpadm` command to configure a VRRP router. The results of all the subcommands of `vrrpadm` command are persistent except for the `vrrpadm show-router` command. For example, the VRRP router that is created by the `vrrpadm create-router` command persists across reboots. For more information, see the `vrrpadm`(1M) man page.

You need to have the `solaris.network.vrrp` authorization, which is part of the Network Management profile, to configure the VRRP router.

Note - The read-only operation initiated by the `vrrpadm show-router` command does not require `solaris.network.vrrp` authorization.

Caution - When you use VRRP with the Oracle Solaris bundled IP Filter, you must check whether the incoming or the outgoing IP traffic is allowed for the standard VRRP multicast address, `224.0.0.18/32` by using the `ipfstat -io` command. If the traffic is not allowed, both the master and backup VRRP routers will be in the `MASTER` state. Therefore, you must add corresponding rules to IP filter configuration for each of the VRRP router. For more information, see "Troubleshooting Issues With VRRP and the Oracle Solaris Bundled IP Filter" in "Troubleshooting Network Administration Issues in Oracle Solaris 11.2 ".

Creating a VRRP VNIC for Layer 2 VRRP

VNICs are virtual network interfaces configured on top of a system's physical network adapter and are essential components of network virtualization. A physical interface can have more than one VNIC. For more information about VNICs, see "Managing Network Virtualization and Network Resources in Oracle Solaris 11.2 ".

Each Layer 2 VRRP router requires a special VRRP VNIC. Use the following command syntax.

```
# dladm create-vnic [-t] [-R root-dir] -l link [-m vrrp -V VRID -A \
{inet | inet6}] [-v VLAN-ID] [-p prop=value[,...]] VNIC
```

This command creates a VNIC with a virtual router MAC address that is defined by the VRRP specification. Use the VNIC address type, `vrrp`, to specify the VRID and address family. The address family is either `inet` or `inet6`, which refers to either IPv4 or IPv6 addresses. For example:

```
# dladm create-vnic -m vrrp -V 21 -A inet6 -l net0 vnic0
```

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

Note - You can also create a VRRP VNIC by using the `-f` option with the `vrrpadm` command. For more information, see "Creating a VRRP Router" on page 37.

Creating a VRRP Router

The `vrrpadm create-router` command creates a Layer 2 or Layer 3 VRRP router with the specified VRID and address family, along with other specified parameters. For more information, see the `vrrpadm`(1M) man page.

To create a VRRP router, use the following syntax:

```
# vrrpadm create-router [-T {l2 | l3}] [-f] -V VRID -I ifname \
-A [inet | inet6] [-a assoc-IPaddress] [-P primary-IPaddress] \
[-p priority] [-i adv-interval] [-o flags] router-name
```

`-T l2 | l3`
: Specifies the type of the router. You can set the type to one of the following values. The default is `l2`.
  - `l2` – L2 type VRRP router
  - `l3`– L3 type VRRP router

`-f`
: (L2 VRRP only) Specifies the creation of the VRRP VNIC with an L2 VRRP router. When you specify the `-f` option, the `vrrpadm` command checks whether the VRRP VNIC with the specified VRID and address family exists. A VRRP VNIC is created only if it does not already exist. The system generates the name of the VRRP VNIC with the naming convention: `vrrp-`*VRID_ifname_*`v`*4* | *6*. The `-f` option does not have any effect when you are creating a Layer 3 VRRP router.

`-V` *VRID*
: The virtual router identifier that defines the VLAN when associated with the address family.

`-I` *ifname*
: The interface on which the VRRP router is configured. For a Layer 2 VRRP, the interface can be a physical link, a VLAN, or an aggregation. For a Layer 3 VRRP, the interface can also include an IPMP interface, a DHCP managed interface, and an InfiniBand interface. This link determines the LAN in which this VRRP router is running.

`-A [inet | inet6]`
: The address family, either `inet` or `inet6`, which refers to either IPv4 or IPv6 addresses.

`-a` *assoc-IPaddress*
: Specifies the comma-separated list of IP addresses.

  You can specify the IP address in any of the following formats:
  - *IP-address*[/*prefix-length*]
  - *hostname*[/*prefix-length*]
  - `linklocal`

    If you specify `linklocal`, an IPv6 link-local `vrrp` address is configured based on the VRID of the associated virtual router. The `linklocal` form applies only to IPv6 VRRP routers. You can combine the `-a` option with the `-f` option so that the VNIC is created and plumbed automatically.

`-P` *primary-IPaddress*
: Specifies the VRRP primary IP address that is used to send the VRRP advertisement.

`-p` *priority*	The priority of the specified VRRP router used for master selection. The default value is `255`. The router with the highest priority value is selected as the master router.
`-i` *adv-interval*	The advertisement interval in milliseconds. The default value is `1000`.
`-o` *flags*	The pre-empt and accept modes of the VRRP router. The values are `preempt` or `un_preempt`, or `accept` or `no_accept`. By default, the pre-empt and accept modes are set to `preempt` and `accept` respectively.
router-name	The *router-name* is the unique identifier of this VRRP router. The permitted characters in a router name are alphanumeric (`a-z`, `A-Z`, `0-9`), and underscore (_). The maximum length of a router name is 31 characters.

EXAMPLE 4-1 Creating a Layer 2 VRRP Router

The following example shows how to create a router over a datalink `net0`.

```
# dladm create-vnic -m vrrp -V 12 -A inet -l net0 vnic1
# vrrpadm create-router -V 12 -A inet -p 100 -I net0 l2router1
# vrrpadm show-router l2router1
NAME       VRID  TYPE   IFNAME AF   PRIO ADV_INTV MODE  STATE  VNIC
l2router1  12    L2     net0   IPv4 100  1000     e-pa- BACK   vnic1
```

An L2 VRRP router `l2router1` is created over the datalink `net0` with an IPv4 address family and VRID `12`. For information about the `vrrpadm show-router` command, see "Displaying Layer 2 and Layer 3 VRRP Router Configurations" on page 42.

EXAMPLE 4-2 Creating a Layer 3 VRRP Router

The following example shows how to create an L3 VRRP router over an IPMP interface named `ipmp0`.

```
# vrrpadm create-router -V 6 -I ipmp0 -A inet -T l3 l3router1
# vrrpadm show-router
NAME      VRID TYPE IFNAME AF   PRIO ADV_INTV MODE  STATE VNIC
l3router1 6    L3   ipmp0  IPv4 255  1000     eopa- INIT  --
```

An L3 VRRP router `l3router1` is created over the IPMP interface `ipmp0` with an IPv4 address family and VRID `6`. For information about the `vrrpadm show-router` command, see "Displaying Layer 2 and Layer 3 VRRP Router Configurations" on page 42.

Configuring the Virtual IP Address for Layer 2 and Layer 3 VRRP Routers

To configure the IP address for an L2 VRRP router, you must configure the virtual IP address of type `vrrp` over the VRRP VNIC that is associated with it.

To configure the virtual IP address for an L3 VRRP router, you must use an IP address of type `vrrp` on the same IP interface over which the L3 VRRP router is configured.

Note - To configure an IPv6 address, you must have created the VRRP VNIC or the L3 VRRP router by specifying the address family of the router as `inet6`.

To configure a virtual IP address for a VRRP router, use the following syntax:

```
# ipadm create-addr [-t] -T vrrp [-a local=addr[/prefix-length]] \
 [-n router-name].... addr-obj | interface
```

`-t` — Specifies that the configured address is temporary and that the changes apply only to the active configuration.

`-T vrrp` — Specifies that the configured address is of the type `vrrp`.

`-n` *router-name* — The `-n` *router-name* option is optional for an L2 VRRP router because the VRRP router name can be derived from the VRRP VNIC interface on which the IP addresses are configured.

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

Note - You can also configure virtual IP addresses by using the `-a` option with the `vrrpadm` command. For more information, see "Creating a VRRP Router" on page 37.

EXAMPLE 4-3 Configuring Virtual IP Address for an L2 VRRP Router

You can use the `vrrp` type IP address to configure the virtual IP addresses for an L2 VRRP router. The following example shows how to create the virtual IP address for `l2router1`.

```
# ipadm create-ip vrrp_vnic1
# ipadm create-addr -T vrrp -n l2router1 -a 192.168.82.8/24 vrrp_vnic1/vaddr1
```

The following example shows how to create an IPv6 link-local `vrrp` IP address for `V6vrrp_vnic1/vaddr1`.

```
# ipadm create-ip V6vrrp_vnic1
# ipadm create-addr -T vrrp V6vrrp_vnic1/vaddr1
```

To configure the IPv6 link-local `vrrp` type IP address for an VRRP router, you do not need to specify the local address. An IPv6 link-local `vrrp` type IP address is created based on the VRID of the associated VRRP router.

