Managing Network Virtualization and Network Resources in Oracle® Solaris 11.2
Ce logiciel et la documentation qui l’accompagne sont protégés par les lois sur la propriété intellectuelle. Ils sont concédés sous licence et soumis à des restrictions d’utilisation et de divulgation. Sauf disposition de votre contrat de licence ou de la loi, vous ne pouvez pas copier, reproduire, traduire, diffuser, modifier, breveter, transmettre, distribuer, exposer, exécuter, publier ou afficher le logiciel, même partiellement, sous quelque forme et par quelque procédé que ce soit. Par ailleurs, il est interdit de procéder à toute ingénierie inverse du logiciel, de le désassembler ou de le décompiler, excepté à des fins d’interopérabilité avec des logiciels tiers ou tel que prescrit par la loi.

Les informations fournies dans ce document sont susceptibles de modification sans préavis. Par ailleurs, Oracle Corporation ne garantit pas qu’elles soient exemptes d’erreurs et vous invite, le cas échéant, à lui en faire part par écrit.

Si ce logiciel, ou la documentation qui l’accompagne, est concédé sous licence au Gouvernement des États-Unis, ou à toute entité qui délivre la licence de ce logiciel ou l’utilise pour le compte du Gouvernement des États-Unis, la notice suivante s’applique:

U.S. GOVERNMENT END USERS. Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

Ce logiciel ou matériau a été développé pour un usage général dans le cadre d’applications de gestion des informations. Ce logiciel ou matériel n’est pas conçu ni n’est destiné à être utilisé dans des applications à risque, notamment dans des applications pouvant causer des dommages corporels. Si vous utilisez ce logiciel ou matériel dans le cadre d’applications dangereuses, il est de votre responsabilité de prendre toutes les mesures de secours, de sauvegarde, de redondance et autres mesures nécessaires à son utilisation dans des conditions optimales de sécurité. Oracle Corporation et ses affiliés déclinent toute responsabilité quant aux dommages causés par l’utilisation de ce logiciel ou matériel pour ce type d’applications.

Oracle et Java sont des marques déposées d’Oracle Corporation et/ou de ses affiliés. Tout autre nom mentionné peut correspondre à des marques appartenant à d’autres propriétaires qu’Oracle.

Intel et Intel Xeon sont des marques ou des marques déposées d’Intel Corporation. Toutes les marques SPARC sont utilisées sous licence et sont des marques ou des marques déposées de SPARC International, Inc. AMD, Opteron, le logo AMD et le logo AMD Opteron sont des marques ou des marques déposées d’Advanced Micro Devices. UNIX est une marque déposée d’The Open Group.

Ce logiciel ou matériau et la documentation qui l’accompagne peuvent fournir des informations ou des liens donnant accès à des contenus, des produits et des services émanant de tiers. Oracle Corporation et ses affiliés déclinent toute responsabilité ou garantie expresse quant aux contenus, produits ou services émanant de tiers. En aucun cas, Oracle Corporation et ses affiliés ne sauraient être tenus pour responsables des pertes subies, des coûts occasionnés ou des dommages causés par l’accès à des contenus, produits ou services tiers, ou à leur utilisation.
# Contents

**Using This Documentation** ................................................................................................................. 9

**1 Introduction to Network Virtualization and Network Resource Management** ............................................ 11

What's New in Managing Network Virtualization and Network Resources in Oracle Solaris 11.2 ................................................................. 11


Overview of Virtual Networks ...................................................................................................................... 13

Components of a Virtual Network ................................................................................................................ 14

How a Virtual Network Works ...................................................................................................................... 17

Using Virtual Extensible Local Area Network ............................................................................................ 18

Using Edge Virtual Bridging ........................................................................................................................ 18

Who Should Implement Virtual Networks? .................................................................................................. 18

Overview of Network Resource Management ............................................................................................ 19

Network Resource Management by Using Datalink Properties ............................................................... 20

Network Resource Management by Using Flows ....................................................................................... 20

Benefits of Network Resource Management ............................................................................................. 20

**2 Creating and Managing Virtual Networks** .......................................................................................... 23

Configuring the Components of a Virtual Network ..................................................................................... 23

Commands for Configuring the Components of a Virtual Network ............................................................ 23

▼ How to Configure VNICs and Etherstub .................................................................................................... 24

▼ How to Configure VNICs With VLAN IDs .............................................................................................. 26

Building Virtual Networks .......................................................................................................................... 28

▼ How to Configure a Zone for the Virtual Network .................................................................................. 28

▼ How to Reconfigure a Zone to Use a VNIC ............................................................................................ 30

▼ How to Temporarily Create VNICs in Zones ......................................................................................... 33

▼ How to Configure a Private Virtual Network ........................................................................................... 34

Managing VNICs ......................................................................................................................................... 37

Displaying VNICs ......................................................................................................................................... 37

Modifying the VLAN IDs of VNICs ................................................................................................................ 41
### Contents

- Modifying VNIC MAC Addresses ............................................................... 42
- Migrating VNICs .................................................................................. 44
- Deleting VNICs .................................................................................. 46
- Using Single Root I/O Virtualization With VNICs ........................................ 48
  - Enabling the SR-IOV Mode of Datalinks .............................................. 49
  - Creating VF VNICs ......................................................................... 50
  - Migrating VF VNICs ....................................................................... 51
  - Displaying VFs Information ............................................................. 52

3 Configuring Virtual Networks by Using Virtual Extensible Local Area Networks .......................................................... 55
  - Overview of VXLANs ......................................................................... 55
  - Advantages of Using VXLANs ............................................................ 56
  - VXLAN Naming Convention ............................................................... 57
  - VXLAN Topology ............................................................................... 57
  - Using VXLAN With Zones ................................................................. 59
  - Planning a VXLAN Configuration ....................................................... 61
  - VXLAN Requirements ...................................................................... 61
  - Configuring a VXLAN ................................................................. 62
    - How to Configure a VXLAN ............................................................ 62
  - Displaying VXLAN Information ........................................................ 66
  - Deleting a VXLAN ........................................................................... 66
  - Assigning a VXLAN to a Zone ........................................................... 67
    - How to Assign a VXLAN to a Zone .................................................. 67
  - Use Case: Configuring a VXLAN Over a Link Aggregation ................. 68

4 Administering Server-Network Edge Virtualization by Using Edge Virtual Bridging ....................................................... 73
  - EVB Support in Server-Network Edge Virtualization ......................... 74
    - Reflective Relay .............................................................................. 74
    - Automated VNIC Configuration in the Network ............................... 74
  - Improving Network and Server Efficiency by Using EVB .................. 75
  - Installing EVB .................................................................................. 78
    - How to Install EVB ...................................................................... 78
  - Controlling Switching Between VMs Over the Same Physical Port ....... 79
    - Enabling the VMs to Communicate Through an External Switch .......... 79
    - Using LLDP to Manage the Communication Between VMs ............... 82
  - Exchanging VNIC Information by Using VDP ................................... 84
    - How VDP Exchanges VNIC Information ............................................ 85
5 About Elastic Virtual Switches .......................................................... 93
   Overview of the Elastic Virtual Switch (EVS) Feature ................................. 93
   Virtual Switches in Oracle Solaris ............................................................ 94
   What Is the Oracle Solaris Elastic Virtual Switch Feature? ......................... 95
   Benefits of Using EVS ........................................................................... 97
   Elastic Virtual Switch Resources ............................................................. 98
   Namespace Management in EVS ................................................................ 100
   EVS Components ......................................................................................... 100
   EVS Manager ...................................................................................... 101
   EVS Controller .................................................................................... 102
   EVS Clients ........................................................................................ 104
   EVS Nodes ......................................................................................... 104
   EVS Administrative Commands ...................................................................... 104
      evsadm Command ................................................................................. 105
      evsstat Command ............................................................................... 107
      dladm Command .................................................................................. 107
      zonecfg Command ............................................................................... 107
      Restrictions for Administering VNICs Connected to an Elastic Virtual Switch ......................................................................................................................... 108
      Automatically Generated VXLAN Datalinks ........................................ 108
   Mandatory Packages for Using EVS .......................................................... 109
   How EVS Works With Zones ......................................................................... 109
   Security Requirements for Using EVS .......................................................... 110

6 Administering Elastic Virtual Switches .................................................. 113
   EVS Administration Tasks ........................................................................ 113
   Planning an Elastic Virtual Switch Configuration ......................................... 114
   Creating and Administering an EVS Controller ......................................... 114
      Mandatory Packages for an EVS Controller ......................................... 115
      Commands for Configuring an EVS Controller ....................................... 116
      Configuring an EVS Controller ............................................................... 118
   Configuring Elastic Virtual Switches .......................................................... 125
7 Managing Network Resources ................................................................. 157
  Managing Network Resources by Using Datalink Properties ...................... 157
  Commands for Allocating Resources in Datalinks ........................................ 158
  Managing NIC Rings ......................................................................................... 158
  Allocating Rings in MAC Clients ................................................................. 159
  Allocating Rings in VLANs ........................................................................... 159
  Commands for Configuring Rings ................................................................. 160
  Displaying Ring Use and Ring Assignments on a Datalink ......................... 161
  Configuring Clients and Allocating Rings .................................................... 162
  Managing Pools and CPUs ............................................................................ 167
  Working With Pools and CPUs ..................................................................... 168
  Configuring a CPU Pool for a Datalink ......................................................... 170
  Allocating CPUs to a Datalink ..................................................................... 171
  Managing Network Resources by Using Flows .............................................. 172
  Commands for Resource Allocation in Flows ................................................. 173
  Configuring Flows ........................................................................................ 174
  Use Case: Managing Network Resources by Setting Datalink and Flow Properties ................................................................. 176

8 Monitoring Network Traffic and Resource Usage ........................................... 181
  Commands for Monitoring Network Traffic Statistics ...................................... 183
  Displaying Network Traffic Statistics of Links .............................................. 183
  Displaying Network Traffic Statistics of Network Devices ............................. 184
  Displaying Network Traffic Statistics of Datalinks ....................................... 187
  Displaying Network Traffic Statistics of Link Aggregations .......................... 188
Using This Documentation

- **Overview** – Describes how to configure the Oracle Solaris virtual networking features and monitor network traffic. It also describes the different processes that are used to manage network resources.
- **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](http://www.oracle.com/pls/topic/lookup?ctx=E36784).

**Access to Oracle Support**


**Feedback**

Provide feedback about this documentation at [http://www.oracle.com/goto/docfeedback](http://www.oracle.com/goto/docfeedback).
Introduction to Network Virtualization and Network Resource Management

This chapter provides an overview of network virtualization and network resource management in Oracle Solaris.

This chapter contains the following topics:

- “What’s New in Managing Network Virtualization and Network Resources in Oracle Solaris 11.2” on page 11
- “Overview of Virtual Networks” on page 13
- “Overview of Network Resource Management” on page 19

What’s New in Managing Network Virtualization and Network Resources in Oracle Solaris 11.2

For existing customers, this section highlights the key changes in this release:

- **Oracle Solaris Elastic Virtual Switch (EVS) feature** – Oracle Solaris network virtualization capabilities are expanded to enable managing virtual switches directly. The Oracle Solaris Elastic Virtual Switch feature provides virtual networking infrastructure within a data center or a multitenant cloud environment to interconnect virtual machines that reside on multiple servers. Virtual machines connected to the same elastic virtual switch can communicate with each other. EVS enables centralized management of virtual switches on multiple hosts and hence VNICs connected to the elastic virtual switch. For more information, see Chapter 5, “About Elastic Virtual Switches”. For more information about how to administer elastic virtual switches, see Chapter 6, “Administering Elastic Virtual Switches”.

- **Support for Virtual Extensible Local Area Network (VXLAN)** – Oracle Solaris supports the VXLAN technology that provides isolation methods to support virtualization in large data centers. It enables migration of virtual machines between physical servers belonging to different Layer 2 networks in a cloud environment. For more information, see Chapter 3, “Configuring Virtual Networks by Using Virtual Extensible Local Area Networks”.

Support for Single Root I/O Virtualization (SR-IOV) – This feature enables the creation of a virtual function (VF) based VNIC on a network device that supports SR-IOV. For more information, see “Using Single Root I/O Virtualization With VNICs” on page 48.

Temporarily creating virtual network interface cards (VNICs) in zones – You can create temporary VNICs directly in a non-global zone from a global zone. You must use the -t option with the dladm create-vnic command to create a temporary VNIC. Temporary VNICs persist until the next reboot of the zone. In addition to temporarily creating VNICs, you can also temporarily create VLANs and IP over InfiniBand (IPoIB) partitions in zones. For more information, see “How to Temporarily Create VNICs in Zones” on page 33.

Communication between VNICs by using an external switch – With the Oracle Solaris 11.2 reflective relay feature, the traffic between the local Oracle Solaris zones or Oracle VMs sharing the same underlying physical NIC can be forced to be always sent to the physical network instead of the host virtual switch. Communication between these entities is subject to policies configured on the external switch that support the reflective relay feature. For more information, see “Controlling Switching Between VMs Over the Same Physical Port” on page 79.

Enhancements for monitoring network traffic statistics – You can use the enhanced dlstat and flowstat commands to effectively monitor network traffic statistics. The enhancements for monitoring network traffic statistics are:
- Network traffic statistics are displayed with the current time.
- Network traffic statistics are displayed and refreshed based on the specified interval and count values.
- Network traffic statistics are displayed in rates per second based on the specified interval value.


Flows configuration changes – You can use the enhanced flowadm add-flow command to configure flows on a datalink based on an additional number of attributes and their newer combinations, which help you to selectively organize network packets that are received from different ports, transport protocols, and IP addresses. For more information, see “Managing Network Resources by Using Flows” on page 172.

In addition to the bandwidth property used for managing flows, you can use the flowadm set-flowprop command to set the priority property for flows. You can prioritize flows by setting the priority property. The new read-only property, hwflow, enables you to see how a flow is instantiated. For more information, see “Configuring Flows” on page 174.

Displaying multiple MAC addresses associated with VNICs – You can use the enhanced dladm show-vnic command to display multiple MAC addresses associated with VNICs. For more information, see “Displaying VNICs With Multiple MAC Addresses” on page 38.

System-created VNICs – In addition to the VNICs that you can create by using the dladm create-vnic command, the system also creates VNICs, which are called system-created
What Is Network Virtualization and Network Resource Management?

Network virtualization is the process of combining hardware network resources and software network resources into a single administrative unit. This single administrative unit is known as a virtual network.

Network resource management is the process of managing and allocating resources for networking processes. You can assign the resources differently depending on the amount of network traffic that is being processed. By managing and allocating resources according to the actual need, you increase the system's efficiency when processing packets.

Network virtualization is optimized when used effectively with network resource management. You can provide systems and users with controlled sharing of the hardware and software networking resources, thus increasing the efficiency of virtual networking processes. Network virtualization with network resource management helps you to manage flow control, improve system performance, and configure the network utilization needed to achieve operating system (OS) virtualization, utility computing, and server consolidation.

Overview of Virtual Networks

A virtual network is a network that emulates a physical network and is a combination of hardware and software network resources. A virtual network is the end product of network virtualization.

Virtual networks are classified into two broad types: external and internal.
External virtual networks consist of several local networks that are administered by software as a single entity. The building blocks of classic external virtual networks are switch hardware and virtual local area network (VLAN) software technology. Examples of external virtual networks include large corporate networks and data centers. For more information about VLANs, see Chapter 3, “Configuring Virtual Networks by Using Virtual Local Area Networks,” in “Managing Network Datalinks in Oracle Solaris 11.2”.

Internal virtual networks consist of one system using virtual machines or zones whose network interfaces are configured over at least one physical network interface card (NIC). Those network interfaces are called virtual network interface cards or virtual NICs (VNICs). These virtual machines or zones can communicate with each other as though they were on the same local network, effectively becoming a virtual network on a single host. The chapters in this document focus on the internal virtual network.

A special type of internal virtual network is the private virtual network. Private virtual networks are different from virtual private networks (VPNs). VPN creates a secure point-to-point link between two endpoint systems. The private virtual network is a virtual network on a system that cannot be accessed by external networks. The isolation of this internal network from other external networks is achieved by configuring VNICs over a pseudo NIC called an etherstub. For more information, see “Etherstub” on page 15.

Components of a Virtual Network

A virtual network has the following components:

- Virtual Network Interface Card (VNIC)
- Virtual switch
- Etherstub
- Zone

Virtual Network Interface Card (VNIC)

A VNIC is an L2 entity or virtual network device that behaves just like a physical NIC when configured. You configure a VNIC over an underlying datalink to share it between multiple zones or VMs. In addition, the system’s resources treat VNICs as if they were physical NICs. All physical Ethernet interfaces support the creation of VNICs. For more information about how to configure a VNIC, see “How to Configure VNICs and Etherstubs” on page 24.

A VNIC has an automatically generated MAC address. Depending on the network interface in use, you can assign a MAC address to a VNIC other than the automatically generated MAC address. For more information, see “Modifying VNIC MAC Addresses” on page 42.
Virtual Switch

A *virtual switch* is an entity that facilitates communication between virtual machines (VMs). The virtual switch loops traffic between virtual machines (inter-VM traffic) within the physical machine and does not send this traffic out on the wire. A virtual switch is implicitly created whenever you create a VNIC on top of an underlying datalink. The VNICs configured with the VMs need to be on the same VLAN or VXLAN for inter-VM communication. Virtual switches can be managed by EVS. For information about EVS, see Chapter 5, “About Elastic Virtual Switches”.

As per Ethernet design, if a switch port receives an outgoing packet from the host connected to that port, that packet cannot go to a destination on the same port. This Ethernet design is a limitation for systems that are configured with virtual networks because the virtual networks share the same NIC. This Ethernet design limitation is overcome by using the virtual switches, which enable VMs to communicate with one another.

In certain cases, communication between VMs in a system might require the use of a switch. For example, communication between VMs might need to be subjected to access control lists (ACLs) that are configured on the switch. By default, a switch cannot send packets on the same port where the packets are received. Therefore, reflective relay is enabled on the switch for communication between VMs that use a switch. Reflective relay enables the switch to forward the packets on the same port where the packets are received. For more information, see Reflective Relay.

Etherstub

An *etherstub* is a pseudo Ethernet NIC that is configured at the datalink layer (L2) of the Oracle Solaris network stack. You can create VNICs over an etherstub instead of over a physical NIC. With etherstubs, you can construct a private virtual network that is isolated both from the other virtual networks on the system and from the external network. For example, etherstubs can be used to create a network environment whose access is limited only to your developers and not to the entire network.

The following figure shows a private virtual network based on the etherstub.
This figure shows etherstub0 over which VNIC1, VNIC2, and VNIC3 are configured. Each VNIC is assigned to a zone. The private virtual network based on the etherstub cannot be accessed by external networks. For more information, see “How to Configure a Private Virtual Network” on page 34.

**Zone**

A *zone* is a virtualized operating system environment created within a single instance of the Oracle Solaris operating system. Zones provide an isolated and protected environment for running applications. Etherstubs and VNICS are only a part of the virtualization features of Oracle Solaris. By assigning VNICS or etherstubs for use by Oracle Solaris zones, you can create a network within a single system. For more information about zones, see “Introduction to Oracle Solaris Zones”.
How a Virtual Network Works

The following figure shows the working of a virtual network and its components in a system.

**FIGURE 1-2  Working of a Virtual Network**

The figure shows a single system with one NIC. The NIC is configured with three VNICS. Each VNIC is assigned to a zone. Zone 1, Zone 2, and Zone 3 are the three zones configured for use in the system. The zones communicate with each other and with the external network by using their respective VNICS. The three VNICS connect to the underlying physical NIC through the
virtual switch. The function of a virtual switch is equivalent to the function of a physical switch as both provide connectivity to the systems.

When a virtual network is configured, a zone sends traffic to an external host in the same way as a system without a virtual network. Traffic flows from the zone, through the VNIC to the virtual switch, and then to the physical interface, which sends the data to the network.

The zones can also exchange traffic with one another inside the system if all the VNICs configured to the zones are part of the same VLAN. For example, packets pass from Zone 1 through its dedicated VNIC 1. The traffic then flows through the virtual switch to VNIC 3. VNIC 3 then passes the traffic to Zone 3. The traffic never leaves the system, and therefore never violates the Ethernet restrictions.

Alternatively, you can create a virtual network based on the etherstub. Etherstubs are entirely software based and do not require a network interface as the basis for the virtual network.

Oracle also provides the Oracle Enterprise Manager Ops Center for managing some aspects of network virtualization, for example, the ability to create virtual networks inside a virtual data center. For more information about the Oracle Enterprise Manager Ops Center, see the documentation library at http://www.oracle.com/pls/topic/lookup?ctx=oc122&id=OPCCM.

Using Virtual Extensible Local Area Network

Virtual extensible Local Area Network (VXLAN) is a network virtualization technology that provides scalability and network isolation for virtual networks. VXLAN addresses the 4K limitation of virtual local area network (VLAN) and also reduces the demand of virtualization on physical infrastructure such as switches. For more information, see Chapter 3, “Configuring Virtual Networks by Using Virtual Extensible Local Area Networks”.

Using Edge Virtual Bridging

Edge virtual bridging (EVB) enables a host to exchange information related to virtual links on a system with an external switch. EVB is used to exchange information about all the virtual links behind a port where as DCB is used to exchange information about the port. For more information about EVB, see Chapter 4, “Administering Server-Network Edge Virtualization by Using Edge Virtual Bridging”.

Who Should Implement Virtual Networks?

If you need to consolidate resources on Oracle's Sun servers, consider building virtual networks. You can achieve better utilization of the available resources by consolidating various
applications on a few servers. You can then use virtual networking to provide connectivity between the applications.

Consolidators at ISPs, telecommunication companies, and large financial institutions can consolidate their servers because of the following hardware resources:

- Powerful NICs with substantial bandwidth and hardware support, for example support for NIC rings and virtual functions (VFs)
- Powerful physical machines with greater random access memory (RAM) and central processing units (CPUs)

You can replace many systems with a single system that has multiple zones or virtual machines, without significantly losing separation, security, and flexibility.

For a demonstration of the benefits of network virtualization, see Consolidating the Data Center With Network Virtualization (http://download.oracle.com/otndocs/tech/OTN_Demos/data-center-consolidation.html).

**Overview of Network Resource Management**

In Oracle Solaris, quality of service (QoS) is obtained more easily and dynamically by managing network resources. Network resource management is comparable to creating dedicated lanes for traffic. When you combine different resources to provide to the specific types of network packets, those resources form a network lane for those packets. Resources can be assigned differently for each network lane. For example, you can allocate more resources to a lane where network traffic is the heaviest. By configuring network lanes where resources are distributed according to the actual need, you increase the system’s efficiency in processing network packets. For more information about network lanes, see “Overview of Monitoring Network Traffic Statistics of Datalinks and Flows” on page 181.

The following network resources are used to increase the system's efficiency in processing packets:

- **Bandwidth** – You can limit the bandwidth of the datalink according to the actual need of the networking processes supported by the datalink.
- **Priority** – You can prioritize the order in which the packets are processed. The latency is reduced for the packets with higher priority because they are processed ahead of the other packets.
- **NIC rings** – If a NIC supports ring allocation, its transmit and receive rings can be dedicated for use by datalinks. For more information, see “Managing NIC Rings” on page 158.
- **CPU pools** – Pools of CPUs are created and associated with specific zones. These pools can be further assigned to datalinks to manage the network processes of their associated zones. For more information, see “Managing Pools and CPUs” on page 167.
- **CPUs** – On a system with multiple CPUs, you can dedicate a given number of CPUs for specific network processing. For more information, see “Managing Pools and CPUs” on page 167.

Network resources on a system can be managed by using either datalink properties or flows.

**Network Resource Management by Using Datalink Properties**

Managing network resources by using datalink properties improves the system's efficiency in processing packets. You can set datalink properties when you create the link. Alternatively, you can set datalink properties later, for example, after studying resource usage over time and determining how to better allocate the resource. By setting datalink properties that pertain to network resources, you can decide how much of a given resource can be used for the networking processes. The procedures for allocating resources apply to the virtual network as well as the physical network. For more information about datalink properties and how to configure them, see “Managing Network Resources by Using Datalink Properties” on page 157.

**Network Resource Management by Using Flows**

A flow is a customized way of categorizing network packets based on a single attribute or a combination of attributes. The attributes that serve as the basis for creating flows are derived from the information in a network packet's header. After setting datalink properties for network resource management, flows can be used to further control how resources are used to process network packets. Flows alone can also be used to manage network resources without setting datalink properties.

Using flows for managing resources involves the following steps:

1. Creating a flow based on a single attribute or a combination of attributes.
2. Customizing a flow's use of resources by setting properties that pertain to network resources. Currently, only bandwidth and priority properties can be set to a flow.

For more information about configuring flows, see “Managing Network Resources by Using Flows” on page 172.

**Benefits of Network Resource Management**

By using network resource management, you can isolate, prioritize, track, and control data traffic on an individual system without the complex QoS rule definitions.
Network resource management is helpful for the following tasks:

- Provisioning the network
- Establishing service-level agreements
- Billing clients
- Diagnosing security problems
CHAPTER 2

Creating and Managing Virtual Networks

This chapter describes tasks for configuring the components of a virtual network, building virtual networks, and managing VNICs in a single system. For an introduction to virtual networks, see Chapter 1, “Introduction to Network Virtualization and Network Resource Management”.

This chapter contains the following topics:

- “Configuring the Components of a Virtual Network” on page 23
- “Building Virtual Networks” on page 28
- “Managing VNICs” on page 37
- “Using Single Root I/O Virtualization With VNICs” on page 48

Configuring the Components of a Virtual Network

In Oracle Solaris, VNICs and etherstubs are the basic components of a virtual network. This section describes the steps to configure these components in preparation for building the virtual network. For a description of these components, see “Components of a Virtual Network” on page 14.

Commands for Configuring the Components of a Virtual Network

To create VNICs, use the `dladm create-vnic` command.

```
# dladm create-vnic -l link [-v vid] VNIC
```

`link` The name of the link over which the VNIC is configured.

`vid` The VLAN ID of the VNIC if you want to create the VNIC as a VLAN. To configure a VNIC with a VLAN ID, see “How to Configure VNICs With VLAN IDs” on page 26. For more information about VLANs, see Chapter 3, “Configuring Virtual Networks by Using Virtual Local...
How to Configure VNICs and Etherstubs

Area Networks,” in “Managing Network Datalinks in Oracle Solaris 11.2”.

VNIC

The name of the VNIC. For the guidelines about how to create customized names, see “Rules for Valid Link Names” in “Configuring and Administering Network Components in Oracle Solaris 11.2”.

You can configure other properties for a VNIC, such as MAC addresses and CPUs to be associated with the VNIC. For a list of these properties, see the `dladm(1M)` man page. Certain property modifications work only with VNICs. For example, with the `dladm create-vnic` command, you can configure a MAC address as well as assign a VLAN ID to create a VNIC as a VLAN. However, you cannot configure a MAC address directly for a VLAN by using the `dladm create-vlan` command.

You can create only one VNIC at a time over a datalink. Like datalinks, VNICs have link properties that you can further configure as needed. For information about the different types of link properties, see “Network Resource Management by Using Datalink Properties” on page 20.

In addition to the VNICs that you can create by using the `dladm create-vnic` command, the system also creates VNICs known as system-created VNICs that help in virtual network I/O for Oracle VM Server for SPARC vnet. The system-created VNICs follow the naming convention `<entity>-<name>`, where entity refers to the system entity that created the VNIC and name refers to the VNIC name within the system entity. The user-created VNIC name cannot contain a hyphen (-). Only a system-created VNIC contains a hyphen (-), which helps you to differentiate between a system-created VNIC and a user-created VNIC. You cannot modify, rename, plumb, or delete system-created VNICs. For more information, see Oracle VM Server for SPARC 3.1 Administration Guide.

You can use the `dlstat` and `snoop` commands to monitor network traffic on system-created VNICs. You can also create flows over system-created VNICs by using the `flowadm` command. Flows help you not only manage network resources but also to monitor network traffic statistics. You can monitor network traffic statistics on flows by using the `flowstat` command. For more information about flows, see “Configuring Flows” on page 174.

To create etherstubs, use the `dladm create-etherstub` command.

```
# dladm create-etherstub ethersub
```

where `ethersub` is the name of the etherstub that you want to create.

▼ How to Configure VNICs and Etherstubs

The VNIC connects the virtual network to the external network. The VNIC also enables the zones to communicate with one another through the virtual switch that is automatically created with the VNIC. For a virtual network to host traffic internally between zones, an external LAN,
and the Internet, each zone must have its own VNIC. Therefore, you must repeat this procedure as many times as the number of zones that belong to the virtual network.

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

2. **(Optional) Create an etherstub.**
   
   ```
   # dladm create-etherstub etherstub
   ```
   
   Perform this step only if you are creating a private virtual network. For a description of a private virtual network, see “Overview of Virtual Networks” on page 13. For more information about how to configure a private virtual network, see “How to Configure a Private Virtual Network” on page 34.

   Like a datalink, you can name the etherstub in any way that is meaningful to your network setup. For the guidelines about how to create customized names, see “Rules for Valid Link Names” in “Configuring and Administering Network Components in Oracle Solaris 11.2 ”.

3. **Create a VNIC.**
   
   ```
   # dladm create-vnic -l link [-v vid] VNIC
   ```
   
   If you are creating the VNIC for a private virtual network, then substitute `etherstub` for `link`. Include the `-v` option in the command only if you are creating the VNIC as a VLAN. For more information about creating the VNIC as a VLAN, see “How to Configure VNICs With VLAN IDs” on page 26.

4. **Create an IP interface over the VNIC.**
   
   ```
   # ipadm create-ip interface
   ```
   
   `interface` The VNIC that you created in the previous step.

5. **Assign a static IP address to the VNIC interface.**
   
   ```
   # ipadm create-addr -a address interface
   ```
   
   `-a address` Specifies the IP address, which can be in Classless Inter-Domain Routing (CIDR) notation.

   The static IP address can be either IPv4 or IPv6 addresses. For more information, see “How to Configure an IPv4 Interface” in “Configuring and Administering Network Components in Oracle Solaris 11.2 ”.

6. **(Optional) Verify the VNIC that has been created.**
   
   ```
   # dladm show-link
   ```
How to Configure VNICs With VLAN IDs

Example 2-1 Configuring a VNIC

This example shows how to configure vnic1 over the datalink net0.

```
# dladm create-vnic -l net0 vnic1
# ipadm create-ip vnic1
# ipadm create-addr -a 192.168.0.10/24 vnic1
# dladm show-link
```

Link          Class     MTU    State    Over
---
net0          phys      1500   up       --
vnic1         vnic      1500   up       net0

How to Configure VNICs With VLAN IDs

You can configure VNICs with VLAN IDs to host VLAN traffic. If a VNIC needs to be a part of a VLAN and receive traffic for that VLAN, then you need to assign the VLAN ID of that VLAN to the VNIC. You also set the link property `vlan-announce` to propagate the VLAN configurations of each individual VNIC to the network.

Unlike a regular VLAN link, the VNIC configured as a VLAN has its own MAC address. For information about regular VLANs, see Chapter 3, “Configuring Virtual Networks by Using Virtual Local Area Networks,” in “Managing Network Datalinks in Oracle Solaris 11.2.”

This procedure contains only the steps to create the VNIC with a VLAN ID and to set the appropriate properties that enable the VNIC to service VLAN traffic. Although the intermediary ports and switches are automatically updated when you enable the `vlan-announce` property, the intermediary ports and switches must be separately configured to define VLANs at these points.

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

2. **Create a VNIC with a VLAN ID.**
   ```
   # dladm create-vnic -l link -v vid VNIC
   ```

3. **(Optional) Broadcast the VNIC's VLAN configuration to the network.**
   ```
   # dladm set-linkprop -p vlan-announce=gvrp link
   ```
   This step enables a GARP VLAN Registration Protocol (GVRP) client system that automatically registers VLAN IDs with attached switches. By default, the `vlan-announce` property is set to `off` and no VLAN broadcast messages are sent to the network. After you set the property to `gvrp`, the VLAN configuration for that link is propagated to enable automatic VLAN port configuration of the network devices. VLAN traffic can then be accepted and
forwarded by these devices. For more information about GVRP, see "Configuring GVRP," in Sun Ethernet Fabric Operating System.

4. **(Optional) Set the gvrp-timeout property to configure the wait period between VLAN broadcasts.**

   ```
   # dladm set-linkprop -p gvrp-timeout=time link
   ```

   `time` Refers to the value of the gvrp-timeout property in milliseconds. The default value is 250 milliseconds. A system with a heavy load might require a shorter interval when rebroadcasting VLAN information. This property enables you to adjust the interval.

5. **(Optional) Display the value of the properties vlan-announce and gvrp-timeout.**

   ```
   # dladm show-linkprop -p vlan-announce,gvrp-timeout
   ```

   **Example 2-2 Configuring a VNIC as a VLAN**

   This example shows how to create a VNIC named vnic0 on the datalink net0 with a VLAN ID 123 and how to enable the VLAN configuration to be announced to the network.

   ```
   # dladm create-vnic -l net0 -v 123 vnic0
   # dladm set-linkprop -p vlan-announce=gvrp net0
   # dladm set-linkprop -p gvrp-timeout=250 net0
   # dladm show-linkprop -p vlan-announce,gvrp-timeout net0
   ```

   The output shows the following information:

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>vlan-announce</td>
<td>rw</td>
<td>gvrp</td>
<td>gvrp</td>
<td>off</td>
<td>off,gvrp</td>
</tr>
<tr>
<td>net0</td>
<td>gvrp-timeout</td>
<td>rw</td>
<td>250</td>
<td>250</td>
<td>250</td>
<td>100-100000</td>
</tr>
</tbody>
</table>

   The output shows the following information:

   - **LINK** Physical datalink, identified by a name.
   - **PROPERTY** Permissions of the link. A link can have several properties.
   - **PERM** Permissions of the property, which can be one of the following:
     - `ro` refers to read only permission of the link property.
     - `rw` refers to read and write permissions of the link property.
   - **VALUE** Current (or persistent) link property value. If the value is not set, it is shown as `--`. If it is unknown, the value is shown as `?`.
   - **DEFAULT** Default value of the link property. If the link property has no default value, `--` is shown.
   - **POSSIBLE** A comma-separated list of the values that the link property can have. If the possible values are unknown or unbounded, `--` is shown.
Building Virtual Networks

You must create a zone to build a virtual network. You can create any number of zones that you require based on the system support. Each zone has its own virtual interface. The zones in the system can communicate with each other. The virtual network as a whole connects to destinations on the larger external network.

To build a virtual network, you have to configure etherstubs or VNICs and also configure zones. Although these are independent sets of procedures, both must be performed to complete the building of the virtual network.

The procedures in this section are based on the following assumptions:

- The virtual network on the system consists of three zones and are in different stages of configuration. The first zone is created as a new zone, the second zone already exists on the system and needs to be reconfigured to use a VNIC, and the third zone is designated to be a private virtual network and needs to be enabled to send the network traffic beyond the system.
- The system's physical interface is configured with the IP address 192.168.3.70.
- The router's IP address is 192.168.3.25.

When building the virtual network, some steps are performed in the global zone and some steps are performed in a non-global zone. For clarity, the prompts in the examples after each step indicate in which zone a specific command is issued. However, the actual path that the prompts display might vary depending on the prompts specified for your system.


How to Configure a Zone for the Virtual Network

This procedure explains how to configure a new zone with a new VNIC. Note that only the steps related to network virtualization are included in the procedure. For more information about how to configure zones, see Chapter 1, “How to Plan and Configure Non-Global Zones,” in “Creating and Using Oracle Solaris Zones”.

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 VNIC.**
   For more information, see “How to Configure VNICS and Etherstubs” on page 24.

3. **Create the zone.**
   ```
   global# zonecfg -z zone
   ```
   
   `zone` Refers to the name of the zone.

   Make sure that you assign the VNIC that you previously created as the zone's physical interface. By default, the `ip-type` parameter of the `zone` is set to `exclusive`.

4. **Verify and commit the changes that you have implemented and then exit the zone.**
   ```
   zonecfg:zone> verify
   zonecfg:zone> commit
   zonecfg:zone> exit
   ```

5. **Install the zone.**
   ```
   global# zoneadm -z zone install
   ```

6. **Start the zone.**
   ```
   global# zoneadm -z zone boot
   ```

7. **After the zone completely boots up, log in to the zone.**
   ```
   global# zlogin -C zone
   ```

8. **Specify the information as you are prompted.**
   You can specify most of the information by selecting from a list of choices. Usually, the default options suffice. To configure the virtual network, you must specify or verify the following information:
   
   - Host name of the zone, for example, `zone1`
   - IP address of the zone which is based on the IP address of the zone's VNIC
   - Whether IPv6 should be enabled
   - Whether the system with the virtual network is part of a subnet
   - Netmask of the IP address
   - Default route, which can be the IP address of the physical interface on which the virtual network is built

   After you have provided the required information, the zone restarts.
Alternatively, you can configure an exclusive-IP zone with an automatic VNIC called the anet resource. For more information, see “How to Configure the Zone” in “Creating and Using Oracle Solaris Zones”.

**Example 2-3 Configuring a Zone for the Virtual Network**

In this example, zone1 is created for the virtual network and vnic1 is attached as the physical interface. Note that only the zone parameters that are relevant to the creation of a virtual network are listed.

```
global # zonecfg -z zone1
zonecfg:zone1> create
zonecfg:zone1> set zonepath=/export/home/zone1
zonecfg:zone1> set autoboot=true
zonecfg:zone1> add net
zonecfg:zone1:net> set physical=vnic1
zonecfg:zone1:net> end
zonecfg:zone1> verify
zonecfg:zone1> commit
zonecfg:zone1> exit
```

```
global# zoneadm -z zone1 install
.
.
.
```

```
global# zoneadm -z zone1 boot
```

```
```

```
```

To configure the network, the following information is supplied:

- **Hostname:** zone1
- **IP address:** 192.168.3.80
- **System part of a subnet:** Yes
- **Netmask:** 255.255.255.0
- **Enable IPv6:** No
- **Default route:** 192.168.3.70
- **Router IP address:** 192.168.3.25

▼ **How to Reconfigure a Zone to Use a VNIC**

This procedure refers to the second zone in the virtual network. This zone already exists but its current configuration prevents it from becoming a part of the virtual network. Specifically, the zone's IP type is a shared type and its current interface is net0. Both of these configurations must be changed.

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

2. **Create the VNIC.**
   
   ```
   global# dladm create-vnic -l link VNIC
   ```
   
   You will configure the VNIC's interface later in this procedure.