EXAMPLE 4-4 Configuring the Virtual IP Address for an L3 VRRP Router

The following example shows how to configure the virtual IP address for `l3router1`.

```
# ipadm create-ip ipmp0
# ipadm create-addr -T vrrp -n l3router1 -a 172.16.82.8/24 ipmp0/vaddr1
```

The following example shows how to configure an IPv6 link-local `vrrp` type IP address for the L3 VRRP router `l3V6router1`.

```
# ipadm create-ip ipmp1
# ipadm create-addr -T vrrp -n l3V6router1 ipmp1/vaddr0
```

Enabling and Disabling a VRRP Router

A VRRP router is enabled by default when you first create it. You can disable a VRRP router and re-enable it. The interface over which the VRRP router is created (specified with the `-I` option when the router is created with `vrrpadm create-router`) must exist when the router is enabled. Otherwise, the enable operation fails. For an L2 VRRP router, if the router's VRRP VNIC does not exist, the router is not effective. The syntax is as follows:

```
# vrrpadm enable-router router-name
```

At times, you might need to temporarily disable a VRRP router to make configuration changes and then re-enable the router. The syntax for disabling a router is as follows:

```
# vrrpadm disable-router router-name
```

Modifying a VRRP Router

The `vrrpadm modify-router` command changes the configuration of a specified VRRP router. You can modify the priority, the advertisement interval, the pre-empt mode, and the accept mode of the router. The syntax is as follows:

```
# vrrpadm modify-router [-p priority] [-i adv-interval] [-o flags] router-name
```

Displaying Layer 2 and Layer 3 VRRP Router Configurations

The `vrrpadm show-router` command shows the configuration and status of a specified VRRP router. For more information, see the `vrrpadm`(1M) man page. The syntax is as follows:

```
# vrrpadm show-router [-P | -x] [-p] [-o field[,...]] [router-name]
```

EXAMPLE 4-5 Displaying a Layer 2 VRRP Router Configuration

The following examples show the `vrrpadm show-router` command output.

```
# vrrpadm show-router vrrp1
NAME VRID TYPE  IFNAME  AF   PRIO ADV_INTV MODE  STATE  VNIC
vrrp1 1   L2    net1    IPv4 100  1000     e-pa- BACK   vnic1
```

`NAME` Name of the VRRP router.

`VRID` VRID of the VRRP router.

`TYPE` The type of VRRP router, which is either L2 or L3.

`IFNAME` The interface on which the VRRP router is configured. For an L2 VRRP router, the interface can be a physical Ethernet interface, a VLAN, or an aggregation.

`AF` The address family of the VRRP router. It can be either IPv4 or IPv6.

`PRIO` The priority of the VRRP router, which is used for master selection.

`ADV_INTV` The advertisement interval displayed in milliseconds.

`MODE` A set of flags that are associated with the VRRP router and include the following possible values:

- `e` – Specifies that the router is enabled.
- `p` – Specifies that the mode is `preempt`.
- `a` – Specifies that the mode is `accept`.
- `o` – Specifies that the router is the virtual address owner.

`STATE` The current state of the VRRP router. The possible values are: `INIT` (initialize), `BACK` (backup), and `MAST` (master).

In this example, information about the specified VRRP router `vrrp1` is displayed.

```
# vrrpadm show-router -x vrrp1
NAME  STATE PRV_STAT STAT_LAST VNIC  PRIMARY_IP VIRTUAL_IPS
vrrp1 BACK  MAST     1m17s     vnic1 10.0.0.100 10.0.0.1
```

`PRV_STAT` The previous state of the VRRP router.

`STAT_LAST` Time since the last state transition.

`PRIMARY_IP` The primary IP address selected by the VRRP router.

`VIRTUAL_IPS` The virtual IP addresses configured on the VRRP router.

In this example, additional information about the router, such as the primary IP address selected by the VRRP router, virtual IP address configured on the VRRP router, and the previous state of the VRRP router is displayed.

```
# vrrpadm show-router -P vrrp1
NAME  PEER       P_PRIO  P_INTV  P_ADV_LAST  M_DOWN_INTV
vrrp1 10.0.0.123 120     1000    0.313s      3609
```

`PEER` The primary IP address of the peer VRRP router.

`P_PRIO` The priority of the peer VRRP router, which is part of the advertisement received from the peer.

`P_INTV` The advertisement interval (in milliseconds), which is part of the advertisements received from the peer.

`P_ADV_LAST` Time since the last received advertisement from the peer.

`M_DOWN_INTV` Time interval (in milliseconds) after which the master router is declared down.

The `-P` option is used only when the VRRP router is in the backup state.

EXAMPLE 4-6 Displaying the L3 VRRP Router on a System

```
# vrrpadm show-router
NAME   VRID  TYPE  IFNAME  AF    PRIO  ADV_INTV  MODE   STATE   VNIC
l3vr1  12    L3    net1    IPv6  255   1000      eopa-  INIT    –
```

In this example, the L3 VRRP router `l3vr1` is configured over the interface `net1`.

Displaying IP Addresses That Are Associated With VRRP Routers

You can display the IP address associated with a VRRP router by using the `ipadm show-addr` command. The `ROUTER` field in the output of the `ipadm show-addr` command displays the name of the VRRP router that is associated with a specific `vrrp` type IP address.

For the `vrrp` type IP address of an L2 VRRP, the name of the VRRP router is derived from the VRRP VNIC over which the IP address is configured. If you issue the `ipadm show-addr` command before you create the L2 router for a VRRP VNIC, the `ROUTER` field displays `?`. For the `vrrp` type IP address of an L3 VRRP, the `ROUTER` field always displays the specified router name. For other types of IP addresses, the `ROUTER` field is not applicable and `--` is displayed.

EXAMPLE 4-7 Displaying IP Addresses That Are Associated With VRRP Routers

```
# ipadm show-addr -o addrobj,type,vrrp-router,addr
ADDROBJ              TYPE     VRRP-ROUTER   ADDR
lo0/v4               static   --               127.0.0.1/8
net1/p1              static   --               192.168.11.10/24
net1/v1              vrrp     l3router1        192.168.81.8/24
vrrp_vnic1/vaddr1    vrrp     l2router1        192.168.82.8/24
lo0/v6               static   --               ::1/128
```

In this example, `l3router1` is associated with the `vrrp` type IP address `192.168.81.8/24` and `l2router1` is associated with the `vrrp` type IP address `192.168.82.8/24`.

The output shows the following information:

`ADDROBJ` The name of the address object.

`TYPE` The type of the address object, which can be one of the following:

- `from-gz`
- `static`
- `dhcp`
- `addrconf`
- `vrrp`

`VRRP-ROUTER` The name of the VRRP router.

`ADDR` The numeric IPv4 or IPv6 address.

Deleting a VRRP Router

The `vrrpadm delete-router` command deletes a specified VRRP router. The syntax is as follows:

```
# vrrpadm delete-router router-name
```

Note - The VRRP VNIC, the `vrrp` type IP address, and the primary IP address that are created by using the `-f`, `-a`, `-P` options of the `vrrpadm create-router` command respectively are not deleted as a result of the `vrrpadm delete-router` command. You must explicitly delete them by using the corresponding `ipadm` and `dladm` commands.

Controlling Gratuitous ARP and NDP Messages

When a backup router becomes a master VRRP router, VRRP sets a flag on all the virtual IP addresses associated with the master router and therefore the virtual IP addresses are protected. If there are no conflicts for the virtual IP addresses, several gratuitous ARP and neighbor advertisement messages are sent to advertise the new mapping between the virtual IP address and the MAC address of the new master.

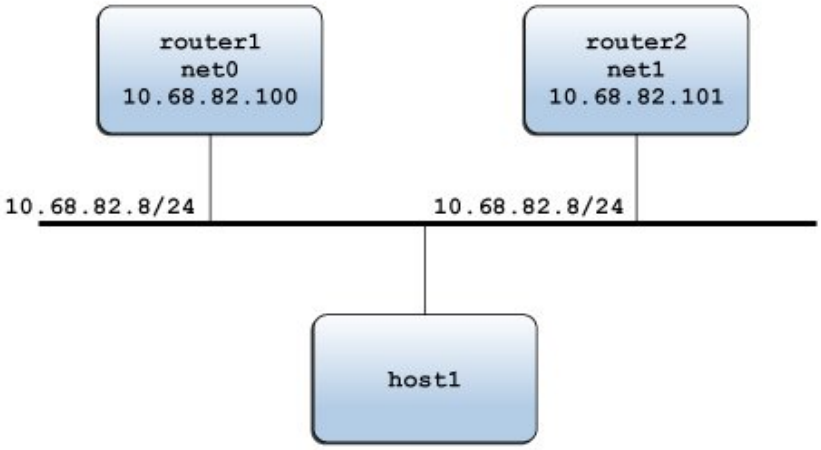
To control the number of messages sent and the interval between the advertisement of messages, you can use the following IP protocol properties:

- `arp_publish_count`
- `arp_publish_interval`
- `ndp_unsolicit_count`
- `ndp_unsolicit_interval`

For more information about the IP protocol properties, see “IP Tunable Parameters Related to Duplicate Address Detection” in “Oracle Solaris 11.2 Tunable Parameters Reference Manual ”.

Use Case: Configuring a Layer 2 VRRP Router

The following figure shows a typical VRRP configuration.



In this example, the IP address `10.68.82.8` is configured as the default gateway for `host1`. This IP address is the virtual IP address that is protected by the virtual router that consists of two VRRP routers: `router1` and `router2`. At any given time, only one of the two routers serves as the master router, assuming the responsibilities of the virtual router and forwarding packets that come from `host1`.

Assume that the VRID of the virtual router is `12`. The following examples show the commands that are used to configure the example VRRP configuration on `router1` and `router2`. `router1` is the owner of the virtual IP address `10.68.82.8` and its priority is the default value (`255`). `router2` is the standby router whose priority is `100`.

For more information about the commands that are used for configuring VRRP, see the `vrrpadm`(1M), `dladm`(1M), and `ipadm`(1M) man pages.

For `router1`:

1. Install VRRP package.

   ```
   # pkg install vrrp
   ```

2. Create the VNIC `vnic0` over `net0` with the VRID value as `12`.

   ```
   # dladm create-vnic -m vrrp -V 12 -A inet -l net0 vnic0
   ```

3. Create a VRRP router `vrrp1` over `net0`.

   ```
   # vrrpadm create-router -V 12 -A inet -I net0 vrrp1
   ```

4. Configure IP interfaces `vnic0` and `net0`.

   ```
   # ipadm create-ip vnic0

   # ipadm create-addr -T vrrp -a 10.68.82.8/24 vnic0/router1
   ```

```
# ipadm create-ip net0

# ipadm create-addr -T static -a 10.68.82.100/24 net0/router1
```

5. Display the router information for `vrrp1`.

```
# vrrpadm show-router -x vrrp1
NAME  STATE   PRV_STAT STAT_LAST  VNIC  PRIMARY_IP    VIRTUAL_IPS
vrrp1 MASTER  INIT     14.444s    vnic0 10.68.82.100  10.68.82.8
```

Similarly, for `router2`:

1. Create the VNIC `vnic1` over `net1` with the VRID value as `12`.

```
# dladm create-vnic -m vrrp -V 12 -A inet -l net1 vnic1
```

2. Create a VRRP router `vrrp2` over `net1`.

```
# vrrpadm create-router -V 12 -A inet -I net1 -p 100 vrrp2
```

3. Configure IP interfaces on `vnic1` and `net1`.

```
# ipadm create-ip vnic1

# ipadm create-addr -T vrrp -a 10.68.82.8/24 vnic1/router2

# ipadm create-ip net1

# ipadm create-addr -T static -a 10.68.82.101/24 net1/router2
```

4. Display the router information for `vrrp2`.

```
# vrrpadm show-router -x vrrp2
NAME  STATE    PRV_STAT  STAT_LAST  VNIC  PRIMARY_IP    VIRTUAL_IPS
vrrp2 BACKUP   INIT      2m32s      vnic1 10.68.82.101  10.68.82.8
```

By using the configuration of `router1` as an example, you must configure at least one IP address over `net0`. This IP address of `router1` is the primary IP address, which is used to send the VRRP advertisement packets.

```
# vrrpadm show-router -x vrrp1
NAME  STATE   PRV_STAT STAT_LAST  VNIC  PRIMARY_IP    VIRTUAL_IPS
vrrp1 MASTER  INIT     14.444s    vnic1 10.68.82.100  10.68.82.8
```

♦ ♦ ♦ CHAPTER 5

Overview of an Integrated Load Balancer

This chapter describes the components of ILB components and modes in which the ILB operates, such as Direct Server Return (DSR) mode and Network Address Translator (NAT) mode.

This chapter contains the following topics:

For more information about ILB, see "Integrated Load Balancer Overview" on page 13.

ILB Components

ILB is managed by the Service Management Facility (SMF) service `svc:/network/loadbalancer/ilb:default`. For more information about SMF, see "Managing System Services in Oracle Solaris 11.2 ". The three major components of ILB are:

- `ilbadm` command-line interface (CLI) – You can use the CLI to configure load-balancing rules, perform optional health checks, and view statistics.
- `libilb` configuration library – `ilbadm` and third-party applications can use the functionality implemented in `libilb` for ILB administration.
- `ilbd` daemon – This daemon performs the following tasks:
  - Manages persistent configuration across reboots and package updates
  - Provides serial access to the ILB kernel module by processing the configuration information and sending it to the ILB kernel module for execution
  - Performs health checks and sends the results to the ILB kernel module so that the load distribution is properly adjusted

ILB Operation Modes

ILB supports stateless DSR and NAT modes of operation for IPv4 and IPv6 in single-legged and dual-legged topologies.

Direct Server Return Mode

In DSR mode, ILB balances the incoming requests to the back-end servers but allows the return traffic from the servers to the clients to bypass it. However, if you set up ILB to be used as a router for a back-end server, the response from the back-end server to the client is routed through the system that is running ILB. ILB's current implementation of DSR does not provide TCP connection tracking, making it stateless. With stateless DSR, ILB does not save any state information of the processed packets except for basic statistics. Being stateless, the performance is comparable to the normal IP forwarding performance. The DSR mode is best suited for connectionless protocols.

Advantages:

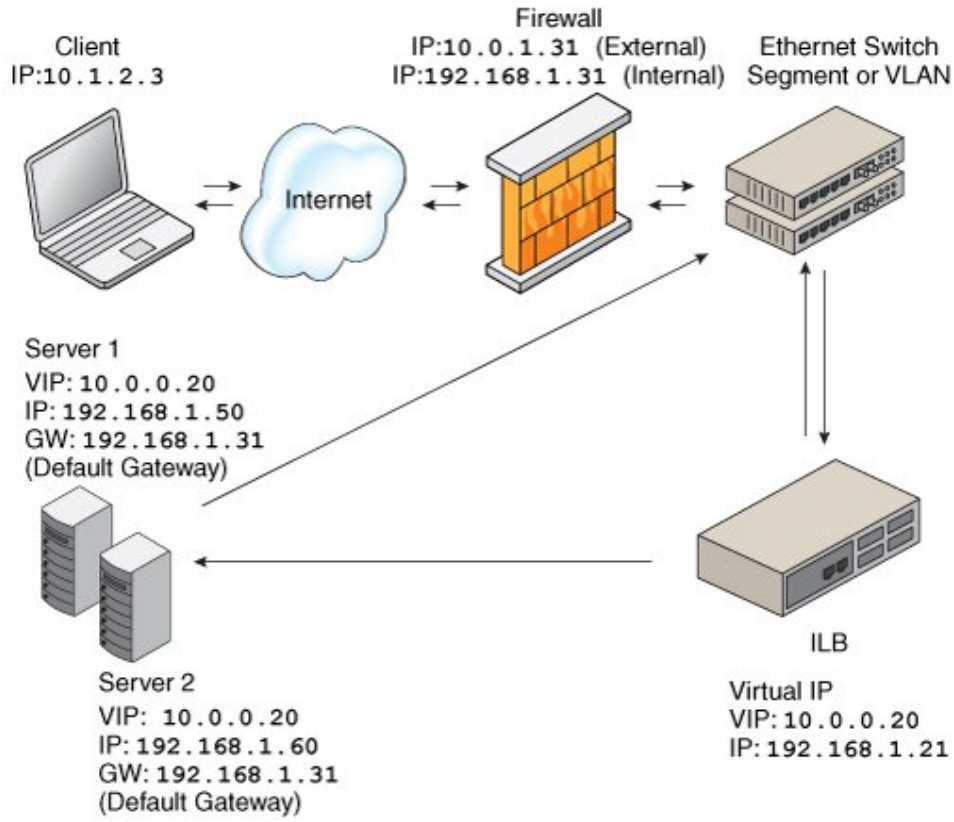
- DSR provides better performance than NAT because only the destination MAC address of packets is changed and servers respond directly to clients.
- There is full transparency between the server and the client. The servers see a connection directly from the client IP address and reply to the client through the default gateway.

Disadvantages:

- The back-end server must respond to both its own IP address (for health checks) and the virtual IP address (for load-balanced traffic).
- Being stateless, adding or removing servers causes connection disruption.

The following figure shows the implementation of ILB in the DSR mode.

FIGURE 5-1 Direct Server Return Topology



In this figure, both back-end servers are in the same subnet (`192.168.1.0/24`) as the ILB box. The servers are also connected to the router so that they can reply directly to clients after receiving a request forwarded by the ILB box.

Network Address Translator Mode

ILB uses NAT in stand-alone mode strictly for load-balancing functionality. In this mode, ILB rewrites the header information and handles the incoming as well as the outgoing traffic. ILB operates in both the half-NAT and full-NAT modes. However, full-NAT also rewrites the source IP address, making it appear to the server that all connections are originating from the load

balancer. NAT does provide TCP connection tracking (meaning that it is stateful). The NAT mode provides additional security and is best suited for Hypertext Transfer Protocol (HTTP) or Secure Sockets Layer (SSL) traffic.