3. **Change the zone's IP type from shared to exclusive.**
   
   ```
   global# zonecfg -z zone
   zonecfg:zone> set ip-type=exclusive
   ```

4. **Change the zone's interface to use a VNIC.**
   
   ```
   zonecfg:zone> remove net physical=NIC
   zonecfg:zone> add net
   zonecfg:zone:net> set physical=VNIC
   zonecfg:zone:net> end
   ```

5. **Verify and commit the changes that you have implemented and then exit the zone.**
   
   ```
   zonecfg:zone> verify
   zonecfg:zone> commit
   zonecfg:zone> exit
   ```

6. **Reboot the zone.**
   
   ```
   global# zoneadm -z zone reboot
   ```

7. **Log in to the zone.**
   
   ```
   global# zlogin zone
   ```

8. **In the zone, create an IP interface over the VNIC that is now assigned to the zone.**
   
   ```
   zone# ipadm create-ip interface
   ```

9. **Configure the VNIC with a static IP address or a Dynamic Host Configuration Protocol (DHCP) IP address.**

   - **Assign a static IP address.**
     
     ```
     zone# ipadm create-addr -a address interface
     ```

     `-a address` Specifies the IP address, which can be in CIDR notation.
How to Reconfigure a Zone to Use a VNIC

- Assign a DHCP IP address.

  
  zone# ipadm create-addr -T dhcp interface

10. Exit the zone.

  
  zone# exit

11. From the global zone, add the address information to the /etc/hosts file.

Example 2-4 Reconfiguring a Zone to Use a VNIC

In this example, zone2 already exists as a shared zone. The zone also uses the primary interface of the system rather than a virtual link. You need to modify zone2 to use vnic2. To use vnic2, zone2's IP type must first be changed to exclusive. Note that some of the output is truncated to focus on the relevant information that relates to virtual networks.

  global# dladm create-vnic -l net0 vnic2
  
  global# zonecfg -z zone2
  zonesfg:zone2> set ip-type=exclusive
  zonesfg:zone2> remove net physical=net0
  zonesfg:zone2> add net
  zonesfg:zone2> set physical=vnic2
  zonesfg:zone2:net> end
  zonesfg:zone2> verify
  zonesfg:zone2> commit
  zonesfg:zone2> exit
  global# zoneadm -z zone2 reboot
  
  global# zlogin zone2
  zone2# ipadm create-ip vnic2
  zone2# ipadm create-addr -a 192.168.3.85/24 vnic2
  ipadm: vnic2/v4
  
  zone2# exit
  
  global# pfedit /etc/hosts
  
  ::1   localhost
  127.0.0.1 localhost
  192.168.3.70 loghost #For net0
  192.168.3.80 zone1 #using vnic1
  192.168.3.85 zone2 #using vnic2
How to Temporarily Create VNICs in Zones

VNICs can be created directly in a non-global zone from a global zone by specifying the link as `zone/link`. This method creates the VNIC directly in the namespace of the non-global zone. The `-t` option is used to specify that the VNIC is temporary. Temporary VNICs persist until the next reboot of the zone. The global zone and other non-global zones can also have VNICs with the same name. VNICs can be created only temporarily by using this method.

In addition to temporarily creating VNICs, you can also temporarily create VLANs and IP over InfiniBand (IPoIB) partitions. See the `dladm(1M)` man page for complete instructions.

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

2. **Create and boot a non-global zone from the global zone.**
   ```
global# zoneadm -z zone boot
   ```

3. **Create a temporary VNIC for the non-global zone.**
   ```
global# dladm create-vnic -t -l link zone/VNIC
   ```
   `-t` Specifies that the VNIC is temporary. Temporary VNICs persist until the next reboot of the zone. This option must be specified if the VNIC is created in a non-global zone's namespace.
   `-l` Specifies the link, which can be a physical link or an etherstub.

   For an example of the command syntax that you would use to create a VLAN or IPoIB partition in a non-global zone from a global zone, see Example 2-5.

4. **Verify that the VNIC was created in the zone.**
   ```
global# dladm show-link -Z
   ```

5. **Log in to the zone.**
   ```
global# zlogin zone
   ```

6. **Verify that the VNIC was successfully created.**
   ```
zone# dladm show-link
   ```
How to Configure a Private Virtual Network

Example 2-5  Temporarily Creating VNICS, VLANs, and IP-over-IB Partitions in Zones

The following example shows how to create a VNIC named vnic1 in a non-global zone from the global zone.

```
 global# zoneadm -z zone1 boot
 global# dladm create-vnic -t -l net0 zone1/vnic1
 global# dladm show-link -Z
```

```
+--------+----------+------+-+-----+-------+
| LINK   | ZONE     | CLASS| MTU | STATE| OVER  |
| net0   | global   | phys | 1500| up   | --    |
| zone1/vnic1 | zone1 | vnic | 1500| down | net0  |
```

The following example shows the output of the `dladm show-link` command from zone1.

```
 zone1# dladm show-link
```

```
+--------+----------+-------+-----+-----+-------+
| LINK   | CLASS    | MTU   | STATE| OVER|
| vnic1  | vnic     | 1500  | down | ?    |
```

The following example shows how to create a VLAN named vlan3 in a non-global zone from a global zone.

```
 global# dladm create-vlan -t -l net0 -v 3 zone1/vlan3
```

The `-v` option specifies the VLAN-ID of the VLAN over the Ethernet link.

The following example shows how to create an IPoIB partition named part1 in a non-global zone from a global zone.

```
 global# dladm create-part -t -l net1 -P FFFF zone1/part1
```

The `-P` option specifies the partition key that is used for creating a partition link.

How to Configure a Private Virtual Network

This procedure explains how to create a private virtual network and enable it to send network traffic beyond the system. Although the zone is part of the virtual network, it is inaccessible from external systems. To enable the isolated zone to send network traffic beyond the system, you must use network address translation (NAT). NAT translates the VNIC’s private IP addresses to routeable IP addresses of the physical network interface. However, the private IP addresses are not visible from the external network. For more information about NAT, see “Using IP Filter’s NAT Feature” in “Securing the Network in Oracle Solaris 11.2”.

The use of etherstubs constitutes the main difference between a virtual network and a private virtual network. In a private virtual network, the VNICs that are assigned to the zones are configured over an etherstub and are isolated from network traffic that flows through the system.

This procedure assumes that the zone already exists, but currently does not have any associated interface.
1. **Become an administrator.**
   For more information, see “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”.

2. **Create the etherstub.**
   ```
   global# dladm create-etherstub etherstub
   ```

3. **Create a VNIC over the etherstub.**
   ```
   global# dladm create-vnic -l etherstub VNIC
   ```
   You will configure the VNIC’s interface later in this procedure.

4. **Assign the VNIC to the zone.**
   ```
   global# zonecfg -z zone
   zonecfg:zone> add net
   zonecfg:zone:net> set physical=VNIC
   zonecfg:zone:net> end
   ```

5. **Verify and commit the changes that you have implemented and then exit the zone.**
   ```
   zonecfg:zone> verify
   zonecfg:zone> commit
   zonecfg:zone> exit
   ```

6. **Reboot the zone.**
   ```
   global# zoneadm -z zone reboot
   ```

7. **Log in to the zone.**
   ```
   global# zlogin zone
   ```

8. **In the zone, create an IP interface over the VNIC that is now assigned to the zone.**
   ```
   zone# ipadm create-ip interface
   ```

9. **Configure the VNIC with a static IP address or a DHCP IP address.**
   - **Assign a static IP address.**
     ```
     zone# ipadm create-addr -a address interface
     ```
   - **Assign a DHCP IP address.**
     ```
     zone# ipadm create-addr -T dhcp interface
     ```
10. Exit the zone.
   
   ```
   zone# exit
   ```

11. From the global zone, add the address information to the `/etc/hosts` file.

12. From the global zone, set the primary interface to perform IP forwarding.
   
   ```
   global# ipadm set-ifprop -p forwarding=on -m ipv4 primary-interface
   ```

   **Note** - In Oracle Solaris, the primary interface is the physical datalink of a NIC.

13. From the global zone, configure network address translation (NAT) in the `/etc/ipnat.conf` file for the primary interface.

14. Start the IP filter service to enable NAT.
   
   ```
   global# svcadm enable network/ipfilter
   ```

15. Reboot the zone.
   
   ```
   global# zoneadm -z zone3 reboot
   ```

**Example 2-6 Configuring a Private Virtual Network**

In this example, zone3 is configured to be isolated as a private network. NAT and IP forwarding are also configured to allow the private virtual network to send packets outside the host while still concealing its private address from the external network. The zone is already configured with an exclusive IP type. However, no IP interface is assigned to the zone.

```plaintext
   global# dladm create-etherstub ether0
   global# dladm create-vnic -l ether0 vnic3
   global# zonecfg -z zone3
   zonecfg:zone3> add net
   zonecfg:zone3:net> set physical=vnic3
   zonecfg:zone3:net> end
   zonecfg:zone3> verify
   zonecfg:zone3> commit
   zonecfg:zone3> exit

   global# zoneadm -z zone3 reboot
   global# zlogin zone3
   zone3# ipadm create-ip vnic3
   zone3# ipadm create-addr -a 192.168.0.10/24 vnic3
   ipadm: vnic3/v4
   zone3# exit

   global# pfedit /etc/hosts
   ::1 localhost
```
Managing VNICS

This section describes tasks that you can perform on VNICS after performing basic configuration. For information about how to perform basic configuration of VNICS, see “How to Configure VNICS and Etherstubs” on page 24.

You can modify the VLAN ID, the MAC address, and the underlying datalink of a VNIC. Modifying the underlying datalink means moving a VNIC to another datalink. You can either globally modify the attribute of all the VNICS on a datalink or selectively modify the attribute of only specified VNICS.

This section covers the following topics:

- “Displaying VNICS” on page 37
- “Modifying the VLAN IDs of VNICS” on page 41
- “Modifying VNIC MAC Addresses” on page 42
- “Migrating VNICS” on page 44
- “Deleting VNICS” on page 46

Displaying VNICS

To obtain information about the VNICS on your system, use the `dladm show-vnic` command.

EXAMPLE 2-7 Displaying VNICS on a System

```
# dladm show-vnic
LINK  OVER  SPEED  MACADDRESS   MACADDRTYPE  VIDS
vnic1  net0  1000  2:8:20:c2:39:38  random  123
vnic2  net0  1000  2:8:20:5f:84:ff  random  456
```
The output shows the following information:

- **LINK**: Virtual datalink, identified by a name.
- **OVER**: Physical or virtual datalink over which the VNIC is configured.
- **SPEED**: Maximum speed of the VNIC, in megabits per second.
- **MACADDRESS**: MAC address of the VNIC.
- **MACADDRTYPE**: MAC address type of the VNIC, which can be one of the following:
  - random – The random address assigned to the VNIC
  - factory – The factory MAC address of the NIC used by the VNIC
  - fixed – The MAC address assigned by the user
- **VID**: VLAN ID of the VNIC.

You can use any `dladm` command that shows information about datalinks to include information about VNICs if they exist on the system. For example, the `dladm show-link` command displays VNICs with other datalinks. You can use the `dladm show-linkprop` command to display the properties of VNICs.

To obtain information about the datalink property of a single VNIC, specify the VNIC in the following command syntax:

```
# dladm show-linkprop [-p property] vnic
```

**EXAMPLE 2-8 Displaying VNICs That Are Attached to Zones**

In this example, information is displayed for the primary datalink and VNICs that are attached to the zones. The primary datalink `net0` is attached to the global zone. The VNICs, `vnic1` and `vnic2`, are attached to `zone1` and `zone2` respectively.

```
# dladm show-link -Z

<table>
<thead>
<tr>
<th>LINK</th>
<th>ZONE</th>
<th>CLASS</th>
<th>MTU</th>
<th>STATE</th>
<th>OVER</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>global</td>
<td>phys</td>
<td>1500</td>
<td>up</td>
<td></td>
</tr>
<tr>
<td>zone1/vnic1</td>
<td>zone1</td>
<td>vnic</td>
<td>1500</td>
<td>up</td>
<td>net0</td>
</tr>
<tr>
<td>zone2/vnic2</td>
<td>zone2</td>
<td>vnic</td>
<td>1500</td>
<td>up</td>
<td>net0</td>
</tr>
</tbody>
</table>
```

**Displaying VNICs With Multiple MAC Addresses**

Multiple MAC addresses are associated with system-created VNICs in Oracle VM Server for SPARC and the `anet` resources in Oracle Solaris Kernel Zones. In Oracle VM Server for SPARC, you need to create a `vnet` with the `alt-mac-addrs` property to support VNICs and zones inside a guest domain. In this case, the system automatically creates a VNIC with
multiple MAC addresses. These multiple MAC addresses are obtained from the vnet that you created. For more information, see *Oracle VM Server for SPARC 3.1 Administration Guide*.

To support zones or VNICs inside kernel zones, you configure the anet resources with multiple MAC addresses. You use the zonecfg command to specify multiple MAC addresses to the anet resources created for network access in kernel zones. For more information, see the *solaris-kz*(5) man page. For information about configuring kernel zones, see “Creating and Using Oracle Solaris Kernel Zones”.

When multiple MAC addresses are associated with VNICs, one MAC address is used by the virtual network driver. You can use the remaining MAC addresses to create VNICs inside kernel zones or a guest domain. For example, if a VNIC is associated with three MAC addresses, one MAC address is assigned for the virtual network driver. Hence, you can create only two VNICs with the remaining two MAC addresses.

You can use the following command to display multiple MAC addresses associated with VNICs:

```
# dladm show-vnic -m
```

**EXAMPLE 2-9** Displaying VNICs With Multiple MAC Addresses in Kernel Zones

```
# dladm show-vnic -m

LINK       OVER MACADDRESSES MACADDPRTYPES  VIDS
gz_vnic0    net0  2:8:20:d7:27:9d  random         0
zone1/net0  net0  2:8:20:70:52:9   random         0
             2:8:20:cd:4c fixed
             2:8:20:78:db:3 random
zone1/net1  net0  0:1:2:3:4:5  fixed          0
             0:1:2:3:4:6 fixed
```

In this example, kernel zone zone1 has two anet resources: net0 and net1. Both resources have more than one MAC address configured. Therefore, inside kernel zone zone1, you can create up to two VNICs on top of the virtual NIC driver zvnet associated with datalink net0. You can create only one VNIC on top of the virtual NIC driver zvnet associated with datalink net1.

**EXAMPLE 2-10** Displaying System-Created VNICs With Multiple MAC Addresses

```
# dladm show-vnic -m

LINK       OVER MACADDRESSES MACADDPRTYPES  VIDS
ldoms-vsw0.vport0 net1 0:14:4f:fb:e1:8f  fixed          0,21
             0:14:4f:fb:8:6b:9 fixed
             0:14:4f:fa:48:7f fixed
ldoms-vsw0.vport1 net1 0:14:4f:f9:1b:8d fixed    45,44
             0:14:4f:f9:27:4 fixed
```

In this example, you can create up to two VNICs on top of the guest domain's virtual network driver vnet associated with ldoms-vsw0.vport0. You can create up to one VNIC on top of the virtual NIC driver vnet associated with ldoms-vsw0.vport1.
Displaying the Physical and Virtual Link State of Datalinks

The physical link state of a datalink identifies whether the physical device has connectivity with the external network. If the cable is plugged in and the state of the port on the other end of the cable is up, then the physical device has connectivity with the external network.

You can use the following commands to display the physical link state of a datalink:

```bash
# dladm show-phys [link]
# dladm show-ether [link]
```

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

**EXAMPLE 2-11** Displaying the Physical Link State of Datalinks

The following example displays the physical link state of datalinks on a system by using the `dladm show-phys` command.

```bash
# dladm show-phys
LINK       MEDIA        STATE      SPEED  DUPLEX    DEVICE
net1       Ethernet     down       0      unknown   e1000g1
net2       Ethernet     down       0      unknown   e1000g2
net3       Ethernet     down       0      unknown   e1000g3
net0       Ethernet     up         1000   full      e1000g0
```

The following example displays the physical link state of datalinks on a system by using the `dladm show-ether` command.

```bash
# dladm show-ether
LINK       PTYPE    STATE    AUTO  SPEED-DUPLEX    PAUSE
net1       current  down     yes   0M              bi
net2       current  down     yes   0M              bi
net3       current  down     yes   0M              bi
net0       current  up       yes   1G-f            bi
```

When multiple VNICS are created over a NIC, a virtual switch is created internally to enable VNICS and the primary datalink to communicate when they are on the same VLAN. These datalinks can communicate with each other even if the physical datalink has no connection with the external network. This forms the virtual link state of the datalink, which can be up, down, or unknown. The virtual link state of a datalink identifies whether a datalink has connectivity with internal networks within the system even if the physical cable is unplugged.

You use the following command to display the virtual link state of a datalink:

```bash
# dladm show-link [link]
```

**EXAMPLE 2-12** Displaying the Virtual Link State of Datalinks

This example displays the virtual link state of datalinks on a system.
Modifying the VLAN IDs of VNICs

VNICs can be configured as VLANs. You need to modify the VLAN IDs of VNICs on a datalink when you want the VNICs to host a specific VLAN's traffic.

The `dladm` subcommand that you use depends on whether you are modifying VLANs or VNICs configured as VLANs:

- For VLANs that are created with the `dladm create-vlan` command, use the `dladm modify-vlan` command. To display these VLANs, use the `dladm show-vlan` command.
- For VLANs that are created with the `dladm create-vnic` command, use the `dladm modify-vnic` command. To display these VNICs, including those with VLAN IDs, use the `dladm show-vnic` command.

You can modify the VLAN ID of a single VNIC or multiple VNICs that are configured on the datalink. You can also modify the VLAN IDs of VNICs as a group by configuring all the VNICs with the same VLAN ID.

- If only one VNIC is configured on the datalink, use the following command syntax to modify the VLAN ID of the VNIC:

  ```
  # dladm modify-vnic -v vid -L link
  ```

  where `vid` is the new VLAN ID that you assign to the VNIC.

**EXAMPLE 2-13** Modifying the VLAN ID of a VNIC on a Datalink

In this example, the VLAN ID of `vnic0` that is configured over the datalink `net0` is modified.

```
# dladm modify-vnic -v 123 -L net0
# dladm show-vnic
```

- If multiple VNICs are configured on the datalink, use the following command syntax to modify the VLAN IDs of the VNICs:
Because each VLAN ID is unique for VNICs on the same datalink, you must change the VLAN IDs one at a time.

**EXAMPLE 2-14** Modifying the VLAN ID of Multiple VNICs on a Datalink

In this example, the VLAN IDs of vnic0, vnic1, and vnic2 are modified.

```plaintext
# dladm modify-vnic -v 123 vnic0
# dladm modify-vnic -v 456 vnic1
# dladm modify-vnic -v 789 vnic2
```

If each VNIC is configured on a different datalink, use the following command syntax to modify the VLAN ID of VNICs as a group:

```plaintext
# dladm modify-vnic -v vid VNIC,VNIC,[...]
```

**EXAMPLE 2-15** Modifying the VLAN IDs of VNICs as a Group

In this example, the VLAN IDs of vnic0, vnic1, and vnic2 are modified as a group. These VNICs are configured over the datalinks net0, net1, and net2 respectively.

```plaintext
# dladm modify-vnic -v 123 vnic0,vnic1,vnic2
```

### Modifying VNIC MAC Addresses

Any VNIC that a user creates can only have one MAC address. You can modify the MAC address by using the `dladm modify-vnic` command. You can configure the VNICs created for kernel zones with one or more MAC addresses.

You can modify the existing MAC address of a VNIC configured on a datalink. You can either modify the MAC addresses of all the VNICs or selectively modify the MAC addresses of the specified VNICs. You can also modify the VLAN ID and the MAC address of a VNIC simultaneously.
To modify the MAC address of a VNIC, use the following command syntax:

```
# dladm modify-vnic -m MAC-address VNIC
```

where `MAC-address` is the new MAC address that you want to assign to the VNIC.

**EXAMPLE 2-16** Modifying the MAC Address of a VNIC

In this example, `vnic0` is assigned a specific MAC address.

```
# dladm modify-vnic -m 3:8:20:5f:84:ff vnic0
# dladm show-vnic
```

```
+----+-------+----+----------------+-------------+-----+
| LINK | OVER  | SPEED | MACADDRESS      | MACADDRTYPE | VIDS |
+----+-------+----+----------------+-------------+-----+
| vnic0 | net0 | 1000 | 3:8:20:5f:84:ff | fixed        | 0   |
+----+-------+----+----------------+-------------+-----+
```

To modify the MAC addresses of all the VNICs on a datalink, use the following command syntax:

```
# dladm modify-vnic -m random -L link
```

In this command syntax, the `-m random` option is equivalent to the `-m auto` option. The MAC address is assigned automatically to the VNICs on a random basis.

**EXAMPLE 2-17** Modifying the MAC Addresses of All the VNICs on a Datalink

In this example, the MAC addresses of all the VNICs configured over the datalink `net0` are automatically modified on a random basis.

```
# dladm modify-vnic -m random -L net0
# dladm show-vnic
```

```
+----+-------+----+----------------+-------------+-----+
| LINK | OVER  | SPEED | MACADDRESS      | MACADDRTYPE | VIDS |
+----+-------+----+----------------+-------------+-----+
| vnic0 | net0 | 1000 | 2:8:20:22:9d:bb | random       | 0   |
| vnic1 | net0 | 1000 | 2:8:20:72:2e:9  | random       | 0   |
| vnic2 | net0 | 1000 | 2:8:20:2f:e5:83 | random       | 0   |
+----+-------+----+----------------+-------------+-----+
```

To modify the MAC addresses of VNICs on a selective basis, use the following command syntax:

```
# dladm modify-vnic -m random VNIC,VNIC,[...]
```

For both the global and selective modifications, you specify `random` for the `-m` option.

**EXAMPLE 2-18** Modifying the MAC Addresses of VNICs on Selective Basis

In this example, the MAC addresses of `vnic0` and `vnic2` that are configured over the datalink `net0` are selectively modified.

```
# dladm modify-vnic -m random vnic0,vnic2
# dladm show-vnic
```

```
+----+-------+----+----------------+-------------+-----+
| LINK | OVER  | SPEED | MACADDRESS      | MACADDRTYPE | VIDS |
+----+-------+----+----------------+-------------+-----+
| vnic0 |      |      | 2:8:20:22:9d:bb | random       | 0   |
| vnic1 |      |      | 2:8:20:72:2e:9  | random       | 0   |
| vnic2 |      |      | 2:8:20:2f:e5:83 | random       | 0   |
+----+-------+----+----------------+-------------+-----+
```
Managing VNICs

To modify the VLAN ID and the MAC address of a VNIC simultaneously, use the following command syntax:

```bash
# dladm modify-vnic -m random -v vid VNIC
```

**Caution** - Modifying multiple attributes of the VNICs globally might cause unexpected behavior with the VNICs. Instead, modify the multiple attributes of the VNICs separately.

**EXAMPLE 2-19**  Modifying the VLAN ID and the MAC Address of a VNIC

In this example, the VLAN ID and the MAC address of `vnic0` are modified simultaneously.

```bash
# dladm modify-vnic -m random -v 123 vnic0
# dladm show-vnic vnic0
```

**Migrating VNICs**

You can move one or more VNICs from one underlying datalink to another underlying datalink without deleting and reconfiguring the VNICs. The underlying datalink can be a physical link, a link aggregation, or an etherstub.

You usually migrate a VNIC in any of the following situations:

- When you need to replace the existing NIC with a new NIC
- When the target NIC has more bandwidth than the existing NIC
- When the target NIC implements certain features in hardware, such as a large receive offload (LRO), a large segment offload (LSO), and checksum

To successfully migrate VNICs, the target datalink to which the VNICs are moved must be able to accommodate the datalink properties of the VNICs. If those properties are not supported, then migration fails and the user is notified. After a successful migration, all the applications that use the VNICs continue to operate normally, provided that the target datalink is connected to the network.

Certain hardware-dependent properties might change after a VNIC migration, such as the datalink state, link speed, and MTU size. The values of these properties are inherited from the datalink to which the VNICs are migrated. You can migrate all the VNICs that are configured
over a datalink or selectively migrate the specified VNICS. You can also migrate the VNICS and modify their VLAN IDs simultaneously.

- To migrate all the VNICS configured over the source link to the target link, use the following command syntax:

  ```bash
  # dladm modify-vnic -l target-link -L source-link
  
  -l target-link  Refers to the link over which the VNICS are migrated
  -L source-link  Refers to the link over which the VNICS were previously configured
  ```

**EXAMPLE 2-20** Migrating All the VNICS From a Source Link to a Target Link

In this example, all the VNICS from the source link `ether0` are moved to the target link `net1`.

```bash
# dladm modify-vnic -l net1 -L ether0
# dladm show-vnic

<table>
<thead>
<tr>
<th>LINK</th>
<th>OVER</th>
<th>SPEED</th>
<th>MACADDRESS</th>
<th>MACADDRTYPE</th>
<th>VIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic0</td>
<td>net1</td>
<td>1000</td>
<td>2:8:20:c2:39:38</td>
<td>random</td>
<td>321</td>
</tr>
<tr>
<td>vnic1</td>
<td>net1</td>
<td>1000</td>
<td>2:8:20:5f:84:ff</td>
<td>random</td>
<td>656</td>
</tr>
<tr>
<td>vnic2</td>
<td>net1</td>
<td>1000</td>
<td>2:8:20:5f:84:ff</td>
<td>random</td>
<td>0</td>
</tr>
<tr>
<td>vnic3</td>
<td>net0</td>
<td>1000</td>
<td>2:8:20:5f:84:ff</td>
<td>random</td>
<td>345</td>
</tr>
</tbody>
</table>
```

- To migrate the specified VNICS configured over the source link to the target link, use the following command syntax:

  ```bash
  # dladm modify-vnic -l target-link VNINC,VNIC,[...]
  
  To perform selective VNIC migration, you need to specify only the target link.
  ```

**EXAMPLE 2-21** Migrating Specified VNICS From a Source Link to a Target Link

In this example, `vnic0`, `vnic1`, and `vnic2` are selectively moved to the target link `net1` from the source link `net0`.

```bash
# dladm modify-vnic -l net1 vnic0,vnic1,vnic2
# dladm show-vnic

<table>
<thead>
<tr>
<th>LINK</th>
<th>OVER</th>
<th>SPEED</th>
<th>MACADDRESS</th>
<th>MACADDRTYPE</th>
<th>VIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic0</td>
<td>net1</td>
<td>1000</td>
<td>2:8:20:c2:39:38</td>
<td>random</td>
<td>321</td>
</tr>
<tr>
<td>vnic1</td>
<td>net1</td>
<td>1000</td>
<td>2:8:20:5f:84:ff</td>
<td>random</td>
<td>656</td>
</tr>
<tr>
<td>vnic2</td>
<td>net1</td>
<td>1000</td>
<td>2:8:20:5f:84:ff</td>
<td>random</td>
<td>0</td>
</tr>
<tr>
<td>vnic3</td>
<td>net0</td>
<td>1000</td>
<td>2:8:20:5f:84:ff</td>
<td>random</td>
<td>345</td>
</tr>
</tbody>
</table>
```

- To modify the VLAN IDs of the VNICS configured over the source link and migrate them to the target link simultaneously, use the following command syntax:

  ```bash
  # dladm modify-vnic -l target-link -v vid VNINC
  
  To assign new VLAN IDs, you must migrate the VNICS one at a time.
EXAMPLE 2-22  Migrating and Modifying the VLAN IDs of VNICs

In this example, vnic0, vnic1, and vnic2 are migrated to the target datalink net1. With the migration, the VLAN IDs of all the VNICs are also modified simultaneously.

```
# dladm modify-vnic -l net1 -v 123  vnic0
# dladm modify-vnic -l net1 -v 456  vnic1
# dladm modify-vnic -l net1 -v 789  vnic2
```

```
  LINK      OVER      SPEED         MACADDRESS         MACADDRTYPE        VIDS
  vnic0     net1      1000         2:8:20:c2:39:38     random             123
  vnic1     net1      1000         2:8:20:5f:84:ff     random             456
  vnic2     net1      1000         2:8:20:5f:84:ff     random             789
```

When you migrate VNICs from the source link to the target link, randomly assigned MAC addresses are unaffected and retained by their respective VNICs after migration. See Example 2-22.

However, the MAC address will change if the VNIC is using a factory MAC address from the source link. If you do not specify a MAC address during migration, the factory MAC address of the VNIC is replaced by a randomly assigned MAC address. If you specify a MAC address with `-m` during migration, the factory MAC address of the VNIC is replaced by the specified MAC address.

You have multiple MAC addresses associated with VNICs created by kernel zones. When you migrate VNICs created by kernel zones, all the multiple MAC addresses associated with VNICs are migrated to the target NIC.

# Deleting VNICs

This section describes how to delete a VNIC.

## How to Delete a VNIC

1. **Become an administrator.**
2. **(Optional) Check whether the VNIC is busy.**
   
   You can delete a VNIC only when it is not busy. A VNIC can be busy for multiple reasons. You need to perform the following steps to check whether the VNIC busy:
   
   - Check whether the VNIC is plumbed and associated with an IP address.
     
     ```
     # ipadm show-if
     # ipadm show-addr
     ```
If the VNIC is plumbed and associated with IP addresses, remove the IP interface.

```
ipadm delete-ip interface
```

- **Check whether there are any flows configured over the VNIC.**

  ```
  flowadm
  ```

  If flows are configured over the VNIC, remove the flow.

  ```
  flowadm remove-flow flowname
  ```

- **Check whether the VNIC is assigned to a zone.**

  ```
  dladm show-link -Z
  ```

  For more information about how to delete a VNIC that is attached to a zone, see “How to Delete a VNIC Attached to a Zone” on page 47.

- **Check whether the VNIC is created by the system.**

  ```
  dladm show-vnic
  ```

  Only a system-created VNIC contains a hyphen (-), which helps you to differentiate between a system-created VNIC and a user-created VNIC. You cannot modify, rename, plumb, or delete system-created VNICs.

- **Check whether the VNIC is snooped.**

  ```
  snoop
tshark
  ```

  If the VNIC is snooped by using the snoop command, kill the process.

  ```
  pkill snoop
  ```

  If the VNIC is snooped by using the tshark command, kill the process.

  ```
  pkill tshark
  ```

3. **Delete the VNIC.**

   ```
   dladm delete-vnic VNIC
   ```

### How to Delete a VNIC Attached to a Zone

This procedure assumes that the VNIC is attached to a zone. You must be in the global zone to perform this procedure.
1. **Halt the zone.**

   ```
global# zoneadm -z zone halt
   ```

   **Note** - To determine the links used by a zone, use the `dladm show-link` command.

2. **Remove or detach the VNIC from the zone.**

   ```
global# zonecfg -z zone remove net physical=VNIC
   ```

3. **Delete the VNIC from the system.**

   ```
global# dladm delete-vnic VNIC
   ```

4. **Reboot the zone.**

   ```
global# zoneadm -z zone boot
   ```

**Example 2-23** Deleting a VNIC Attached to a Zone

In this example, vnic1 is removed from zoneB and from the system.

```
global# dladm show-link
LINK            CLASS   MTU    STATE   OVER
net0            phys    1500   up      --
net2            phys    1500   up      --
net1            phys    1500   up      --
net3            phys    1500   up      --
zezoneA/net0    vnic    1500   up      net0
nzezoneB/net0    vnic    1500   up      net0
vnic0           vnic    1500   up      net1
nzezoneA/vnic0   vnic    1500   up      net1
vnic1           vnic    1500   up      net1
nzezoneB/vnic1   vnic    1500   up      net1

global# zoneadm -z zoneB halt

global# zonecfg -z zoneB remove net physical=vnic1

global# dladm delete-vnic vnic1

global# zoneadm -z zoneB boot
```
Enabling the SR-IOV Mode of Datalinks

In Oracle Solaris, you can associate the virtual function (VF) of a network device that supports SR-IOV with a VNIC or a VLAN. A VF VNIC is a VNIC that owns a dedicated VF. A VF VNIC differs from a regular VNIC in the sharing of resources. A regular VNIC needs to share resources with other regular VNICs, but a VF VNIC need not share resources. Each VF is a separate hardware resource for the VF VNIC.

You can create VF VNICs only over datalinks that support the SR-IOV mode. By default, the SR-IOV mode of a datalink is disabled. You can enable the SR-IOV mode of a datalink by setting the `iov` property to `on`. For information about creating VF VNICs after you enable the SR-IOV mode of a datalink, see “Creating VF VNICs” on page 50.

You can check the SR-IOV mode of a datalink by specifying the link property `iov` with the `dladm show-linkprop` command. If the value under the EFFECTIVE column of the output is `off`, the SR-IOV mode of the datalink is disabled.

The following example shows how you can check the SR-IOV mode of the datalink `net0`.

```
# dladm show-linkprop -p iov net0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>iov</td>
<td>rw</td>
<td>auto</td>
<td>off</td>
<td>auto</td>
<td>auto,on,off</td>
</tr>
</tbody>
</table>

In this example, the SR-IOV mode of the datalink `net0` is disabled. The output shows the following information:

- **VALUE**: Specifies the value that you have set for the `iov` link property. If you have not modified the `iov` link property, the default value of the `iov` link property is `auto`. The value of `auto` means that the OS determines whether the SR-IOV mode is enabled by default on a particular physical datalink.

- **EFFECTIVE**: The actual SR-IOV mode of the datalink. By default, all SRIOV-capable NICs show the value `off` under the EFFECTIVE column.

You can enable the SR-IOV mode of the datalink `net0` by setting the `iov` property to `on` as follows:

```
# dladm set-linkprop -p iov=on net0
# dladm show-linkprop -p iov net0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>iov</td>
<td>rw</td>
<td>on</td>
<td>on</td>
<td>auto</td>
<td>auto,on,off</td>
</tr>
</tbody>
</table>

Similarly, you can disable the SR-IOV mode of a datalink by setting the `iov` link property to `off`. For more information about the `dladm` command, see the `dladm(1M)` man page.
Creating VF VNICs

To create a VF VNIC on a datalink, you need to enable the SR-IOV mode of a datalink. For more information, see “Enabling the SR-IOV Mode of Datalinks” on page 49. After you enable the SR-IOV mode of a datalink, VFs are automatically allocated to VNICs when you create VNICs by using the `dladm create-vnic` command. Similarly, VFs are automatically allocated to VLANs when you create VLANs by using the `dladm create-vlan` command.

You can also explicitly specify whether a VF needs to be allocated to a VNIC or a VLAN by specifying the `iov` VNIC link property with the `dladm create-vnic` or the `dladm create-vlan` commands.

You use the following command syntax to explicitly create a VF VNIC:

```bash
# dladm create-vnic [-p iov=value] -l link VNIC
```

When you are creating a VF VNIC, specifying the `iov` VNIC link property is optional. If you do not specify the `iov` VNIC link property, then the default value `inherit` is assigned to this property. You can specify the following values for the `iov` VNIC link property:

- **inherit**
  - Default value of the `iov` VNIC link property. Determines whether a VF needs to be allocated based on the effective `iov` property value of the underlying datalink:
    - off – Does not allocate a VF for a VNIC.
    - on – Tries to allocate a VF for a VNIC. If not possible, a regular VNIC is created.

- **on**
  - Allocates a VF. If a VF is not found, the creation of a VNIC fails.

- **off**
  - Creates a VNIC without a VF.

The effective value of a datalink property is the value displayed under the EFFECTIVE column when you use the `dladm show-linkprop` command for a datalink.

The difference between the `iov` VNIC link property and other datalink properties is that you can specify the `iov` VNIC link property only when you are creating a VNIC or a VLAN. You cannot modify the `iov` VNIC link property after you create a VNIC or a VLAN.

The `iov` VNIC link property has an effective value that indicates whether a VF is allocated for the VNIC or VLAN. The value `on` under the EFFECTIVE column means that the VF is allocated and the value `off` under the EFFECTIVE column means that the VF is not allocated.
EXAMPLE 2-24  Creating a VF VNIC

The following example shows how to create the VF VNIC vfvnic1 and the regular VNIC vnic1 on the datalink net0 by explicitly specifying the iov VNIC link property. This example assumes that you have enabled the SR-IOV mode of the datalink net0.

```
# dladm show-linkprop -p iov net0
LINK PROPERTY PERM VALUE EFFECTIVE DEFAULT POSSIBLE
net0 iov rw on on auto auto,on,off
# dladm create-vnic -l net0 vfvnic1
# dladm show-linkprop -p iov vfvnic1
LINK PROPERTY PERM VALUE EFFECTIVE DEFAULT POSSIBLE
vfvnic1 iov r- inherit on inherit inherit,on,off
# dladm create-vnic -p iov=off -l net0 vnic1
# dladm show-linkprop -p iov vnic1
LINK PROPERTY PERM VALUE EFFECTIVE DEFAULT POSSIBLE
vnic1 iov r- off off inherit inherit,on,off
```

This example provides the following information:

- You need to set the iov property for the datalink net0 to on before you create the VF VNICs.
- If you do not specify a value for the iov property when creating a VNIC, then the default value inherit is assigned to the iov property. The VF VNIC vfvnic1 is created with a VF.
- If you explicitly specify the value off for the iov property when creating a VNIC, a regular VNIC is created without a VF even though the iov property of the underlying datalink net0 is on. The VNIC vnic1 is created without a VF.

Migrating VF VNICs

You can move VF VNICs or VF VLANs from one datalink to another datalink. Note the following requirements:

- The target datalink must support SR-IOV and the iov property must be set to on. For more information about how to check the status of the iov property for a datalink, see “Enabling the SR-IOV Mode of Datalinks” on page 49.
- A VF must be available on the target datalink. For more information about how to check the number of VFs available on a datalink, see “Displaying VFs Information” on page 52.

If these requirements are not met, then the VF VNIC is migrated to the target datalink as a regular VNIC without a VF.

If you migrate a VF VNIC, that was created by specifying iov=inherit, the migration succeeds even if the target datalink does not support the iov property or the iov property is disabled. If
you try to migrate a VF VNIC, that was created with `iov=on`, the migration succeeds only if the SR-IOV mode is enabled on the target datalink.

For more information about how to migrate a VNIC, see “Migrating VNICs” on page 44.

### Displaying VFs Information

You can display information about the availability of VFs on a datalink by using the following command:

```
# dladm show-phys -V
```

The output shows the following information:

<table>
<thead>
<tr>
<th>LINK</th>
<th>VFS-AVAIL</th>
<th>VFS-INUSE</th>
<th>FLAGS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of VFs available on a datalink that can be assigned to a VNIC. If the datalink does not support SR-IOV, VFS-AVAIL is shown as --.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VFS-INUSE</td>
<td>Number of VFs that are used by a datalink. If the datalink does not support SR-IOV, VFS-INUSE is shown as --.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLAGS</td>
<td>The l flag indicates that the datalink is managed by Oracle VM Server for SPARC.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### EXAMPLE 2-25 Displaying VFs Information for Datalinks

```
# dladm show-phys -V
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>VFS-AVAIL</th>
<th>VFS-INUSE</th>
<th>FLAGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>30</td>
<td>1</td>
<td>-----</td>
</tr>
<tr>
<td>net1</td>
<td>0</td>
<td>0</td>
<td>l-----</td>
</tr>
<tr>
<td>net2</td>
<td>--</td>
<td>--</td>
<td>-----</td>
</tr>
</tbody>
</table>

In this example, the datalink net0 has 30 available VFs and one VF in use. The datalink net1 has zero (0) available VFs and it is currently being used by Oracle VM Server for SPARC. The datalink net2 does not support SR-IOV.