Advantages:

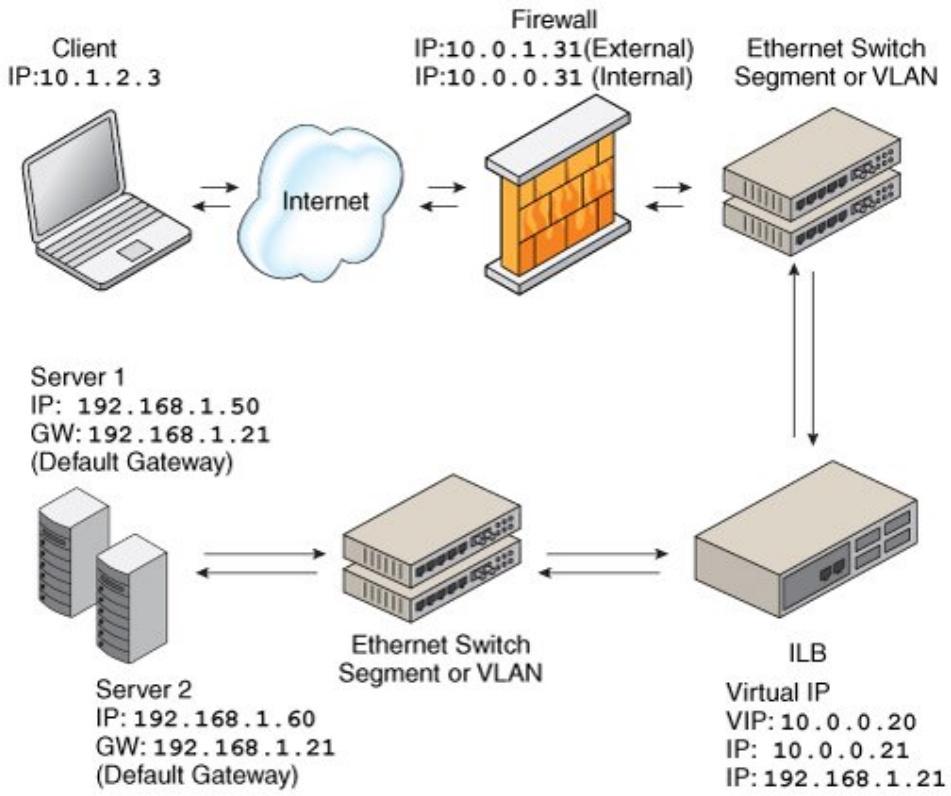
- Works with all back-end servers by changing the default gateway to point to the load balancer
- Adding or removing servers without connection disruption is possible because the load balancer maintains the connection state

Disadvantages:

- Slower performance than DSR because processing involves manipulation of the IP header and servers send responses to the load balancer
- All the back-end servers must use the load balancer as a default gateway

The general implementation of the NAT mode is as shown in the following figure.

FIGURE 5-2 Network Address Translation Topology



In this case, all requests to the VIP go through the ILB and are forwarded to the back-end servers. All the replies from the back-end servers pass through the ILB for NAT.

Caution - The NAT code path that is implemented in ILB differs from the code path that is implemented in the IP Filter feature of Oracle Solaris. Do *not* use both these code paths simultaneously.

Half-NAT Load-Balancing Mode

In the half-NAT mode of the ILB operation, ILB rewrites only the destination IP address in the header of the packets. If you are using the half-NAT implementation, you cannot connect to a VIP address of the service from the same subnet on which the server resides. The following table shows the IP addresses of the packets flowing between the client and ILB, and between ILB and back-end servers.

TABLE 5-1 Request Flow and Response Flow for the Half-NAT Implementation When the Server and Client Are on Different Networks

	Request Flow	Source IP Address	Destination IP Address
1.	Client –> ILB	Client	VIP of ILB
2.	ILB –> Server	Client	Server
	Response Flow		
3.	Server –> ILB	Server	Client
4.	ILB –> Client	VIP of ILB	Client

If you connect the client system to the same network as that of the servers, the intended server responds directly to the client, and the fourth step in the table does not occur. Therefore, the source IP address for the server response to the client is invalid. When the client sends a connection request to the load balancer, the response occurs from the intended server. From this point onwards, the IP stack of the client correctly drops all the responses. For this scenario, the request flow and response flow proceed as shown in the following table.

TABLE 5-2 Request Flow and Response Flow for the Half-NAT Implementation When the Server and Client Are on the Same Network

	Request Flow	Source IP Address	Destination IP Address
1.	Client –> ILB	Client	VIP of ILB
2.	ILB –> Server	Client	Server
	Response Flow		
3.	Server –> Client	Server	Client

Full-NAT Load-Balancing Mode

In the full-NAT implementation of the ILB operation, the source and destination IP addresses are rewritten to ensure that the traffic goes through the load balancer in both directions. The

full-NAT mode makes it possible to connect to the VIP from the same subnet that the servers are on.

The following table depicts the IP addresses of the packets flowing between a client and ILB, and between ILB and a back-end server by using the full-NAT mode. No special default route using the ILB box is required in the servers. Note that the full-NAT mode requires the administrator to set aside one IP address or a range of IP addresses to be used by ILB as source addresses to communicate with the back-end servers. Assume that the addresses used belong to subnet C. In this scenario, ILB behaves as a proxy.

TABLE 5-3 Request Flow and Response Flow for the Full-NAT Implementation

	Request Flow	Source IP Address	Destination IP Address
1.	Client –> ILB	Client	VIP of ILB
2.	ILB –> Server	Interface address of the load balancer (subnet C)	Server
	Response Flow		
3.	Server –> ILB	Server	Interface address of the ILB (subnet C)
4.	ILB –> Client	VIP of ILB	Client

How ILB Works

This section describes the process of ILB, which involves processing a request from a client to the VIP, forwarding the request to a back-end server, and processing the response.

The client-to-server packet processing involves the following steps:

1. ILB receives an incoming request that is sent by the client to a VIP address and matches the request to a load-balancing rule.
2. If ILB finds a matching load-balancing rule, it uses a load-balancing algorithm to forward the request to a back-end server depending on the mode of operation.
   - In the DSR mode, ILB replaces the MAC header of the incoming request with the MAC header of the selected back-end server.
   - In the half-NAT mode, ILB replaces the destination IP address and the transport protocol port number of the incoming request with that of the selected back-end server.
   - In the full-NAT mode, ILB replaces the source IP address and the transport protocol port number of the incoming request with the load-balancing rule's NAT source address. ILB also replaces the destination IP address and the transport protocol port number of the incoming request with that of the selected back-end server.
3. ILB forwards the modified incoming request to the selected back-end server.

The server-to-client packet processing involves the following steps:

1. The back-end server sends a reply to ILB in response to the incoming request from the client.
2. ILB's action after receiving the response from the back-end server is based on the mode of operation.
   - In the DSR mode, the response from the back-end server bypasses ILB and goes directly to the client. However, if ILB is also used as a router for the back-end server, then the response from the back-end server to the client is routed through the system running ILB.
   - In the half-NAT and full-NAT modes, ILB matches the response from the back-end server to the incoming request, and replaces the changed IP address and the transport protocol port number with those from the original incoming request. ILB then forwards the response to the client.

♦ ♦ ♦ CHAPTER 6

Configuring and Managing the Integrated Load Balancer

ILB is configured on the L3 and L4 layers of the network protocol stack. This chapter describes the tasks to install ILB, enable or disable ILB, define server groups and back-end servers in ILB, and export and import ILB configurations by using the `ilbadm` command. For more information, see the `ilbadm`(1M) man page. For more information about configuring your ILB for high availability, see Chapter 7, “Configuring ILB for High Availability”.

For information about how to deploy an Oracle Solaris integrated load balancer, see Deploying the Oracle Solaris Integrated Load Balancer in 60 Minutes (http://www.oracle.com/technetwork/systems/hands-on-labs/hol-deploy-ilb-60mmin-2137812.html). For information about how to add high availability to your application by using Oracle Solaris Zones and the ILB in Oracle Solaris, see How to Set Up a Load-Balanced Application Across Two Oracle Solaris Zones (http://www.oracle.com/technetwork/articles/servers-storage-admin/loadbalancedapp-1653020.html).

This chapter contains the following topics:

Installing ILB

ILB has kernel and userland installations. ILB kernel installation is performed automatically as a part of the Oracle Solaris installation. However, you must perform the ILB userland installation by using the following command:

```
# pkg install ilb
```

Configuring ILB by Using the Command-Line Interface

The ILB CLI is located in the `/usr/sbin/ilbadm` directory. The CLI includes subcommands to configure load-balancing rules, server groups, and health checks. It also includes subcommands to display statistics as well as view configuration details. You must set up user authorization for ILB configuration subcommands except for the view subcommands, such as `ilbadm show-rule`, `ilbadm show-server`, and `ilbadm show-healthcheck`. You must have the `solaris.network.ilb.config` RBAC authorization to execute the ILB configuration subcommands.

- To find out how to assign the authorization to an existing user, see Chapter 3, "Assigning Rights in Oracle Solaris," in "Securing Users and Processes in Oracle Solaris 11.2 ".
- You can also provide the authorization when creating a new user account on the system.

  The following example creates a user `ilbadm` with group ID `10`, user ID `1210`, and with the authorization to administer ILB in the system.

  ```
  # useradd -g 10 -u 1210 -A solaris.network.ilb.config ilbadmin
  ```

  The `useradd` command adds a new user to the `/etc/passwd`, `/etc/shadow`, and `/etc/user_attr` files. The `-A` option assigns the authorization to the user.

The subcommands can be divided into two categories:

- **Configuration subcommands** – These subcommands enable you to perform the following tasks:
  - Create and delete load-balancing rules
  - Enable and disable load-balancing rules
  - Create and delete server groups
  - Add and remove servers from a server group
  - Enable and disable back-end servers
  - Create and delete server health checks for a server group within a load-balancing rule

---

Note - You must have privileges to administer the configuration subcommands. To create the appropriate role and assign the role to a user, see "Creating a Role" in "Securing Users and Processes in Oracle Solaris 11.2 ".