You can display the VF devices assigned to VNICs on a system by using the following command:

```
# dladm show-vnic -V
```

The output shows the following information:

<table>
<thead>
<tr>
<th>LINK</th>
<th>Name of the VNIC.</th>
</tr>
</thead>
</table>
VF-ASSIGNED  VF device assigned to the VNIC. If the VNIC does not have a VF, VF-ASSIGNED is shown as --.

**EXAMPLE 2-26**  Displaying VF Devices Assigned to VNICS

```
# dladm show-vnic -V
LINK  VF-ASSIGNED
vnic1  ixgbevf0
vnic2  --
vnic3  ixgbevf1
```

In this example, the VF device ixgbevf0 is assigned to vnic1. The VNIC vnic2 does not have an allocated VF device. The VF device ixgbevf1 is assigned to vnic3.
Configuring Virtual Networks by Using Virtual Extensible Local Area Networks

Traditional network isolation methods, such as virtual local area networks (VLANs) are not adequate to support virtualization in large data centers. As cloud environments are also tightly coupled with the underlying physical networks, virtual machines cannot be migrated between physical servers that belong to different physical Layer 2 networks. Oracle Solaris supports the virtual extensible local area network (VXLAN) technology that addresses such virtualization issues in a large virtualized data center or cloud environment.

This chapter provides an overview of deploying VXLANs and describes how to configure them. It also discusses how VXLANs can be used with other technologies, for example, zones.

This chapter contains the following topics:

- “Overview of VXLANs” on page 55
- “Advantages of Using VXLANs” on page 56
- “VXLAN Naming Convention” on page 57
- “VXLAN Topology” on page 57
- “Using VXLAN With Zones” on page 59
- “Configuring a VXLAN” on page 62
- “Displaying VXLAN Information” on page 66
- “Deleting a VXLAN” on page 66
- “Assigning a VXLAN to a Zone” on page 67
- “Use Case: Configuring a VXLAN Over a Link Aggregation” on page 68

Overview of VXLANs

In a cloud environment, physical servers might be located in different Layer 2 networks. For example, a cloud might span physical servers that are in different geographical locations. In such cases, creating virtual machines (VMs) or tenants over a Layer 2 network restricts the number of physical servers that you can use for provisioning these VMs. You can use physical
servers in different Layer 2 networks for provisioning VMs. However, as the migration between different servers is restricted to the same Layer 2 network, the utilization of the physical resource is not optimized.

VXLAN is a Layer 2 technology that enables you to create a Layer 2 network on top of a Layer 3 network, thereby providing further network isolation. VXLAN provides a virtual Layer 2 network that stretches over multiple physical Layer 2 networks. Therefore, provisioning resources in a cloud environment is not restricted to a single physical Layer 2 network. Physical servers can be a part of a VXLAN network as long as they are connected by IPv4 or IPv6 networks.

You can use the VXLAN technology with the Elastic Virtual Switch (EVS) feature of Oracle Solaris to create a large number of virtual networks. For information about how to use VXLAN with the EVS feature to create a virtual network, see “Use Case: Configuring an Elastic Virtual Switch for a Tenant” on page 151. For more information, see Chapter 5, “About Elastic Virtual Switches” and Chapter 6, “Administering Elastic Virtual Switches”.

VXLAN provides isolated Layer 2 segment that is identified by the VXLAN segment ID or VXLAN network identifier (VNI). All VMs in the same VXLAN segment belong to the same virtual Layer 2 broadcast domain.

Communication in VXLANs is similar to that in isolated VLANs. Hence, only VMs that are in the same VXLAN segment can talk to each other. VMs that are not in the same VXLAN segment cannot communicate with each other.

**Advantages of Using VXLANs**

VXLAN provides the following advantages:

- Increases scalability in virtualized cloud environments as the VXLAN ID is 24 bits, which enables you to create up to 16 million isolated networks. This overcomes the limitation of VLANs having the 12 bits VLAN ID, which enables you to create a maximum of 4094 isolated networks.
- Enables you to use the Layer 3 features of the underlying network.
- The virtual Layer 2 network is abstracted from the underlying physical network. As a result, the virtual network is not visible to the physical network and provides the following benefits:
  - Removes the need to have additional physical infrastructure. For example, the forwarding table of the external switch does not grow with the increase in the VMs behind the physical port on the server.
  - Reduces the scope of MAC address duplication to VMs that exists in the same VXLAN segment. The MAC address can overlap when the addresses are not a part of the same VXLAN segment.
In a VXLAN, only the MAC address of the datalink that belong to the same VXLAN segment or VNI must be unique. This is similar to a VLAN where the VLAN ID and the MAC address must have a unique combination.

**VXLAN Naming Convention**

In Oracle Solaris, a VXLAN endpoint is represented by a VXLAN datalink. This VXLAN datalink is associated with an IP address (IPv4 or IPv6) and a VXLAN network identifier (VNI). Even though multiple VXLAN datalinks can use the same IP address, the combination of the IP address and VNI must be unique. You can configure a VXLAN datalink with an optional multicast address, which is used for discovering the peer VXLAN endpoints on the same VNI and also to implement broadcast within a VXLAN segment. VXLAN datalinks in the same VNI must be configured with the same multicast address. For more information about the requirements of a VXLAN, see “VXLAN Requirements” on page 61.

Every VXLAN datalink is associated with a VXLAN segment ID, or a VNI. The convention for naming VXLAN datalinks is same as the convention that is used for links or VLANs. For information about providing valid datalink names, see “Rules for Valid Link Names” in “Configuring and Administering Network Components in Oracle Solaris 11.2”.

**VXLAN Topology**

VXLAN enables you to organize systems on a Layer 3 network within their own VXLAN segments.

The following figure illustrates a VXLAN network that is configured over multiple physical servers.
FIGURE 3-1 VXLAN Topology
The figure shows three virtualized hosts attached to an IP network infrastructure. There are three VXLAN overlay networks identified by the VXLAN segment IDs or VNIs, 60, 20, and 22. The VMs VM1 and VM6 are on the overlay network identified by the VNI 60, the VMs VM2 and VM3 are on the overlay network identified by the VNI 20, and the VMs VM4 and VM5 are on the overlay network identified by the VNI 22.

**Using VXLAN With Zones**

You can assign VNICs that are created over VXLAN datalinks to zones. VXLAN datalinks are created by specifying a VNI and these VXLAN datalinks belong to the VXLAN segment that is identified by that VNI. For example, if you specify the VNI as 20 when you create the VXLAN datalink, then that datalink belongs to the VXLAN segment identified by the VNI 20. VNICs that are created over VXLAN datalinks are a part of the VXLAN segment.

The following figure shows two virtualized Oracle Solaris hosts attached to an IP network infrastructure with two VXLAN overlay networks identified by the VNIs, 20 and 60.
You can create zones that are a part of a VXLAN segment in the following ways:

- Create a VNIC over a VXLAN and assign the VNIC to the zone. For more information, see “Configuring a VXLAN” on page 62.
- Assign the VXLAN as the underlying link for the zone’s anet (VNIC) resource. For more information, see “Assigning a VXLAN to a Zone” on page 67.

In any case, the VNIC that is created in a zone is a part of a VXLAN segment identified by the underlying VXLAN datalink. For more information about zones, see “Introduction to Oracle Solaris 11.2 Virtualization Environments”.
Assigning VNICs to VXLAN links is similar to creating a VLAN link and assigning it to a zone. For more information about creating a VLAN and assigning it to a zone, see “How to Configure a VLAN” in “Managing Network Datalinks in Oracle Solaris 11.2”.

Planning a VXLAN Configuration

Planning a VXLAN configuration, includes the following steps:

1. Determine the virtual network topology in a physical network. For example, if you are hosting a service that consists of several VMs on different servers, you can assign a VXLAN segment for these VMs. The VMs in this VXLAN segment can communicate with each other but not with the other VMs that are not in this VXLAN segment.

2. Verify that the physical servers are connected through an IP interface and that IP multicasting is enabled on the physical network.

3. Create a numbering scheme for the VXLAN segments. For example, you can assign the VXLAN segments (VNIs) based on the application hosted by the VMs.

4. Create a VXLAN datalink by specifying the IP address and the VXLAN segment ID. Optionally, you can assign the VXLAN segments with their own multicast address.

5. Create VNICs over VXLAN datalinks and assign the VNICs to zones.

   Alternatively, you can assign the VXLAN links as the underlying link for the zone's anet link.

VXLAN Requirements

Before using a VXLAN, check whether you have met the following requirements:

- Ensure that IP multicasting is supported on the network. If IP multicasting is not supported, VMs in the VXLAN cannot communicate with each other.

- If the VXLAN includes servers in different IP subnets, then multicast routing must be supported across the subnets. If multicasting routing is not supported, only the VMs over the VXLANs on the same IP subnet can communicate with each other and VMs over VXLANs on different IP subnets, for example, geographically dispersed data centers cannot communicate with each other.

For more information about naming conventions of a VXLAN datalink, see “VXLAN Naming Convention” on page 57.
Configuring a VXLAN

The following procedure assumes that the zones are already created on the system. For information about zone configuration, see Chapter 1, “How to Plan and Configure Non-Global Zones,” in “Creating and Using Oracle Solaris Zones”.

❖ How to Configure a VXLAN

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

2. **Determine the IP addresses that are available on the system.**
   
   # ipadm show-addr

3. **Create the VXLAN datalink by specifying the IP address or IP interface.**

   - **To create the VXLAN by specifying the IP address:**
     
     # dladm create-vxlan -p prop=value VXLAN-LINK
     
     -p prop=value Specifies a comma-separated list of VXLAN datalink properties that can be set to the specified values on the VXLAN datalink that you create. You set the following properties:
     - ```
     - addr – Specifies the IPv4 or IPv6 address for the VXLAN network. This address can be a specific address or a combination of address/prefix length.
     ```
     - ```
     - vni – Specifies the network identifier of the VXLAN segment. You can specify a number between 0 and 16777215.
     ```
     - ```
     - mgroup – (Optional) Specifies the multicast group name. You can specify this option only if the VXLAN segment has its own multicast group.
     ```
     
     VXLAN-LINK Name of the VXLAN.

   - **To create the VXLAN by specifying the IP interface:**
     
     # dladm create-vxlan -p prop=value
     
     -p prop=value Specifies a comma-separated list of VXLAN datalink properties that can be set to the specified values on the VXLAN datalink that you create. You set the following properties:
interface – Specifies the IP interface for the VXLAN network.

vni – Specifies the network identifier of the VXLAN segment. You can specify a number between 0 and 16777215.

VXLAN Name of the VXLAN.

When you specify the IP interface and the IP version, the VXLAN datalink is created over an available IP address of the version that is specified on that interface. For example, if you have an IP address 10.10.10.1 configured over net0, a VXLAN datalink is created over 10.10.10.1. By default, an IP version is an IPv4 address. However, if you need an IPv6 address, you must specify the version by using the ipvers property.

Note - You can create VXLAN datalinks on IP addresses that are hosted on physical aggregated links (trunk or DLMP aggregation) or IPoIB links. However, you cannot create VXLAN datalinks on IP addresses hosted on IPMP, a virtual network interface, or loopback interfaces.

4. Verify the VXLAN that you created.

   # dladm show-vxlan

5. Create a VNIC over the VXLAN datalink.

   # dladm create-vnic -l VXLAN-LINK VNIC

   You can create VLAN VNIC over a VXLAN datalink. To create a VLAN VNIC, you must specify the -f (force) option. For information, see “How to Configure VNICs With VLAN IDs” on page 26.

6. Configure an IP interface over the VNIC directly or by assigning the VNIC to a zone first.

   ■ Configure an IP interface over the VNIC.

      # ipadm create-ip VNIC

      # ipadm create-addr -a address VNIC

   ■ Assign the VNIC to a zone and configure an IP interface over the VNIC within the zone.

      a. Assign the VNIC with the zone's interface.

         zonecfg:zone> add net
         zonecfg:zone:net> set physical=VNIC
         zonecfg:zone:net> end
b. Verify and commit the changes that you have implemented and then exit the zone.

```
zonecfg:zone> verify
zonecfg:zone> commit
zonecfg:zone> exit
```

c. Reboot the zone.

```
global# zoneadm -z zone reboot
```

d. Log in to the zone.

```
global# zlogin zone
```

e. In the zone, create an IP interface over the VNIC that is now assigned to the zone.

```
zone# ipadm create-ip interface
```

f. Configure the VNIC with a valid IP address.

If you are assigning a static address to the VNIC, you would type the following:

```
zone# ipadm create-addr -a address interface

-a address         Specifies the IP address, which can be in CIDR notation.
```

g. Exit the zone.

For information about the `dladm` and `ipadm` commands, see the `dladm(1M)` and `ipadm(1M)` man pages.

**Example 3-1  Creating a VXLAN and Configuring an IP Interface for the VNIC Created Over the VXLAN**

1. Check the available IP addresses on the system.

   ```
   # ipadm show-addr net4
   ADDROBJ  TYPE  STATE   ADDR
   net4/v4  static ok   10.10.11.1/24
   ```

2. Create a VXLAN datalink in VXLAN segment 10.

   ```
   # dladm create-vxlan -p addr=10.10.11.1,vni=10 vxlan1
   ```

3. Verify the VXLAN link that you created.

   ```
   # dladm show-vxlan
   LINK  ADDR   VNI  MGROUP
   ```
vxlan1  10.10.11.1  10  224.0.0.1

Because you have not specified a multicast address, this VXLAN segment uses the All Host multicast address, which addresses all the hosts on the same network segment.

4. Check the VXLAN link information.

   # dladm show-link vxlan1
   LINK  CLASS  MTU  STATE  OVER
   vxlan1  vxlan  1440  up  --

   vxlan1 is created and the link state is up.

5. Create a VNIC over vxlan1.

   # dladm create-vnic -l vxlan1 vnic1

6. Verify the VNIC that you created.

   # dladm show-vnic
   LINK  OVER  SPEED  MACADDRESS  MACADDRTYPE  VIDS
   vnic1  vxlan1  10000  2:8:20:fe:58:d4  random  0

7. Configure an IP interface over the VNIC.

   # ipadm create-ip vnic1

   # ipadm create-addr -T static -a local=10.10.12.1/24 vnic1/v4

   # ipadm show-addr vnic1
   ADDROBJ  TYPE  STATE  ADDR
   vnic1/v4  static  ok  10.10.12.1/24

You have successfully created a VXLAN by specifying the IP address. You have created a VNIC over the VXLAN and configured the IP interface.

Example 3-2  Assigning the VNIC Created Over a VXLAN to a Zone and Configuring an IP Interface

This example assumes that you have completed steps 1 to 6 in Example 3-1.

After you create the VNIC, assign the VNIC to a zone and configure the IP interface.

   global# zonecfg -z zone2
   zonecfg:zone2> add net
   zonecfg:zone2> set physical=vnic1
   zonecfg:zone2> end
   zonecfg:zone2> verify
   zonecfg:zone2> commit
   zonecfg:zone2> exit
   global# zoneadm -z zone2 reboot
   global# zlogin zone2
You have assigned the VNIC to a zone and then configured the IP interface over the VNIC.

Displaying VXLAN Information

You can use the `dladm show-link` command to view generic link information about VXLAN links. To view information that is specific to a VXLAN, use the `dladm show-vxlan` command.

```
# dladm show-link
LINK    CLASS  MTU  STATE  OVER
net6    phys   1500  down  --
net0    phys   1500  up    --
net2    phys   1500  unknown --
net3    phys   1500  unknown --
net4    phys   1500  up    --
vxlan1  vxlan  1440  up    --
vnic1   vnic   1440  up    vxlan1
```

```
# dladm show-vxlan vxlan1
LINK     ADDR     VNI  MGROUP
vxlan1   10.10.11.1 10  224.0.0.1
```

Deleting a VXLAN

To delete a VXLAN link, use the `dladm delete-vxlan` command. Before deleting a VXLAN link, you must ensure that there are no VNICs configured on that VXLAN link by using the `dladm show-link` command.

Become an administrator and issue the following command:

```
# dladm delete-vxlan VXLAN
```

For example, if you want to delete `vxlan1`, type the following command:

```
# dladm delete-vxlan vxlan1
```
Assigning a VXLAN to a Zone

You can create zones that are a part of a VXLAN segment by assigning VXLAN as an underlying link to the zone's anet resource. For information about configuring a zone, see “Creating and Using Oracle Solaris Zones”.

 <*Problem*>

**How to Assign a VXLAN to a Zone**

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

2. **Determine the available IP addresses on the system.**

   ```
   # ipadm show-addr
   ```

3. **Create the VXLAN by specifying the IP address.**

   ```
   # dladm create-vxlan -p prop=value VXLAN-LINK
   ```

4. **Verify the VXLAN that you created.**

   ```
   # dladm show-vxlan
   ```

5. **Configure the zone by assigning the VXLAN that you created as the underlying link for the zone's anet.**

   ```
   global# zonectf -z zone
   zonectf:zone2> add anet
   zonectf:zone2:net> set linkname=datalink
   zonectf:zone2:net> set lower-link=VXLAN-LINK
   zonectf:zone2:net> end
   zonectf:zone2> verify
   zonectf:zone2> commit
   zonectf:zone2> exit
   global# zonadm -z zone reboot
   ```

   VXLAN is assigned as the underlying link for the zone's anet.

   **Example 3-3** Assigning a VXLAN to a Zone's anet

   ```
   # ipadm show-addr net4
   ADDROBJ TYPE STATE ADDR
   net4/v4 static ok 10.10.11.1/24 2
   # dladm create-vxlan -p addr=10.10.11.1,vni=10 vxlan1
   ```
Use Case: Configuring a VXLAN Over a Link Aggregation

# dladm show-vxlan

<table>
<thead>
<tr>
<th>LINK</th>
<th>ADDR</th>
<th>VNI</th>
<th>MGROUP</th>
</tr>
</thead>
<tbody>
<tr>
<td>vxlan1</td>
<td>10.10.11.1</td>
<td>10</td>
<td>224.0.0.1</td>
</tr>
</tbody>
</table>

Because you have not specified a multicast address, this VXLAN segment uses the All Host multicast address, which addresses all the hosts on the same network segment.

# dladm show-link vxlan1

<table>
<thead>
<tr>
<th>LINK</th>
<th>CLASS MTU</th>
<th>STATE</th>
<th>OVER</th>
</tr>
</thead>
<tbody>
<tr>
<td>vxlan1</td>
<td>vxlan</td>
<td>1440 up</td>
<td>--</td>
</tr>
</tbody>
</table>

vxlan1 is created and the link state is up.

global# zonecfg -z zone2
zonecfg:zone2> add anet
zonecfg:zone2:net> set linkname=net1
zonecfg:zone2:net> set lower-link=vxlan1
zonecfg:zone2:net> end
zonecfg:zone2> verify
zonecfg:zone2> commit
zonecfg:zone2> exit
global# zoneadm -z zone2 reboot

vxlan1 is assigned as the underlying link for the zone's anet.

When the zone boots up, net1 is created in zone2 over vxlan1.

Use Case: Configuring a VXLAN Over a Link Aggregation

The following use case shows how to accomplish the following:

- Create a DLMP aggregation
- Configure an IP address over the aggregation
- Create two VXLANs over the aggregation
- Configure two zones with VXLAN datalinks as the lower links

For information about link aggregation, see Chapter 2, “Configuring High Availability by Using Link Aggregations,” in “Managing Network Datalinks in Oracle Solaris 11.2.”

The following figure shows VXLAN configuration over a DLMP aggregation.
When an aggregated port or an external switch fails, VXLAN datalinks over the aggregation continue to exist as long as at least one port and a switch is functional, thereby providing network high availability during failover. For example, if net0 fails, then DLMP aggregation shares the remaining port net1, between VXLAN datalinks. The distribution among the aggregated ports occurs transparently to the user and independently of the external switches connected to the aggregation.
1. Become an administrator.
   For more information, see “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”.

2. Display datalink information to identify the datalinks for aggregation.

   ```
   # dladm show-link
   LINK   CLASS  MTU  STATE  OVER
   net0   phys   1500 up     --
   net1   phys   1500 up     --
   net2   phys   1500 up     --
   ```

3. Ensure that the datalinks that you want to aggregate do not have IP interfaces configured over the link. Delete the interface if any interface is configured on any of the links.

   ```
   # ipadm show-if
   IFNAME   CLASS  STATE  ACTIVE  OVER
   lo0      loopback ok yes     --
   net0     ip      ok no      --
   # ipadm delete-ip net0
   ```

4. Create a DLMP aggregation with the links `net0` and `net1`.

   ```
   # dladm create-aggr -m dlmp -l net0 -l net1 dlmp0
   ```

5. Configure an IP interface on top of the aggregation `dlmp0`.

   ```
   # ipadm create-ip dlmp0
   # ipadm create-addr -T static -a local=10.10.10.1 dlmp0/v4
   ```

6. Create two VXLANs by specifying the IP address that is configured over the aggregation and also specify the VNI, which is the network identifier of the VXLAN segment.

   ```
   # dladm create-vxlan -p addr=10.10.10.1,vni=20 vxlan20
   # dladm create-vxlan -p addr=10.10.10.1,vni=60 vxlan60
   ```

   Both VNIs are configured with the default multicast address.

7. Configure the zone `VM1` with the VXLAN datalink `vxlan20` as the lower-link.

   ```
   global# zonecfg -z VM1
   zonecfg:VM1> add anet
   zonecfg:VM1:net> set linkname=net0
   zonecfg:VM1:net> set lower-link=vxlan20
   zonecfg:VM1:net> end
   zonecfg:VM1> verify
   zonecfg:VM1> commit
   zonecfg:VM1> exit
   global# zoneadm -z VM1 reboot
   ```

8. Configure the zone `VM2` with the VXLAN datalink `vxlan60` as the lower-link.
Use Case: Configuring a VXL AN Over a Link Aggregation

global# zonecfg -z VM2
zonecfg:VM2> add anet
zonecfg:VM2:net> set linkname=net0
zonecfg:VM2:net> set lower-link=vxlan60
zonecfg:VM2:net> end
zonecfg:VM2> verify
zonecfg:VM2> commit
zonecfg:VM2> exit
global# zoneadm -z VM2 reboot

The net0 and net1 datalinks are aggregated into DLMP aggregation, dlmp0 and an IP address 10.10.10.1 is configured for the aggregation. The VXL ANs, vxlan20 and vxlan60 are created over the specified IP address 10.10.10.1, which is configured for the aggregation. The VXL AN, vxlan20 is created in the VXL AN segment 20 and the VXL AN, vxlan60 is created in the VXL AN segment 60. The zone VM1 is configured with the VXL AN datalink, vxlan20 as the lower link and the zone VM2 is configured with the VXL AN datalink, vxlan60 as the lower link.
Chapter 4 • Administering Server-Network Edge Virtualization by Using Edge Virtual Bridging

A server-network edge exists at the connection between a server port and its first hop switch port. Network configurations such as virtual local area network (VLAN) and Link Aggregation Control Protocol (LACP) must be the same on the server port and the switch port at this edge. You can use Data Center Bridging Capability Exchange (DCBX) to automate the configuration on the server and the switch port. For more information, see Chapter 6, “Managing Converged Networks by Using Data Center Bridging,” in “Managing Network Datalinks in Oracle Solaris 11.2”.

With server virtualization, multiple virtual ports are associated with the virtual machines (VMs) behind the server port instead of only one server port connected to a switch port. Server virtualization imposes the following additional requirements on the server-network edge:

- Switching between the virtual machines through the external switch so that inter-VM traffic is subjected to policies configured on the switch
- Extending the virtual port properties into the network

Oracle Solaris supports edge virtual bridging (EVB), which is an evolving IEEE standard that addresses these requirements.

This chapter contains the following topics:

- “EVB Support in Server-Network Edge Virtualization” on page 74
- “Improving Network and Server Efficiency by Using EVB” on page 75
- “Installing EVB” on page 78
- “Controlling Switching Between VMs Over the Same Physical Port” on page 79
- “Exchanging VNIC Information by Using VDP” on page 84
- “Displaying VDP and ECP State and Statistics” on page 86
- “Changing the Default EVB Configuration” on page 88
EVB Support in Server-Network Edge Virtualization

A virtualized server might contain multiple virtual NICs over the same physical link. You can assign these VNICs to VMs. Traditionally, a switch does not transmit packets back on the same link on which it receive the packets. Packets between VMs are looped back by the virtual switch within the host itself. Therefore, any policies that are configured on the external switch are not applied to inter-VM packets. With the support for EVB, Oracle Solaris and the switch enable inter-VM packets to be switched by the external switch after enforcing any policies on the inter-VM packets. For more information about VNICs, see Chapter 2, “Creating and Managing Virtual Networks”.

In addition, Oracle Solaris with the support of EVB can exchange information about VNICs with the switch. This exchange of information enables the switch to automatically configure the VNIC properties such as bandwidth limits, bandwidth shares, and MTU on the network. In the absence of this feature, the server administrator and the network administrator must coordinate with each other to make changes on the switch every time a VNIC is created, modified, or deleted on the server. Extending the VNIC properties into the network leads to an efficient use of networking resources based on VNIC properties. For example, enforcing a bandwidth limit on packets after they arrive at the host is not very helpful because the packets might have already used up the link bandwidth.

Reflective Relay

Reflective relay is a feature that enables VMs that are using the VNICs over the same physical NIC to communicate through the external switch. The switch must support this capability. In Oracle Solaris, LLDP is extended to include an EVB type-length value (TLV) unit, which is used to determine if the switch supports reflective relay capability and to enable or disable reflective relay capability on the switch. Therefore, you can automate the detection and configuration of this capability on the switch by using LLDP only if the switch supports LLDP and EVB TLV unit. Otherwise, reflective relay feature must be manually configured on the switch. For information about how to manually configure reflective relay, refer to the switch manufacturer's documentation.

For more information about the reflective relay support in Oracle Solaris, see “Controlling Switching Between VMs Over the Same Physical Port” on page 79. For more information about the LLDP TLV units, see “Information the LLDP Agent Advertises” in “Managing Network Datalinks in Oracle Solaris 11.2”.

Automated VNIC Configuration in the Network

Oracle Solaris uses the Virtual Station Interface Discovery and Configuration Protocol (VDP) defined in IEEE 802.1Qbg to exchange VNIC information with the switch. If the switch supports VDP, then VNIC properties are automatically configured on the switch. This is similar to the host and switch exchanging physical link properties by using DCBX. When a VNIC is created, modified, or deleted, a VDP exchange is initiated between the host and the switch. This exchange enables the switch to allocate resources for the packets destined to the VNIC based on properties of the VNIC.

For more information about exchange of VNIC information between a system and an external switch in Oracle Solaris, see “Exchanging VNIC Information by Using VDP” on page 84 and “How VDP Exchanges VNIC Information” on page 85.

Improving Network and Server Efficiency by Using EVB

This section provides an example to show how you can increase server and network efficiency when you enable EVB on a server.

This example assumes that the server hosts two applications in a cloud environment on the same physical machine.

- Applications are hosted on a cloud as separate virtual machines (VM1 and VM2) on a physical machine. The VNICs VNIC1 and VNIC2 are configured for VM1 and VM2 respectively.
- Clients (Client 1 and Client 2) with an account can access the applications.
- The virtual machines (VM1 and VM2) share the resources of the physical system and the bandwidth on link L2.
- The clients are connected to the switch by using the link L1. The switch is connected to the NIC by using the link L2.
- Predetermined SLA determines the assignment of the resource for the virtual machines. The following (L2) bandwidth usage is included for SLAs of the virtual machines:
  - VM1 is running a high priority Transmission Control Protocol (TCP) service. So, SLA for VM1 has the maximum bandwidth limit of 8 Gbps.
  - VM2 is running a User Datagram Protocol (UDP) service that is not high priority. So, SLA for VM2 has the maximum bandwidth limit of 3 Gbps.

The following figure shows the applications hosted on a server.
When you enable EVB on the server and the switch, the server exchanges the VNIC information with the switch through the same physical switch port as shown in the following figure.
The following table shows the efficiency of the server before and after enabling EVB on the server and switch.

**TABLE 4-1**  Efficiency of the Server Without EVB and With EVB

<table>
<thead>
<tr>
<th>Server Efficiency Without EVB</th>
<th>Server Efficiency With EVB</th>
</tr>
</thead>
<tbody>
<tr>
<td>The server regulates incoming traffic from the clients for bandwidth enforcement.</td>
<td>The switch regulates the traffic destined to the server.</td>
</tr>
<tr>
<td>System resources are used, thereby affecting the system and network performance.</td>
<td>System resources are not used to process the bandwidth, thereby improving the system efficiency.</td>
</tr>
<tr>
<td>In this example, when the clients (Client 1 and Client 2) need to utilize the services simultaneously, the clients use the bandwidth of link L2 and server resources. The server enforces the SLA on the VNICS for VM1 and VM2 to regulate the inbound and the outbound traffic of the clients. However, network performance and bandwidth usage are affected in the following ways:</td>
<td>When EVB is enabled on the server and the switch, system efficiency increases in the following ways:</td>
</tr>
<tr>
<td>■ Traffic from the clients (Client 1 and Client 2) use the bandwidth of link L2 without any restrictions. Also, if there is a bandwidth limit configured on the host, packets that use the bandwidth of L2 might be dropped on the host, which results in inefficient use of the bandwidth.</td>
<td>■ SLA configured on the VNICS of the server are reflected on the switch.</td>
</tr>
<tr>
<td>■ VM1 provides a high priority TCP service and VM2 provides UDP service that is not high priority. Regulating VM1’s bandwidth on the server causes TCP to respond, hence impacting VM1’s use of</td>
<td>■ Switch regulates the traffic towards VM1 and VM2 based on the configured bandwidth and therefore helps to utilize the bandwidth of link L2 appropriately, thereby providing network efficiency. Because the switch regulates the bandwidth, the server does not have to process bandwidth on the receive side, thereby providing server efficiency.</td>
</tr>
</tbody>
</table>
Installing EVB

You must install the EVB package to use EVB on your system.

▼ How to Install EVB

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 EVB package is installed.**
   
   ```bash
   # pkg info evb
   ```

3. **If the EVB package is not installed, install the package.**
   
   ```bash
   # pkg install evb
   ```

4. **Verify whether the service is enabled.**
   
   ```bash
   # svcs vdp
   ```

5. **If the service is not enabled, enable the service.**
   
   ```bash
   # svcadm enable vdp
   ```

The default EVB configuration is automatically enabled after EVB package installation. By accepting the default EVB configuration, the system can immediately exchange the information about any VNIC that you configure on the system with the external switch.

**See Also**
- To know more about exchanging VNIC information, the protocols that are used for exchanging the VNIC information, and EVB components, see “Exchanging VNIC Information by Using VDP” on page 84.
Controlling Switching Between VMs Over the Same Physical Port

You can use the `vswitchmode` `datalink` property to control switching between VMs over the same physical port. The three possible values are:

- **local** – Enables the network traffic between VMs over the same physical NIC to be exchanged internally. This is the default mode.
- **remote** – Enables the network traffic between VMs over the same physical NIC to be exchanged through the external switch.
- **auto** – Uses LLDP to determine whether reflective relay is supported on the external switch. If reflective relay is supported on the external switch, network traffic between VMs is exchanged through the external switch. Otherwise, network traffic between VMs is exchanged internally.

Enabling the VMs to Communicate Through an External Switch

When you have multiple VNICs configured over the same physical NIC, you can set the `vswitchmode` `datalink` property to `remote` to send the network traffic externally through the switch. However, the external switch must be configured in the reflective relay mode. The switch configuration that enables reflective relay is specific to the switch type. For more information, refer to the switch manufacturer's documentation.

The following figure shows a sample system with a 10G Ethernet link that is connected to an external switch and hosting two zones (VMs) that are running services for the same customer.
Because the two zones, Zone1 and Zone2, are running services for the same customer, the communication between the two zones can occur internally without any restrictions. Hence, the traffic between VNIC1 and VNIC2 can be exchanged internally.

You would check the existing value of the vswitchmode property for the physical NIC net5 as follows:

```
# dladm show-linkprop -p vswitchmode net5
```

```
LINK PROPERTY   PERM  VALUE  EFFECTIVE  DEFAULT  POSSIBLE
net4 vswitchmode rw local local local,remote,auto
```

The output displays the value `local` for the VALUE and the EFFECTIVE fields. This value indicates that the communication between the zones is internal.
In this example, assume that the two zones, Zone1 and Zone2, need to run services for different customers and the external switch has an access control list (ACL) configured that controls the network traffic for these services. Therefore, they must not communicate internally and the network traffic between VNIC1 and VNIC2 must be exchanged externally through a switch.

Hence, you must disable the internal communication between the zones by setting the vswitchmode property to remote as follows:

```
# dladm set-linkprop -p vswitchmode=remote net5
```

```
# dladm show-linkprop -p vswitchmode net5
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net5</td>
<td>vswitchmode</td>
<td>rw</td>
<td>remote</td>
<td>remote</td>
<td>local</td>
<td>local,remote,auto</td>
</tr>
</tbody>
</table>

**Note** - The external switch must be configured for reflective relay before you set the vswitchmode to remote.

Because you set the vswitchmode property to remote to disable the internal communication of the VNICs, the network traffic between the VNICs is sent through the external switch as shown in the following figure.
Using LLDP to Manage the Communication Between VMs

You can use LLDP for the automatic configuration of communication between VMs. LLDP configures the exchange of network traffic to be internal or external based on whether the
external switch supports reflective relay. To use LLDP, set the vswitchmode datalink property to auto. First, you must ensure the following:

- The LLDP package is installed.
  To check whether the LLDP package is installed, use the following command:
  
  ```
  # pkg info lldp
  ```

- The LLDP service is online.
  To check whether the LLDP service is online, use the following command:
  
  ```
  # svcs lldp
  ```

- EVB is enabled in the dot1-tlv TLV unit.

- LLDP mode is both for the NIC.
  In the example, to check whether EVB is enabled in the dot1-tlv TLV unit and the LLDP mode is both, you would use the following command:
  
  ```
  # lldpadm show-agentprop -p mode,dot1-tlv net5
  ```

To set the vswitchmode datalink property to auto:

```
# dladm set-linkprop -p vswitchmode=auto net5
```

When you set the vswitchmode datalink property to auto, you can use the output of the `dladm show-linkprop` command to check whether the communication between the VMs is internal or through an external switch.

```
# dladm show-linkprop -p vswitchmode net5
```

Since the value of the EFFECTIVE field of the output is remote, LLDP has enabled reflective relay on the external switch and the communication between the VMs is through the external switch.

For more information about LLDP, see Chapter 5, “Exchanging Network Connectivity Information With Link Layer Discovery Protocol,” in “Managing Network Datalinks in Oracle Solaris 11.2.”
Exchanging VNIC Information by Using VDP

VNIC (VSI) information is exchanged between the system (station) and the external switch (bridge) by using the VSI discovery and configuration protocol (VDP). The VDP type-length value (TLV) units are exchanged by using the Edge Control Protocol (ECP), which reliably transmits the VDP packets between the peers. The VDP TLV units are exchanged when you create or delete a VNIC.

The following EVB components enable the system to advertise the VNIC (VSI) information to the external switch:

- A VSI profile consists of link properties that have been configured for the specific VNIC. Therefore, a system can have as many VSI profiles as there are configured VNICS.
- The VSI identifier uniquely identifies a VSI instance. In Oracle Solaris, this VSI instance is the MAC address of the VNIC (VSI). The VSI Type ID and VSI Version identify the profile within a given VSI Manager ID.
- The VSI Manager manages multiple VSI profiles on the system by mapping the VSI Type ID - VSI Version with a specific set of VNIC properties. Oracle Solaris has defined a default VSI Manager, oracle_v1, as a 3-byte encoding. This 3-byte encoding is used as the VSI Type ID by an Oracle Solaris host in the VDP packet.
- A VSI Manager ID identifies the VSI Manager that is relevant to a specific VSI Type ID - VSI Version pair. The VSI Manager ID is represented as an IPv6 address. Oracle Solaris has defined a default VSI Manager ID, ORACLE_VSIMGR_V1.

**Note** - Currently, there are no defined standards for defining a VSI profile and its specific properties. The definition of VSI types is vendor-specific and is closely linked to a VSI Manager ID.

This oracle_v1 encoding supports the following properties:

- Bandwidth limit
- Bandwidth share
- Link speed of the underlying link
- Maximum transmission unit (MTU) of the VNIC

In Oracle Solaris, the system encodes the link information by using the oracle_v1 encoding and then transmits the information to the external switch. After the information is received by the switch, it decodes the encoded information by using the same oracle_v1 encoding.

By default, an Oracle Solaris host sends the following elements to the external switch:

- Oracle VSI Manager – oracle_v1
- VSI Type ID – VNIC properties encoded by using oracle_v1 encoding
In Oracle Solaris, the VNIC information exchange mechanism is as follows:

1. The external switch is configured to support the Oracle VSI Manager, oracle_v1.
2. The external switch uses oracle_v1 to determine the properties encoded in the VSI Type ID.
3. The external switch applies the property configuration on packets for that VNIC.

An Oracle organization-specific OUI TLV unit follows the VSI Manager ID TLV to indicate that it is the Oracle-specific VSI Manager ID. The absence of the Oracle-specific TLV unit in the response from the switch indicates to the Oracle Solaris host that the switch does not support Oracle VSI Manager (encodings). Oracle Switch ES1-24 supports the Oracle VSI Manager, oracle_v1. For more information about configuration of EVB on Oracle Switch ES1-24, see *Sun Ethernet Fabric Operating System, EVB Administration Guide*.

**Note** - In addition to supporting the VDP and ECP protocols, to interoperate with Oracle Solaris system, external switches must also support ORACLE_VSIMGR_V1, which is the default Oracle VSI Manager ID, and the Oracle organizationally unique identifier (OUI) TLV (subtype VDP_ORACLEOUI_VSIMGR_SUBTYPE, which is used to carry the encoding information).

## How VDP Exchanges VNIC Information

A VNIC information exchange works as follows:

The system sends an association request (ASSOC) to the external switch by specifying the VNIC and its associated profile. The external switch responds to the association request with a success or failure response. The system can subsequently send a disassociation request (DEASSOC) to the external switch, which removes the association for a VNIC. For information about how to display and obtain the state of the request for a VNIC, see “Displaying VDP and ECP State and Statistics” on page 86.

When you create a VNIC, the VDP exchange occurs as follows:

1. A VDP association (ASSOC) request TLV unit containing the information about the VNIC is sent to the external switch by the system.
2. The external switch receives the VDP (ASSOC) TLV unit and obtains the VNIC information by using the VSI Type ID, VSI Version, and VSI Manager ID.
3. The external switch applies the property configuration for the VNIC.
4. The external switch sends a VDP association (ASSOC) response TLV unit to the system stating that the external switch has configured properties for the VNIC.
When you delete a VNIC, VDP exchange occurs as follows:

1. A VDP disassociation (DEASSOC) request TLV unit containing the VSI ID is sent to the external switch by the system.
2. The external switch receives the VDP (DEASSOC) TLV unit and obtains the VSI ID of the VSI that is deleted.
3. The external switch removes the configuration for the deleted VNIC.
4. The external switch sends a VDP disassociation (DEASSOC) response TLV unit to the system.

**Note** - In Oracle Solaris, the VDP supports only ASSOC and DEASSOC VDP requests.

### Displaying VDP and ECP State and Statistics

You can display information about the VDP state for physical Ethernet links if EVB is enabled on the system and also if VDP packets are being exchanged for VNICs. To display information only for a single link, specify that link in the command. Otherwise, VDP information for all the Ethernet links is displayed.

#### Displaying the VDP State and Statistics

To display the VDP state, type the following command:

```
# dladm show-ether -P vdp
```

<table>
<thead>
<tr>
<th>VSI</th>
<th>LINK</th>
<th>VSIIID</th>
<th>VSI-TYPEID</th>
<th>VSI-STATE</th>
<th>CMD-PENDING</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic1</td>
<td>net0</td>
<td>2:8:20:22:3c:6b</td>
<td>98/0</td>
<td>ASSOC</td>
<td>NONE</td>
</tr>
<tr>
<td>vnic2</td>
<td>net0</td>
<td>2:8:20:90:7f:ef</td>
<td>96/0</td>
<td>ASSOC</td>
<td>NONE</td>
</tr>
</tbody>
</table>

**VSI-STATE** shows the status of the VDP exchange with the peer. Possible values are:

- **TIMEDOUT** – The peer has not responded to the VDP requests.
- **ASSOC** – The peer processed the request successfully.
- **DEASSOC** – Either the host or the peer has rejected the request. The peer can reject the request if it is not able to determine the profile or the properties specified. The host can reject the exchange of VDP packets if it is using oracle_v1 encoding and the peer does not include the Oracle OUI in its response.

The sample output shows that two VSIs (VNICs) are configured over the link net0. Their specific VSI IDs refer to their respective MAC addresses. The **VSI-TYPE ID** for VNICs, vnic1 and vnic2 are generated from their respective properties (bandwidth limit and MTU) and the encoding is defined by oracle_v1.
To obtain statistics about the outgoing or incoming VDP packets, type the following command:

```bash
# dlstat show-ether -P vdp net1
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>IPKTS</th>
<th>OPKTS</th>
<th>KeepAlives</th>
</tr>
</thead>
<tbody>
<tr>
<td>net1</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

**Displaying the Link Properties**

You use the `-p` option of the `dladm show-linkprop` command to display link properties.

The following example shows how to display the link properties for `vnic1` and `vnic2`.

```bash
# dladm show-linkprop -p maxbw,mtu vnic1
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic1</td>
<td>maxbw</td>
<td>rw</td>
<td>100</td>
<td>100</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>vnic1</td>
<td>mtu</td>
<td>rw</td>
<td>1500</td>
<td>1500</td>
<td>1500</td>
<td>1500</td>
</tr>
</tbody>
</table>

```bash
# dladm show-linkprop -p maxbw,mtu vnic2
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic2</td>
<td>maxbw</td>
<td>rw</td>
<td>20</td>
<td>20</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>vnic2</td>
<td>mtu</td>
<td>rw</td>
<td>1500</td>
<td>1500</td>
<td>1500</td>
<td>1500</td>
</tr>
</tbody>
</table>

**Displaying ECP State and Statistics**

VDP uses ECP to exchange messages. The following example shows state of ECP that is specific to the physical link `net0`.

```bash
# dladm show-ether -P ecp net0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>MAX-RETRIES</th>
<th>TIMEOUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>3</td>
<td>164</td>
</tr>
</tbody>
</table>

**MAX-RETRIES** Specifies the number of times ECP transmits a packet when it does not get an acknowledgement from the peer.

**TIMEOUT** Specifies the interval (in milliseconds) before retransmitting a packet. The time interval that ECP waits for an acknowledgment before retransmitting a packet.

To obtain the statistics for a physical link, type the following command:

```bash
# dlstat show-ether -P ecp
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>IPKTS</th>
<th>OPKTS</th>
<th>IERRORS</th>
<th>OERRORS</th>
<th>RETRANSMITS</th>
<th>TIMEOUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Changing the Default EVB Configuration

By default, you need not change the default EVB configuration. In most cases, you can install EVB and use the default EVB configuration to exchange the information about any VNIC that you configure on the system with the external switch. However, if you want to completely take control and manage EVB configuration on the host and the network, then you can change the default configuration.

When you use the default Oracle Solaris VSI Manager ID, ORACLE_VSIMGR_V1 the system automatically generates the VSI Type ID for the VNICs that you create. Therefore, there is no need to set the datalink properties, such as vsi-typeid and vsi-vers. However, if you are not using the default VSI Manager ID, you must set the datalink properties that are related to EVB by using the dladm set-linkprop command. To set datalink properties that are related to EVB, the external switch must be able to communicate with the system and retrieve properties for a given set of VSI Type ID and VSI Version.

Use the default Oracle VSI Manager ID when using EVB so that the Oracle VSI Manager can automatically generate VSI Type IDs and VSI Version for the VSI profiles of the system.

You can configure the following datalink properties that are related to EVB:

- **vsi-mgrid** – Specifies the VSI Manager ID that is set for a physical link or a VNIC. If this property is not set for a VNIC, the default value, ORACLE_VSIMGR_V1, of the underlying physical link is used.
  
  If you explicitly set the vsi-mgrid property, then you also need to explicitly set the VSI Type ID and VSI Version. In addition, you also need to explicitly configure these properties on the datalinks.

**Note** - In Oracle Solaris, when you manually configure the VSI Manager ID, VSI Type ID, and VSI Version, the corresponding VNIC properties are not automatically configured.

- **vsi-mgrid-enc** – Indicates the encoding that is associated with the VSI Manager ID. By default, this property is set to oracle_v1. If you do not want to associate oracle_v1 with the VSI Manager ID, set this property value to none. When you set the value none, also make sure that you configure the VSI Manager ID, VSI Type ID, and VSI Version manually because they will not be automatically generated.

- **vsi-typeid** – Specifies a VSI Type ID. A VSI Type ID pairs with a VSI Version to be associated with a VSI profile. This 3-byte value is automatically generated if you use the default values for vsi-mgrid and vsi-mgrid-enc. Otherwise, you must explicitly specify a value for this property.

- **vsi-vers** – Specifies a VSI Version. The VSI Version pairs with a VSI Type ID to be associated with a VSI profile. This 1-byte value is automatically generated if you use the
default values for vsi-mgrid and vsi-mgrid-enc. Otherwise, you must explicitly specify a value for this property.

You can display EVB-related properties by using the `dladm show-linkprop` command. You can obtain the effective values of the VNIC-related link properties from their respective EFFECTIVE field values of the properties. For more information, see Example 4-2.

For more information about the EVB components, see “Exchanging VNIC Information by Using VDP” on page 84. For more information about EVB, see the `evb(7P)` man page.

How to Change the Default EVB Configuration

You must configure the vsi-mgrid and vsi-mgrid-enc properties only on the physical link. The other EVB-related properties, such as vsi-typeid and vsi-vers, must be configured on a VNIC.

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

2. Create a VNIC by using the datalink properties mentioned in the profile database.
   ```bash
   # dladm create-vnic -l datalink -p maxbw=maxbw-value,priority=priority-value VNIC
   ```

3. Set the encoding that is associated with the VSI Manager ID to none on the physical link because you are not using the default Oracle VSI Manager ID.
   ```bash
   # dladm set-linkprop -p vsi-mgrid-enc=none datalink
   ```

4. Set the VSI Manager ID on the physical link with an IPv6 address.
   ```bash
   # dladm set-linkprop -p vsi-mgrid=IPv6-address datalink
   ```

5. Set the VSI Type ID and VSI Version for the VNIC that you have created.
   ```bash
   # dladm set-linkprop -p vsi-typeid=VSI-Type-ID,vsi-vers=VSI-Version VNIC
   ```

6. Verify the properties that are set for the VNIC.
   ```bash
   # dladm show-linkprop VNIC
   ```

Example 4-1 Setting EVB-Related Datalink Properties

The following example shows how to set datalink properties that are related to EVB. This example uses a system with a profile that you can access by using an IPv6 address, IP1.
Assume that the VSI Manager ID, IP1 has the following profiles defined:

- VSI Type ID: 2
- VSI Version: 1
- Datalink properties: maxbw=20, priority=5

1. Create a VNIC by using the datalink properties mentioned in the profile.

   ```
   # dladm create-vnic -l net0 -p maxbw=20,priority=5 vnic1
   ```

2. Set the encoding that is associated with the VSI Manager ID to none on the physical link net0 because you are not using the default Oracle VSI Manager ID.

   ```
   # dladm set-linkprop -p vsi-mgrid-enc=none net0
   ```

3. Set the VSI Manager ID on the physical link net0 with the IPv6 address IP1.

   ```
   # dladm set-linkprop -p vsi-mgrid=IP1 net0
   ```

4. Set the VSI Type ID and VSI Version for vnic1.

   ```
   # dladm set-linkprop -p vsi-typeid=2,vsi-vers=1 vnic1
   ```

5. Verify the properties that are set for vnic1.

   ```
   # dladm show-linkprop vnic1
   ```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic1</td>
<td>vsi-typeid</td>
<td>rw</td>
<td>2</td>
<td>2</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>vnic1</td>
<td>vsi-vers</td>
<td>rw</td>
<td>1</td>
<td>1</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>vnic1</td>
<td>vsi-mgrid</td>
<td>rw</td>
<td>IP1</td>
<td>IP1</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>vnic1</td>
<td>vsi-mgrid-enc</td>
<td>rw</td>
<td>--</td>
<td>none</td>
<td>oracle_v1</td>
<td>none,oracle_v1</td>
</tr>
</tbody>
</table>

The VDP ASSOC TLV unit for vnic1 contains the following information:

- VSI Manager ID = IP1
- VSI Type ID = 2
- VSI Version = 1

**Example 4-2** Displaying EVB-Related Datalink Properties on a Physical Link

The following example displays EVB-related properties on the physical link.

```
# dladm show-linkprop -p vsi-mgrid,vsi-mgrid-enc net4
```
The output displays the default configuration of EVB in Oracle Solaris. By using the oracle_v1 encoding, the VSI Type ID and VSI version are automatically generated from the properties that are configured on the VNICS.

Example 4-3 Displaying EVB-Related Properties on a VNIC

The following example displays EVB-related properties on a VNIC.

```
# dladm show-linkprop vnic0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic0</td>
<td>vsi-typeid</td>
<td>rw</td>
<td>--</td>
<td>94</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>vnic0</td>
<td>vsi-vers</td>
<td>rw</td>
<td>--</td>
<td>0</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>vnic0</td>
<td>vsi-mgrid</td>
<td>rw</td>
<td>--</td>
<td>::</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>vnic0</td>
<td>vsi-mgrid-enc</td>
<td>rw</td>
<td>--</td>
<td>oracle_v1</td>
<td>oracle_v1</td>
<td>none,oracle_v1</td>
</tr>
</tbody>
</table>

The output displays the effective encoding for vnic0 as oracle_v1. In turn, the EFFECTIVE value for vsi-typeid 94 is automatically generated and effective for vnic0.
About Elastic Virtual Switches

Starting with the Oracle Solaris 11.2 release, you can use the Oracle Solaris Elastic Virtual Switch (EVS) feature to manage multiple virtual switches that are spread across several physical machines. This chapter provides an overview of the elastic virtual switch feature in Oracle Solaris and includes the following topics:

- “Overview of the Elastic Virtual Switch (EVS) Feature” on page 93
- “EVS Components” on page 100
- “EVS Administrative Commands” on page 104
- “Mandatory Packages for Using EVS” on page 109
- “How EVS Works With Zones” on page 109
- “Security Requirements for Using EVS” on page 110

Overview of the Elastic Virtual Switch (EVS) Feature

Today’s data centers include multiple physical servers hosting several virtual machines (VMs) that are connected by a network fabric. Provisioning networking for VMs in a data center is a challenge for administrators, as it includes virtual networking between VMs, managing the MAC address and IP address, and administering VLANs and VXLANs. The additional challenge apart from ensuring internal and external network connectivity for VMs is to provision and enforce service-level agreements (SLAs) for the VMs and applications within VMs. These SLAs include bandwidth limits and priorities. Data center administrators also need to provide isolation between multiple tenants sharing a common network infrastructure.

To meet these requirements, Oracle Solaris network virtualization capabilities enable administrators to manage virtual switches across a data center. The virtual switches are exposed as first-class operating system abstractions. These virtual switches, also known as elastic virtual switches, span multiple physical servers and enable system administrators to manage them as a single virtual switch.
Virtual Switches in Oracle Solaris

The virtual switch is an entity that facilitates communication between virtual machines. In Oracle Solaris, a virtual switch is automatically or implicitly created when you create a VNIC over a datalink, such as a link aggregation, a physical NIC, or an etherstub. The virtual switch loops traffic between VMs (inter-VM traffic) within the physical machine and does not send this traffic out on the wire. All VMs need to exist on the same Layer 2 segment to communicate with each other. For more information, see “Virtual Switch” on page 15.

In releases prior to Oracle Solaris 11.2, virtual switches were indirectly managed through the datalinks over which the VNICS were created. Starting with the Oracle Solaris 11.2 release, virtual switches can be managed by EVS. You can create a virtual switch explicitly and specify a name, assign virtual ports (VPort) to the virtual switch, and associate it with a block of IP addresses. You can set properties such as priority, maximum bandwidth, class of service (CoS), MAC address, and IP address for the virtual ports. You can also configure default SLAs on a per-virtual-switch basis.

Note - Virtual switches that are implicitly created as a part of the VNIC creation continue to exist and function the same in this release as in previous releases. EVS does not replace the existing implicit virtual switch.

The following figure shows the elastic virtual switch EVS0 in a single compute node.
What Is the Oracle Solaris Elastic Virtual Switch Feature?

The Oracle Solaris Elastic Virtual Switch (EVS) feature enables you to create and administer a virtual switch that spans one or more compute nodes. These compute nodes are the physical machines that host VMs. An elastic virtual switch is an entity that represents explicitly created virtual switches that belong to the same Layer 2 (L2) segment. An elastic virtual switch provides network connectivity between VMs connected to it from anywhere in the network.

Note - In EVS, all references to the term virtual machines (VMs) specifically refer to Oracle Solaris Zones and Oracle Solaris Kernel Zones.
An elastic virtual switch can span across multiple hosts. These virtual switches are described as "elastic" because they have the capability to span into the host and span out of the host. The elastic virtual switch spans into the host when you connect the VNICS of the hosts to the elastic virtual switch. When you delete these VNICS, the elastic virtual switch spans out of the hosts.

An elastic virtual switch represents an isolated L2 segment, and the isolation is implemented through VLANs or VXLANs. For information about how you can implement an elastic virtual switch with a VLAN, see “Use Case: Configuring an Elastic Virtual Switch” on page 146. For information about how you implement an elastic virtual switch with a VXLAN, see “Use Case: Configuring an Elastic Virtual Switch for a Tenant” on page 151.

For information about administering VLANs, see Chapter 3, “Configuring Virtual Networks by Using Virtual Local Area Networks,” in “Managing Network Datalinks in Oracle Solaris 11.2”. For information about administering VXLANs, see Chapter 3, “Configuring Virtual Networks by Using Virtual Extensible Local Area Networks”.

Every elastic virtual switch is associated with a name, virtual ports, and a block of IP addresses. You can create, monitor, and control the virtual switch resources. For more information, see Chapter 6, “Administering Elastic Virtual Switches”.

The following figure shows two elastic virtual switches (EVS1 and EVS2) between two compute nodes. The VMs that are provisioned on these compute nodes are connected through the elastic virtual switches that span across the two compute nodes. Each compute node connects to the same network fabric through a datalink. The datalink is also known as an uplink port. The datalinks on these compute nodes connect the virtual switch to the external network. The VNICS is connected to the elastic virtual switch through a virtual port (VPort). The VNICS inherit properties that are associated with the virtual ports such as MAC address, IP address, and SLAs.
FIGURE 5-2 Elastic Virtual Switches Between Compute Nodes

In this figure, the VMs VM1, VM2, and VM6 can communicate with each other through the elastic virtual switch EVS1. The VMs VM3, VM4, and VM5 can communicate with each other through the elastic virtual switch EVS2. For more information, see “How to Configure an Elastic Virtual Switch” on page 128.

Benefits of Using EVS

In a data center environment that hosts several virtual machines, EVS makes some of the network administration tasks simpler by providing the following benefits:

- Creates a virtual network between VMs that are on multiple servers thus providing network connectivity
Overview of the Elastic Virtual Switch (EVS) Feature

- Supports addition of virtual ports with custom SLAs
- Provides network isolation by using VLANs or VXLANs
- Supports multitenant virtual networks that share the same underlying infrastructure
- Integrated with Oracle Solaris Zones and Oracle Solaris Kernel Zones
- Provides centralized management of:
  - MAC address and IP address for the virtual ports
  - SLAs on a per-virtual-switch or per-virtual-port basis
  - Monitoring runtime network traffic statistics of the virtual ports

Elastic Virtual Switch Resources

An elastic virtual switch is associated with the following main resources: an IP network and a virtual port.

**IP Network**

An IP network, also known as an IPnet, represents a block of IPv4 or IPv6 addresses with a default router for the block. This block of IPv4 or IPv6 addresses is also known as the subnet. You can associate only one IPnet to an elastic virtual switch. All VMs that connect to the elastic virtual switch through a virtual port are assigned an IP address from the IPnet that is associated with the elastic virtual switch.

You can also manually assign an IP address to a VM by setting the IP address property, ipaddr, for the VPort. This IP address must be within the subnet range of the IPnet. For more information about how to add an IPnet to the elastic virtual switch, see “How to Configure an Elastic Virtual Switch” on page 128.

**Virtual Port**

A virtual port, also known as a VPort, represents the point of attachment between the VNIC and an elastic virtual switch. When a VNIC connects to a VPort, the VNIC inherits the network configuration parameters that the VPort encapsulates, such as the following:

- SLA parameters such as maximum bandwidth, class of service, and priority
- MAC address
- IP address
When you create a VPort, a randomly generated MAC address and the next available IP address from the associated IPnet are assigned to the VPort. The randomly generated MAC address has a default prefix consisting of a valid IEEE OUI with the local bit set. You can also specify the IP address and the MAC address when you add a VPort by using the `evsadm add-vport` command. For more information about how to add a VPort, see “How to Configure an Elastic Virtual Switch” on page 128.

**Note** - You do not always need to add a virtual port to an elastic virtual switch. When a VNIC is created, you can specify only the name of the elastic virtual switch to which the VNIC must connect. In such cases, the EVS controller generates a system virtual port. These virtual ports follow the naming convention `sys-vportname`, for example, `sys-vport0`. The system virtual port inherits the elastic virtual switch properties.

The following table shows the VPort properties.

<table>
<thead>
<tr>
<th>VPort Property</th>
<th>Description</th>
<th>Possible Values</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cos</td>
<td>Specifies the 802.1p priority on outbound packets on the VPort.</td>
<td>0 - 7</td>
<td>--</td>
</tr>
<tr>
<td>maxbw</td>
<td>Specifies the full-duplex bandwidth for the VPort.</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>priority</td>
<td>Specifies the relative priority for the VPort.</td>
<td>high, medium, or low</td>
<td>medium</td>
</tr>
<tr>
<td>ipaddr</td>
<td>Specifies the IP address associated with the virtual port. You can assign the IP address only when you create the VPort.</td>
<td>--</td>
<td>If you do not specify the IP address for the VPort, the EVS controller automatically selects an IP address from the IPnet associated with the elastic virtual switch.</td>
</tr>
<tr>
<td>macaddr</td>
<td>Specifies the MAC address associated with the VPort. You can assign the MAC address only when you create the VPort.</td>
<td>--</td>
<td>If you do not specify the MAC address for the VPort, the EVS controller generates a random MAC address for the VPort.</td>
</tr>
<tr>
<td>evs</td>
<td>A read-only property that represents the elastic virtual switch with which the VPort is associated.</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>tenant</td>
<td>A read-only property that represents the tenant with which the VPort is associated.</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>
You cannot modify the properties evs and tenant because they are read-only properties. For more information about the VPort properties, see the evsadm(1M) man page.

**Namespace Management in EVS**

The elastic virtual switches and their resources are logically grouped together. Each logical group is called a **tenant**. The defined resources for the elastic virtual switch within a tenant are not visible outside that tenant’s namespace. The tenant acts as a container to hold all the tenant’s resources together. For more information about how to create an elastic virtual switch with a tenant, see “How to Configure an Elastic Virtual Switch” on page 128.

You do not need to specify the tenant name for any EVS operation. The default tenant name is `sys-global` and all the EVS operations occur in this namespace.

**EVS Components**

EVS has the following components:

- EVS manager
- EVS controller
- EVS clients
- EVS nodes

The following figure shows the components of EVS.
In this figure, the EVS manager and the EVS controller are two separate hosts. The EVS nodes EVS-Node1, EVS-Node2, and EVS-Node3 are three hosts whose VNICS or zone's VNIC anet resources connect to an elastic virtual switch.

**EVS Manager**

The EVS manager is the entity that communicates with the EVS controller to define the L2 network topologies and the IP addresses that must be used on these L2 networks. The EVS manager communicates with the EVS controller by using the evsadm command. The EVS manager and the EVS controller can also be on the same compute node.

*Note* - The L2 network topologies are the network segments and each segment forms a single broadcast domain, which is implemented by using VLANs or VXLANs.
You can perform EVS operations on the EVS manager after you install the `service/network/evs` package and specify the EVS controller by using the `evsadm set-prop` command. The controller property is specified in the `ssh://[user@example-controller.com` format. For more information, see Chapter 6, “Administering Elastic Virtual Switches”.

### EVS Controller

The EVS controller provides functionality for the configuration and administration of an elastic virtual switch and all the resources associated with it. You must set up only one physical machine as the EVS controller in a data center.

You specify the EVS controller by using the `controller` property with the `evsadm set-prop` command. The controller property is saved in the `svc:/network/evs:default` SMF service and therefore is persistent across system boots.

The EVS controller is associated with properties that you can configure by using the `evsadm set-controlprop` command. To implement the L2 segments across physical machines, you need to configure the properties of an EVS controller with information such as available VLAN IDs, available VXLAN segment IDs, or an uplink port for each EVS node. For more information about how to configure the EVS controller and set properties for it, see “Creating and Administering an EVS Controller” on page 114.

**Note** - You can also push the EVS controller information to each of the EVS nodes in the data center by using SMF site profiles and the Auto Install (AI) service. For more information about SMF, see “Managing System Services in Oracle Solaris 11.2”. For more information about AI service, see “Working With Install Services” in “Installing Oracle Solaris 11.2 Systems”.

The following table shows the properties that you can configure for the EVS controller.

<table>
<thead>
<tr>
<th>EVS Controller Property</th>
<th>Description</th>
<th>Possible Values</th>
<th>Default Value</th>
</tr>
</thead>
</table>
| `l2-type`               | Defines how an elastic virtual switch is implemented across physical machines.  
**Note** - When you change the `l2-type` property, the elastic virtual switches that are created prior to change are not affected. Only the elastic virtual switches that are created after the change have the | `vlan` or `vxlan`     | `vlan`        |
### EVS Controller Property
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Possible Values</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>updated l2-type</td>
<td>updated l2-type property. This behavior means that L2 segments based on VLAN and VXLAN can coexist in an EVS controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>vlan-range</td>
<td>A comma-separated list of VLAN ID ranges that are used for creating an elastic virtual switch. One VLAN ID is associated with each elastic virtual switch.</td>
<td>1 - 4094</td>
<td>--</td>
</tr>
<tr>
<td>vxlan-range</td>
<td>A comma-separated list of VXLAN segment number ranges that are used for creating an elastic virtual switch. One VXLAN segment number is associated with each elastic virtual switch.</td>
<td>0 - 16777215</td>
<td>--</td>
</tr>
<tr>
<td>vxlan-addr</td>
<td>Specifies the IP address over which the VXLAN datalink must be created. You can also set the vxlan-addr property to a subnet.</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>vxlan-mgroup</td>
<td>Specifies the multicast address that you need to use while creating the VXLAN datalinks.</td>
<td>--</td>
<td>If you do not specify the multicast address, the VXLAN datalink uses the All Host address.</td>
</tr>
<tr>
<td>vxlan-ipvers</td>
<td>Specifies the IP version of the address that you need to use for the IP interface that hosts VXLAN datalinks.</td>
<td>v4 or v6</td>
<td>v4</td>
</tr>
<tr>
<td>uplink-port</td>
<td>Specifies the datalink that you need to use for VLANS or VXLANs.</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The controller properties that you set for an EVS controller are applicable to the entire data center. However, you can override the values of the controller properties uplink-port and vxlan-addr on a per-host basis.

For example, suppose that when you set the controller properties, you set the uplink-port property to the datalink net2, which is used to create VNICs or VXLANs on every EVS node in the data center. However, if an EVS node in the data center has the datalink net1 as the only interface, you would need to override the global value net2 with a per-host value as follows:

```
# evsadm set-controlprop -h host1 -p uplink-port=net1
```

For more information, see “How to Configure an EVS Controller” on page 123.
If you do not specify a value for a controller property, the property is reset to the default value, as shown in Example 6-2. For more information about the EVS controller properties, see the `evsadm(1M)` man page.

**EVS Clients**

The `dladm` and `zonecfg` commands are the EVS clients. You can define the L2 network topologies through the `evsadm` command by using the elastic virtual switch, IPnet, and VPorts. You can use the `dladm` command to connect the VNICs to the L2 network topologies or the `zonecfg` command to connect the VNIC `anet` resource, thereby connecting the zones to the L2 network topologies.

**Note** - The `evsadm` command is the EVS manager that defines L2 network topologies.

When VNICs are created for the elastic virtual switch by using the `dladm` command or the `zonecfg` command, the configuration information for VNICs is retrieved from the EVS controller.

You can perform EVS operations on the EVS client after you install the `service/network/evs` package and specify the EVS controller by using the `controller` property with the `evsadm set-prop` command. The `controller` property is specified in the `ssh://[user@]example-controller.com` format. For more information, see Chapter 6, “Administering Elastic Virtual Switches”.

**EVS Nodes**

EVS nodes are hosts whose VNICs or zone's VNIC `anet` resources connect to an elastic virtual switch. You can use commands such as `dladm` and `zonecfg` to specify VNICs that need to be connected to an elastic virtual switch. For more information, see “Creating a VNIC for an Elastic Virtual Switch” on page 129.

**EVS Administrative Commands**

You manage an elastic virtual switch by using the following administrative commands:

- `evsadm`
- evsstat
- dladm
- zonecfg

For information about how to configure an elastic virtual switch, see “How to Configure an Elastic Virtual Switch” on page 128.

**evsadm Command**

You use the `evsadm` command to communicate with the EVS controller and manage the elastic virtual switch, IPnet, and VPorts. This section describes the subcommands you use to perform activities with this command. For more information, see the `evsadm(1M)` man page.

**evsadm Subcommands for Managing an Elastic Virtual Switch**

The `evsadm` subcommands for managing a virtual switch are:

- **create-evs**
  Creates an elastic virtual switch

- **delete-evs**
  Deletes an elastic virtual switch

- **show-evs**
  Displays information about an elastic virtual switch

- **set-evsprop**
  Enables you to set the `maxbw` and `priority` properties for an elastic switch
  For more information about these properties, see “Setting Properties for an Elastic Virtual Switch” on page 133.

- **show-evsprop**
  Displays the properties of the elastic virtual switch

**evsadm Subcommands for Managing an IPnet**

The `evsadm` subcommands for managing an IPnet are:

- **add-ipnet**
  Adds an IPnet to the elastic virtual switch and enables you to set the `subnet` and `defrouter` properties
  For more information about these properties, see “Adding an IPnet to an Elastic Virtual Switch” on page 127.
remove-ipnet  Removes an IPnet
show-ipnet    Displays information about an IPnet

**evsadm Subcommands for Managing a VPort**

The `evsadm` subcommands for managing a virtual port are:

- **add-vport**      Adds a VPort
- **remove-vport**   Removes a VPort
- **show-vport**     Displays information about a VPort
- **set-evsprop**    Enables you to set the following properties for a VPort:
  - `cos`
  - `maxbw`
  - `priority`
  For more information about these properties, see Table 5-1.
- **show-vportprop** Displays the properties of the VPort
- **reset-vport**    Resets a VPort

**evsadm Subcommands for Managing EVS Client Properties**

The `evsadm` subcommands for managing EVS client properties are:

- **set-prop**      Enables you to set the controller property
- **show-prop**     Displays EVS client properties

**evsadm Subcommands for Managing EVS Controller Properties**

The `evsadm` subcommands for managing EVS controller properties are:

- **set-controlprop** Enables you to set the following properties for the controller:
  - `l2-type`
  - `vlan-range`
  - `vxlan-range`
■ vxlan-mgroup
■ vxlan-addr
■ vxlan-ipvers
■ uplink-port

For more information about these properties, see Table 5-2.

show-controlprop Displays the properties of the EVS controller

### evsstat Command

The `evsstat` command displays the network traffic statistics for all the VPorts in a data center or for all the VPorts of the specified elastic virtual switch. It also reports the statistics of VNICs associated with the VPorts. For more information, see “Monitoring Elastic Virtual Switches” on page 143. For more information about the `evsstat` command, see the `evsstat(1M)` man page.

### dladm Command

You can administer the VNICs connected to an elastic virtual switch by using the following `dladm` commands:

- `dladm create-vnic` command – Enables you to create a VNIC and specify the elastic virtual switch name to which you need to connect the VNIC. Optionally, you can specify the VPort of the elastic virtual switch.
- `dladm show-vnic` command – Enables you to display the elastic virtual switch information for a specific VNIC. The output of the `dladm show-vnic` command also displays the fields TENANT, EVS, and VPORT. However, these fields are not visible from within a zone.

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

For more information about how to configure a VNIC for an elastic virtual switch, see “How to Create a VNIC for an Elastic Virtual Switch” on page 129.

### zonecfg Command

You use the enhanced `zonecfg` command to configure a zone's VNIC anet resource for an elastic virtual switch. You can set the following properties for the VNIC anet resource:
EVS Administrative Commands

- **tenant** – Specifies the name of the tenant. If you do not specify a value when configuring a zone, the system assigns the default value, `sys-global`.
- **vport** – Specifies the name of the VPort. If you do not specify a value when configuring a zone, the system generates a VPort for the elastic virtual switch and the VPort inherits the elastic virtual switch properties.
- **evs** – Specifies the name of an elastic virtual switch to which you must connect the VNIC `anet` resource.

For more information about the `anet` resource, see the `anet` description in “Resource Type Properties” in “Introduction to Oracle Solaris Zones”.

**Note** - Zone configuration must include the tenant name, elastic virtual switch name, and VPort name by which a VPort in a data center is uniquely identified. For more information about the zone configuration, see “Creating and Using Oracle Solaris Zones”.

For more information about how to configure the VNIC `anet` resource for an elastic virtual switch, see “Creating a VNIC `anet` Resource for an Elastic Virtual Switch” on page 130. For more information about the `zonecfg` command, see the `zonecfg(1M)` man page.

**Restrictions for Administering VNICS Connected to an Elastic Virtual Switch**

The following restrictions apply on the VNICS that you create and connect to an elastic virtual switch by using the `dladm create-vnic` command or the `zonecfg` command:

- You cannot rename the VNICS by using the `dladm rename-link` command.
- You cannot change the properties of such VNICS by using the `dladm set-linkprop` or `dladm reset-linkprop` commands.
- You cannot modify these VNICS by using the `dladm modify-vnic` command.

**Automatically Generated VXLAN Datalinks**

If you implement Layer 2 segments for elastic virtual switches by using VXLANs, EVS automatically creates VXLAN datalinks on the EVS nodes that hosts VNICS for the elastic virtual switch. These datalinks are known as automatically generated VXLAN datalinks and follow the naming convention `evs-vxlansegment-ID`, where `evs` is the entity that created the datalink. For example, the name `evs-vxlan200` indicates that 200 is the VXLAN ID and `evs` is the entity that has created this datalink. You can use the `dladm show-vxlan` command to display
the automatically generated VXLAN datalinks. For more information, see “Displaying VXLAN Information” on page 66.

You cannot use the dladm subcommands on automatically generated VXLAN datalinks to delete or rename the datalink. However, you can temporarily set the datalink properties by using the dladm set-linkprop command and the dladm reset-linkprop command.

Mandatory Packages for Using EVS

You need to install the following packages before using EVS:

- **pkg:/service/network/evs**
  You need to install the core package pkg:/service/network/evs on the EVS manager, EVS controller, and EVS nodes. This package contains the following components:
  - evsadm
  - evsstat
  - SMF service (svc:/network/evs:default) – This SMF service has the controller property that holds the hostname or the IP address of the EVS controller. The EVS client uses the hostname or the IP address to communicate with the EVS controller. You use the evsadm set-prop command to manage the controller property.

  When you install the pkg:/service/network/evs package, a new user, evsuser is created. The evsuser is a specific user with the Elastic Virtual Switch Administration rights profile. This profile provides all the required authorizations and privileges to perform EVS operations.

- **pkg:/system/management/rad/module/rad-evs-controller**
  You need to install this package only on the system that acts as an EVS controller. You must use only one controller to manage all the elastic virtual switches in a data center. This package contains the SMF service, svc:/network/evs-controller:default. This SMF service has properties that capture information that is necessary for implementing L2 segments across physical machines. You use the evsadm set-controlprop command to manage the controller properties.

  For more information, see “Mandatory Packages for an EVS Controller” on page 115.

How EVS Works With Zones

You can connect the VNIC anet resource to an elastic virtual switch by using the properties associated with the zonecfg command. Oracle Solaris Zones and Oracle Solaris Kernel Zones support the EVS feature.
Kernel zones support VNICs that you create for the elastic virtual switch. The VNIC that you create inside the kernel zone works only if the VNIC uses the factory MAC addresses that are associated with the zvnet driver. Because a VNIC that you create for the elastic virtual switch inherits the MAC address associated with the VPort of the elastic virtual switch, you must create the VPort for the elastic virtual switch by setting the macaddr property to the factory MAC address of the zvnet driver.

You use the following command syntax to explicitly specify the factory MAC address:

```bash
# evsadm add-vport -p macaddr=factory-MAC-addr-zvnet EVS-name/VPort-name
```

In the kernel zone, you can connect the VNIC to the VPort that is created by using this command. For information about kernel zones, see “Creating and Using Oracle Solaris Kernel Zones”.

## Security Requirements for Using EVS

To perform EVS operations, you need to be superuser or a user with the Elastic Virtual Switch Administration rights profile. You can also create a user and assign the Elastic Virtual Switch Administration rights profile to the user. For more information, see “Securing Users and Processes in Oracle Solaris 11.2”.

**Note** - In a multitenant EVS setup, individual tenants cannot manage their own elastic virtual switches and their resources because per-tenant user authorizations for each user is not supported. The entire EVS domain must have a single administrator who manages resources of all the tenants.

The following example shows how to create user1 with the Elastic Virtual Switch Administration rights profile.

```bash
# useradd -P "Elastic Virtual Switch Administration" user1
```

The following example shows how to add the Elastic Virtual Switch Administration rights profile to the existing user user1.