- **View subcommands** – These subcommands enable you to perform the following tasks:
  - View configured load-balancing rules, server groups, and health checks
  - View packet forwarding statistics
  - View the NAT connection table
  - View health check results
  - View the session persistence mapping table

Note - You do not need privileges to administer the view subcommands.

For more information about the `ilbadm` subcommands, refer to the `ilbadm`(1M) man page.

Enabling or Disabling ILB

This section describes how to enable ILB after it has been installed or disable ILB if the ILB services are not required.

▼ How to Enable ILB

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in "Securing Users and Processes in Oracle Solaris 11.2 ".

2. **Enable the appropriate forwarding service, either IPv4 or IPv6 or both of them.**

   This command produces no output when successful.

   ```
   # ipadm set-prop -p forwarding=on ipv4
   # ipadm set-prop -p forwarding=on ipv6
   ```

3. **Enable the ILB service.**

   ```
   # svcadm enable ilb
   ```

4. **Verify that the ILB service is enabled.**

   ```
   # svcs ilb
   ```

   This command displays information about service instances as recorded in the service configuration repository.

▼ How to Disable ILB

When the ILB services are not required you can disable ILB.

1. **Become an administrator.**

For more information, see "Using Your Assigned Administrative Rights" in "Securing Users and Processes in Oracle Solaris 11.2 ".

2. **Disable the ILB service.**

```
# svcadm disable ilb
```

3. **Verify that the ILB service is disabled.**

```
# svcs ilb
```

This command displays information about service instances as recorded in the service configuration repository.

Next Steps After the ILB service is disabled, you must disable IP forwarding if not required.

Managing an ILB

You can set up an ILB by defining the server groups, monitoring the health checks of ILB, and creating ILB rules after you have enabled the ILB.

This section has the following topics:

Defining Server Groups and Back-End Servers in ILB

This section describes how to create an ILB server group and add back-end servers to the server group. When a server is added by using either the `create-servergroup` or the `add-server` subcommands, server IDs are generated by the system. The server IDs are unique within the server group. For information about the `ilbadm` subcommands, see the `ilbadm`(1M) man page.

Creating an ILB Server Group

To create an ILB server group, first identify the servers that are to be included in the server group. Servers can be specified by their host name or IP addresses and optional ports. Then as an administrator, run the following command:

```
# ilbadm create-servergroup -s servers=server1,server2,server3  servergroup
```

Unique server IDs prefixed with a leading underscore (_) are generated for each server added.

Note - A server can have multiple server IDs if it belongs to multiple server groups.

Adding Back-End Servers to an ILB Sever Group

To add a back-end server to a server group, become an administrator and run the following command:

```
# ilbadm add-server -s server=server1[,server2...] servergroup
```

Server specifications must include a host name or IP address and can also include an optional port or a range of ports. Server entries with the same IP address are disallowed within a server group. Unique server IDs prefixed with a leading underscore (_) are generated for each server added.

Note - IPv6 addresses must be enclosed in square brackets.

EXAMPLE 6-1 Creating an ILB Server Group and Adding Back-End Servers

The following example creates a server group called `webgroup` with three back-end servers.

```
# ilbadm create-servergroup -s \
servers=192.168.89.11,192.168.89.12,192.168.89.13 webgroup
# ilbadm show-servergroup
SGNAME         SERVERID            MINPORT MAXPORT IP_ADDRESS
webgroup       _webgroup.0         --      --      192.168.89.11
webgroup       _webgroup.1         --      --      192.168.89.12
webgroup       _webgroup.2         --      --      192.168.89.13
```

The following example creates a server group called `webgroup1` and adds three back-end servers to the server group.

```
# ilbadm create-servergroup webgroup1
# ilbadm add-server -s server=[2001:0db8:7::feed:6]:8080,\
[2001:0db8:7::feed:7]:8080,[2001:0db8:7::feed:8]:8080 webgroup1
```

Enabling or Disabling a Back-End Server in an ILB Server Group

First identify the IP address, host name, or server ID of the back-end server you want to re-enable or disable. You must associate the server group with a rule before the servers in the server group can be enabled or disabled.

A server can have multiple server IDs if it belongs to multiple server groups. You must specify a server ID to re-enable or disable the server for the specific rules that are associated with the server ID.

- To disable an enabled server, type the following command:

  ```
  # ilbadm disable-server server1
  ```

  The selected server, which is enabled, is disabled. The kernel does not forward traffic to this server.
- To re-enable the disabled server, type the following command:

  ```
  # ilbadm enable-server server1
  ```

  The selected server, which is disabled, is re-enabled.
- To display the state of the server, type the following command:

  ```
  # ilbadm show-server [[-p] -o field[,field...]] [rulename]
  ```

Note - A server displays the state as enabled or disabled only when the server group that the server belongs to is associated with a rule.

EXAMPLE 6-2 Disabling and Re-enabling a Back-End Server in an ILB Server Group

In the following example, a server with server ID `_websg.1` is disabled and then re-enabled.

```
# ilbadm enable-server _websg.1
# ilbadm disable-server _websg.1
```

Deleting a Back-End Server From an ILB Server Group

You remove a back-end server from one ILB server group or from all server groups by using the `ilbadm remove-server` command. First, Identify the server ID of the server that you want to remove from a server group.

```
ilbadm show-servergroup -o all
```

The server ID is a unique name for the IP address that is assigned to a system when the server is added to a server group.

Then, delete the server.

```
# ilbadm remove-server -s server=server-ID  server-group
```

If the server is being used by a NAT or half-NAT rule, disable the server by using the `disable-server` subcommand before removal. For more information, see "Enabling or Disabling a Back-End Server in an ILB Server Group" on page 62. When a server is disabled, it enters the connection-draining state. Periodically check the NAT table by using the `ilbadm show-nat` command to see whether the server still has connections. After all the connections are drained (the server is not displayed in the `show-nat` command output), you can remove the server by using the `remove-server` command.

If the `conn-drain` timeout value is set, the connection-draining state will be completed upon conclusion of the timeout period. The default value of `conn-drain` timeout is `0`, which means that the connection-draining waits until a connection is gracefully shut down.

EXAMPLE 6-3 Deleting a Back-End Server From an ILB Server Group

The following example removes the server with server ID `_sg1.2` from server group `sg1`.

```
# ilbadm remove-server -s server=_sg1.2 sg1
```

Deleting ILB Server Groups

This section describes how to delete an ILB server group. You cannot delete a server group that is used by any active rule.

First, display all the available information about server groups.

```
# ilbadm show-servergroup -o all
sgname      serverID       minport     maxport     IP_address
specgroup   _specgroup.0   7001        7001        192.168.68.18
specgroup   _specgroup.1   7001        7001        192.168.68.19
test123     _test123.0     7002        7002        192.168.67.18
test123     _test123.1     7002        7002        192.168.67.19
```

Type the following command:

```
# ilbadm delete-servergroup servergroup
```

If the server group is in use by an active rule, the deletion fails.

The following example removes the server group called `webgroup`.

```
# ilbadm delete-servergroup webgroup
```

Monitoring Health Checks in ILB

ILB provides the following optional types of server health checks:

- Built-in ping probes
- Built-in TCP probes
- Built-in UDP probes
- User-supplied custom tests that can run as health checks

By default, ILB does not perform any health checks. You can specify health checks for each server group when creating a load-balancing rule. You can configure only one health check per load-balancing rule. As long as a virtual service is enabled, the health checks on the server group that is associated with the enabled virtual service start automatically and are repeated periodically. The health checks stop as soon as the virtual service is disabled. The previous health check states are not preserved when the virtual service is re-enabled.

When you specify a TCP, UDP, or custom test probe for running a health check, ILB sends a ping probe, by default, to determine whether the server is reachable before it sends the specified TCP, UDP, or custom test probe to the server. If the ping probe fails, the corresponding server is disabled with the health check status `unreachable`. If the ping probe succeeds but the TCP, UDP, or custom test probe fails, the server is disabled with the health check status `dead`.

You can disable the default ping probe except for the UDP probe. The ping probe is always the default probe for UDP health checks.

Creating a Health Check

You can create a health check and assign the health check to a server group when creating a load-balancing rule. In the following example, two health check objects, `hc1` and `hc-myscript`, are created. The first health check uses the built-in TCP probe. The second health check uses a custom test, `/var/tmp/my-script`.

```
# ilbadm create-healthcheck -h hc-timeout=3,\
hc-count=2,hc-interval=8,hc-test=tcp hc1
# ilbadm create-healthcheck -h hc-timeout=3,\
hc-count=2,hc-interval=8,hc-test=/var/tmp/my-script hc-myscript
```

The arguments are as follows:

`hc-timeout`	Specifies the timeout when the health check is considered to have failed if it does not complete.
`hc-count`	Specifies the number of attempts to run the `hc-test` health check.
`hc-interval`	Specifies the interval between consecutive health checks. To avoid sending probes to all servers at the same time, the actual interval is randomized between `0.5 * hc-interval` and `1.5 * hc-interval`.
`hc-test`	Specifies the type of health check. You can specify the built-in health checks, such as `tcp`, `udp`, and `ping` or an external health check, which has to be specified with the full path name.

Note - The port specification for `hc-test` is specified with the `hc-port` keyword in the `create-rule` subcommand. For more information, see the `ilbadm`(1M) man page.

A user-supplied custom test can be a binary or a script.

- The test can reside anywhere on the system. You must specify the absolute path when using the `create-healthcheck` subcommand.

  When you specify the test (for example, `/var/tmp/my-script`) as part of the health check specification in the `create-rule` subcommand, the `ilbd` daemon forks a process and executes the test as follows:

  ```
  /var/tmp/my-script $1 $2 $3 $4 $5
  ```

  The arguments are as follows:

`$1`	VIP (literal IPv4 or IPv6 address)
`$2`	Server IP (literal IPv4 or IPv6 address)
`$3`	Protocol (UDP, TCP as a string)
`$4`	Numeric port range (the user-specified value for `hc-port`)
`$5`	Maximum time (in seconds) that the test must wait before returning a failure. If the test runs beyond the specified time, it might be stopped, and the test is considered failed. This value is user-defined and specified in `hc-timeout`.

- The user-supplied test, does not have to use all the arguments, but it *must* return one of the following:
  - Round-trip time (RTT) in microseconds
  - 0 if the test does not calculate RTT

- -1 for failure

By default, the health check test runs with the following privileges: `PRIV_PROC_FORK`, `RIV_PROC_EXEC`, and `RIV_NET_ICMPACCESS`.

If a broader privilege set is required, you must implement `setuid` in the test. For more details on the privileges, refer to the `privileges`(5) man page.

Listing Health Checks

To obtain detailed information about configured health checks, issue the following command:

```
# ilbadm show-healthcheck
HCNAME      TIMEOUT COUNT   INTERVAL DEF_PING TEST
hc1         3       2       8        Y        tcp
hc2         3       2       8        N        /var/usr-script
```

Displaying Health Check Results

You use the `ilbadm list-hc-result` command to obtain health check results. If a rule or a health check is not specified, the subcommand lists all the health checks.

The following example displays the health check results associated with a rule called `rule1`.

```
# ilbadm show-hc-result rule1
RULENAME   HCNAME     SERVERID   STATUS   FAIL LAST      NEXT      RTT
rule1      hc1        _sg1:0     dead     10   11:01:19  11:01:27  941
rule1      hc1        _sg1:1     alive    0    11:01:20  11:01:34  1111
```

Note - The `show-hc-result` command displays the health check result only when the rules have associated health checks.

The `LAST` column of the output shows the time a health check was done on a server. The `NEXT` column shows the time at which the next health check will be done.

Deleting a Health Check

You delete a health check by using the `ilbadm delete-healthcheck` command. The following example deletes a health check called `hc1`.

```
# ilbadm delete-healthcheck hc1
```

Configuring ILB Rules

This section describes how you can use the `ilbadm` command to create, delete, and list the load-balancing rules.

ILB Algorithms

ILB algorithms control traffic distribution and provide various characteristics for load distribution and server selection.

ILB provides the following algorithms for the two modes of operation:

- Round-robin – In a round-robin algorithm, the load balancer assigns the requests to a server group on a rotating basis. After a server is assigned a request, the server is moved to the end of the list.
- *src-IP* hash – In the source IP hash method, the load balancer selects a server based on the hash value of the source IP address of the incoming request.
- *src-IP, port* hash – In the source IP, port hash method, the load balancer selects a server based on the hash value of the source IP address and the source port of the incoming request.
- *src-IP, VIP* hash – In the source IP, VIP hash method, the load balancer selects a server based on the hash value of the source IP address and the destination IP address of the incoming request.

Creating an ILB Rule

In ILB, a virtual service is represented by a load-balancing rule and is defined by the following parameters:

- Virtual IP address
- Transport protocol: TCP or UDP
- Port number (or a port range)
- Load-balancing algorithm
- Load-balancing mode (DSR, full-NAT, or half-NAT)
- Server group consisting of a set of back-end servers
- Optional server health checks that can be executed for each server in the server group

- Optional port to use for health checks

Note - You can specify health checks on a particular port or on any port that the `ilbd` daemon randomly selects from the port range for the server.

- Rule name to represent a virtual service

Before you can create a rule, you must do the following:

- Create a server group that includes the appropriate back-end servers. For information, see "Defining Server Groups and Back-End Servers in ILB" on page 60.
- Create a health check to associate the server health check with the rule. For information, see "Creating a Health Check" on page 64.
- Identify the VIP, port, and optional protocol that are to be associated with the rule.
- Identify the operation you want to use (DSR, half-NAT, or full-NAT).
- Identify the load-balancing algorithm to be used. For more information, see "ILB Algorithms" on page 67.

You create an ILB rule by using the `ilbadm create-rule` command. For more information about using the `ilbadm create-rule` command, see the `ilbadm`(1M) man page.

The syntax is as follows:

```
# ilbadm create-rule -e -i vip=IPaddr,port=port,protocol=protocol \
-m lbalg=lb-algorithm,type=topology-type,proxy-src=IPaddr1-IPaddr2,\
pmask=value -h hc-name=hc1 -o servergroup=sg rule1
```

Note - The `-e` option enables the rule that is being created, which would otherwise be disabled by default.

EXAMPLE 6-4 Creating a Full-NAT Rule With Health Check Session Persistence

This example creates a health check called `hc1` and a server group called `sg1`. The server group consists of two servers, each with a range of ports. The last command creates and enables a rule called `rule1` and associates the rule to the server group and the health check. This rule implements the full-NAT mode of operation. Note that the creation of the server group and health check must precede the creation of the rule.

```
# ilbadm create-healthcheck -h hc-test=tcp,hc-timeout=2,\
hc-count=3,hc-interval=10 hc1
# ilbadm create-servergroup -s server=192.168.0.10:6000-6009,192.168.0.11:7000-7009 sg1
# ilbadm create-rule -e -p -i vip=10.0.0.10,port=5000-5009,\
```

```
protocol=tcp -m lbalg=rr,type=NAT,proxy-src=192.168.0.101-192.168.0.104,pmask=24 \
-h hc-name=hc1 -o servergroup=sg1 rule1
```

When you create persistent mapping, subsequent requests for connections, packets, or both, to a virtual service with a matching source IP address of the client are forwarded to the same back-end server. The prefix length in Classless Inter-Domain Routing (CIDR) notation is a value between `0-32` for IPv4 and `0-128` for IPv6.

When creating a half-NAT or a full-NAT rule, specify the value for the `connection-drain` timeout. The default value of `conn-drain` timeout is `0`, which means that connection draining keeps waiting until a connection is gracefully shut down.

Listing ILB Rules

To list the configuration details of a rule, issue the following command. If no rule name is specified, information is provided for all rules.

```
# ilbadm show-rule
RULENAME        STATUS   LBALG           TYPE    PROTOCOL VIP           PORT
rule-http       E        hash-ip-port    NAT     TCP      10.0.0.1      80
rule-dns        D        hash-ip         NAT     UDP      10.0.0.1      53
rule-abc        D        roundrobin      NAT     TCP      2001:db8::1   1024
rule-xyz        E        ip-vip          NAT     TCP      2001:db8::1   2048-2050
```

Deleting an ILB Rule

You use the `ilbadm delete-rule` command to delete a rule. Add the `-a` option to delete all rules. The following example deletes the rule called `rule1`.

```
# ilbadm delete-rule rule1
```

Use Case: Configuring an ILB

This section describes the steps for setting up ILB to use a half-NAT topology to load balance traffic among two servers. See the NAT topology implementation in “ILB Operation Modes” on page 50.

1. Become an administrator.

   For more information, see “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2 ”.

2. Set up the server group in ILB.

   The two servers are `192.168.1.50` and `192.169.1.60`. Create a server group `srvgrp1`, consisting of these two servers by typing the following command. For more information about setting up a server group in ILB, see "Creating an ILB Server Group" on page 60.

   ```
   # ilbadm create-sg -s servers=192.168.1.50,192.168.1.60 srvgrp1
   ```

3. Set up the back-end servers.

   The back-end servers are set up to use ILB as the default router in this scenario. Run the following command on both servers:

   ```
   # route add -p default 192.168.1.21
   ```

   After executing this command, start the server applications on both servers. Assume that it is a TCP application listening on port 5000. For more information on setting up back-end servers, see "Adding Back-End Servers to an ILB Sever Group" on page 61.

4. Set up a simple health check called `hc-srvgrp1`. Create the health check by typing the following command:

   ```
   # ilbadm create-hc -h hc-test=tcp,hc-timeout=3,\
   hc-count=3,hc-interval=60 hc-srvgrp1
   ```

   A simple TCP level health check is used to detect if the server application is reachable. This check is done every 60 seconds. The health check tries at most three times and waits for at most 3 seconds between trials to see whether a server is healthy. If all three trials fail, it marks the server as `dead`. For more information about monitoring and creating health checks, see "Monitoring Health Checks in ILB" on page 64.

5. Set up an ILB rule by typing the following command:

   ```
   # ilbadm create-rule -e -p -i vip=10.0.2.20,port=5000 -m \
   lbalg=rr,type=half-nat,pmask=32 \
   -h hc-name=hc-srvgrp1 -o servergroup=srvgrp1 rule1_rr
   ```

   Persistence (with 32 bits mask) is used in this rule. The load balance algorithm is `round robin`. For information about different ILB algorithms, see "ILB Algorithms" on page 67. The server group `srvgrp1` is used and the health check mechanism used is `hc-srvgrp1`. For more information about creating ILB rules, see "Creating an ILB Rule" on page 67.