```bash
# usermod -P +"Elastic Virtual Switch Administration" user1
```

When you set the EVS controller, you must specify the user who has the Elastic Virtual Switch Administration rights profile. For example, you must specify user1 when you set the EVS controller as follows:

```bash
# evsadm set-prop -p controller=ssh://user1@example-controller.com
```

For more information, see “Configuring an EVS Controller” on page 118.
Note - You can also use evsuser that is created when you install the pkg:/service/network/evs package. The user, evsuser, is assigned with the Elastic Virtual Switch Administration rights profile. This profile provides all the required authorizations and privileges to perform EVS operations.
Chapter 6 • Administering Elastic Virtual Switches

This chapter describes tasks for administering elastic virtual switches and their resources. For general information, see Chapter 5, “About Elastic Virtual Switches”.

This chapter contains the following topics:

- “EVS Administration Tasks” on page 113
- “Planning an Elastic Virtual Switch Configuration” on page 114
- “Creating and Administering an EVS Controller” on page 114
- “Configuring Elastic Virtual Switches” on page 125
- “Administering Elastic Virtual Switches, IPnets, and VPorts” on page 131
- “Monitoring Elastic Virtual Switches” on page 143
- “Example Use Cases for Elastic Virtual Switches” on page 145

EVS Administration Tasks

This section provides the following information for accomplishing EVS administration tasks:

- “How to Configure an EVS Controller” on page 123
- “How to Configure an Elastic Virtual Switch” on page 128
- “Creating a VNIC for an Elastic Virtual Switch” on page 129
- “Displaying Elastic Virtual Switch Information” on page 132
- “Setting Properties for an Elastic Virtual Switch” on page 133
- “Displaying Properties of an Elastic Virtual Switch” on page 134
- “Removing an IPnet” on page 136
- “Displaying IPnets” on page 136
- “Setting Properties for a VPort” on page 138
- “Displaying Properties of a VPort” on page 138
- “Displaying VPorts” on page 140
- “Removing a VPort” on page 141
- “How to Delete an Elastic Virtual Switch” on page 142
- “Monitoring Elastic Virtual Switches” on page 143
Planning an Elastic Virtual Switch Configuration

Planning an elastic virtual switch configuration includes the following actions:

1. Installing the mandatory packages on the EVS controller, EVS manager, and EVS nodes. You must install these packages for each of these components separately. For more information, see “Mandatory Packages for Using EVS” on page 109.

2. Setting up the SSH authentication with the preshared public key for evsuser between the following components in the EVS setup:
   - EVS manager and the EVS controller
   - Each EVS node and the EVS controller
   - EVS controller and each EVS node
   For more information, see “Setting Up SSH Authentication” on page 118.

3. Specify the EVS controller by setting the controller property. You must specify the host name or IP address of the EVS controller on the EVS nodes, EVS manager, and EVS controller. For more information, see “Configuring an EVS Controller” on page 118.

4. Configuring the EVS controller, which involves:
   a. Setting the properties for the EVS controller.
   b. Verifying the properties that are set for the EVS controller.
   For more information, see “How to Configure an EVS Controller” on page 123.

5. Configuring the elastic virtual switch by using the EVS manager, which involves:
   a. Creating the elastic virtual switch.
   b. Adding the IPnet to the elastic virtual switch.
   c. Adding the VPort to the elastic virtual switch.
   d. Verifying the configured elastic virtual switch.
   For more information, see “How to Configure an Elastic Virtual Switch” on page 128.

6. Creating VNICs on the EVS nodes and connecting the VNICs to the elastic virtual switch, which involves:
   a. Creating VNICs by using the dladm command or creating VNIC anet resources by using the zonecfg command and connecting them to the elastic virtual switch.
   b. Verifying the VNICs that are connected to the elastic virtual switch.
   For more information, see “Creating a VNIC for an Elastic Virtual Switch” on page 129.

Creating and Administering an EVS Controller

An EVS controller provides functionality for the configuration and administration of an elastic virtual switch and all the resources associated with it. You must set properties for an EVS...
controller, which captures information necessary for implementing Layer 2 segments across physical machines. For more information, see “EVS Controller” on page 102.

Planning for an EVS controller includes the following considerations:

- Determine whether you are implementing the elastic virtual switch by using a VLAN, VXLAN, or both.
  - If you use a VLAN to implement the elastic virtual switch, you need to set the properties uplink-port and vlan-range.
  - If you use a VXLAN to implement the elastic virtual switch, you need to set the properties vxlan-range and uplink-port or vxlan-addr. Optionally, you can also set the properties vxlan-mgroup and vxlan-ipvers.

**Note** - After you create an elastic virtual switch, you cannot modify the EVS controller properties for that elastic virtual switch. Any modifications to the EVS controller properties are reflected in the new elastic virtual switches that you create.

- If the compute nodes do not have the same datalink, then for every compute node, you need to specify the datalink for the uplink-port property.
  
  For example, consider two compute nodes, host1 with the datalink net2 and host2 with the datalink net3. You need to specify the datalinks of both the hosts when you set the uplink-port property as follows:

  ```
  # evsadm set-controlprop -h host1 -p uplink-port=net2
  # evsadm set-controlprop -h host2 -p uplink-port=net3
  ```

**Mandatory Packages for an EVS Controller**

You must use only one controller to manage all the elastic virtual switches in a data center. You must install the `pkg:/service/network/evs` package and the `pkg:/system/management/rad/module/rad-evs-controller` package on the system that acts as an EVS controller.

Use the following commands to install the packages:

```
# pkg install evs
# pkg install rad-evs-controller
```

After you install the rad-evs-controller package, you need to restart the rad:local service to load the EVS controller by using the following command:

```
# svcadm restart rad:local
```
Commands for Configuring an EVS Controller

This section describes how to perform the following tasks for an EVS controller:

- Setting the EVS controller
- Displaying the EVS controller
- Setting the properties for the EVS controller
- Displaying the properties of the EVS controller

Setting the EVS Controller

You use the `evsadm set-prop` command to set the EVS controller on a host. The command syntax is:

```
# evsadm set-prop -p controller=[value[,...]]
```

This command sets the values of a property for the host where the command is executed. The only supported property is `controller`, which can be of the format `ssh://[user@]evs-controller-host-name` or `ssh://[user@]evs-controller-IP-address`.

Displaying the EVS Controller

You use the `evsadm show-prop` command to display the EVS controller. The command syntax is:

```
# evsadm show-prop [-c] [-o field[,...]] [-p controller[,...]]
```

- `-p controller` Specifies the EVS controller to which the RAD clients must connect.
- `-o field[,...]` Specifies a case-insensitive, comma-separated list of output fields to display. You can specify the following fields, which appear as columns in the output:
  
  | all    | Displays all the output fields |
  | PROPERTY | Name of the property         |
  | PERM    | Permission of the property, which is either r\* or r- |
  | VALUE   | Value of the property        |
  | DEFAULT | Default value of the property |
Display using a stable machine-parseable format. You need to specify the -o option with the -c option.

For an example that shows how to display the EVS controller, see Example 6-1.

### Setting Properties for an EVS Controller

You use the `evsadm set-controlprop` command to set the properties for the EVS controller. The command syntax is:

```
# evsadm set-controlprop [-h host] -p prop=[value[,...]]
```

- **-h host** Specifies the host for which the property is set.

- **-p prop** Specifies the name of the controller property that is set for an EVS controller. If the property takes multiple values, you must specify the values with a comma as the delimiter. You must specify only one property at a time. If the value is not specified, the property is reset to the default value. For more information about the properties that you can set for an EVS controller, see Table 5-2.

### Displaying Properties of an EVS Controller

You use the `evsadm show-controlprop` command to display the properties of an EVS controller. The command syntax is:

```
# evsadm show-controlprop [-c] -o field[,...] [-p prop[,...]]
```

This command displays the current values of one or more properties for the EVS controller. If properties are not specified for the EVS controller, then all the existing properties for the controller are displayed. For more information about the controller properties, see Table 5-2.

- **-o field[,...]** Specifies a case-insensitive, comma-separated list of output fields to display. You can specify the following fields, which appear as columns in the output:
  
  - `all` Displays all the output fields.
  - `PROPERTY` Name of the property.
  - `PERM` Permission of the property, which is either `rw` or `r-`.
  - `VALUE` Value of the property.
Creating and Administering an EVS Controller

DEFAULT

Default value of the property.

HOST

If the value is --, then the property is global and applicable to all the hosts. Otherwise, the property is applicable to the particular host.

For an example that shows how to display the properties for the EVS controller, see Example 6-1.

Configuring an EVS Controller

You must configure only one compute node as an EVS controller in your network and then set the EVS controller on each EVS node so that the EVS nodes can communicate with the EVS controller. However, you need to set the properties for the EVS controller only once from any node that can communicate with the EVS controller. You use the evsadm set-controlprop command to set the properties for the EVS controller. For more information, see “How to Configure an EVS Controller” on page 123.

You can also reset the properties for an EVS controller. Example 6-2 shows how to reset a property for an EVS controller. For information about the EVS controller and its properties, see “EVS Controller” on page 102.

To simplify the configuration of an elastic virtual switch, you need to connect as evsuser. When you install the mandatory EVS package (service/network/evs), a special user, evsuser, is created and assigned with the Elastic Virtual Switch Administration rights profile. This profile contains all the authorizations and privileges to perform the EVS operations. To use evsuser, you need to set the controller property as follows:

```
# evsadm set-prop -p controller=ssh://evsuser@evs-controller-hostname-or-IP-address
```

In addition, you must set up the SSH authentication by using the preshared public key between the host where you run the evsadm command and the EVS controller.

---

**Note** - To perform the EVS operations, you need to be superuser or a user that has the Elastic Virtual Switch Administration rights profile. For more information, see “Security Requirements for Using EVS” on page 110.

---

**Setting Up SSH Authentication**

You need SSH authentication with the preshared public key for the evsadm command to communicate with the EVS controller non-interactively and securely. You need to set up
the SSH authentication with the preshared public key for evsuser between the following components in the EVS setup:

- **EVS manager and EVS controller** – Append the public key of the administrator or the user running the evsadm command on the EVS manager in the `/var/user/evsuser/.ssh/authorized_keys` file on the EVS controller.

- **EVS nodes and EVS controller** – Append the public key of the root user on each EVS node in the `/var/user/evsuser/.ssh/authorized_keys` file on the EVS controller. You need to append these public keys because the zoneadmd daemon runs as root. This daemon connects to the EVS controller and retrieves configuration information for the VNIC `anet` resource. For more information, see the `zoneadmd(1M)` man page.

- **EVS controller and EVS nodes** – Append the public key of evsuser on the EVS controller in the `/var/user/evsuser/.ssh/authorized_keys` file on each EVS node as the EVS controller communicates with each of the EVS node for setting VPort properties.

The following figure shows the setting up of SSH authentication between the EVS components.
After you set up the SSH authentication, you need to specify the EVS controller. The assumption is that the controller property is set to ssh://evsuser@evs-controller.example.com on the EVS nodes, EVS manager, and EVS controller.

The following procedures show how to set up the SSH authentication.
How to Set Up SSH Authentication Between an EVS Node and the EVS Controller

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

2. **Generate a RSA key pair in the EVS node.**
   
   ```
   evs-node# ssh-keygen -t rsa
   Generating public/private rsa key pair.
   Enter file in which to save the key (/root/.ssh/id_rsa):
   Enter passphrase (empty for no passphrase):
   Enter same passphrase again:
   Your identification has been saved in /root/.ssh/id_rsa.
   Your public key has been saved in /root/.ssh/id_rsa.pub.
   The key fingerprint is:
   ```

3. **Copy the public key from the /root/.ssh/id_rsa.pub file in the EVS node to the /var/user/evsuser/.ssh/authorized_keys file in the EVS controller.**

4. **Log in to the EVS controller as evsuser from the EVS node to verify whether the SSH authentication is set up.**
   
   ```
   evs-node# ssh evsuser@evs-controller
   The authenticity of host 'evs-controller (192.168.100.10)' can't be established.
   Are you sure you want to continue connecting (yes/no)? yes
   Warning: Permanently added 'evs-controller' (RSA) to the list of known hosts.
   Last login: Wed Jun 11 14:36:28 2014 from evs-controller
   Oracle Corporation SunOS 5.11 11.2 April 2014
   evsuser@evs-controller$
   ```

   The output shows that you can log in to the EVS controller as evsuser without a password from the EVS node.

How to Set Up SSH Authentication Between the EVS Manager and the EVS Controller

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

2. **Generate a RSA key pair in the EVS manager.**
   
   ```
   evs-manager# ssh-keygen -t rsa
   Generating public/private rsa key pair.
   ```
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:

3. Copy the public key from the /root/.ssh/id_rsa.pub file in the EVS manager to the /var/user/evsuser/.ssh/authorized_keys file in the EVS controller.

4. Log in to the EVS controller as evsuser from the EVS manager to verify whether the SSH authentication is set up.

evs-manager# ssh evsuser@evs-controller
The authenticity of host 'evs-controller (192.168.100.10)' can't be established.
Are you sure you want to continue connecting [yes/no]? yes
Warning: Permanently added 'evs-controller' (RSA) to the list of known hosts.
Oracle Corporation SunOS 5.11 11.2 April 2014
evsuser@evs-controller$

The output shows that you can log in to the EVS controller as evsuser without a password from the EVS manager.

How to Set Up SSH Authentication Between the EVS Controller and an EVS Node

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

2. Become the user, evsuser, in the EVS controller.

evs-controller# su - evsuser
For more information, see “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”.

3. Generate a RSA key pair in the EVS controller for evsuser.

evsuser@evs-controller$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/var/user/evsuser/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/user/evsuser/.ssh/id_rsa.
Your public key has been saved in /var/user/evsuser/.ssh/id_rsa.pub.
The key fingerprint is:
4. **Copy the public key from the** `/var/user/evsuser/.ssh/id_rsa.pub` **file in the EVS controller to the** `/var/user/evsuser/.ssh/authorized_keys` **file in the EVS node.**

5. **Log in to the EVS node as** `evsuser` **from the EVS controller to verify whether the SSH authentication is set up.**

   ```
evsuser@evs-controller$ ssh evsuser@evs-node
   The authenticity of host 'evs-node (192.168.100.20)' can't be established.
   Are you sure you want to continue connecting (yes/no)? yes
   Warning: Permanently added 'evs-node' (RSA) to the list of known hosts.
   Last login: Wed Jun 11 14:40:28 2014 from evs-node
   evsuser@evs-node$
   ```

   The output shows that you can log in to the EVS node as `evsuser` without a password from the EVS controller.

**Caution** - If you do not set up the SSH authentication during the EVS setup, the `evsadm` command cannot communicate with the EVS controller non-interactively and securely.

---

**How to Configure an EVS Controller**

**Before You Begin**

Set up the SSH authentication with the preshared keys between the host where you run the `evsadm` command and the EVS controller.

1. **Become an administrator or user with the Elastic Virtual Switch Administration rights profile.**

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

2. **Set the EVS controller.**

   ```
   # evsadm set-prop [-p controller=[value[,...]]]
   ```

   This command sets the values of a property for the host where the command is executed. The only supported property is controller, which can be of the format `ssh://[user@]evs-controller-host-name` or `ssh://[user@]evs-controller-IP-address`.

3. **(Optional) Display the configured EVS controller.**

   ```
   # evsadm show-prop [-c] -o field[,...] [-p controller[,...]]
   ```
4. Set the properties for the EVS controller.

   `# evsadm set-controlprop [-h host] -p prop=value[...,[...]]`

   For more information, see “Setting Properties for an EVS Controller” on page 117.

5. (Optional) Display the properties of an EVS controller.

   `# evsadm show-controlprop [-c] [-o field[...]] [-p prop[...]]`

   For more information, see “Displaying Properties of an EVS Controller” on page 117.

Example 6-1 Configuring an EVS Controller

The following example shows how to configure the host `s11-server` as the EVS controller, whose L2 segments are created by using a VXLAN.

```
# evsadm set-prop -p controller=ssh://evsuser@s11-server
# evsadm show-prop
PROPERTY PERM    VALUE                      DEFAULT
controller rw     ssh://evsuser@s11-server   --

# evsadm set-controlprop -p l2-type=vxlan
# evsadm set-controlprop -p vxlan-range=10000-20000
# evsadm set-controlprop -p vxlan-addr=192.168.10.0/24
# evsadm set-controlprop -h s11-server -p uplink-port=net3
# evsadm set-controlprop -h s11-client -p uplink-port=net4
# evsadm show-controlprop
PROPERTY PERM VALUE               DEFAULT HOST
l2-type     rw   vxlan               vlan                --
uplink-port rw   net3                --                  s11-server
uplink-port rw   net4                --                  s11-client
vxlan-range rw   --                  --                  --
vlan-range-avail r- --                  --                  --
vxlan-addr   rw   192.168.10.0/24 0.0.0.0   --
vxlan-ipv6   rw   v4                 v4                  --
vxlan-mgroup rw   0.0.0.0             0.0.0.0             --
vxlan-range  rw   10000-20000         --                  --
vxlan-range-avail r   10000-20000         --                  --
```

In this example, the `vxlan-range-avail` property displays the VXLAN IDs (10000-20000) that are available for implementing elastic virtual switches. An IP interface that is part of the subnet 192.168.10.0/24 is used to create the VXLAN links on the EVS nodes.

The following example shows how to configure a host with the IP address 192.168.100.1 as the EVS controller, whose L2 segments are created by using a VLAN.

```
# evsadm set-prop -p controller=ssh://evsuser@192.168.100.1
# evsadm set-controlprop -p l2-type=vlan
# evsadm set-controlprop -p vlan-range=200-300,400-500
```
# evsadm set-controlprop -p uplink-port=net2
# evsadm set-controlprop -h host2.example.com -p uplink-port=net3
# evsadm set-controlprop -h host3.example.com -p uplink-port=net4

The output shows that the VLAN IDs 200-300 and 400-500 are set aside for elastic virtual switches. The datalink net2 is uplink-port on all the hosts except for host2.example.com and host3.example.com. On host2, the datalink net3 is used as uplink-port and on host3, the datalink net4 is used as uplink-port.

Example 6-2  Resetting Properties for an EVS Controller

The following example shows how to reset the controller property uplink-port.

# evsadm show-controlprop -p uplink-port
PROPERTY     PERM VALUE      DEFAULT       HOST
uplink-port  rw   net2       --            --

# evsadm set-controlprop -p uplink-port=

# evsadm show-controlprop -p uplink-port
PROPERTY     PERM VALUE      DEFAULT       HOST
uplink-port  rw   --         --            --

Configuring Elastic Virtual Switches

An elastic virtual switch is a virtual switch that spans one or more physical machines and represents an isolated L2 segment. The isolation is implemented either through VLANs or VXLANs. You can connect the VNICs or anet resources of the EVS nodes to the elastic virtual switch, thus providing network connectivity between the EVS nodes. For more information, see “What Is the Oracle Solaris Elastic Virtual Switch Feature?” on page 95.

When you plan to configure an elastic virtual switch, you need to understand your virtual topology. Determine how many L2 segments you need and the IPnet information for each network including the subnet and the default router. In addition, you might need to determine the number of virtual ports that you need to configure for the elastic virtual switch and properties that you need to specify for virtual ports.

Mandatory Package for an Elastic Virtual Switch

You must install the pkg:/service/network/evs package on the system that acts as EVS clients and EVS nodes.

Use the following command to install the package:
# pkg install evs

## Commands for Configuring an Elastic Virtual Switch

This section describes how to perform the following tasks to configure an elastic virtual switch:

- Creating an elastic virtual switch
- Adding an IPnet to an elastic virtual switch
- Adding a VPort to an elastic virtual switch

### Creating an Elastic Virtual Switch

You use the `evsadm create-evs` command to create an elastic virtual switch. The command syntax is:

```
# evsadm create-evs [-T tenant-name] [-p {prop=value[,...]}[,...]} EVS-switch-name
```

- **-T tenant-name** Specifies the tenant. If you specify a tenant, then the elastic virtual switch is created within the namespace of that tenant. Otherwise, the elastic virtual switch is created in the default tenant `sys-global`. A tenant is a read-only property that represents the tenant with which an elastic virtual switch is associated.

- **-p prop** Specifies a comma-separated list of properties that you can set to the specified values on the elastic virtual switch. You can set the following properties:
  - **maxbw** - Sets the full-duplex bandwidth for the ports of the elastic virtual switch. The bandwidth is specified as an integer with a scale suffix (K, M, or G for Kbps, Mbps, and Gbps). If units are not specified, the input value is read as Mbps. There is no default bandwidth limit.
  - **priority** - Sets the relative priority for the ports of the elastic virtual switch. The possible values are `high`, `medium`, or `low`. The default value is `medium`. The priority is not reflected in any protocol priority fields on the wire but is used for packet processing scheduling within the system. A VPort with a high priority offers more latency depending on the availability of system resources.

**EVS-switch-name** Specifies the name of the elastic virtual switch.

For an example that shows how to create an elastic virtual switch, see Example 6-3.
Adding an IPnet to an Elastic Virtual Switch

You use the `evsadm add-ipnet` command to add an IPnet to an elastic virtual switch. The command syntax is:

```
# evsadm add-ipnet [-T tenant-name] -p subnet=value{(prop=value[,...])} EVS-switch-name/IPnet-name
```

- `-T tenant-name`: Specifies the name of the tenant. If you specify the tenant name, the IPnet is associated with the EVS in the tenant namespace.

- `-p prop`: A comma-separated list of IPnet properties that you must set for the specific elastic virtual switch.

The supported properties for an IPnet are:

- `subnet`: Mandatory. Represents the block of either IPv4 or IPv6 addresses. You must specify the `subnet` property when you add an IPnet. Otherwise, adding an IPnet fails.

- `defrouter`: Optional. Specifies the gateway's IP address for the given subnet. When `defrouter` is not specified, the first address in the range is selected as the default router IP address.

```
EVS-switch-name/IPnet-name
```

For more information about IPnet properties, see the `evsadm(1M)` man page. For an example that shows how to add an IPnet to an elastic virtual switch, see Example 6-3.

Adding a VPort to an Elastic Virtual Switch

You use the `evsadm add-vport` command to add a VPort to an elastic virtual switch. The command syntax is:

```
# evsadm add-vport [-T tenant-name] [-p (prop=value[,...])...] EVS-switch-name/VPort-name
```

- `-p prop`: Specifies a comma-separated list of VPort properties that you can set for the VPort. For more information about the supported VPort properties, see Table 5-1.

```
EVS-switch-name/VPort-name
```

For an example that shows how to add a VPort to an elastic virtual switch, see Example 6-3.
How to Configure an Elastic Virtual Switch

Before You Begin
You need to set the EVS controller on the compute node on which you want to configure the elastic virtual switch. For information, see step 2 in “How to Configure an EVS Controller” on page 123.

1. **Become an administrator or user with the Elastic Virtual Switch Administration rights profile.**
   For more information, see “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”.

2. **Create an elastic virtual switch.**
   
   ```bash
   # evsadm create-evs [-T tenant-name] [-p {prop=value[,...],[...]}[,..]] EVS-switch-name
   ```
   For more information, see “Creating an Elastic Virtual Switch” on page 126.

   **Note** - If you set a property explicitly for a virtual port, that property value overrides the corresponding elastic virtual switch property value.

3. **Add an IPnet to an elastic virtual switch.**
   
   ```bash
   # evsadm add-ipnet [-T tenant-name] -p subnet=value[,{prop=value[,...],[...]}[,...]] EVS-switch-name/IPnet-name
   ```
   For more information, see “Adding an IPnet to an Elastic Virtual Switch” on page 127.

4. **(Optional) Add a VPort to an elastic virtual switch.**
   
   ```bash
   # evsadm add-vport [-T tenant-name] [-p {prop=value[,...],[...]}[,...]] EVS-switch-name/VPort-name
   ```
   When a VPort is added to the elastic virtual switch, it is assigned a random MAC address and an IP address from the IPnet address range. Therefore, you must first add an IPnet to the elastic virtual switch and then add the VPort. For more information about the `evsadm add-vport` command, see “Adding a VPort to an Elastic Virtual Switch” on page 127.

   **Note** - You do not need to always add a virtual port to an elastic virtual switch. When a VNIC is created, you can specify only the name of the elastic virtual switch to which the VNIC must connect. In such cases, the EVS controller generates a system virtual port. These virtual ports follow the naming convention `sys-vportname`, for example, `sys-vport0`. The system virtual port inherits the elastic virtual switch properties.
5. **(Optional) Display the configured elastic virtual switch.**
   
   ```bash
   # evsadm
   ```

**Example 6-3** Configuring an Elastic Virtual Switch

The following example shows how to create the elastic virtual switch ORA, add the IPnet ora_ipnet, and add the VPort vport0 to the elastic virtual switch.

```bash
# evsadm create-evs ORA
# evsadm add-ipnet -p subnet=192.168.10.0/24 ORA/ora_ipnet
# evsadm add-vport ORA/vport0
# evsadm

<table>
<thead>
<tr>
<th>NAME</th>
<th>TENANT</th>
<th>STATUS</th>
<th>VNIC</th>
<th>IP</th>
<th>HOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA</td>
<td>sys-global</td>
<td>idle</td>
<td>--</td>
<td>ora_ipnet</td>
<td>--</td>
</tr>
<tr>
<td>vport0</td>
<td>--</td>
<td>free</td>
<td>--</td>
<td>192.168.10.2/24</td>
<td>--</td>
</tr>
</tbody>
</table>
```

The following example shows how to create the elastic virtual switch ORA with the tenant tenantA, add the IPnet ora_ipnet, and add the VPort vport0 to the elastic virtual switch.

```bash
# evsadm create-evs -T tenantA ORA
# evsadm add-ipnet -T tenantA -p subnet=192.168.10.0/24 ORA/ora_ipnet
# evsadm add-vport -T tenantA ORA/vport0
# evsadm

<table>
<thead>
<tr>
<th>NAME</th>
<th>TENANT</th>
<th>STATUS</th>
<th>VNIC</th>
<th>IP</th>
<th>HOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA</td>
<td>tenantA</td>
<td>idle</td>
<td>--</td>
<td>ora_ipnet</td>
<td>--</td>
</tr>
<tr>
<td>vport0</td>
<td>--</td>
<td>free</td>
<td>--</td>
<td>192.168.10.2/24</td>
<td>--</td>
</tr>
</tbody>
</table>
```

**Creating a VNIC for an Elastic Virtual Switch**

The `dladm` and `zonecfg` commands now enable you to create VNICs for an elastic virtual switch.

**How to Create a VNIC for an Elastic Virtual Switch**

**Before You Begin**

You must set the controller property on the EVS node by using the `evsadm set-prop` command. For more information, see “How to Configure an EVS Controller” on page 123.

1. **Become an administrator or user with the Elastic Virtual Switch Administration rights profile.**
   
   For more information, see “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2 ”.

2. **Configure a VNIC for an elastic virtual switch.**
How to Create a VNIC for an Elastic Virtual Switch

# dladm create-vnic -t -c EVS-switch-name[/VPort-name] [-T tenant-name] VNIC-name

- t Specifies that the VNIC is temporary.
- c EVS-switch-name[/VPort-name] Specifies the name of the elastic virtual switch to which you must connect the VNIC. If you specify the name of the VPort, the VNIC is connected to the specified VPort. If you do not specify the VPort name, the system automatically generates a VPort and assigns the VPort to the VNIC. After the VNIC is connected to an elastic virtual switch, the VNIC either inherits the properties from the specified elastic virtual switch or VPort.
- T tenant-name Specifies the name of the tenant that owns the elastic virtual switch. If the tenant is not specified, then the system assumes default sys-global tenant.

VNIC-name The name of the VNIC.

3. (Optional) Display information about the VNICs connected to an elastic virtual switch.

# dladm show-vnic -c

The -c option displays the information about VNICs connected to an elastic virtual switch.

Example 6-4 Creating a VNIC for an Elastic Virtual Switch

This example shows how to create a temporary VNIC vnic1 and connect the VNIC to the elastic virtual switch ORA and VPort vport0.

# dladm create-vnic -t -c ORA/vport0 vnic1
# dladm show-vnic -c

Creating a VNIC anet Resource for an Elastic Virtual Switch

You can use the enhanced zonecfg command to configure a zone's VNIC anet resource for an elastic virtual switch.

You can set the following properties for the anet resource when you are configuring a zone:

- tenant – Specifies the name of the tenant. If a value is not specified when configuring a zone, the system assigns the default value, sys-global tenant.
vport – Specifies the name of the VPort. If a value is not specified when configuring a zone, a system VPort is automatically generated for the elastic virtual switch and the VPort inherits the elastic virtual switch properties.

evs – Specifies the name of an elastic virtual switch to which you must connect the anet resource.

A VPort in a data center is uniquely identified by the tenant name, elastic virtual switch name, and VPort name. For more information, see “Creating and Using Oracle Solaris Zones”.

EXAMPLE 6-5 Creating a VNIC anet Resource for an Elastic Virtual Switch

This example shows how to create a zone that has a VNIC anet resource evszone/net1, which is connected to ORA and vport0 of the tenant tenantA.

```
# zonecfg -z evszone
Use 'create' to begin configuring a new zone
zonecfg:evszone> create
create: Using system default template 'SYSdefault'
zonecfg:evszone> set zonepath=/export/zones/evszone
zonecfg:evszone> set tenant=tenantA
zonecfg:evszone> add anet
zonecfg:evszone:net> set evs=ORA
zonecfg:evszone:net> set vport=vport0
zonecfg:evszone:net> end
zonecfg:evszone> exit
# zoneadm -z evszone install
# zoneadm -z evszone boot
# zlogin -C evszone
# dladm show-vnic -c

LINK   TENANT   EVS  VPORT   OVER MACADDRESS   VIDS
evszone/net1  tenantA  ORA  vport0  net2 2:8:20:89:a1:97  200
```

When evszone boots, the VNIC anet evszone/net1 is associated with the MAC address, IP address, and SLA properties of the VPort ORA/vport0. For more information about configuring a zone's VNIC anet resources for an elastic virtual switch, see “Use Case: Configuring an Elastic Virtual Switch” on page 146.

Administering Elastic Virtual Switches, IPnets, and VPorts

This section describes how to administer an elastic virtual switch, an IPNet, and a VPort. For more information about how to configure an elastic virtual switch, IPNet, and VPort, see “Configuring Elastic Virtual Switches” on page 125.
Administering an Elastic Virtual Switch

This section describes how to perform the following tasks for an elastic virtual switch:

- Displaying information about an elastic virtual switch
- Setting properties for an elastic virtual switch
- Displaying elastic virtual switch properties

Displaying Elastic Virtual Switch Information

You use the `evsadm show-evs` command to display elastic virtual switch information. The command syntax is:

```
# evsadm show-evs [-f {fname=value[,...]}[,...]] [-L] [-c] [-o field[,...]] [EVS-switch-name]
```

- `-f {fname=value[,...]}[,...]` A comma-separated name-value pair used to filter the output (row selection). If multiple filters are specified, then the displayed output is a result of an AND operation among the filters. If the filter value is multi-valued, then the displayed output is a result of an OR operation among the filter values. The supported filters are:
  - `tenant`
  - `evs`
  - `host`
  - `ipnet`
  - `vport`

- `-L` Displays the VLAN ID or VXLAN segment ID associated with an elastic virtual switch.

- `-o field[,...]` Specifies a case-insensitive, comma-separated list of output fields to display. You can specify the following fields, which appear as columns in the output:
  - `all` Displays all the output fields.
  - `EVS` Name of the elastic virtual switch.
  - `TENANT` Name of the tenant that owns the elastic virtual switch.
  - `STATUS` Status of the elastic virtual switch, whether it is idle or busy. The elastic virtual switch is busy
Administering Elastic Virtual Switches, IPnets, and VPorts

Chapter 6 • Administering Elastic Virtual Switches

if it has at least one VPort that has a VNIC connected to it.

**NVPORTS**  
Number of virtual ports associated with the elastic virtual switch.

**IPNETS**  
The list of IP networks associated with the EVS. Currently only one IP network can be associated with an elastic virtual switch.

**HOST**  
The list of hosts that the elastic virtual switch spans across multiple servers.

**EXAMPLE 6-6**  
Displaying Elastic Virtual Switch Information

The following example displays information for the elastic virtual switch ORA.

```bash
# evsadm show-evs ORA
```

<table>
<thead>
<tr>
<th>EVS</th>
<th>TENANT</th>
<th>STATUS</th>
<th>NVPORTS</th>
<th>IPNETS</th>
<th>HOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA</td>
<td>sys-global</td>
<td>busy</td>
<td>1</td>
<td>ora_ipnet</td>
<td>s11-client</td>
</tr>
</tbody>
</table>

The following example displays the VLAN ID associated with the elastic virtual switch ORA.

```bash
# evsadm show-evs -L
```

<table>
<thead>
<tr>
<th>EVS</th>
<th>TENANT</th>
<th>VID</th>
<th>VNI</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA</td>
<td>tenantA</td>
<td>200</td>
<td>--</td>
</tr>
</tbody>
</table>

The output shows the following information:

- **EVS**: Name of the elastic virtual switch
- **TENANT**: Name of the tenant that owns the elastic virtual switch
- **VID**: VLAN ID used to implement the elastic virtual switch
- **VNI**: VXLAN segment ID used to implement the elastic virtual switch

**Setting Properties for an Elastic Virtual Switch**

You use the `evsadm set-evsprop` command to set properties for an elastic virtual switch. The command syntax is:

```bash
# evsadm set-evsprop [-T tenant-name] -p prop=value[,...] EVS-switch-name
```

- **-p prop**  
  Sets the values of a property on the specified elastic virtual switch.

EVS supports the following properties:
Administering Elastic Virtual Switches, IPnets, and VPorts

- **maxbw** – Sets the full-duplex bandwidth for all the virtual ports that connect to the specified elastic virtual switch. The bandwidth is specified as an integer with a scale suffix (K, M, or G for Kbps, Mbps, and Gbps). If no units are specified, the input value is read as Mbps. The default is no bandwidth limit.

- **priority** – Sets the default priority for all the virtual ports that connect to the specified elastic virtual switch. The possible values are high, medium, or low. The default value is medium. The priority is not reflected in any protocol priority fields on the wire but is used for packet processing scheduling within the system. A VPort with a high priority offers a better latency depending on the availability of system resources.

**EXAMPLE 6-7** Setting Properties for an Elastic Virtual Switch

This example shows how to set properties for the elastic virtual switch ORA.

```bash
# evsadm set-evsprop -p maxbw=200 ORA
# evsadm set-evsprop -p priority=high ORA
```

**Displaying Properties of an Elastic Virtual Switch**

You use the `evsadm show-evsprop` command to display the properties of an elastic virtual switch. The command syntax is:

```bash
# evsadm show-evsprop [-f {fname=value[,...]}{,...}] [[-c] -o field[,...] \ [-p prop[,...]] [EVS-switch-name]
```

- **-f**
  
  `{fname=value[,...]}{,...}`
  
  A comma-separated name-value pair used to filter the output (row selection). If multiple filters are specified, then the displayed output is a result of an AND operation among the filters. If the filter value is multivalued, then the displayed output is a result of an OR operation among the filter values. The supported filters are:

  - **tenant** – Filter the elastic virtual switch properties by the tenant name
  - **evs** – Filter the elastic virtual switch properties by the elastic virtual switch name
  - **host** – Filter the elastic virtual switch properties by the host name

**Example 6-8** shows output based on the filter value.

- **-o**
  
  `field[,...]`
  
  Specifies a case-insensitive, comma-separated list of output fields to display. You can specify the following fields, which appear as columns in the output:

  - **name**
  - **value**
  - **tenant**
  - **evs**
  - **host**
  - **maxbw**
  - **priority**
all

Displays all the output fields.

**EVS**

Name of the elastic virtual switch.

**TENANT**

Name of the tenant that owns the elastic virtual switch.

**PROPERTY**

Name of the elastic virtual switch property.

**PERM**

The read or write permissions of the property. The value shown is either `r-` or `rw`.

**VALUE**

The current property value. If the value is not set, it is shown as `--`. If the value is unknown, it is shown as `?`.

**DEFAULT**

The default value of the property. If the property has no default value, `--` is shown.

**POSSIBLE**

A comma-separated list of possible values for the property. If the possible values are unknown or unbounded, `--` is shown.

### Example 6-8 Displaying Elastic Virtual Switch Properties

The following example displays the properties configured for the elastic virtual switch ORA.

```
# evsadm show-evsprop ORA

<table>
<thead>
<tr>
<th>EVS</th>
<th>TENANT</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA</td>
<td>sys-global</td>
<td>maxbw</td>
<td>rw</td>
<td>200</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>ORA</td>
<td>sys-global</td>
<td>priority</td>
<td>rw</td>
<td>high</td>
<td>medium</td>
<td>low,medium,high</td>
</tr>
<tr>
<td>ORA</td>
<td>sys-global</td>
<td>tenant</td>
<td>r-</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>
```

The following example displays the output for the elastic virtual switches HR and ORA. In this example, the `evs` filter is specified to obtain the output for elastic virtual switches HR and ORA.

```
# evsadm show-evsprop -f evs=HR,ORA

<table>
<thead>
<tr>
<th>EVS</th>
<th>TENANT</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR</td>
<td>tenantA</td>
<td>maxbw</td>
<td>rw</td>
<td>300</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>HR</td>
<td>tenantA</td>
<td>priority</td>
<td>rw</td>
<td>--</td>
<td>medium</td>
<td>low,medium,high</td>
</tr>
<tr>
<td>HR</td>
<td>tenantA</td>
<td>tenant</td>
<td>r-</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>ORA</td>
<td>sys-global</td>
<td>maxbw</td>
<td>rw</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>ORA</td>
<td>sys-global</td>
<td>priority</td>
<td>rw</td>
<td>--</td>
<td>medium</td>
<td>low,medium,high</td>
</tr>
<tr>
<td>ORA</td>
<td>sys-global</td>
<td>tenant</td>
<td>r-</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>
```
Administering an IPnet Configuration

This section describes how to perform the following tasks for an IPnet after you add an IPnet for an elastic virtual switch:

- Removing an IPnet configured for an elastic virtual switch
- Displaying information about IPnets

Removing an IPnet

You use the `evsadm remove-ipnet` command to remove an IPnet configured for the elastic virtual switch. The command syntax is:

```
# evsadm remove-ipnet [-T tenant-name] EVS-switch-name/IPnet-name
```

This command removes the specified IPnet from the specified elastic virtual switch. You cannot remove an IPnet if any one of the VPorts is in use. A VPort is in use if it has a VNIC connected to it.

**EXAMPLE 6-9 Removing an IPnet Configured for an Elastic Virtual Switch**

This example shows how to remove the IPnet `ora_ipnet` from the elastic virtual switch `ORA`.

```
# evsadm remove-ipnet ORA/ora_ipnet
```

Displaying IPnets

You use the `evsadm show-ipnet` command to display information about IPnets managed by the EVS controller or for the specified IPnet. The command syntax is:

```
# evsadm show-ipnet [-f {fname=value[,...]}[,...]] [(-c) -o field[,...]] [IPnet-name]
```

- `-f` A comma-separated name-value pair used to filter the output (row selection). If multiple filters are specified, then the displayed output is a result of an AND operation among the filters. If the filter value is multivalued, then the displayed output is a result of an OR operation among the filter values. The supported filters are `tenant`, `evs`, `ipnet`, and `host`.

- `-o field[,...]` Specifies a case-insensitive, comma-separated list of output fields to display. You can specify the following fields, which appear as columns in the output:
Administering Elastic Virtual Switches, IPnets, and VPorts

<table>
<thead>
<tr>
<th>all</th>
<th>Displays all the output fields.</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>Name of the IPnet along with the name of the elastic virtual switch with which it is associated.</td>
</tr>
<tr>
<td>IPNET</td>
<td>Name of the IPnet.</td>
</tr>
<tr>
<td>EVS</td>
<td>Name of the elastic virtual switch.</td>
</tr>
<tr>
<td>TENANT</td>
<td>The name of the tenant that owns the elastic virtual switch.</td>
</tr>
<tr>
<td>SUBNET</td>
<td>Represents the subnet (either IPv4 or IPv6) for this IPnet.</td>
</tr>
<tr>
<td>START</td>
<td>Start address of the IP address range.</td>
</tr>
<tr>
<td>END</td>
<td>End address of the IP address range.</td>
</tr>
<tr>
<td>DEFR ROUTER</td>
<td>The IP address of the default router for the given IPnet.</td>
</tr>
<tr>
<td>AVAIL RANGE</td>
<td>A comma-separated list of available IP addresses that can be assigned to VPort.</td>
</tr>
</tbody>
</table>

**EXAMPLE 6-10** Displaying IPnet for an Elastic Virtual Switch

This example displays the IPnet configured for the elastic virtual switch ORA.