Displaying ILB Statistics

This section describes how to use the `ilbadm` command to obtain information such as the printing statistics for a server or statistics for a rule. You can also display NAT table information and the session persistence mapping table.

Displaying Statistical Information

Use the `ilbadm show-statistics` command to view load distribution details as shown in the following example.

```
# ilbadm show-statistics
PKT_P   BYTES_P   PKT_U   BYTES_U   PKT_D   BYTES_D
9       636        0        0        0        0
```

`PKT_P` Packets processed

`BYTES_P` Bytes processed

`PKT_U` Unprocessed packets

`BYTES_U` Unprocessed bytes

`PKT_D` Packets dropped

`BYTES_D` Bytes dropped

Displaying the NAT Connection Table

Use the `ilbadm show-nat` command to display the NAT connection table. Note that the relative positions of elements in consecutive runs of this command are not significant. For example, if you execute the `ilbadm show-nat 10` command twice, you might not see the same 10 items each time you execute, especially on a busy system. If a count value is not specified, the entire NAT connection table is displayed.

EXAMPLE 6-5 NAT Connection Table Entries

The following example displays five entries from the NAT connection table.

```
# ilbadm show-nat 5
UDP: 124.106.235.150.53688 > 85.0.0.1.1024 >>> 82.0.0.39.4127 > 82.0.0.56.1024
UDP: 71.159.95.31.61528 > 85.0.0.1.1024 >>> 82.0.0.39.4146 > 82.0.0.55.1024
UDP: 9.213.106.54.19787 > 85.0.0.1.1024 >>> 82.0.0.40.4114 > 82.0.0.55.1024
UDP: 118.148.25.17.26676 > 85.0.0.1.1024 >>> 82.0.0.40.4112 > 82.0.0.56.1024
UDP: 69.219.132.153.56132 > 85.0.0.1.1024 >>> 82.0.0.39.4134 > 82.0.0.55.1024
```

The format of the entries is as follows:

```
T: IP1 > IP2 >>> IP3 > IP4
```

`T` Transport protocol used in this entry

`IP1`	Client's IP address and port
`IP2`	VIP and port
`IP3`	In half-NAT mode, the client's IP address and port. In full-NAT mode, the client's IP address and port.
`IP4`	Back-end server's IP address and port.

Displaying the Session Persistence Mapping Table

Use the `ilbadm show-persist` command to display the session persistence mapping table.

EXAMPLE 6-6 Session Persistence Mapping Table Entries

The following example displays five entries from the session persistence mapping table.

```
# ilbadm show-persist 5
rule2: 124.106.235.150 --> 82.0.0.56
rule3: 71.159.95.31 --> 82.0.0.55
rule3: 9.213.106.54 --> 82.0.0.55
rule1: 118.148.25.17 --> 82.0.0.56
rule2: 69.219.132.153 --> 82.0.0.55
```

The format of entries is as follows:

```
R: IP1 --> IP2
```

`R`	Rule that the persistence entry is tied to.
`IP1`	Client's IP address.
`IP2`	Back-end server's IP address.

Importing and Exporting Configurations

The export and import subcommands are used to move a configuration from one system to another. For example, if you want to set up a back up of ILB using VRRP to have a active-passive configuration, you can just export the configuration to a file and the import it in the back up system. The `ilbadm export` command exports the current ILB configuration to a user-specified file. This information can then be used as input for the `ilbadm import` command.

The `ilbadm import` command deletes the existing configuration before importing unless specifically instructed to retain it. Omission of a file name instructs the command to read from `stdin` or write to `stdout`.

To export an ILB configuration, use the `export-config` command. The following example exports the current configuration into the file `/var/tmp/ilb_config` in a format suitable for importing by using the `import` command.

```
# ilbadm export-config /var/tmp/ilb_config
```

To import an ILB configuration, use the `import-config` command. The following example reads the contents of the file `/var/tmp/ilb_config` and overrides the existing configuration.

```
# ilbadm import-config /var/tmp/ilb_config
```

♦ ♦ ♦ CHAPTER 7

Configuring ILB for High Availability

This chapter describes the high availability (HA) configuration of ILB by using the VRRP feature. The ILB is configured for high availability by using the DSR and half-NAT topologies. The half-NAT and DSR topologies use the VRRP to protect the virtual IP address of an ILB rule. However, in half-NAT topology, the VRRP is also used to protect the IP address of the primary load balancer that is facing the back end servers. This helps to make sure that when the primary load balancer fails, the back end servers switch to use the standby (passive) load balancer.

For more information about VRRP, see Chapter 3, “Using Virtual Router Redundancy Protocol” and for more information about how to configure and manage ILB, see Chapter 6, “Configuring and Managing the Integrated Load Balancer”.

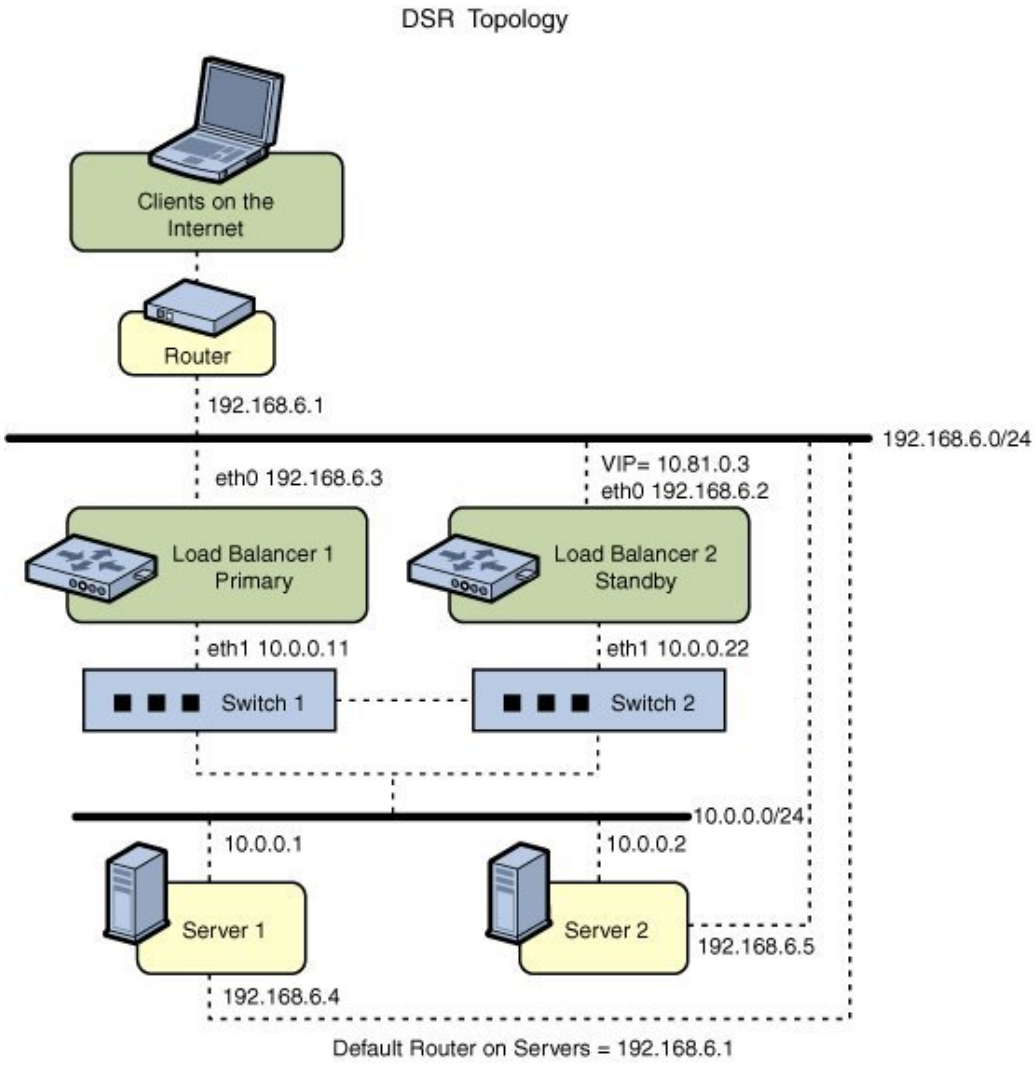
This chapter contains the following topics:

Configuring ILB for High Availability By Using the DSR Topology

You can set up two load balancers, one as the primary load balancer and the other as the standby load balancer. The primary load balancer acts as the master router and standby (passive) load balancer acts as the back up router. The virtual IP address of an ILB rule acts as the virtual router IP address. The VRRP subsystem checks if the primary load balancer has failed. If the primary load balancer fails, the standby load balancer assumes the role of the primary load balancer.

The following figure shows the DSR topology for configuring the ILB connections to achieve HA.

FIGURE 7-1 ILB for HA Configuration by Using DSR Topology



All VIPs on Load Balancers are configured on interfaces facing subnet 192.168.6.0/24.