```
# evsadm show-ipnet
NAME          TENANT     SUBNET          DEFR ROUTER    AVAIL RANGE
ORA/ora_ipnet sys-global 192.168.10.0/24 192.168.10.1 192.168.10.3-192.168.10.254
```

**Administering VPort Configuration**

This section describes how to perform the following tasks for a VPort:

- Setting properties for a VPort
- Displaying properties associated with a VPort
- Displaying information about VPorts
- Resetting a VPort
- Removing a VPort
Setting Properties for a VPort

You use the `evsadm set-vportprop` command to set properties for a VPort. The command syntax is:

```
# evsadm set-vportprop [-T tenant-name] -p prop=value[,...] EVS-switch-name/VPort-name
```

- `-T tenant-name` Specifies the name of the tenant.
- `-p prop=value[,...]` Specifies the values of a property for the specified VPort. If the VPort has a VNIC connected to it, then setting the property on that VPort results in change of VNIC's property. For information about VPort properties, see Table 5-1.

**Note** - You cannot change the property of the system VPort. For more information about the system VPort, see “How to Configure an Elastic Virtual Switch” on page 128.

---

**EVS-switch-name/VPort-name** Specifies the name of the elastic virtual switch or the VPort for which the properties are set.

**Note** - You cannot modify the `ipaddr`, `macaddr`, `evs`, and `tenant` properties after you have created the VPort.

---

**EXAMPLE 6-11** Setting a Property for a VPort

This example shows how to set the maximum bandwidth property to 1G for `HR/vport0`.

```
# evsadm set-vportprop -p maxbw=1G HR/vport0
```

Displaying Properties of a VPort

You use the `evsadm show-vportprop` command to display properties of a VPort. The command syntax is:

```
# evsadm show-vportprop [-f {fname=value[,...]}[,...] [-c] -o field[,...]] [EVSwitch-name]/[VPort-name]
```

This command shows the current values of one or more properties for either all VPorts or the specified VPort. If VPort properties are not specified, then all available VPort properties are displayed. For information about the VPort properties, see Table 5-1.
[-t (name=value[,...]) [...]]

A comma-separated name-value pair used to filter the output (row selection). If multiple filters are specified, then the displayed output is a result of an AND operation among the filters. If the filter value is multivalued, then the displayed output is a result of an OR operation among the filter values. The supported filters are:
- tenant – Filter the VPort properties by the tenant name
- EVS – Filter the VPort properties by the elastic virtual switch name
- vport – Filter the VPort properties by the VPort name
- host – Filter the VPort properties by the host name

-o field[,...]

Specifies a case-insensitive, comma-separated list of output fields to display. You can specify the following fields, which appear as columns in the output:

- all
  Displays all the output fields.
- NAME
  Name of the VPort with the name of the elastic virtual switch with which the VPort is associated in the format EVS-switch-name/VPort-name.
- TENANT
  Name of the tenant that owns the elastic virtual switch.
- PROPERTY
  Name of the VPort property.
- PERM
  The read or write permissions of the property. The value shown is either r- or rw.
- VALUE
  The current property value. If the value is not set, it is shown as -. If it is unknown, the value is shown as ?.
- DEFAULT
  The default value of the property. If the property has no default value, -- is shown.
- POSSIBLE
  A comma-separated list of possible values for the property. If the values span a numeric range, min-max might be shown as shorthand. If the possible values are unknown or unbounded, -- is shown.

**EXAMPLE 6-12** Displaying VPort Properties

This example displays the VPort properties for the VPort vport0.

```
# evsadm show-vportprop ORA/vport0
```
Displaying VPorts

You use the `evsadm show-vport` command to display VPorts. The command syntax is:

```
# evsadm show-vport [-f {fname=value[,...]}[,...]] [(-c) -o field[,...]] \
[EVSwitch-name/VPort-name]
```

- `-f` A comma-separated name-value pair used to filter the output (row selection). If multiple filters are specified, then the displayed output is a result of an AND operation among the filters. If the filter value is multivalued, then the displayed output is a result of an OR operation among the filter values. The supported filters are:
  - `tenant` – Filter the VPort list by the tenant name
  - `EVS` – Filter the VPort list by the elastic virtual switch name
  - `vport` – Filter the VPort list by the VPort name
  - `host` – Filter the VPort list by the host name

- `-o field[,...]` Specifies a case-insensitive, comma-separated list of output fields to display. You can specify the following fields, which appear as columns in the output:
  - `all` Displays all the output fields.
  - `NAME` Name of the VPort with the name of the elastic virtual switch with which it is associated in the format `EVS-switch-name/VPort-name`.
  - `TENANT` Name of the tenant that owns the elastic virtual switch.
  - `STATUS` Displays whether the VPort is in use or free. A VPort is in use if the VPort is associated with a VNIC. Otherwise, the VPort is free.
  - `VNIC` Name of the VNIC associated with the VPort.
HOST Name of the host that has the VNIC associated with the VPort.

EXAMPLE 6-13  Displaying VPort Information

This example displays information about the VPort vport0.

```
# evsadm show-vport
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>TENANT</th>
<th>STATUS</th>
<th>VNIC</th>
<th>HOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA/vport0</td>
<td>sys-global</td>
<td>used</td>
<td>vnic1</td>
<td>s11-client</td>
</tr>
</tbody>
</table>

**Resetting a VPort**

When you delete a VNIC associated with a VPort, the state of the VPort is **free**. The VPort can be in the **used** state even if you delete the VNIC that is associated with the VPort in the following situations:

- The EVS node is unable to reach the EVS controller when you delete the VNIC in the EVS node.
- The VNIC associated with the VPort is not deleted before you reboot the EVS node.

To reset the state of a VPort to **free**, use the `evsadm reset-vport` command. The command syntax is:

```
# evsadm reset-vport [-T tenant-name] EVS-switch-name/VPort-name
```

**Removing a VPort**

If a VNIC is associated with the VPort, then the removal of the VPort fails. Therefore, you must first check whether a VNIC is associated with the VPort that you want to remove by using the `evsadm show-vport` command. You use the `evsadm remove-vport` command to remove a VPort from an elastic virtual switch. The command syntax is:

```
# evsadm remove-vport [-T tenant-name] EVS-switch-name/VPort-name
```

This command removes the specified VPort. When a VPort is removed, the IP address and the MAC address associated with the VPort are released.

EXAMPLE 6-14  Removing a VPort

This example shows how to remove the VPort vport0 configured for the elastic virtual switch ORA.

```
# evsadm remove-vport -T tenantA ORA/vport0
```
Deleting an Elastic Virtual Switch

This section describes how to delete an elastic virtual switch. You can delete an elastic virtual switch only when all the VPorts of an elastic virtual switch are free. Therefore, VPorts must not be associated with VNICS.

How to Delete an Elastic Virtual Switch

1. Become an administrator or user with the Elastic Virtual Switch Administration rights profile.
   For more information, see “Using Your Assigned Administrative Rights” in “Securing Users and Processes in Oracle Solaris 11.2”.

2. Check whether VPorts are used by the elastic virtual switch.
   
   ```bash
   # evsadm show-evs
   ```
   You cannot delete an elastic virtual switch if a VPort is in use. A VPort is in use if a VNIC is connected to the VPort. The STATUS field in the `evsadm show-evs` command output displays whether an elastic virtual switch is busy or idle.
   
   If a VPort is in use, you need to delete the VNIC associated with the VPort as follows:
   
   ```bash
   # dladm delete-vnic VNIC
   ```

3. Delete the elastic virtual switch.
   
   ```bash
   # evsadm delete-evs [-T tenant-name] EVS-switch-name
   ```
   This command deletes the specified elastic virtual switch and all the VPorts and the IPnet associated with the elastic virtual switch.

Example 6-15 Deleting an Elastic Virtual Switch

The following example shows how to delete the elastic virtual switch ORA.

```bash
# evsadm show-evs
EVS   TENANT    STATUS NVPORTS IPNETS    HOST
ORA   sys-global idle 0      ora_ipnet --
# evsadm delete-evs ORA
# evsadm show-evs ORA
evsadm: failed to show EVS(s): evs not found
```

The following example shows how to delete the elastic virtual switch EVS1, which is busy.

```bash
# evsadm show-evs EVS1
```
Monitoring Elastic Virtual Switches

You can monitor network traffic statistics for the virtual ports of an elastic virtual switch to obtain the following information:

- The amount of network traffic that is sent and received by a VM, which provides information about the workload on the VM.
- The number of packets that are dropped inbound (idrops) and outbound (odrops). These values provide information about faulty networks.
- The amount of network traffic that is sent and received by all the VMs on a compute node, which helps you to perform capacity planning.

You use the evsstat command to monitor elastic virtual switches. The evsstat command reports runtime statistics for each VPort of the elastic virtual switch. It also reports the statistics of VNICs associated with the VPorts. For more information about EVS and virtual ports, see the evsadm(1M) man page.

The evsstat command is a Remote Administration Daemon (RAD) client, and it communicates with a remote EVS controller to execute all the evsstat subcommands. Before using the evsstat command, you must specify a resolvable hostname or the IP address of the EVS controller by using the evsadm set-prop command. The command syntax is:

```
# evsadm set-prop -p controller=ssh://[username@]hostname-or-IP-address
```

In addition, you must set up SSH authentication by using the preshared public key between the host where you run the evsstat command and the EVS controller. You need SSH authentication with the preshared public key for the evsstat command to communicate with the EVS controller non-interactively and securely. For more information, see “Setting Up SSH Authentication” on page 118.

The command syntax for evsstat is:

```
# evsstat [-f {fname=value[,...]}[,...] [-c] -o field[,...] [-u R|K|M|G|T|P] \
[VES-switch-name]/VPort-name] [interval] [count]
```
EVS-switch-name Specifies the name of the elastic virtual switch whose statistics you want to monitor. If the name of the elastic virtual switch is not specified, statistics for all elastic virtual switches are displayed.

VPort-name Specifies the name of the VPort whose statistics you want to monitor. The statistics are displayed only for the VNIC connected to the specified VPort. You must specify the name of the elastic virtual switch and then specify the name of the VPort.

-f {fname=val[,...]} […] A comma-separated name-value pair used to filter the output (row selection). If multiple filters are specified, then the displayed output is a result of an AND operation among the filters. If the filter value is multivalued, then the displayed output is a result of an OR operation among the filter values. The supported filters are tenant, evs, and host.

-o field[,...]] Specifies a case-insensitive, comma-separated list of output fields to display. You can specify the following fields, which appear as columns in the output:

- vport
- evs
- tenant
- vnic
- host
- ipkts
- rbytes
- opkts
- idrops
- odrops

-u R|K|M|G|T|P Specifies the unit in which the statistics are displayed. If not specified, then different units, as appropriate, are used to display the statistics, using the format xy.zU, where x, y, and z are numbers and U is the appropriate unit. The supported units are:

- R – Raw count
- K – Kilobits
- M – Megabits
- G – Gigabits
- T – Terabits
- P – Petabits

interval Specifies the time in seconds at which you want to refresh the network statistics.
count Specifies the number of times to refresh the statistics. You must specify the interval and then specify the count.

**EXAMPLE 6-16 Monitoring Elastic Virtual Switches**

The following example displays statistics for all elastic virtual switches.

```bash
# evsstat
VPORT    EVS   TENANT       IPKTS   RBYTES   OPKTS   OBYTES
sys-vport0 ORA   sys-global   101.88K   32.86M   40.16K   4.37M
sys-vport2 ORA   sys-global   4.50M     6.78G    1.38M    90.90M
sys-vport0 HR    sys-global   132.89K   12.25M   236      15.82K
sys-vport1 HR    sys-global   144.47K   13.32M   247      16.29K
```

The following example displays statistics for the specified elastic virtual switch, evs0.

```bash
# evsstat ORA
VPORT    EVS   TENANT       IPKTS   RBYTES   OPKTS   OBYTES
sys-vport0 ORA   sys-global   101.88K   32.86M   40.16K   4.37M
sys-vport2 ORA   sys-global   4.50M     6.78G    1.38M    90.90M
```

The following example displays statistics for the specified VPort, evs0/sys-vport2.

```bash
# evsstat ORA/sys-vport2
VPORT    EVS   TENANT       IPKTS   RBYTES   OPKTS   OBYTES
sys-vport2 ORA   sys-global   4.50M     6.78G    1.38M    90.90M
```

The following example shows the statistics of a VPort with an interval value of 1 second and count value of 3. The statistics are refreshed three times with an interval of one second.

```bash
# evsstat ORA/sys-vport2 1 3
VPORT    EVS   TENANT       IPKTS   RBYTES   OPKTS   OBYTES
sys-vport2 ORA   sys-global   4.50M     6.78G    1.38M    90.90M
sys-vport2 ORA   sys-global   4.50M     6.78G    1.38M    90.90M
sys-vport2 ORA   sys-global   4.50M     6.78G    1.38M    90.90M
```

The following example shows the statistics for the specified output fields.

```bash
# evsstat -o vport,evs,vnic,host,ipkts,opkts
VPORT    EVS   VNIC      HOST    IPKTS   OPKTS
sys-vport0 ORA   vnic0   host1   101.88K   40.16K
sys-vport2 ORA   vnic0   host2   4.50M     1.38M
sys-vport0 HR    vnic1   host1   132.89K   236
sys-vport1 HR    vnic1   host2   144.47K   247
```

---

**Example Use Cases for Elastic Virtual Switches**

This section provides example use cases that describes how to configure an elastic virtual switch.
Use Case: Configuring an Elastic Virtual Switch

**Objective** – This use case shows how to set up an elastic virtual switch (EVS1) across two compute nodes.

In this use case, you connect the VNIC vnic0 on CN1 and the VNIC anet of the zone z1 to the elastic virtual switch EVS1 so that they are a part of the same L2 segment and they can communicate with each other on a VLAN. The following figure shows the elastic virtual switch (EVS1) across two compute nodes.

**FIGURE 6-2** Elastic Virtual Switch Configuration

The figure shows a network with four nodes that contains the following components:

- Two compute nodes (CN1 and CN2)
Example Use Cases for Elastic Virtual Switches

- Zone z1 on CN2 with the VNIC anet resource (z1/net0)
- VNIC vnic0 on CN1
- A node that acts as an EVS controller (evs-controller.example.com)
- A node that acts as an EVS manager on which you need to run the evsadm command (MANAGER)
- A VLAN to implement the elastic virtual switch EVS1
- uplink-port, which specifies the datalink that is used for the VLAN

**Note** - All the four nodes can be on a single machine. The EVS controller and EVS manager can be on the same machine.

---

### Planning for the Elastic Virtual Switch Setup

1. Install the mandatory EVS packages.
   
   For information about the required packages, see “Mandatory Packages for Using EVS” on page 109.

   **Note** - The evsuser is a specific user that is created when you install the pkg:/service/network/evs package. The user, evsuser, is assigned with the Elastic Virtual Switch Administration rights profile. This profile provides all the required authorizations and privileges to perform EVS operations.

2. Set up SSH authentication with the preshared public key for evsuser between the following components in the EVS setup:
   - The EVS manager and the EVS controller
   - Each EVS node and the EVS controller
   - The EVS controller and each EVS node
   
   For more information, see “Setting Up SSH Authentication” on page 118.

   **Note** - This use case assumes that the controller property is set to ssh://evsuser@evs-controller.example.com on the EVS node, EVS manager, and EVS controller.

3. Configure the EVS controller.
   
   a. Specify a compute node as an EVS controller in your network and then set the EVS controller on each compute node so that the compute nodes can communicate with the EVS controller. Note that you can set the controller properties from any compute node that can communicate with the EVS controller. For more information, see “Configuring Elastic Virtual Switches” on page 125.
b. Specify the properties \texttt{l2-type}, \texttt{vlan-range}, and \texttt{uplink-port}. Otherwise, you cannot create the elastic virtual switch.

4. Create an elastic virtual switch. You must associate an IPnet and add a VPort to the elastic virtual switch.

5. Create a temporary VNIC on \texttt{CN1} and connect the VNIC to the VPort of the elastic virtual switch.

6. Create a VNIC anet resource on the zone \texttt{z1} and connect it to the elastic virtual switch.

\section*{EVS Manager Operations}

1. Set the EVS controller.

\begin{verbatim}
EVS Manager Operations

1. Set the EVS controller.

   \texttt{MANAGER\# evsadm set-prop \textasciitilde p controller=ssh://evsuser@evs-controller.example.com}

2. Set the EVS controller properties.

   a. Set the type of L2 topology that must be used for the elastic virtual switch.

      \texttt{MANAGER\# evsadm set-controlprop \textasciitilde p l2-type=vlan}

   b. Set the VLAN range.

      \texttt{MANAGER\# evsadm set-controlprop \textasciitilde p vlan-range=200-300}

   c. Specify the datalinks (\texttt{uplink-port}) that are used for the VLAN.

      \texttt{MANAGER\# evsadm set-controlprop \textasciitilde p uplink-port=net2}

      \texttt{MANAGER\# evsadm set-controlprop \textasciitilde h CN2 \textasciitilde p uplink-port=net3}

\end{verbatim}

Note - You can configure the EVS controller from any node in the data center as long as you can connect to the EVS controller and have the required authorizations. For more information, see “Security Requirements for Using EVS” on page 110.

3. Verify the controller properties.

\begin{verbatim}
EVS Manager Operations

3. Verify the controller properties.

   \texttt{MANAGER\# evsadm show-controlprop \textasciitilde p l2-type,vlan-range,uplink-port}

\begin{tabular}{llll}
\hline
NAME  & VALUE & DEFAULT & HOST \\
\hline
l2-type & vlan & vlan & -- \\
vlan-range & 200-300 & -- & -- \\
uplink-port & net2 & -- & -- \\
uplink-port & net3 & -- & CN2 \\
\hline
\end{tabular}


   \texttt{MANAGER\# evsadm create-evs EVS1}

5. Add the IPnet EVS1\_ipnet to EVS1.

   \texttt{MANAGER\# evsadm add-ipnet \textasciitilde p subnet=192.168.100.0/24 EVS1/EVS1_ipnet}

\end{verbatim}
6. Add the VPort vport0 to EVS1.

**MANAGER# evsadm add-vport EVS1/vport0**

You do not need to always add a virtual port to an elastic virtual switch. When a VNIC is created, you can specify only the name of the elastic virtual switch to which the VNIC must connect. In such cases, the EVS controller generates a system virtual port. These virtual ports follow the naming convention `sys-vportname`, for example, `sys-vport0`. The system virtual port inherits the elastic virtual switch properties.

7. Verify the elastic virtual switch that is created.

**MANAGER# evsadm**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TENANT</th>
<th>STATUS</th>
<th>VNIC</th>
<th>IP</th>
<th>HOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVS1</td>
<td>sys-global</td>
<td>--</td>
<td>--</td>
<td>EVS1_ipnet</td>
<td>--</td>
</tr>
<tr>
<td>vport0</td>
<td>--</td>
<td>free</td>
<td>--</td>
<td>192.168.100.2/24</td>
<td>--</td>
</tr>
</tbody>
</table>

**Note** - Because the tenant name is not specified, the default tenant name, `sys-global` is used by the elastic virtual switch `EVS1`. You can specify the tenant name by using the `-T` option when you create an elastic virtual switch. For more information, see “How to Configure an Elastic Virtual Switch” on page 128.

8. Check the MAC address and the IP address associated with EVS1/vport0.

**MANAGER# evsadm show-vportprop -p macaddr,ipaddr EVS1/vport0**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TENANT</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVS1/vport0</td>
<td>sys-global</td>
<td>ipaddr</td>
<td>r-</td>
<td>192.168.100.2/24</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>EVS1/vport0</td>
<td>sys-global</td>
<td>macaddr</td>
<td>r-</td>
<td>2:8:20:3c:78:bd</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The VNIC that connects to vport0 will inherit the IP address and MAC address. The IP address that is assigned for vport0 is the next available IP address from the IPnet `EVS1_ipnet` and the MAC address is randomly generated for vport0.

9. Check the VLAN ID associated with the elastic virtual switch EVS1.

**MANAGER# evsadm show-evs -L**

<table>
<thead>
<tr>
<th>NAME</th>
<th>TENANT</th>
<th>VID</th>
<th>VNI</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EVS1</td>
<td>sys-global</td>
<td>200</td>
<td>--</td>
</tr>
</tbody>
</table>

**Compute Node CN1 Operations**

1. Specify the EVS controller.

**CN1# evsadm set-prop -p controller=ssh://evsuser@evs-controller.example.com**

2. Create a temporary VNIC vnic0 and connect it to EVS1/vport0.

**CN1# dladm create-vnic -t -c EVS1/vport0 vnic0**
3. Verify the VNIC that is created.

```
CN1# dladm show-vnic -c
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>TENANT</th>
<th>EVS</th>
<th>VPORT</th>
<th>OVER</th>
<th>MACADDRESS</th>
<th>VIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic0</td>
<td>sys-global</td>
<td>EVS1</td>
<td>vport0</td>
<td>net2</td>
<td>2:8:20:3c:78:bd</td>
<td>200</td>
</tr>
</tbody>
</table>

The MAC address of vnic0 maps to the MAC address of the VPort.

4. Check the allowed IP addresses for vnic0.

```
CN1# dladm show-linkprop -p allowed-ips vnic0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic0</td>
<td>allowed-ips</td>
<td>192.168.100.2</td>
<td>192.168.100.2</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The allowed-ips property is set to the IP address associated with the VPort. With this setting, you cannot create any other IP address on vnic0 other than 192.168.100.2.

5. Create an IP interface for vnic0 and assign 192.168.100.2 as the IP address.

```
# ipadm create-ip -t vnic0
# ipadm create-addr -t -a 192.168.100.2 vnic0
```

**Compute Node CN2 Operations**

1. Specify the EVS controller.

```
CN2# evsadm set-prop -p controller=ssh://evsuser@evs-controller.example.com
```

2. Configure the VNIC anet resource for the zone z1 and connect it to the elastic virtual switch.

```
CN2# zonecfg -z z1
Use 'create' to begin configuring a new zone
zonecfg:z1> create
create: Using system default template 'SYSdefault'
zonecfg:z1> set zonepath=/export/zones/z1
zonecfg:z1> select anet linkname=net0
zonecfg:z1:anet> set evs=EVS1
zonecfg:z1:anet> end
zonecfg:z1> commit
zonecfg:z1> exit
```

3. Install and boot the zone z1.

```
CN2# zonesadm -z z1 install
CN2# zonesadm -z z1 boot
```

4. Log in to the zone z1 and complete the zone configuration.

```
CN2# zlogin -C z1
```
For more information about zone configuration, see “Creating and Using Oracle Solaris Zones”.

5. Verify the VNIC anet that is created.

```
CN2# dladm show-vnic -c
LINK       TENANT       EVS    VPORT        OVER   MACADDRESS        VIDS
z1/net0    sys-global   EVS1   sys-vport0   net2   2:8:20:1a:c1:e4   200
```

Because the VPort was not specified when you created the VNIC anet resource, the EVS controller creates a system VPort, sys-vport0, for the VNIC anet resource.

6. Display the information that is related to the VPort.

```
CN2# evsadm show-vport -o all
NAME            TENANT     STATUS VNIC    HOST MACADDR         IPADDR
EVS1/sys-vport0 sys-global used   z1/net0 CN2  2:8:20:1a:c1:e4 192.168.100.3/24
```

The VNIC anet resource is plumbed and assigned the VPort's IP address.

7. Verify the IP address of the VNIC anet resource, z1/net0.

```
CN2# zlogin z1 ipadm
NAME              CLASS/TYPE    STATE        UNDER      ADDR
lo0               loopback      ok           --         --
lo0/v4         static        ok           --         127.0.0.1/8
lo0/v6         static        ok           --         ::1/128
net0              ip            ok           --         --
net0/v4     inherited     ok           --         192.168.100.3/24
```

**Use Case: Configuring an Elastic Virtual Switch for a Tenant**

**Objective** – This use case shows how to set up an elastic virtual switch (HR) across two compute nodes for a tenant.

In this use case, you connect the VNIC vnic0 on CN1 and the VNIC anet of the zone z1 to the elastic virtual switch HR, so that they are a part of the same L2 segment and they can communicate with each other on a VXLAN. The VNICS are part of the tenant tenantA. The following figure shows the EVS setup.
The figure shows a network with four nodes that contains the following components:

- Two compute nodes (CN1 and CN2)
- Zone z1 on CN2 with a VNIC anet resource
- VNIC vnic0 on CN1
- A node that acts as an EVS controller, CONTROLLER
- A node that acts as an EVS manager on which you need to run the evsadm command, MANAGER
- A VXLAN to implement the elastic virtual switch HR
- uplink-port that specifies the datalink that is used for the VXLANs
Planning for the Elastic Virtual Switch Setup

1. Install the mandatory EVS packages. For information about the required packages, see “Mandatory Packages for Using EVS” on page 109.

Note - The evsuser is a specific user that is created when you install the pkg:/service/network/evs package. The user, evsuser, is assigned with the Elastic Virtual Switch Administration rights profile. This profile provides all the required authorizations and privileges to perform EVS operations.

2. Set up SSH authentication with the preshared public key for evsuser between the following components in the EVS setup:
   - The EVS manager and the EVS controller
   - Each EVS node and the EVS controller
   - The EVS controller and each EVS node
   For more information, see “Setting Up SSH Authentication” on page 118.

Note - This use case assumes that the controller property is set to ssh://evsuser@evs-controller.example.com on each of the EVS node, EVS manager, and EVS Controller.

3. Configure the EVS controller and set the controller properties.
   a. Set the EVS controller on all the compute nodes and then set the controller properties that specify how to implement the elastic virtual switch across the compute nodes.
   b. Specify the properties l2-type, vxlan-range, and uplink-port. Otherwise, you cannot create the elastic virtual switch.

4. Create an elastic virtual switch. You must associate an IPnet and add a VPort to the elastic virtual switch.

5. Create a temporary VNIC on CN1 and connect the VNIC to the VPort of the elastic virtual switch.

6. Create a VNIC anet on the zone z1 and connect the VNIC anet resource to the elastic virtual switch.

EVS Manager Operations

1. Set the EVS controller.

   MANAGER# evsadm set-prop -p controller=ssh://evsuser@evs-controller.example.com

2. Set the EVS controller properties.
   a. Set the type of L2 topology that must be used for the elastic virtual switch. This example uses a VXLAN.
MANAGER# evsadm set-controlprop -p l2-type=vxlan

b. Set the VXLAN range.

MANAGER# evsadm set-controlprop -p vxlan-range=200-300

c. Specify the datalinks (uplink-port) that are used for the VXLAN.

MANAGER# evsadm set-controlprop -p uplink-port=net2

MANAGER# evsadm set-controlprop -h CN2 -p uplink-port=net3

Note - You can configure the controller from any node in the data center as long as you can connect to the EVS controller and have the required authorizations. For more information, see “Security Requirements for Using EVS” on page 110.

3. Verify the EVS controller properties.

MANAGER# evsadm show-controlprop -p l2-type,vxlan-range,uplink-port

<table>
<thead>
<tr>
<th>NAME</th>
<th>VALUE</th>
<th>DEFAULT</th>
<th>HOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>l2-type</td>
<td>vxlan</td>
<td>vlan</td>
<td>--</td>
</tr>
<tr>
<td>vxlan-range</td>
<td>200-300</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>uplink-port</td>
<td>net2</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>uplink-port</td>
<td>net3</td>
<td>--</td>
<td>CN2</td>
</tr>
</tbody>
</table>

4. Create the elastic virtual switch HR for the tenant tenantA.

MANAGER# evsadm create-evs -T tenantA HR

5. Add the IPnet hr_ipnet to the elastic virtual switch HR.

MANAGER# evsadm add-ipnet -T tenantA -p subnet=192.168.100.0/24 HR/hr_ipnet

6. Add the VPort vport0 to the elastic virtual switch HR.

MANAGER# evsadm add-vport -T tenantA HR/vport0

7. Verify the elastic virtual switch that was created for the tenant tenantA.

MANAGER# evsadm

<table>
<thead>
<tr>
<th>NAME</th>
<th>TENANT</th>
<th>STATUS</th>
<th>VNIC</th>
<th>IP</th>
<th>HOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR</td>
<td>tenantA</td>
<td>--</td>
<td>--</td>
<td>hr_ipnet</td>
<td>--</td>
</tr>
<tr>
<td>vport0</td>
<td>--</td>
<td>free</td>
<td>--</td>
<td>192.168.100.2/24</td>
<td>--</td>
</tr>
</tbody>
</table>

8. Check the MAC address and the IP address associated with HR/vport0.

MANAGER# evsadm show-vportprop -p macaddr,ipaddr HR/vport0

<table>
<thead>
<tr>
<th>NAME</th>
<th>TENANT</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR/vport0</td>
<td>tenantA</td>
<td>ipaddr</td>
<td>r</td>
<td>192.168.100.2/24</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>HR/vport0</td>
<td>tenantA</td>
<td>macaddr</td>
<td>r</td>
<td>2:8:20:0d:da:10</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

9. Check the VXLAN segment ID associated with the elastic virtual switch HR.
**Compute Node CN1 Operations**

1. Specify the EVS controller.

   ```
   CN1# evsadm set-prop -p controller=ssh://evsuser@evs-controller.example.com
   ```

2. Create a temporary VNIC vnic0 and connect it to the elastic virtual switch HR/vport0.

   ```
   CN1# dladm create-vnic -t -T tenantA -c HR/vport0 vnic0
   ```

3. Verify the VNIC that was created.

   ```
   CN1# dladm show-vnic -c
   ```

4. Check the allowed IP addresses for vnic0.

   ```
   CN1# dladm show-linkprop -p allowed-ips vnic0
   ```

5. Create an IP interface for vnic0 and assign 192.168.100.2 as the IP address.

   ```
   # ipadm create-ip -t vnic0
   # ipadm create-addr -t -a 192.168.100.2 vnic0
   ```

6. Check the automatically generated VXLAN datalink.

   ```
   CN1# dladm show-vxlan
   ```

**Compute Node CN2 Operations**

1. Specify the EVS controller.

   ```
   CN2# evsadm set-prop -p controller=ssh://evsuser@evs-controller.example.com
   ```

2. Configure the VNIC anet for the zone z1 and connect it to the elastic virtual switch.
Example Use Cases for Elastic Virtual Switches

3. Install and boot the zone z1.

```bash
cn2# zoneadm -z z1 install
cn2# zoneadm -z z1 boot
```

4. Log in to the zone z1 and complete the zone configuration.

```bash
cn2# zlogin -C z1
```

For more information about zone configuration, see “Creating and Using Oracle Solaris Zones”.

5. Verify the VNIC anet resource that was created.

```bash
cn2# dladm show-vnic -c
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>TENANT</th>
<th>EVS</th>
<th>VPORT</th>
<th>OVER</th>
<th>MACADDRESS</th>
<th>VIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>z1/net0</td>
<td>tenantA</td>
<td>HR</td>
<td>sys-vport0</td>
<td>evs-vxlan200</td>
<td>2:8:20:1a:c1:e4</td>
<td>0</td>
</tr>
</tbody>
</table>

Because the VPort is not specified, the EVS controller creates a system VPort sys-vport0 for the VNIC anet resource.

6. Display the information that is related to the VPort.

```bash
cn2# evsadm show-vport -o all
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>TENANT</th>
<th>STATUS</th>
<th>VNIC</th>
<th>HOST MACADDR</th>
<th>IPADDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR/sys-vport0</td>
<td>tenantA</td>
<td>used</td>
<td>z1/net0</td>
<td>CN2</td>
<td>192.168.100.3/24</td>
</tr>
</tbody>
</table>

The VNIC anet resource is plumbed and assigned the VPort's IP address.

7. Verify the IP address of the VNIC anet z1/net0.

```bash
cn2# zlogin z1 ipadm
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>CLASS/TYPE</th>
<th>STATE</th>
<th>UNDER</th>
<th>ADDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>lo0</td>
<td>loopback</td>
<td>ok</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>lo0/v4</td>
<td>static</td>
<td>ok</td>
<td>--</td>
<td>127.0.0.1/8</td>
</tr>
<tr>
<td>lo0/v6</td>
<td>static</td>
<td>ok</td>
<td>--</td>
<td>::1/128</td>
</tr>
<tr>
<td>net0</td>
<td>ip</td>
<td>ok</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net0/v4</td>
<td>inherited</td>
<td>ok</td>
<td>--</td>
<td>192.168.100.3/24</td>
</tr>
</tbody>
</table>
Managing Network Resources

This chapter explains how to manage and allocate network resources by using datalink properties and flows. By managing network resources, you can implement IP quality of service (QoS) that enhances the performance of the virtual network and physical network. For an introduction to network resource management, see “Overview of Network Resource Management” on page 19.

This chapter contains the following topics:

- “Managing Network Resources by Using Datalink Properties” on page 157
- “Managing NIC Rings” on page 158
- “Managing Pools and CPUs” on page 167
- “Managing Network Resources by Using Flows” on page 172
- “Use Case: Managing Network Resources by Setting Datalink and Flow Properties” on page 176

Managing Network Resources by Using Datalink Properties

You can allocate network resources to datalinks to increase the system’s efficiency to process packets. You can allocate network resources by setting datalink properties when you create a datalink. Alternatively, you can set datalink properties to an existing datalink. You can set the following datalink properties to allocate network resources to a datalink by using the `dladm` command:

- ```maxbw``` – Specifies the maximum amount of bandwidth that you can allocate to a datalink. For more information, see “Use Case: Managing Network Resources by Setting Datalink and Flow Properties” on page 176.
- ```rxrings``` and ```txrings``` – Specifies the number of receive rings and transmit rings of a NIC that you can assign to a specific datalink. For more information, see “Managing NIC Rings” on page 158.
- ```pool``` – Specifies the name of the CPU pool containing sets of CPU that you can assign to a datalink to manage network processes efficiently. For more information, see “Managing Pools and CPUs” on page 167.
Managing NIC Rings

- **cpus** – Specifies the name of the CPUs that you can assign to a datalink. For more information, see “Managing Pools and CPUs” on page 167.

For a demonstration of managing network resources in Oracle Solaris, see Managing Network Resources Using Oracle Solaris (http://www.oracle.com/webfolder/technetwork/tutorials/tutorial/solaris/11/ManagingNetworkResources/ManagingNetworkResources.htm).

### Commands for Allocating Resources in Datalinks

The following commands are used for allocating network resources in datalinks:

- To simultaneously create a virtual link and allocate resources to it, use the following command syntax:

  ```
  # dladm create-vnic -l link -p prop=value[,...] VNIC
  ```

  - **link** Refers to the name of the link which can be either a physical link or a virtual link.
  - **prop** Refers to the datalink property. For information about the different types of datalink properties that can be set for resource allocation, see “Managing Network Resources by Using Datalink Properties” on page 157.

- To set the property for an existing link, use the following command syntax:

  ```
  # dladm set-linkprop -p prop=value[,...] link
  ```

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

### Managing NIC Rings

On NICs, receive (Rx) rings and transmit (Tx) rings are hardware resources through which the system receives and sends network packets, respectively. By managing and allocating rings according to the network traffic, you increase the system's efficiency for processing packets. For example, you can allocate more number of receive (Rx) rings for a link that is receiving more packets.
Allocating Rings in MAC Clients

MAC clients such as physical datalinks and VNICS are configured over a NIC to enable communication between a system and other network nodes. A MAC client can be either a hardware-based client or a software-based client.

Hardware-based Clients

Clients that have exclusive use of one or more NIC rings are called hardware-based clients. You can assign rings for exclusive use by hardware-based clients depending on the ring allocation supported by the NICs.

Software-based Clients

Clients that do not have exclusive use of NIC rings are called software-based clients. They share rings with other existing software-based clients or with the primary client. The rings that the software-based clients use depend on the number of hardware-based clients that have priority in ring allocation.

Allocating Rings in VLANs

Ring allocation in VLANs differs based on how the VLAN is created.

You can create a VLAN in the following ways:

- By using the `dladm create-vlan` command:
  
  ```
  # dladm create-vlan -l link -v vid VLAN
  ```

  If you create a VLAN by using the `dladm create-vlan` command, it shares the same MAC address as the underlying datalink. Therefore, the VLAN also shares the Rx and Tx rings of the underlying datalink. For more information about configuring VLANs, see “Configuring a VLAN” in “Managing Network Datalinks in Oracle Solaris 11.2 ”.

- By using the `dladm create-vnic` command:
  
  ```
  # dladm create-vnic -l link -v vid VNIC
  ```

  If you create a VLAN as a VNIC by using the `dladm create-vnic` command, it has a different MAC address from its underlying datalink. The allocation of rings for this type of VLAN is independent of the allocation of the underlying datalink. Hence, the VLAN can be assigned its own dedicated rings, assuming that the NIC supports hardware-based clients. For more information about how to assign rings to clients, see “Configuring Clients and Allocating Rings” on page 162.
Commands for Configuring Rings

To configure the rings of a datalink, use the following `dladm` subcommands:

- `# dladm show-linkprop link`

Displays the current values of the datalink properties, including Rx and Tx rings. For an example, see Example 7-1.

The following table describes the ring properties that are displayed by using the `dladm show-linkprop` command.

<table>
<thead>
<tr>
<th>Ring Property</th>
<th>Permission</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>rxringsavail</code></td>
<td>Read only</td>
<td>Indicates the number of Rx rings that you can allocate to hardware-based clients on the physical datalink.</td>
</tr>
<tr>
<td><code>rxhwclntavail</code></td>
<td>Read only</td>
<td>Indicates the number of hardware-based Rx clients that you can create on the physical datalink.</td>
</tr>
<tr>
<td><code>rxrings</code></td>
<td>Read and write</td>
<td>Indicates the number of Rx rings exclusively used by the datalink. You can set this property to one of the three possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>hw</code> indicates that you are configuring a hardware-based client. You can set this value, if the hardware-based Rx clients (<code>rxhwclntavail</code>) on the underlying physical link is greater than zero.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>number</code> indicates the number of rings that you can assign to a datalink. You can set this value, if the Rx rings (<code>rxringsavail</code>) on the underlying physical link is greater than zero.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <code>sw</code> indicates that the datalink is a software-based client.</td>
</tr>
<tr>
<td><code>txringsavail</code></td>
<td>Read only</td>
<td>Indicates the number of Tx rings that you can allocate to hardware-based clients on the physical datalink.</td>
</tr>
<tr>
<td><code>txhwclntavail</code></td>
<td>Read only</td>
<td>Indicates the number of hardware-based Tx clients that you can create on the physical datalink.</td>
</tr>
<tr>
<td><code>txrings</code></td>
<td>Read and write</td>
<td>Indicates the number of Tx rings exclusively used by the datalink. You can set this property to one of the three possible values:</td>
</tr>
</tbody>
</table>
### Managing NIC Rings

#### Chapter 7 • Managing Network Resources

<table>
<thead>
<tr>
<th>Ring Property</th>
<th>Permission</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hw</td>
<td>hw</td>
<td>Indicates that you are configuring a hardware-based client. You can set this value, if the hardware-based Tx clients (txhwclntavail) on the underlying physical link is greater than zero.</td>
</tr>
<tr>
<td>number</td>
<td>number</td>
<td>Indicates the number of rings that you can assign to a datalink. You can set this value, if the Tx rings (txringsavail) on the underlying physical link is greater than zero.</td>
</tr>
<tr>
<td>sw</td>
<td>sw</td>
<td>Indicates that the datalink is a software-based client.</td>
</tr>
</tbody>
</table>

- `# dladm show-phys -H link`
  Displays how the rings of a physical datalink are currently being used by existing clients.

- `# dladm create-vnic -p ring-properties -l link VNIC`
  - `-p ring-properties` Refers to the ring-properties whose values can be set.
  - Creates a client with a specific number of Rx or Tx rings.

- `# dladm set-linkprop -p ring-properties VNIC`
  Allocates rings to a specific client, provided that the rings are available and ring allocation is supported.

## Displaying Ring Use and Ring Assignments on a Datalink

To display the possible values, configured values, and effective values of Rx rings and Tx rings of a datalink, you use the following command syntax:

`# dladm show-linkprop -p rxrings,txrings link`

To display how the rings of a physical datalink are currently being used by clients, you use the following command syntax:

`# dladm show-phys -H link`

### EXAMPLE 7-1  Ring Use and Ring Assignments on a Datalink

The following example shows the ring assignments on the datalink `net4`.

`# dladm show-linkprop net4`

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
</table>

Chapter 7 • Managing Network Resources 161
How to Configure Clients and Allocate Rings

The output shows that the datalink net4 has exclusive use of one Rx ring and one Tx ring. The datalink net4 has seven Rx rings and ten Tx rings that are available for allocation to the clients. You can create three hardware-based Rx clients and three hardware-based Tx clients over the datalink net4.

The following example shows the ring use for the datalink net0.

```
# dladm show-phys -H net0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>RINGTYPE</th>
<th>RINGS</th>
<th>CLIENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>RX</td>
<td>0-1</td>
<td>&lt;default,mcast&gt;</td>
</tr>
<tr>
<td>net0</td>
<td>TX</td>
<td>0-7</td>
<td>&lt;default&gt;net0</td>
</tr>
<tr>
<td>net0</td>
<td>RX</td>
<td>2-3</td>
<td>net0</td>
</tr>
<tr>
<td>net0</td>
<td>RX</td>
<td>4-5</td>
<td>--</td>
</tr>
<tr>
<td>net0</td>
<td>RX</td>
<td>6-7</td>
<td>--</td>
</tr>
</tbody>
</table>

Based on the output, the two Rx rings allocated to net0 are rings 2 and 3. For Tx rings, net0 uses rings 0 through 7.

Configuring Clients and Allocating Rings

This section describes how to configure clients on a datalink based on the type of support for ring allocation.

How to Configure Clients and Allocate Rings

Make sure that you can interpret the output of the `dladm` commands that display datalink ring properties, as explained in “Commands for Configuring Rings” on page 160. This information helps you to configure clients and allocate rings.

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

2. Display the underlying physical datalink's properties.
   ```
   # dladm show-linkprop -p rxringsavail,txringsavail,rxhwclntavail,txhwclntavail link
   ```
Determine the following information from the output of the command:

- Whether the NIC supports hardware-based clients
- The availability of rings to allocate to hardware-based clients
- The availability of hardware-based clients that you can configure on the link

3. **Depending on the information from the previous step, perform one of the following:**
   - **Create the hardware-based client with the following syntax:**
     ```bash
     # dladm create-vnic -p rxrings=value[,txrings=value] -l link VNIC
     ```
     where `value` can be one of the following:
     - `hw` - Indicates that you are configuring a hardware-based client.
     - `number` - Indicates that you are configuring a hardware-based client only. The number refers to the quantity of rings that you can allocate to the client for its exclusive use.
   
   - **Create the software-based client with the following syntax:**
     ```bash
     # dladm create-vnic -p rxrings=sw[,txrings=sw] -l link VNIC
     ```
     Alternatively, if the client was previously created, you can use the `dladm set-linkprop` command to set the ring properties.

4. **(Optional) Verify the ring information of the client that you created.**
   ```bash
   # dladm show-linkprop -p rxrings,txrings VNIC
   ```

5. **(Optional) Verify the link’s rings that are distributed among different clients.**
   ```bash
   # dladm show-phys -H link
   ```

**Example 7-2** Configuring Clients and Allocating Rings on the nxge Device

This example is based on the nxge device and shows how to configure clients and allocate rings on the datalink net5. This example shows how to create the following clients:

- The VNIC vnic2, which is a hardware-based client with exclusive use of Rx and Tx rings.
- The VNIC vnic3, which is a hardware-based client with a fixed number of rings that are set according to the NIC driver's initial configuration.
- The VNIC vnic4, which is a software-based client.

1. **Check whether the physical datalink net5 supports ring allocation for clients.**
   ```bash
   # dladm show-linkprop -p rxringsavail,txringsavail net5
   ```
How to Configure Clients and Allocate Rings

The output shows that the physical datalink net5 has 7 Rx rings and 11 Tx rings that you can assign to the clients over the physical datalink net5.

2. Check the availability of hardware-based clients that you can create over the physical datalink net5.

```bash
# dladm show-linkprop -p rxhwclntavail,txhwclntavail net5
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net5</td>
<td>rxhwclntavail</td>
<td>r-</td>
<td>3</td>
<td>3</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net5</td>
<td>txhwclntavail</td>
<td>r-</td>
<td>4</td>
<td>4</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The output shows that you can create 3 hardware-based Rx clients and 4 hardware-based Tx clients over the datalink net5.

3. Check the existing ring usage over the physical datalink net5.

```bash
# dladm show-phys -H net5
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>RINGTYPE</th>
<th>RINGS</th>
<th>CLIENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>nxge1</td>
<td>RX</td>
<td>0-7</td>
<td>&lt;default,mcast&gt;</td>
</tr>
<tr>
<td>nxge1</td>
<td>TX</td>
<td>0-11</td>
<td>&lt;default&gt;</td>
</tr>
<tr>
<td>nxge1</td>
<td>RX</td>
<td>1-2</td>
<td>vnic2</td>
</tr>
<tr>
<td>nxge1</td>
<td>TX</td>
<td>1-2</td>
<td>vnic2</td>
</tr>
</tbody>
</table>

The output shows that the nxge1 device has eight Rx rings (0-7) and twelve Tx rings (0-11). Because no datalinks are on the nxge1 device, the Rx rings and Tx rings are not assigned to any datalinks. The value <default> in the CLIENTS column means that the Tx rings will be used by the software-based clients. The value <default,mcast> under the CLIENTS column means that the Rx rings will be used by the software-based clients and non-unicast packets.

4. Create the VNIC vnic2 over the datalink net5 with two Rx rings and two Tx rings.

```bash
# dladm create-vnic -l net5 -p rxrings=2,txrings=2 vnic2
```

5. Verify the rings that are assigned to the VNIC vnic2.

```bash
# dladm show-linkprop -p rxrings,txrings vnic2
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic2</td>
<td>rxrings</td>
<td>rw</td>
<td>2</td>
<td>2</td>
<td>--</td>
<td>sw,hw,&lt;1-7&gt;</td>
</tr>
<tr>
<td>vnic2</td>
<td>txrings</td>
<td>rw</td>
<td>2</td>
<td>2</td>
<td>--</td>
<td>sw,hw,&lt;1-11&gt;</td>
</tr>
</tbody>
</table>

6. Verify the ring usage on the physical datalink net5.

```bash
# dladm show-phys -H net5
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>RINGTYPE</th>
<th>RINGS</th>
<th>CLIENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>nxge1</td>
<td>RX</td>
<td>0-3-7</td>
<td>&lt;default,mcast&gt;</td>
</tr>
<tr>
<td>nxge1</td>
<td>TX</td>
<td>0-3-11</td>
<td>&lt;default&gt;</td>
</tr>
<tr>
<td>nxge1</td>
<td>RX</td>
<td>1-2</td>
<td>vnic2</td>
</tr>
<tr>
<td>nxge1</td>
<td>TX</td>
<td>1-2</td>
<td>vnic2</td>
</tr>
</tbody>
</table>
The output shows that the Rx rings allocated to vnic2 are 1 and 2. For Tx rings, vnic2 uses the rings 1 and 2.

7. Check whether you can create additional hardware-based clients over the physical datalink net5.

```shell
# dladm show-linkprop -p rxhwclntavail,txhwclntavail net5
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net5</td>
<td>rxhwclntavail</td>
<td>r-</td>
<td>2</td>
<td>2</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net5</td>
<td>txhwclntavail</td>
<td>r-</td>
<td>3</td>
<td>3</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The output shows that you can create two hardware-based Rx clients and three hardware-based Tx clients over the physical datalink net5.

8. Create the VNIC vnic3, which is a hardware-based client.

```shell
# dladm create-vnic -l net5 -p rxrings=hw,txrings=hw vnic3
```

9. Verify the rings that are assigned to the VNIC vnic3.

```shell
# dladm show-linkprop -p rxrings,txrings vnic3
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic3</td>
<td>rxrings</td>
<td>rw</td>
<td>--</td>
<td>1</td>
<td>--</td>
<td>sw,hw,&lt;1-7&gt;</td>
</tr>
<tr>
<td>vnic3</td>
<td>txrings</td>
<td>rw</td>
<td>hw</td>
<td>hw</td>
<td>--</td>
<td>sw,hw,&lt;1-11&gt;</td>
</tr>
</tbody>
</table>

**Note** - The number of rings that are assigned to a client depends on the network device. One ring is assigned to a client on the device that enables you to explicitly specify the number of rings, for example, the nxge device. For other devices, the number of rings assigned to a client depends on how the device is configured. See Example 7-3.

10. Check whether you can create additional hardware-based clients over the physical datalink net5.

```shell
# dladm show-linkprop -p rxhwclntavail,txhwclntavail net5
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net5</td>
<td>rxhwclntavail</td>
<td>r-</td>
<td>2</td>
<td>2</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net5</td>
<td>txhwclntavail</td>
<td>r-</td>
<td>2</td>
<td>2</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The output shows that you can create 2 hardware-based Rx clients and 2 hardware-based Tx clients over the physical datalink net5.

11. Create the VNIC vnic4, which is a software-based client.

```shell
# dladm create-vnic -l net5 -p rxrings=sw,txrings=sw vnic4
```

12. Verify the ring usage on vnic4.

```shell
# dladm show-linkprop -p rxrings,txrings vnic4
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic4</td>
<td>rxrings</td>
<td>rw</td>
<td>sw</td>
<td>--</td>
<td>--</td>
<td>sw,hw,&lt;1-7&gt;</td>
</tr>
<tr>
<td>vnic4</td>
<td>txrings</td>
<td>rw</td>
<td>sw</td>
<td>--</td>
<td>--</td>
<td>sw,hw,&lt;1-11&gt;</td>
</tr>
</tbody>
</table>
13. Verify the ring usage on the physical datalink net5.

```
# dladm show-phys -H net5
```

```
<table>
<thead>
<tr>
<th>LINK</th>
<th>RINGTYPE</th>
<th>RINGS</th>
<th>CLIENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>nxge1</td>
<td>RX</td>
<td>0,4-7</td>
<td>&lt;default,mcast&gt;,vnic4</td>
</tr>
<tr>
<td>nxge1</td>
<td>TX</td>
<td>0,4-11</td>
<td>&lt;default&gt;,vnic4</td>
</tr>
<tr>
<td>nxge1</td>
<td>RX</td>
<td>1-2</td>
<td>vnic2</td>
</tr>
<tr>
<td>nxge1</td>
<td>RX</td>
<td>3</td>
<td>vnic3</td>
</tr>
<tr>
<td>nxge1</td>
<td>TX</td>
<td>1-2</td>
<td>vnic2</td>
</tr>
<tr>
<td>nxge1</td>
<td>TX</td>
<td>3</td>
<td>vnic3</td>
</tr>
</tbody>
</table>
```

The output shows that vnic4 is software-based client that shares the default set of rings on the physical datalink net5. The VNIC vnic2 is a hardware-based client that has exclusive use of two rings (2-3) and vnic3 is a hardware-based client that has exclusive use of one ring (3).

**Example 7-3 Configuring Clients and Allocating Rings on the ixgbe Device**

This example is based on the ixgbe device and shows how to configure clients and allocate rings on the physical datalink net4.

1. Check the existing ring usage over the physical datalink net4.

```
# dladm show-phys -H net4
```

```
<table>
<thead>
<tr>
<th>LINK</th>
<th>RINGTYPE</th>
<th>RINGS</th>
<th>CLIENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>net4</td>
<td>RX</td>
<td>0-3</td>
<td>&lt;default,mcast&gt;</td>
</tr>
<tr>
<td>net4</td>
<td>RX</td>
<td>4-7</td>
<td>--</td>
</tr>
<tr>
<td>net4</td>
<td>RX</td>
<td>8-11</td>
<td>--</td>
</tr>
<tr>
<td>net4</td>
<td>RX</td>
<td>12-15</td>
<td>--</td>
</tr>
<tr>
<td>net4</td>
<td>TX</td>
<td>0-7</td>
<td>&lt;default&gt;</td>
</tr>
</tbody>
</table>
```

2. Check whether you can create hardware-based clients over the physical datalink net4.

```
# dladm show-linkprop -p rxhwclntavail,txhwclntavail,rxringsavail,txringsavail net4
```

```
<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net4</td>
<td>rxhwclntavail</td>
<td>r-</td>
<td>3</td>
<td>3</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net4</td>
<td>txhwclntavail</td>
<td>r-</td>
<td>0</td>
<td>0</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net4</td>
<td>rxringsavail</td>
<td>r-</td>
<td>0</td>
<td>0</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net4</td>
<td>txringsavail</td>
<td>r-</td>
<td>0</td>
<td>0</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>
```

The output shows that you can create 3 hardware-based Rx clients over the physical datalink net4.

3. Create the VNIC vnic3, which is a hardware-based Rx client.

```
# dladm create-vnic -l net4 -p rxrings=hw vnic3
```
You cannot configure the \texttt{txrings} property for \texttt{vnic3} because the available number of hardware-based Tx clients (\texttt{txhwclntavail}) is zero.

4. Verify the rings that are assigned to the VNIC \texttt{vnic3}.

```bash
# dladm show-linkprop -p rxrings,txrings vnic3
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic3</td>
<td>rxrings</td>
<td>rw</td>
<td>hw</td>
<td>hw</td>
<td>--</td>
<td>sw,hw</td>
</tr>
<tr>
<td>vnic3</td>
<td>txrings</td>
<td>rw</td>
<td>--</td>
<td>8</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

5. Check whether you can create additional hardware-based clients over the physical datalink \texttt{net4}.

```bash
# dladm show-linkprop -p rxhwclntavail,txhwclntavail,rxringsavail,txringsavail net5
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net4</td>
<td>rxhwclntavail</td>
<td>r-</td>
<td>2</td>
<td>2</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net4</td>
<td>txhwclntavail</td>
<td>r-</td>
<td>0</td>
<td>0</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net4</td>
<td>rxringsavail</td>
<td>r-</td>
<td>0</td>
<td>0</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net4</td>
<td>txringsavail</td>
<td>r-</td>
<td>0</td>
<td>0</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The output shows that you can create 2 hardware-based Rx clients over the physical datalink \texttt{net4}.

6. Verify the ring usage on the physical datalink \texttt{net4}.

```bash
# dladm show-phys -H net4
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>RINGTYPE</th>
<th>RINGS</th>
<th>CLIENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>net4</td>
<td>RX</td>
<td>0-3</td>
<td>&lt;default,mcast&gt;</td>
</tr>
<tr>
<td>net4</td>
<td>RX</td>
<td>4-7</td>
<td>vnic3</td>
</tr>
<tr>
<td>net4</td>
<td>RX</td>
<td>8-11</td>
<td></td>
</tr>
<tr>
<td>net4</td>
<td>RX</td>
<td>12-15</td>
<td></td>
</tr>
<tr>
<td>net4</td>
<td>TX</td>
<td>0-7</td>
<td>&lt;default&gt;,vnic3</td>
</tr>
</tbody>
</table>

The output shows that \texttt{vnic3} is a hardware-based Rx client with exclusive use of four rings. For Tx rings, \texttt{vnic3} uses the default set of rings and also shares the rings with other datalinks when they are created on the physical datalink \texttt{net4}.

**Managing Pools and CPUs**

In Oracle Solaris, zone administration includes assigning a pool of CPU resources for non-networking processes by using the \texttt{zonecfg} or \texttt{poolcfg} command. To dedicate that same pool of resources to also manage network processes, use the \texttt{dladm set-linkprop} command to configure a link's pool property. The pool link property enables you to assign a pool of CPUs for the networking processes. With this property, you can better integrate network resource management with CPU allocation and administration in zones.
By setting the `pool` property for a link and assigning the link as the zone's network interface, that link becomes bound to a zone's pool. If the zone is set to become an exclusive zone, then CPU resources in the pool can no longer be used by other links that are not assigned to the zone.

**Note** - A separate property, `cpus`, can be set to assign specific CPUs to a datalink. The `cpus` and `pool` properties are mutually exclusive. You cannot set both properties for a given datalink. To assign CPU resources to a datalink by using the `cpus` property, see “How to Allocate CPUs to a Datalink” on page 171.

For more information about pools within a zone, see Chapter 13, “Creating and Administering Resource Pools Tasks,” in “Administering Resource Management in Oracle Solaris 11.2”. For more information about creating pools and assigning CPU sets to the pools, see the `poolcfg(1M)` man page.

**Working With Pools and CPUs**

The following figure shows how pools work when the `pool` property is assigned to a datalink.

**FIGURE 7-1 pool Property of a VNIC Assigned to a Zone**

In the figure, the system has eight CPUs. When no pools are configured on the system, all the CPUs belong to the *default pool* and are used by the global zone. However, in this example, the pool99 pool has been created and consists of CPU 3 and CPU 4. This pool is associated with `zone1`, which is an exclusive zone. If pool99 is set as a property of `vnic1`, then pool99 becomes dedicated to also manage `vnic1`'s networking processes. After `vnic1` is assigned to
be zone1's network interface, the CPUs in pool99 are reserved to manage both networking and non-networking processes of zone1.

The pool property is dynamic in nature. Zone pools can be configured with a range of CPUs, and the kernel determines which CPUs are assigned to the pool's CPU set. Changes to the pool are automatically implemented for the datalink, which simplifies pool administration for that link. In contrast, assigning specific CPUs to the link by using the cpu property requires you to specify the CPU to be assigned. You have to set the cpu property every time you want to change the CPU components of the pool.

For example, suppose that the system CPU 4 in Figure 7-1 is taken offline. Because the pool property is dynamic, the software automatically associates an additional CPU with the pool. Hence, the pool's original configuration of two CPUs is preserved. For vnic1, the change is transparent. The updated configuration is shown in the following figure.

**FIGURE 7-2** Automatic Reconfiguration of the pool Property

When you use the dladm show-linkprop command to display information for a datalink, the value in the EFFECTIVE column for the pool and cpus datalink properties indicates the current system-selected value of those properties.

The following read-only values are displayed for the pool and cpus properties:

- For the pool datalink property, the value in the EFFECTIVE column indicates the pool that is used for network processes.
- For the cpus datalink property, the value in the EFFECTIVE column indicates the CPUs that are used for network processes. For an example that shows how to display the cpus property for a datalink, see Example 7-5.

To manage the CPU resources of a zone, you do not need to set a datalink's pool property. You can use commands such as zonecfg and poolcfg to configure a zone to use a pool of resources.
When the `cpus` and `pool` link properties are not set for a datalink, the value in the `EFFECTIVE` column of the `pool` and `cpus` properties of the datalinks are set automatically according to the zone configurations when the zone is booted. The default pool is displayed in the `EFFECTIVE` column of the `pool` property and the system selects the value in the `EFFECTIVE` column of the `cpus` property. Therefore, if you use the `dladm show-linkprop` command, the value of the `pool` and `cpus` properties is empty but values are displayed in the `EFFECTIVE` column of the `pool` and `cpus` properties.

You can also directly set the `pool` and `cpu` properties of a datalink to assign a zone's CPU pool for networking processes. After you configure these properties, their values are reflected in the `EFFECTIVE` column of the `pool` and `cpus` properties. However, this alternative method is used less often to manage a zone's network resources.

## Configuring a CPU Pool for a Datalink

This section describes how to set the `pool` property for a datalink either when the link is created or later when the link requires further configuration.

### How to Configure a CPU Pool for a Datalink

**Before You Begin**

You must have completed the following tasks:

- Created a processor set with its assigned number of CPUs
- Created a pool with which the processor set will be associated
- Associated the pool with the processor set

**Note** - For the instructions to complete these prerequisites, see “How to Modify a Configuration” in “Administering Resource Management in Oracle Solaris 11.2”.

1. **Set the link's `pool` property to the pool of CPUs that you created for the zone.**

   - **If the VNIC has not yet been created, use the following syntax:**
     ```
     # dladm create-vnic -l link -p pool=pool VNIC
     ```
   - **If the VNIC exists, use the following syntax:**
     ```
     # dladm set-linkprop -p pool=pool VNIC
     ```
How to Allocate CPUs to a Datalink

Chapter 7 • Managing Network Resources

2. **Set the zone to use the VNIC.**

   ```
   global# zonecfg -z zone
   zonecfg:zone> add net
   zonecfg:zone:net> set physical=VNIC
   zonecfg:zone:net> end
   ```

3. **Verify and commit the changes you have implemented and then exit the zone.**

   ```
   zonecfg:zone> verify
   zonecfg:zone> commit
   zonecfg:zone> exit
   ```

**Example 7-4 Assigning a Link’s CPU Pool to a Zone**

This example shows how a pool is assigned to a zone’s datalink. The scenario is based on the configuration in Figure 7-1. The example assumes that a pool of CPUs named pool99 has already been configured for the zone. The pool is then assigned to a VNIC. Finally, the non-global zone zone1 is set to use the VNIC as the network interface.

```
# dladm create-vnic -l net1 -p pool=pool99 vnic1

# zonecfg -z zone1
zonecfg:zone1> add net
zonecfg:zone1:net> set physical=vnic1
zonecfg:zone1:net> end
zonecfg:zone1> verify
zonecfg:zone1> commit
zonecfg:zone1> exit
```

Allocating CPUs to a Datalink

This section describes how to assign CPU resources to a datalink by configuring the `cpu` property. Unlike rings, you cannot allocate CPUs exclusively for a datalink. You can allocate the same set of CPUs to multiple datalinks.

**How to Allocate CPUs to a Datalink**

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 the CPU assignments for the interface.**

   ```
   # dladm show-linkprop -p cpus link
   ```
3. **Assign CPUs to the link.**

A list of CPUs that process packets for the datalink. Interrupts for the datalink might also be targeted to one of the CPUs in the list.

```bash
# dladm set-linkprop -p cpus=cpu1,cpu2,... link
```

*cpu1,cpu2,...* Refer to the CPU number that you want to assign to the link. You can dedicate multiple CPUs to the link.

4. **(Optional) Display the CPUs that are associated with the link.**

```bash
# dladm show-linkprop -p cpus link
```

**Example 7-5 Allocating CPUs to a Datalink**

This example shows how to dedicate specific CPUs to the datalink `net0`.

```bash
# dladm show-linkprop -p cpus net0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>cpus</td>
<td>rw</td>
<td>--</td>
<td>0-2</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The output shows that the system has implicitly assigned three CPUs (0-2) to the datalink `net0`. However, the CPUs are not exclusively allocated to the datalink `net0`.

```bash
# dladm set-linkprop -p cpus=0,1 net0
# dladm show-linkprop -p cpus net0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>PROPERTY</th>
<th>PERM</th>
<th>VALUE</th>
<th>EFFECTIVE</th>
<th>DEFAULT</th>
<th>POSSIBLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>cpus</td>
<td>rw</td>
<td>0-1</td>
<td>0-1</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The output shows that you have explicitly assigned two CPUs (0-1) to the datalink `net0`. The allocated CPUs will process packets for the datalink `net0`.

---

**Managing Network Resources by Using Flows**

A flow is a customized way of categorizing network packets based on a single attribute or a combination of attributes. Flows enable you to further allocate network resources. For an overview of flows, see “Network Resource Management by Using Flows” on page 20.

Using flows for managing network resources involves the following steps:

1. Creating the flow.

   A flow is created based on a single attribute or a combination of attributes that are derived from the information in a packet's header.

   You can use one of the following attributes to organize packet traffic into a flow:
   - Local IP address.
   - Remote IP address.
Transport protocol name (UDP, TCP, or SCTP).
Differentiated Services Field (DS Field) attribute, which is used for QoS in IPv6 packets only. For more information, see “Managing IP Quality of Service in Oracle Solaris 11.2”.

You can use one of the following combinations of attributes to organize packet traffic into a flow:
- Transport protocol name (UDP, TCP or SCTP) with the local application port number (for example, port 21 for FTP).
- Transport protocol name (UDP, TCP or SCTP) with the remote application port number.
- Transport protocol name (UDP, TCP or SCTP) with the local IP address and the local application port number. This combination of attributes can further include the remote IP address with the remote application port number: Transport protocol name (UDP, TCP or SCTP) + local IP address + local application port number [+ remote IP address [+ remote application port number]].

A flow can be based on only one of these attribute combinations. For example, you can create a flow according to the transport protocol and port that is being used, such as TCP port 21 for FTP, or according to IP addresses, such as packets from a specific source IP address. As the most general case, you can create a flow by specifying the transport protocol, local or remote IP address, and local or remote port. Also, all the flows belonging to the same link must have the same combination of attributes.

2. Customizing the flow’s use of resources by setting properties that pertain to network resources. Currently, bandwidth and priority properties can be associated with flows.

For more information, see “Configuring Flows” on page 174.

Commands for Resource Allocation in Flows

The commands used for allocating network resources in flows are as follows:
- To simultaneously create a flow and add resources to it, use the following command syntax:
  ```
  # flowadm add-flow -l link -a attribute=value[,attribute=value] -p prop=value[,...] flow
  ```
  The set of defined attributes that characterizes the flows constitutes the system's flow control policy. For the list of different attributes that you can use to organize packet traffic into a flow, see “Managing Network Resources by Using Flows” on page 172.
- To set the property of an existing flow, use the following command syntax:
  ```
  # flowadm set-flowprop -p prop=value[,...] flow
  ```
  where prop refers to the flow properties that can be assigned to a flow. The flow properties are the same as the properties that are assigned directly to a link. However, only the
bandwidth and priority properties can be associated with flows. To configure these properties, see “How to Configure Flows” on page 174.

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

Configuring Flows

This section describes how to create a flow and set flow properties.

▼ How to Configure Flows

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

2. (Optional) List the available links to determine the link on which you will configure flows.
   # dladm show-link

3. Verify that IP interfaces over the selected link are properly configured with IP addresses.
   # ipadm show-addr

4. Create flows according to the attribute you have determined for each flow.
   # flowadm add-flow -l link -a attribute=value[,attribute=value] flow

   link Refers to the link on which you are configuring the flow.

   attribute Refers to a single or a combination of attributes that you can use to organize network packets into a flow. For information about the attributes, see “Managing Network Resources by Using Flows” on page 172.

   flow Refers to the name that you assign to the flow.

   For more information about flows and flow attributes, see the flowadm(1M) man page.

5. (Optional) Display the possible range of values for the link's bandwidth.
   # dladm show-linkprop -p maxbw link

   For more information about flows and flow attributes, see the flowadm(1M) man page.
link 
Refers to the datalink on which the flow is configured.

The range of values is listed under the POSSIBLE field of the command's output.

6. Implement resource controls on the flows by setting the appropriate flow properties.

   \# flowadm set-flowprop -p prop=value[,...] flow

You can specify the following flow properties that control resources:

maxbw 
The maximum amount of the link's bandwidth that packets identified with the flow can use. The value you set must be within the allowed range of values for the link's bandwidth.

priority 
The priority with which packets belonging to the specified flow will be processed. The allowed values for the priority property are high, medium, and low. If the priority of a flow is set to high, all the packets belonging to that flow will be processed ahead of other packets on the same link. This property is used to create a flow for applications that are latency sensitive. The default value of this property is medium.

   Note - Currently, setting the priority property to low from medium has no effect.

7. (Optional) Display the flows that you have created over the datalink.

   \# flowadm

   Note - The flowadm command, if used without any subcommand, provides the same information as the flowadm show-flow command.

8. (Optional) Display the property values for a specified flow.

   \# flowadm show-flowprop flow

This command displays maxbw and priority flow properties, plus the read-only hwflow property.

hwflow 
A read-only property that helps you to understand the packet classification in flows. The possible values of this property are on and off. The value of on means that the flow has been offloaded to the NIC and packet classification for the flow is conducted at the hardware level.
This property cannot be used with -p option in flowadm add-flow, flowadm set-flowprop, or flowadm reset-flowprop commands.

**Note** - Currently, only the flows that are defined by specifying all the transport protocols, local or remote IP address, and local or remote port can be assigned the on value for hwflow. Also, not all NICs support the hwflow property.

### Use Case: Managing Network Resources by Setting Datalink and Flow Properties

The following use case is based on a scenario in which you increase a system's efficiency by setting both datalink and flow properties. This use case is based on the configuration shown in the following figure.
The figure shows the following two physical hosts that are connected to each other:

- **Host1** has the following configuration:
  - One non-global zone that functions as a server and router. Two interfaces are assigned to the zone: the net0 interface connects to the Internet and the net1 interface connects to the internal network including the Host2.
Flows are configured over net1 to isolate the traffic and implement control over how packets belonging to the flows use resources. For information about configuring flows, see “Managing Network Resources by Using Flows” on page 172.

Host2 has the following configuration:

- Three non-global zones and their respective VNICS. The VNICS are configured over net0, whose NIC card supports ring allocation. For more information about ring allocation, see “Managing NIC Rings” on page 158.
- Each zone's network processing load is different. In this example, zone1 functions as the HTTP client. The remaining zones, zone2 and zone3, function as the SSH client that tries to access Host1 through secure shell (SSH) protocol. The network traffic for zone1 is higher than zone2 and zone3 and is not time sensitive. However, the network traffic for zone2 and zone3 is low and time sensitive. Therefore, to process the network traffic faster for zone2 and zone3, you need to limit the bandwidth allocated to the network traffic for zone1. If the bandwidth allocated for zone1 is not limited, it will use all the available bandwidth. This leads to the denial of bandwidth to the remaining zones: zone2 and zone3.
- A separate VNIC is configured as a software-based client. For an overview of MAC clients, see “Allocating Rings in MAC Clients” on page 159.

The tasks in this use case involve the following actions:

- Creating a flow and configuring flow control – Flows are created over net1 to create a separate resource control over packets belonging to the flows that are received by net1 of Host1.
- Configuring network resource properties for the VNICS on Host2 – Based on the processing load, each zone's VNIC is configured with a set of dedicated rings. A separate VNIC is also configured without dedicated rings as an example of a software-based client.

**Note** - The use case does not include any procedures for zone configuration. To configure zones, see Chapter 1, “How to Plan and Configure Non-Global Zones,” in “Creating and Using Oracle Solaris Zones.”

1. View information about links and IP interfaces on Host1.

```bash
# ipadm

<table>
<thead>
<tr>
<th>NAME</th>
<th>CLASS/TYPE</th>
<th>STATE</th>
<th>UNDER</th>
<th>ADDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>lo0</td>
<td>loopback</td>
<td>ok</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>lo0/v4</td>
<td>static</td>
<td>ok</td>
<td>--</td>
<td>127.0.0.1/8</td>
</tr>
<tr>
<td>lo0/v6</td>
<td>static</td>
<td>ok</td>
<td>--</td>
<td>:::1/128</td>
</tr>
<tr>
<td>net1</td>
<td>ip</td>
<td>ok</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>net1/v4</td>
<td>static</td>
<td>ok</td>
<td>--</td>
<td>192.168.200.183/24</td>
</tr>
</tbody>
</table>
```
Use Case: Managing Network Resources by Setting Datalink and Flow Properties

Chapter 7 • Managing Network Resources

2. Create the following flows over net1 on Host1:
   - **httpflow** – Contains all the HTTP traffic between zone1 and net1.
     ```
     # flowadm add-flow -l net1 -a transport=tcp,local_ip=192.168.200.103,\       
     local_port=80,remote_ip=192.168.200.110 httpflow
     ```
   - **sshflow** – Contains all the SSH traffic coming in to and going out of net1.
     ```
     # flowadm add-flow -l net1 -a transport=tcp,local_ip=192.168.200.103,\       
     local_port=22 sshflow
     ```

3. Implement resource control on the flows.
   - For **httpflow**, set the maximum bandwidth to 500M.
     ```
     # flowadm set-flowprop -p maxbw=500M httpflow
     ```
   - For **sshflow**, set the priority to high.
     ```
     # flowadm set-flowprop -p priority=high sshflow
     ```

4. Verify the information about the created flows.
   ```
   # flowadm
   FLOW        LINK     PROTO LADDR             LPORT  RADDR             RPORT DSFLD
   httpflow    net1     tcp   192.168.200.103   80     192.168.200.110   --    --
   sshflow     net1     tcp   192.168.200.103   22     --                --    --
   # flowadm show-flowprop
   FLOW         PROPERTY        PERM    VALUE        DEFAULT        POSSIBLE
   httpflow     maxbw           rw      500          --             --
   httpflow     priority        rw      medium       medium         low,medium,high
   httpflow     hwflow          r-      off          --             on,off
   sshflow      maxbw           rw      --           --             --
   sshflow      priority        rw      high         medium         low,medium,high
   sshflow      hwflow          r-      off          --             on,off
   ```

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

5. On Host2, configure VNICs over net0 for each zone.
   ```
   # dladm create-vnic -l net0 vnic0
   # dladm create-vnic -l net0 vnic1
   # dladm create-vnic -l net0 vnic2
   ```

6. Implement resource controls on each VNIC.
   ```
   # dladm set-linkprop -p rxrings=4,txrings=4 vnic0
   # dladm set-linkprop -p rxrings=2,txrings=2 vnic1
   # dladm set-linkprop -p rxrings=1,txrings=1 vnic2
   ```
7. Assign the VNICs to their respective zones.

# zonecfg -z zone1
# zonecfg:zone1> add net
# zonecfg:zone1:net> set physical=vnic0
# zonecfg:zone1:net> end
# zonecfg:zone1> commit
# zonecfg:zone1> exit
# zoneadm -z zone1 reboot

# zonecfg -z zone2
# zonecfg:zone2> add net
# zonecfg:zone2:net> set physical=vnic1
# zonecfg:zone2:net> end
# zonecfg:zone2> commit
# zonecfg:zone2> exit
# zoneadm -z zone2 reboot

# zonecfg -z zone3
# zonecfg:zone3> add net
# zonecfg:zone3:net> set physical=vnic2
# zonecfg:zone3:net> end
# zonecfg:zone3> commit
# zonecfg:zone3> exit
# zoneadm -z zone3 reboot

8. Create a software-based client that shares rings with the primary interface net0.

# dladm create-vnic -p rxrings=sw,txrings=sw -l net0 vnic3

9. Assume pool1, a set of CPUs in Host2, is assigned to zone1. Assign the same pool1 of CPUs to also manage network processes for zone1.

# dladm set-linkprop -p pool=pool1 vnic0
This chapter describes tasks for monitoring network statistics about the use of network resources on datalinks and flows. You configure network accounting on a system to record network traffic statistics in a log file. This statistical information can help you analyze resource allocation for provisioning, consolidation, and billing purposes. This chapter introduces the two commands that you can use to display network traffic statistics: `dlstat` and `flowstat`.

This chapter contains the following topics:

- “Commands for Monitoring Network Traffic Statistics” on page 183
- “Displaying Network Traffic Statistics of Links” on page 183
- “Configuring Network Accounting for Network Traffic” on page 192

**Overview of Monitoring Network Traffic Statistics of Datalinks and Flows**

Packets traverse a path when they flow into or out of a system. On a granular level, packets are received and transmitted through receive (Rx) rings and transmit (Tx) rings of a NIC. Inbound packets from these rings are passed up the network stack for further processing while outbound packets are sent to the network.

You can combine and allocate system resources to manage the network traffic. You can monitor the receive-side and transmit-side network traffic statistics for both datalinks and flows. This chapter focuses primarily on receive-side network traffic statistics on datalinks and flows.

You can configure receive rings, transmit rings, and other resources on datalinks by setting datalink properties. Depending on the network traffic on a datalink, you can assign dedicated hardware rings to a datalink to increase the system's efficiency to process packets. For example, you can allocate more rings to a datalink, where the network traffic is most heavy. For more information about how to allocate hardware rings to a datalink, see “Configuring Clients and Allocating Rings” on page 162.
A datalink might not have dedicated hardware rings because of the following reasons:

- Lack of hardware resources. For example, there might not be rings available that can be exclusively assigned to datalinks.
- Lack of hardware capabilities. For example, the NIC does not expose hardware rings.
- The datalink might not be tied to a lower hardware datalink. For example, when you create VNICs over etherstubs.

Some datalinks might be configured to share rings for the following reasons:

- The datalink might not be performing intensive processes that require dedicated rings.
- The NIC might not support ring allocation.
- The rings are no longer available to be assigned for exclusive use although the datalink supports ring allocation.

The following figure shows the allocation of hardware rings among datalinks.

**FIGURE 8-1  Ring Allocation in Datalinks**

The figure shows the following configuration:
The net0 datalink has 16 hardware rings (0-15) that can be allocated to other datalinks.