How to Configure ILB for High Availability by Using the DSR Topology

You can configure both the primary and standby load balancers to have the same configurations for ILB rule, server group, and health check. You can set up both the load balancers to use the VRRP. Also, set the virtual IP address of the rule to be the virtual router address. The VRRP sub-system then ensures that one of the load balancers is always active.

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in "Securing Users and Processes in Oracle Solaris 11.2 ".

2. **Configure both the primary and standby (passive) load balancers to have the same set up.**

```
# ilbadm create-servergroup -s server=10.0.0.1,10.0.0.2 sg1
# ilbadm create-rule -i vip=10.81.0.3,port=9001 \
-m lbalg=hash-ip-port,type=DSR -o servergroup=sg1 rule1
```

3. **Configure Load Balancer 1 to serve as the primary load balancer.**

```
LB1# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB1# vrrpadm create-router -V 1 -A inet -l eth0 -p 255 vrrp1
LB1# ipadm create-ip vnic1
LB1# ipadm create-addr -d -a 10.81.0.3/24 vnic1
```

   The priority of the `vrrp1` router is set to be `255` by using the `vrrpadm` command. The priority value makes the router the master router and hence the active load balancer.

4. **Configure Load Balancer 2 to serve as the standby load balancer.**

```
LB2# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB2# vrrpadm create-router -V 1 -A inet -l eth0 -p 100 vrrp1
LB2# ipadm create-ip vnic1
LB2# ipadm create-addr -d -a 10.81.0.3/24 vnicl
```

   The preceding configuration provides protection against the following failure scenarios:

   - If Load Balancer 1 fails, Load Balancer 2 becomes the primary load balancer. Load balancer 2 then takes over address resolution for the VIP `10.81.0.3` and handles all the packets from clients with the destination IP address `10.81.0.3`.

     When Load Balancer 1 recovers, Load Balancer 2 returns to standby mode.
   - If one or both of Load Balancer 1's interfaces fail, Load Balancer 2 takes over as the primary load balancer. Load Balancer 2 then takes over address resolution for VIP `10.81.0.3` and handles all the packets from clients with the destination IP address `10.81.0.3`.

When both of Load Balancer 1's interfaces are healthy, Load Balancer 2 returns to standby mode.

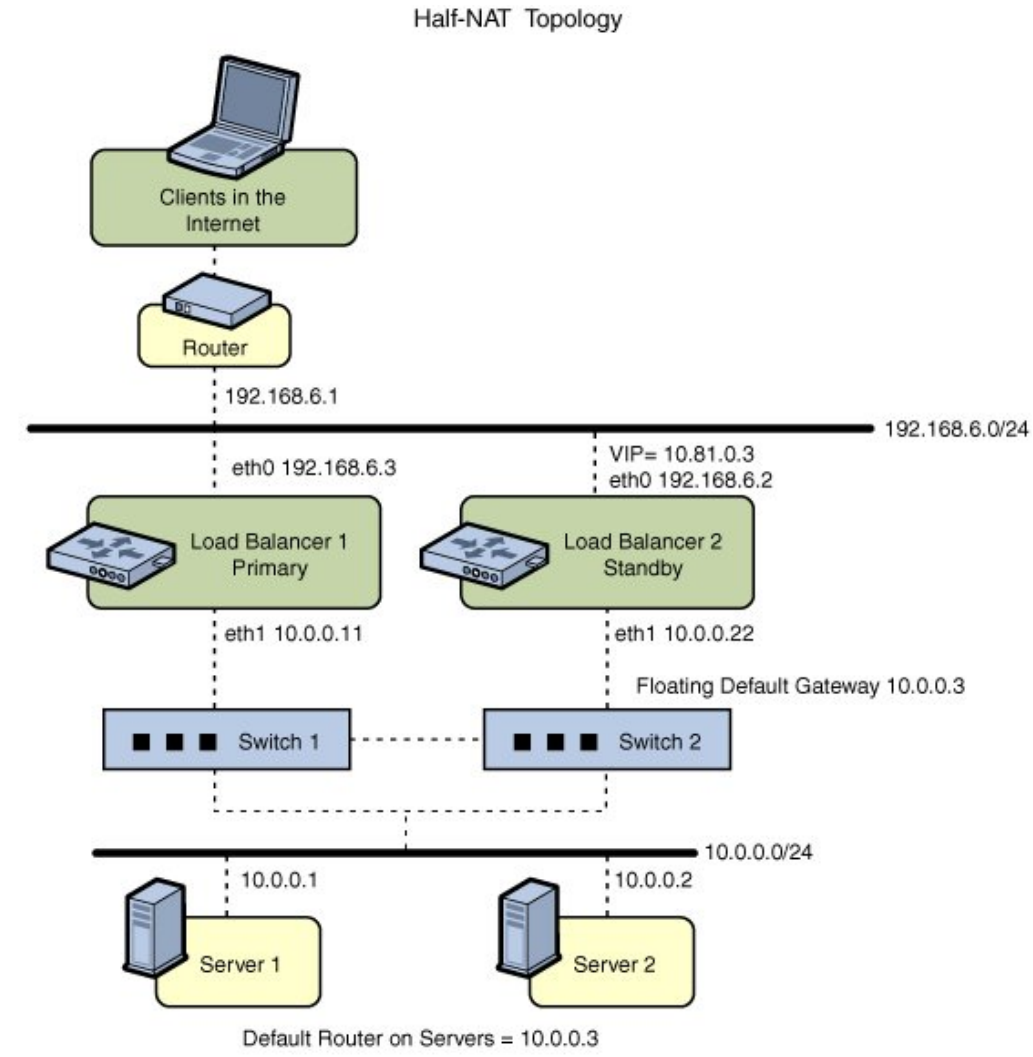
Configuring ILB for High Availability By Using the Half-NAT Topology

This section describes how to set up the ILB connections to achieve HA by using the half-NAT topology. You need to set up two load balancers, one as the primary and the other as the standby. If the primary load balancer fails, the standby load balancer assumes the role of the primary load balancer.

Note - The current implementation of ILB does not synchronize primary and standby load balancers. When the primary load balancer fails and the standby load balancer takes over, the existing connections fail. However, HA without synchronization is still valuable under circumstances when the primary load balancer fails.

The following figure shows the half-NAT topology for configuring the ILB connections to achieve HA.

FIGURE 7-2 ILB for HA Configuration By Using Half-NAT Topology



All VIPs on Load Balancers are configured on interfaces facing subnet 192.168.6.0/24.

How to Configure ILB for High-Availability by Using the Half-NAT Topology

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in "Securing Users and Processes in Oracle Solaris 11.2 ".

2. **Configure both the primary and standby load balancers.**

```
# ilbadm create servergroup -s server=10.0.0.1,10.0.0.2 sg1
# ilbadm create-rule -ep -i vip=10.81.0.3,port=9001-9006,protocol=udp \
-m lbalg=roundrobin,type=HALF-NAT,pmask=24 \
-h hc-name=hc1,hc-port=9006 \
-t conn-drain=70,nat-timeout=70,persist-timeout=70 -o servergroup=sg1 rule1
```

3. **Configure Load Balancer 1 to serve as the primary load balancer.**

```
LB1# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB1# ipadm create-ip vnic1
LB1# ipadm create-addr -d -a 10.81.0.3/24 vnic1
LB1# vrrpadm create-router -V 1 -A inet -l eth0 -p 255 vrrp1
LB1# dladm create-vnic -m vrrp -V 2 -A inet -l eth1 vnic2
LB1# ipadm create-ip vnic2
LB1# ipadm create-addr -d -a 10.0.0.3/24 vnic2
LB1# vrrpadm create-router -V 2 -A inet -l eth1 -p 255 vrrp2
```

4. **Configure Load Balancer 2 to serve as the standby load balancer.**

```
LB2# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB2# ipadm create-ip vnic1
LB2# ipadm create-addr -d -a 10.81.0.3/24 vnic1
LB2# vrrpadm create-router -V 1 -A inet -l eth0 -p 100 vrrp1
LB2# dladm create-vnic -m vrrp -V 2 -A inet -l eth1 vnic2
LB2# ipadm create-ip vnic2
LB2# ipadm create-addr -d -a 10.0.0.3/24 vnic2
LB2# vrrpadm create-router -V 2 -A inet -l eth1 -p 100 vrrp2
```

5. **Add the IP address for the floating default gateway to both servers.**

```
# route add default 10.0.0.3
```

   This configuration provides protection against the following failure scenarios:

   - If Load Balancer 1 fails, Load Balancer 2 becomes the primary load balancer. Load balancer 2 then takes over address resolution for the VIP `10.81.0.3` and handles all the packets from clients with the destination IP address `10.81.0.3`. Load balancer 2 also handles all the packets that are sent to the floating gateway address `10.0.0.3`.

     When Load Balancer 1 recovers, Load Balancer 2 returns to the standby mode.

- If one or both of Load Balancer 1's interfaces fail, Load Balancer 2 takes over as primary load balancer. Load Balancer 2 then takes over address resolution for VIP `10.81.0.3` and handles all packets from clients with the destination IP address `10.81.0.3`. Load balancer 2 also handles all the packets that are sent to the floating gateway address `10.0.0.3`.

  When both of Load Balancer 1's interfaces are healthy, Load Balancer 2 returns to standby mode.

Index

I

L

T

V