- The VNICs vnic1, vnic2, vnic3, and vnic4 are configured over the datalink net0.
- The VNICs vnic1, vnic2, and vnic3 are each assigned four dedicated hardware rings.
- The hardware rings (0-3) are shared between the datalink net0 and the VNIC vnic4. The following example shows the ring allocation for the physical datalink net0.

```
# dladm show-phys -H net0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>RINGTYPE</th>
<th>RINGS</th>
<th>CLIENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>RX</td>
<td>0-3</td>
<td>&lt;default,mcast&gt;,vnic4</td>
</tr>
<tr>
<td>net0</td>
<td>RX</td>
<td>4-7</td>
<td>vnic1</td>
</tr>
<tr>
<td>net0</td>
<td>RX</td>
<td>8-11</td>
<td>vnic2</td>
</tr>
<tr>
<td>net0</td>
<td>RX</td>
<td>12-15</td>
<td>vnic3</td>
</tr>
<tr>
<td>net0</td>
<td>TX</td>
<td>0-7</td>
<td>&lt;default&gt;,vnic4,vnic3,vnic2,vnic1</td>
</tr>
</tbody>
</table>

- You use the dlstat show-phys command to display the network traffic statistics for the physical datalink net0. See Example 8-1.
- You use the dlstat show-link command to display the network traffic statistics for the datalinks net0, vnic1, vnic2, vnic3, and vnic4. See Example 8-7.

## Commands for Monitoring Network Traffic Statistics

The dlstat and flowstat commands enable you to monitor network traffic statistics on datalinks and flows, respectively. These commands are equivalent to the dladm and flowadm commands. The following table compares the functions of the pair of administrative commands to the pair of monitoring commands.

<table>
<thead>
<tr>
<th>Administrative Commands</th>
<th>Monitoring Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>Function</td>
</tr>
<tr>
<td>dladm</td>
<td>Configures and administers datalinks</td>
</tr>
<tr>
<td>flowadm</td>
<td>Configures and administers flows</td>
</tr>
</tbody>
</table>

## Displaying Network Traffic Statistics of Links

You can use the following variants of the dlstat command to display network traffic information.
Displaying Network Traffic Statistics of Links

<table>
<thead>
<tr>
<th>Command</th>
<th>Information Provided</th>
</tr>
</thead>
<tbody>
<tr>
<td>dlstat [link]</td>
<td>Displays inbound and outbound traffic statistics per datalink</td>
</tr>
<tr>
<td>dlstat -rt [link]</td>
<td></td>
</tr>
<tr>
<td>dlstat show-link [link]</td>
<td></td>
</tr>
<tr>
<td>dlstat show-link -rt [link]</td>
<td>Displays inbound and outbound traffic statistics per ring per datalink</td>
</tr>
<tr>
<td>dlstat show-phys [link]</td>
<td>Displays inbound and outbound traffic statistics per network physical device</td>
</tr>
<tr>
<td>dlstat show-phys -rt [link]</td>
<td>Displays inbound and outbound traffic statistics per ring per network physical device</td>
</tr>
<tr>
<td>dlstat show-aggr [link]</td>
<td>Displays inbound and outbound traffic statistics per port per aggregation</td>
</tr>
<tr>
<td>dlstat show-aggr -rt [link]</td>
<td></td>
</tr>
<tr>
<td>dlstat show-bridge [bridge]</td>
<td>Displays inbound and outbound traffic statistics per bridge</td>
</tr>
<tr>
<td>dlstat show-bridge -rt [bridge]</td>
<td></td>
</tr>
</tbody>
</table>

You can use the -r option to display receive-side statistics information or the -t option to display the transmit-side statistics information with the dlstat command. For more information about other options, see the dlstat(1M) man page.

Displaying Network Traffic Statistics of Network Devices

The dlstat show-phys command provides statistics that refer to the physical network device. As shown in Figure 8-1, the dlstat show-phys command operates on the hardware rings which are on the device layer of the network stack.

You can use the following command syntax to display the network traffic statistics on network devices:

```
# dlstat show-phys [-r|t] [-Tu | -Td] [link] [interval [count]]
```

- **-r**
  Displays receive-side network traffic statistics only. You should not specify the -t option with this option.
  If you do not specify the -r option or the -t option, both the transmit-side and receive-side network statistics are displayed.

- **-t**
  Displays transmit-side network traffic statistics only. You should not specify the -r option with this option.
If you do not specify the -\texttt{r} or the -\texttt{t} option, both the transmit-side and receive-side network statistics are displayed.

-\texttt{Tu} Displays the current time in internal representation.

-\texttt{Td} Displays the current time in standard date format.

\textit{link} Name of the datalink whose network statistics you want to monitor. If you do not specify the datalink, then the information about all the configured datalinks on the system are displayed.

\textit{interval} Specifies the time in seconds at which you want to refresh the network statistics.

\textit{count} Specifies the number of times you want the displayed network traffic statistics to be refreshed. If you do not specify the count value, the statistics are refreshed indefinitely.

**EXAMPLE 8-1** Displaying Traffic Statistics for Physical Links on the System

In this example, both incoming and outgoing network traffic on each link on the system is displayed. The number of packets and their byte sizes are displayed.

```
# dlstat show-phys

<table>
<thead>
<tr>
<th>LINK</th>
<th>IPKTS</th>
<th>RBYTES</th>
<th>OPKTS</th>
<th>OBYTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>net5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>net6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>net0</td>
<td>25.57K</td>
<td>5.10M</td>
<td>1.93K</td>
<td>226.05K</td>
</tr>
<tr>
<td>net0</td>
<td>179</td>
<td>26.63K</td>
<td>161</td>
<td>22.75K</td>
</tr>
<tr>
<td>net3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>net4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>net2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>net8</td>
<td>238</td>
<td>137.16K</td>
<td>191</td>
<td>8.41K</td>
</tr>
<tr>
<td>net1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\ldots
```

The output shows the following information:

- **LINK** Physical or virtual datalink, identified by a name
- **IPKTS** Number of inbound packets on the link
- **RBYTES** Number of bytes received on the link
- **OPKTS** Number of outbound packets on the link
- **OBYTES** Number of bytes sent on this link
EXAMPLE 8-2  Displaying Receive-Side Traffic Statistics for Network Devices

In this example, network traffic statistics that are being received are displayed with an interval value of 2 seconds and the count value of 3.

```
# dlstat show-phys -r 2 3
```

```
LINK   TYPE  INDEX    IPKTS   RBYTES
net0   rx      0    8.03M   12.09G
net1   rx      0        0        0
net0   rx      0    8.79K   13.28M
net1   rx      0        0        0
net0   rx      0    8.50K   12.83M
net1   rx      0        0        0
```

Consider the datalinks, net0 and net1 as a set. The first set of datalinks, net0 and net1, show the total number of packets and bytes received. In this example, 8.03M is the total number of packets received and 12.09G is the total number of bytes received by net0. The second set of datalinks, net0 and net1, show the network traffic statistics in rates per second, also known as the normalized value. That is, 8.79K is the normalized value of the packets received by net0 in the interval of 2 seconds. Similarly, the third set of datalinks, net0 and net1, also show the normalized value for the network traffic statistics in the interval of 2 seconds.

EXAMPLE 8-3  Displaying Receive-Side Traffic Statistics for a Network Device

In this example, the incoming traffic statistics for the datalink net0 are displayed.

```
# dlstat show-phys -r net0
```

```
LINK     TYPE     ID    INDEX    IPKTS     RBYTES
net0      rx   local      --        0          0
net0      rx      hw       1        0          0
net0      rx      hw       2    1.73M      2.61G
net0      rx      hw       3        0          0
net0      rx      hw       4    8.44M     12.71G
net0      rx      hw       5    5.68M      8.56G
net0      rx      hw       6    4.99M      7.38G
net0      rx      hw       7        0          0
```

In this example, the net0 datalink has eight receive rings, which are identified under the INDEX field. An even distribution of packets per ring is an ideal configuration that indicates that the rings are properly allocated to links according to the link's load. An uneven distribution indicates a disproportionate distribution of rings per link. The resolution of the uneven distribution depends on whether the NIC supports dynamic ring allocation. If it does, you can redistribute rings per link to process packets more evenly. For more information, see “Managing NIC Rings” on page 158.

EXAMPLE 8-4  Displaying Transmit-Side Traffic Statistics for a Network Device

In this example, the usage of the transmit rings for net0 as a network device is displayed.

```
# dlstat show-phys -t net0
```
Displaying Network Traffic Statistics of Links

### EXAMPLE 8-5   Displaying Traffic Statistics for a Network Device With Time

The following example displays statistics about network traffic for net0 as a network device with internal representation of the current time.

```
# dlstat show-phys -Tu net0
1401652481
```

```
LINK    IPKTS   RBYTES    OPKTS   OBYTES
net0      184   27.14K      165   22.91K
```

The following example displays statistics about network traffic for net0 as a network device with the current time in standard date format.

```
# dlstat show-phys -Td net0
Sun Jun  1 12:54:47 PDT 2014
```

```
LINK    IPKTS   RBYTES    OPKTS   OBYTES
net0      184   27.14K      165   22.91K
```

### Displaying Network Traffic Statistics of Datalinks

You can use the `dlstat show-link` command to display the network traffic statistics for a datalink.

#### EXAMPLE 8-6   Displaying Network Traffic Statistics for a Datalink

This example shows the network traffic statistics for the datalink vnic0.

```
# dlstat show-link vnic0
```

```
LINK    IPKTS   RBYTES    OPKTS   OBYTES
vnic0   3       180       0       0
```

#### EXAMPLE 8-7   Displaying Network Traffic Statistics for a Datalink With Dedicated Hardware Rings

This example shows the receive-side network traffic statistics for the datalink vnic0 that has four dedicated Rx rings. The `hw` value under the `ID` column in the output indicates that the datalink vnic0 has dedicated hardware rings.

```
# dlstat show-link -r vnic0
```

```
LINK    IPKTS   RBYTES    OPKTS   OBYTES
vnic0   3       180       0       0
```
### Displaying Network Traffic Statistics of Links

<table>
<thead>
<tr>
<th>LINK</th>
<th>TYPE</th>
<th>ID</th>
<th>INDEX</th>
<th>IPKTS</th>
<th>RBYTES</th>
<th>INTRS</th>
<th>POLLS</th>
<th>IDROPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic0</td>
<td>rx</td>
<td>local</td>
<td>--</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>vnic0</td>
<td>rx</td>
<td>other</td>
<td>--</td>
<td>64</td>
<td>2.94K</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>vnic0</td>
<td>rx</td>
<td>hw</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>vnic0</td>
<td>rx</td>
<td>hw</td>
<td>9</td>
<td>53</td>
<td>7.97K</td>
<td>53</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>vnic0</td>
<td>rx</td>
<td>hw</td>
<td>10</td>
<td>4</td>
<td>392</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>vnic0</td>
<td>rx</td>
<td>hw</td>
<td>11</td>
<td>153.65K</td>
<td>228.68M</td>
<td>153.65K</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**EXAMPLE 8-8**  
Displaying Transmit-Side Network Traffic Statistics for a Datalink

This example shows the transmit-side network traffic statistics for the datalink vnic0.

```
# dlstat show-link -t vnic0
```

<table>
<thead>
<tr>
<th>LINK</th>
<th>TYPE</th>
<th>ID</th>
<th>INDEX</th>
<th>OPKTS</th>
<th>OBYTES</th>
<th>ODROPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>vnic0</td>
<td>tx</td>
<td>local</td>
<td>--</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>vnic0</td>
<td>tx</td>
<td>other</td>
<td>--</td>
<td>19</td>
<td>798</td>
<td>0</td>
</tr>
<tr>
<td>vnic0</td>
<td>tx</td>
<td>sw</td>
<td>--</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**EXAMPLE 8-9**  
Displaying Network Traffic Statistics for a Datalink Without Dedicated Hardware Rings

This example shows the network traffic statistics for the datalink net6 that does not have dedicated Rx rings. The `sw` value under the `ID` column in the output indicates that the datalink net6 is not configured with dedicated hardware rings.

```
# dlstat show-link -r net6
```

**Displaying Network Traffic Statistics of Link Aggregations**

The `dlstat show-aggr` command shows network packet statistics for each aggregation's ports when traffic traverses the aggregation on the system.

**EXAMPLE 8-10**  
Displaying Network Traffic Statistics for Link Aggregations

```
# dlstat show-aggr
```

In this example, the output indicates the configuration of a link aggregation `aggr0` with two underlying links, `net0` and `net3`. As network traffic is received or sent by the system through...
the aggregation, information about incoming and outgoing packets and their respective sizes is
reported for every port. The ports are identified by the underlying links of the aggregation.

For information about link aggregations, see Chapter 2, “Configuring High Availability by
Using Link Aggregations,” in “Managing Network Datalinks in Oracle Solaris 11.2 ”.

### Displaying Network Traffic Statistics of Bridges

The `dlstat` `show-bridge` command shows network statistics for each bridge and lists the
statistics of the links connected to each bridge.

**EXAMPLE 8-11  Displaying Network Traffic Statistics for Bridges**

In this example, the network statistics for the bridges `rbblue0` and `stbred0` are displayed.

```
# dlstat show-bridge

           BRIDGE        LINK     IPKTS    RBYTES     OPKTS    OBYTES     DROPS  FORWARDS
rbblue0         --     1.93K   587.29K     2.47K     3.30M         0         0
simblue1     72     4.32K     2.12K     2.83M         0        --
simblue2  1.86K   582.97K       348   474.04K         0        --
stbred0         --       975   976.69K     3.44K     1.13M         0        38
simred3    347   472.54K     1.86K   583.03K         0        --
simred4    628   504.15K     1.58K   551.51K         0        --
```

### Displaying Network Traffic Statistics of Flows

Statistics on flows help you to evaluate packet traffic on all the defined flows on a system.
To display the statistics on flows, use the `flowstat` command. For more information, see the
`flowstat(1M)` man page.

Use the following command syntax to display network traffic statistics on flows:

```
# flowstat [-r|-t] [-l link] [-Tu | -Td] [flow] [interval [count]]
```

- **-r** Displays receive-side network traffic statistics only. You should not specify the -t option with this option.
  
  If you do not specify the -r option or -t option, both the transmit-side and receive-side network statistics are displayed.

- **-t** Displays transmit-side network traffic statistics only. You should not specify the -r option with this option.
  
  If you do not specify the -r option or the -t option, both the transmit-side and receive-side network statistics are displayed.
### Displaying Network Traffic Statistics of Flows

- **-l link**  
  Name of the datalink whose network statistics you want to monitor.  
  If you do not specify the datalink, then the information about all the configured flows on the system are displayed.

- **-Tu**  
  Displays the current time in internal representation.

- **-Td**  
  Displays the current time in standard date format.

- **flow**  
  Name of the flow whose network statistics you want to monitor. If you do not specify the flow, then depending on the specified link, all the flow statistics are displayed.

- **interval**  
  Specifies the time in seconds at which you want to refresh the network statistics. If you do not specify the interval value, then the total number of packets and bytes is displayed.

- **count**  
  Specifies the number of times you want the displayed network traffic statistics to be refreshed. If you do not specify the count value, the statistics are refreshed indefinitely.

The following examples show different ways to display information about configured flows on the system.

#### EXAMPLE 8-12  Displaying Network Traffic Statistics for Flows

In this example, network traffic statistics for all the configured flows on the system are displayed with an interval value of 1 second and the count value of 2.

```
# flowstat 1 2

<table>
<thead>
<tr>
<th>FLOW</th>
<th>IPKTS</th>
<th>RBYTES</th>
<th>IDROPS</th>
<th>OPKTS</th>
<th>OBYTES</th>
<th>ODROPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>flow1</td>
<td>1.78M</td>
<td>2.68G</td>
<td>443</td>
<td>889.57K</td>
<td>58.72M</td>
<td>0</td>
</tr>
<tr>
<td>flow2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>flow1</td>
<td>8.31K</td>
<td>12.51M</td>
<td>243</td>
<td>4.22K</td>
<td>280.45K</td>
<td>0</td>
</tr>
<tr>
<td>flow2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Consider the flows, *flow1* and *flow2*, as a set. The first set of flows, *flow1* and *flow2*, show the total number of network traffic statistics received and transmitted by the flows. In this example, 1.78M is the total number of packets received by *flow1*. The second set of flows, *flow1* and *flow2*, show the network statistics in rates per second, also known as the normalized value. In this example, 8.31K is the normalized value of the packets received by *flow1* in the interval of 1 second.

#### EXAMPLE 8-13  Displaying Transmit-Side Traffic Statistics for Flows

In this example, the network traffic statistics about outgoing traffic for all the configured flows on the system are displayed.
EXAMPLE 8-14 Displaying Receive-Side Traffic Statistics for Flows on a Datalink

In this example, incoming network traffic for all the configured flows on the datalink \texttt{net0} are displayed with an interval value of 2 seconds and the count value of 5.

```
# flowstat -r -l net0 2 5
FLOW     IPKTS   RBYTES   IDROPS
flow1    2.38M    3.59G   14.89K
flow2        0        0        0
flow3    8.24K   12.40M      180
flow2        0        0        0
flow3    8.94K   13.47M      206
flow2        0        0        0
flow3    7.43K   11.19M      161
flow2        0        0        0
flow3    8.38K   12.62M      213
flow2        0        0        0
```

Consider the flows, flow1 and flow2, as a set. The first set of flows, flow1 and flow2, show the total number of packets and bytes received by the flows. In this example, 2.38M is the total number of packets received and 3.59G is the total number of bytes received by flow1. The second set of flows, flow1 and flow2, show the network statistics in rates per second, also known as the normalized value. In this example, 8.24K is the normalized value of the packets received by flow1 in the interval of 2 seconds. Similarly, the succeeding sets of flows also show the normalized value for the network traffic statistics in the periodic interval of 2 seconds.

EXAMPLE 8-15 Displaying Traffic Statistics for Flows With Time

The following example displays statistics about incoming traffic on all the flows that are created over the datalink \texttt{net0} with the internal representation of the current time.

```
# flowstat -r -l net0 -Tu
1364380279
FLOW     IPKTS   RBYTES   IDROPS
tcp-flow  183.11K  270.24M        0
udp-flow        0        0        0
```

The following example displays statistics about incoming traffic on all the flows that are created over the datalink \texttt{net0} with the current time in standard date format.

```
# flowstat -r -l net0 -Td
Wednesday, March 27, 2013 04:01:01 PM IST
FLOW     IPKTS   RBYTES   IDROPS
tcp-flow  183.11K  270.24M        0
udp-flow        0        0        0
```
Configuring Network Accounting for Network Traffic

You can use the extended accounting facility to set up network accounting on the system. Network accounting involves capturing statistics about network traffic in a log file. You can maintain records of traffic for tracking, provisioning, consolidation, and billing purposes. Later, you can see the log file to obtain historical information about network use over a period of time.

To set up network accounting, use the extended accounting facility's `acctadm` command. For more information, see the `acctadm(1M)` man page. After you have completed setting up network accounting, use the `flowstat` command to record traffic statistics.

### How to Set Up Network Accounting

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

2. **View the status of the accounting types that can be enabled by the extended accounting facility.**

   ```
   # acctadm [process | task | flow | net]
   ```

   The extended accounting facility can enable four types of accounting. The optional operands of the `acctadm` command correspond to the following accounting types:

   - **process** – Process accounting
   - **task** – Task accounting
   - **flow** – Flow accounting
   - **net** – Network accounting

   **Note** - Network accounting also applies to flows that are managed by the `flowadm` and `flowstat` commands as discussed in “Managing Network Resources by Using Flows” on page 172. Therefore, to set up accounting for these flows, use the net option with the `acctadm` command. Do not use the `flow` option, which enables flow accounting for IPQoS configurations.

   Specifying `net` displays the status of network accounting. If `net` is not used, then the status of all four accounting types is displayed.

3. **Enable the extended accounting for network traffic.**
How to Set Up Network Accounting

Chapter 8 • Monitoring Network Traffic and Resource Usage

# acctadm -e extended -f filename net

where *filename* includes the full path of the log file that captures network traffic statistics. The log file can be created in any directory that you specify.

4. **Verify that extended network accounting has been activated.**

   # acctadm net

**Example 8-16** Setting Up Network Accounting on the System

This example shows how to configure network accounting to capture and display historical traffic information on the system.

View the status of all accounting types as follows:

```
# acctadm
```

- Task accounting: inactive
- Task accounting file: none
- Tracked task resources: none
- Untracked task resources: extended
- Process accounting: inactive
- Process accounting file: none
- Tracked process resources: none
- Untracked process resources: extended,host
- Flow accounting: inactive
- Flow accounting file: none
- Tracked flow resources: none
- Untracked flow resources: extended
- Net accounting: inactive
- Network accounting file: none
- Tracked Network resources: none
- Untracked Network resources: extended

The output shows that network accounting is not active. Therefore, you should enable extended network accounting.

```
# acctadm -e extended -f /var/log/net.log net
# acctadm net
```

```
Net accounting: active
Net accounting file: /var/log/net.log
Tracked net resources: extended
Untracked net resources: none
```

**Displaying Historical Statistics on Network Traffic**

After you have enabled network accounting, you can use the `dlstat` and `flowstat` commands to extract information from the log file.
You must enable extended accounting for the network before you can display historical data about the network. Further, to display historical data about traffic on flows, you must first configure flows on the system, as explained in “Managing Network Resources by Using Flows” on page 172.

Displaying Historical Network Traffic Statistics on Datalinks

You can display historical network traffic statistics on datalinks by using the following command syntax:

```
# dlstat show-link -h [-a] [-f filename] [-d date] [-F format] [-s start-time] [-e end-time] [link]
```

- `-h` Displays a summary of historical information about resource usage by incoming and outgoing packets on datalinks.
- `-a` Displays resource usage on all datalinks, including those that have already been deleted after the data capture.
- `-f filename` Specifies the log file that was defined when network accounting was enabled with the `acctadm` command.
- `-d date` Displays logged information for the specified date.
- `-F format` Displays the data in a specific format that can then be plotted for analysis. Currently, `gnuplot` is the only supported format.
- `-s start-time` Specifies the start time to display the logged information of the network statistics. Use the `MM/DD/YYYY,hh:mm:ss` format. The hour (`hh`) must use 24-hour clock notation. If you do not include the date, then data for the specified time range for the current date is displayed.
- `-e end-time` Specifies the end time to display the logged information of the network statistics. Use the `MM/DD/YYYY,hh:mm:ss` format. The hour (`hh`) must use 24-hour clock notation. If you do not include the date, then data for the specified time range for the current date is displayed.
- `link` Displays historical data for a specified datalink. If you do not use this option, then historical network data for all configured datalinks is displayed.

**EXAMPLE 8-17** Displaying Historical Statistics About Resource Usage on Datalinks

In this example, the historical statistics about network traffic and its use of resources on all the datalinks in a system are displayed.
# dlstat show-link -h -f /var/log/net.log

<table>
<thead>
<tr>
<th>LINK</th>
<th>DURATION</th>
<th>IPKTS</th>
<th>RBYTES</th>
<th>OPKTS</th>
<th>OBYTES</th>
<th>BANDWIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>net0</td>
<td>80</td>
<td>1031</td>
<td>546908</td>
<td>0</td>
<td>0</td>
<td>2.44 Mbps</td>
</tr>
<tr>
<td>net1</td>
<td>100</td>
<td>2045</td>
<td>235977</td>
<td>0</td>
<td>0</td>
<td>9.67 Mbps</td>
</tr>
</tbody>
</table>

Displaying Historical Network Traffic Statistics on Flows

You can display historical network traffic statistics on flows by using the following command syntax:

```
# flowstat -h [-a] [-f filename] [-d date] [-F format] [-s start-time] [-e end-time] [flow]
```

- **-h** Displays a summary of historical information about resource usage by incoming and outgoing packets on configured flows.

- **-a** Displays resource usage on all configured flows, including those that have already been deleted after the data capture.

- **-f filename** Specifies the log file that was defined when network accounting was enabled with the `acctadm` command.

- **-d** Displays logged information for the specified date.

- **-F format** Displays the data in a specific format. Currently, `gnuplot` is the only supported format.

- **-s start-time** Specifies the start time to display the logged information of the network statistics. Use the `MM/DD/YYY,hh:mm:ss` format. The hour (hh) must use 24-hour clock notation. If you do not include the date, then data for the specified time range for the current date is displayed.

- **-e end-time** Specifies the end time to display the logged information of the network statistics. Use the `MM/DD/YYY,hh:mm:ss` format. The hour (hh) must use 24-hour clock notation. If you do not include the date, then data for the specified time range for the current date is displayed.

- **flow** Displays historical data for a specified flow. If you do not use this option, then historical network data for all configured flows is displayed.

**EXAMPLE 8-18**  
Displaying Historical Statistics About Resource Usage on Flows

The following example displays historical statistics of resource usage by traffic on the flows in a system.

```
# flowstat -h -f /var/log/net.log
```
How to Set Up Network Accounting

The following example displays historical statistics of resource usage by traffic on `flowtcp` over a given date and time range.

```
# flowstat -h -s 02/19/2008,10:39:06 -e 02/19/2008,10:40:06 \ 
-f /var/log/net.log flowtcp
```

<table>
<thead>
<tr>
<th>FLOW</th>
<th>DURATION</th>
<th>IPACKETS</th>
<th>RBYTES</th>
<th>OPACKETS</th>
<th>OBYTES</th>
<th>BANDWIDTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>tcp</td>
<td>100</td>
<td>1031</td>
<td>546908</td>
<td>0</td>
<td>0</td>
<td>43.76Kbps</td>
</tr>
<tr>
<td>udp</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.00Mbps</td>
</tr>
</tbody>
</table>

The following example displays historical statistics of resource usage by traffic on `flowtcp` over a given date and time range by using gnuplot format.

```
# flowstat -h -s 02/19/2008,10:39:06 -e 02/19/2008,10:40:06 \ 
-F gnuplot -f /var/log/net.log flowtcp
```

# Time tcp-flow

```
10:39:06 3.23
10:39:26 5.40
10:39:46 0.18
10:40:06 0.00
```
Index

A
access control lists (ACLs), 74
acctadm command, 192
allocating CPUs to a datalink, 171
attributes for configuring flows
  DS field, 173
  local IP address, 172
  remote IP address, 172
  transport protocol with application port number, 173
  transport protocol with application port number and IP address, 173
  transport protocols, 173
automatically generated VXLAN datalinks, 108
configuring flows, 174
  commands, 173
configuring network accounting, 192
controlling communication in VMs, 79
CPU pools, 167
  assigning to links, 170
  datalink, 19
  default pool, 168
  working with, 168
CPUs, 20, 168
  CPU allocation, 171
  creating
    IP interface, 25
  creating VF VNICs, 50
  creating VNICs in zones temporarily, 34

B
bandwidth
  setting on datalinks, 19
  setting on flows, 19, 175

C
changing
  default EVB configuration, 88
commands
  administering EVS, 104
  allocating resources in datalinks, 158
  configuring the components of virtual network, 23
  managing rings, 160
  monitoring network traffic statistics, 183
  resource allocation in flows, 173
components of virtual network
  configuring, 23
configuring
  VXLAN, 62

D
data center consolidation, 19
datalink
  EVB properties, 88
datalink properties
    cpu, 157
    maxbw, 157
    pool, 157
  rxrings and txrings, 157
data links
  allocating bandwidth, 19
  allocating CPU pools, 19
  allocating CPUs, 20
  displaying historical network traffic statistics, 194
  resource control properties, 20, 158
DCBX protocol, 75
deleting a VNIC attached to a zone, 47
deleting VNICs, 46
deleting VXLAN, 66
Index

displaying
  elastic virtual switch information, 125, 128, 132
  EVB-related datalink properties, 90
  EVS controller properties, 123
  IPnet, 136
  physical link state, 40
  VDP and ECP state and statistics, 86
  virtual link state, 40
  VPort, 140
displaying historical network traffic statistics
  on datalinks, 193
  on flows, 193
displaying VFs, 52
displaying VXLAN, 66
dladm command, 104, 107
  create-etherstub, 24, 35
  create-vlan, 159
  create-vnic, 23, 107, 158, 170
  create-vxlan, 62
  delete-vnic, 46
  delete-vxlan, 66
  modify-vnic, 41, 43, 45
  set-linkprop, 26, 158
  show-ether, 86
  show-link, 66
  show-linkprop, 49, 160, 161
  show-phys, 52, 161, 161
  show-vnic, 37, 39, 52, 107
  show-vxlan, 62, 66
dlstat command, 24, 183, 183
  show-aggr, 188
  show-bridge, 189
  show-link, 194
  show-phys, 184
dlstat show-ether command, 87
DS field, 173

e
edge control protocol (ECP), 84
  displaying ECP state and statistics, 87
edge virtual bridging, 74 See EVB
  access control lists, 74
  automatic virtual port configuration, 75
  changing the default EVB configuration, 88
  components, 84
  controlling communication in VMs, 79
  datalink properties, 88
  enabling VMs to communicate through an external switch, 79
  example of displaying the link properties, 87
  example of displaying the VDP state, 86
  example of displaying the VDP statistics, 87
  exchanging VNIC information, 84
  how EVB increases network and server efficiency, 75
  how VNIC information exchange works, 85
  installing, 78
  Oracle VSI Manager, 84
  oracle_v1, 84
  oracle_v1 encoding, 84
  overview, 74
  reflective relay, 74, 74
  server efficiency with and without EVB, 77
  using LLDP to manage communication between VMs, 82
  using VDP, 84
  VDP protocol, 84
  VDP state for Ethernet links, 86
  VSI identifier, 84
  VSI Manager, 84
  VSI Manager ID, 84
  VSI profile, 84
  VSI Type ID, 84, 88
  VSI Version, 84, 88
elastic virtual switch, 94, 108
  administering, 131
  associating a block of IP addresses with, 96
  associating virtual ports with, 96
  configuring based on a VLAN, 146
  configuring based on a VXLAN for a tenant, 151
  creating, 128
  creating a VNIC, 129
  creating a VNIC anet resource, 130
  deleting, 142
  displaying, 128, 132
  displaying properties, 134
  mandatory package, 125
  monitoring, 143
  overview, 93
planning, 114
resources
   IP network, 98
   virtual port, 98
setting properties, 133
enabling
   VMs to communicate through an external switch, 79
etherstubs, 14, 15
   configuring, 24
   creating, 24
private virtual networks, 34
EVB, 18 See edge virtual bridging
EVS
   administration tasks, 113
   benefits, 97
   mandatory packages, 109
   namespace management, 100
   security requirements, 110
   working with zones, 109
EVS clients, 104
EVS components, 100
   EVS clients, 100
   EVS controller, 100
   EVS manager, 100
   EVS nodes, 100
EVS controller, 102
   configuring, 118
   creating and administering, 114
   displaying, 123
   displaying properties, 123
   mandatory packages, 109, 115
   planning, 114
   setting, 123
   setting properties, 123
EVS controller properties, 102
   displaying, 118
   setting, 118
EVS manager
   description of, 101
EVS nodes
   definition of, 104
evsadm
   subcommands for managing a VPort, 106
   subcommands for managing an elastic virtual switch, 105
   subcommands for managing an IPnet, 105
   subcommands for managing EVS client properties, 106
   subcommands for managing EVS controller properties, 106
evsadm command, 104, 105
   add-ipnet, 105, 128
   add-vport, 106, 128
   create-evs, 105, 128
   delete-evs, 105, 142
   remove-ipnet, 106, 136
   remove-vport, 106, 141
   reset-vport, 106, 141
   set-controlprop, 106, 123
   set-evsprop, 105, 106, 133
   set-prop, 101, 102, 106, 123
   set-vportprop, 138
   show-controlprop, 107, 123
   show-evs, 105, 132
   show-evsprop, 105, 134
   show-ipnet, 106, 136
   show-prop, 106, 123
   show-vport, 106, 140
   show-vportprop, 106, 138
evsstat command, 143
   displaying network traffic statistics for elastic virtual switch, 107
example of assigning the VNIC created over a VXLAN to a zone, 65
example of displaying ECP state and statistics, 87
example of displaying EVB-related datalink properties on a physical link, 90
example of displaying EVB-related properties on a VNIC, 91
example of setting EVB-related datalink properties, 89
exchanging VNIC information by using VDP, 84
extended accounting facility, 192
   flow accounting, 192
   network accounting for links and flows, 192
   process accounting, 192
   task accounting, 192
external virtual networks, 13
F
flow control, 172
flowadm command, 172, 175
  add-flow, 173, 174, 179
  help, 174
  set-flowprop, 173, 175, 179
  show-flowprop, 175, 179
flows, 20, 24, 172
  bandwidth setting, 173, 175
  based on attributes, 172
  configuring, 174
  creating, 173, 174
  displaying historical network traffic statistics, 195
  displaying information, 175
  priority setting, 173
  setting properties, 173
flowstat command, 24, 183, 189

G
GARP VLAN Registration Protocol (GVRP), 26

H
hardware rings, 159
hardware-based clients, 159

I
installing
  edge virtual bridging, 78
internal virtual networks, 13
IP address, 98
IP network See IPnet
ipadm command, 178
  create-addr, 25
  create-ip, 25
  displaying IP information, 178
  show-addr, 62
IPnet
  adding to an elastic virtual switch, 128
  administering, 131, 136
  displaying, 136
  removing, 136
IPoIB
  temporarily creating in zones, 34

L
layer 2 network, 56
LLDP, 75
  using LLDP to manage communication between VMs, 82
local application port number, 173
local IP address, 172

M
MAC address, on VNICs, 14
MAC clients, 159
  allocating rings, 162
  configuring, 162
  hardware-based, 159
  primary, 161
  software-based, 159
managing
  an elastic virtual switch, evsadm subcommands for, 105
  EVS client properties, evsadm subcommands for, 106
  EVS controller properties, evsadm subcommands for, 106
  an IPnet, evsadm subcommands for, 105
  a VPort, evsadm subcommands for, 106
managing CPUs, 168
managing network datalinks
  EVB, 18
managing network resources, 157
  using datalink properties, 157
  using flows, 172
managing rings
  commands, 160
migrating
  VF VNICs, 51
  VNICs, 44
monitoring
  network traffic statistics
    commands, 183
    network use, 181
Index

N
network accounting, 192, 192
network lanes, 19
network resource management, 13, 19, 157
  benefits, 20
  by setting datalink and flow properties, 176
  by using datalink properties, 19, 20
  by using flows, 19, 20
  CPU pools, 167
  CPUs, 168
dladm subcommands for implementation, 158
NIC rings, 158
network resources
  bandwidth, 19
  CPU pools, 19
  CPUs, 19
  NIC rings, 19
  priority, 19
network statistics, 183
  historical traffic information, 193
  monitoring network use, 181
network traffic statistics
  displaying for bridges, 189
  displaying for datalinks, 187
  displaying for flows, 189
  displaying for link aggregations, 188
  displaying for links, 183
  displaying for network devices, 184
network virtualization, 13
  data center consolidation, 19
  dladm subcommands for implementation, 158
  etherstubs, 15
  virtual network, 13
  virtual switches, 15
  with network resource management, 13
  zones, 16
NIC rings, 19, 19
  See also ring allocation
  receive and transmit, 158

O
Oracle Solaris Elastic Virtual Switch, 93, 95
  See also EVS
Oracle Solaris Kernel Zones, 109
Oracle VSI Manager, 78, 84
oracle_v1 encoding, 78
  definitions, 84
ORACLE_VSIMGR_V1, 84

P
physical link state
  displaying, 40
primary client, 159, 161
priority
  datalink, 19
  flows, 19, 173
private virtual networks, 13, 15
  configuring with etherstubs, 34
  protocols
    DCBX, 75
    ECP, 84
    VDP, 84
pseudo Ethernet NICs See etherstubs

Q
quality of service (QoS), 19, 157

R
receive rings See Rx rings
reflective relay, 15, 74
remote application port number, 173
remote IP address, 172
ring allocation, 159, 159
  on VLANs, 159
  ring use and ring assignments, 161

S
SCTP, 173
service-level agreement See SLA
single root I/O virtualization See SR-IOV
SLA
  virtual switches and, 94
software-based clients, 159, 159
SR-IOV
  checking for datalinks, 49
  enabling for datalinks, 49
virtual function See VF with VNICs, 48
SSH authentication setting up, 118 subnet, 98 system-created VNICs, 24 displaying, 38

T
TCP, 173
temporarily creating VNICs in zones, 33
temporarily creating VNICs, VLANs, and IPoIB partitions in zones, 34
tenant definition of, 100 transmit rings See Tx rings

U
UDP, 173
uplink port definition of, 96 using LLDP to manage communication between VMs, 82

V
VDP See VSI discovery and configuration protocol VDP statistics, displaying command, 87 VDP TLV, 84 VF information displaying, 52 VF VNICs creating, 50 displaying, 52 migrating, 51 virtual extensible local area network See VXLAN virtual extensible local area networks See VXLANs virtual link state displaying, 40 virtual machines, 94 virtual network, 17, 56 components of, 14 etherstubs, 14 virtual switches, 14 VNICs, 14 zones, 14 virtual network interface cards See VNICs virtual networks, 13, 28 See also VNICs building, 28 configuring a etherstub, 24 configuring a VNIC, 24 configuring zones for, 28 exclusive IP-type zones, 32 external and internal, 13 implementing, 18 private, 13 reconfiguring zones for, 30 zones, 28 virtual NICs See VNICs virtual port, 94 See VPort virtual switch, 14, 15 virtual switches, 94 VLANs, 95, 97 as VNICs, 26 modifying the VLAN ID of a VNIC, 41 temporarily creating in zones, 34 VNI, 57 VNIC anet resource, 104, 107 VNICs, 13, 14, 14 as VLANs, 23 assigning a static IP address, 25 assigning CPU pool resources, 170 assigning to a zone, 31 changing the underlying link, 44 configuring, 24, 28 configuring in the zone, 31 creating, 23 creating a VNIC IP interface, 25 creating temporarily in zones, 33 deleting, 46 displaying information, 37 displaying multiple MAC addresses, 38 managing, 37 migrating, 44 modifying MAC addresses, 42 modifying VLAN IDs, 41 private virtual networks, 34 setting properties, 158
system-created, 24
using SR-IOV, 48
using with zones, 30
with VLAN IDs, 26
VPort
adding to an elastic virtual switch, 128
administering, 131, 137
displaying, 140
displaying properties, 138
removing, 141
resetting, 141
setting properties, 138
VPort properties, 98
IP address, 98
MAC address, 98
SLA
class of service, 98
maximum bandwidth, 98
priority, 98
VSI See virtual station instance
VSI discovery and configuration protocol (VDP)
displaying VDP statistics, 87
VDP state for Ethernet links, 86
VDP TLV, 84
VSI identifier, 84
VSI Manager, 84
oracle_v1, 84
VSI Manager ID
ORACLE_VSIMGR_V1, 84
VSI profile, 84
VSI Type ID, 84
VSI Version, 84
VXLAN, 18
eexample of assigning a VXLAN to a zone's
anet, 67
eexample of creating a VXLAN, 64
VXLAN segment ID, 57, 61
VXLAN segment IDs See VNIs
VXLANs, 55, 95, 97, 108
advantages, 56
anet resource, 59
assigning to a zone, 67
configuring a VXLAN, 62
configuring for zones, 62
deleting, 66
displaying, 66
lower-link, 67
naming convention, 57
overview, 55
planning a configuration, 61
requirements, 61
topology, 57
using with zones, 59
VNIC, 59
VXLAN end points, 57
VXLAN segment, 59
Z
zone
assigning a VXLAN, 67
zoneadm command, 29
zonecfg command, 29, 104, 171, 180
zones, 14, 104
assigning VNICS, 31
configuring for network virtualization, 28
reconfiguring for virtual networks, 30
temporarily creating VNICS, 33
using VXLAN, 59