Logical Domains (LDoms) 1.0
Administration Guide

Sun Microsystems, Inc.
www.sun.com

Part No. 819-6428-11
May 2007, Revision B

Submit comments about this document at: http://www.sun.com/hwdocs/feedback

---

Contents

Figures

Tables

Code Samples

Preface

The *Logical Domains (LDoms) 1.0 Administration Guide* provides detailed information and procedures that describe the overview, security considerations, installation, configuration, modification, and execution of common tasks for the Logical Domains Manager 1.0 software on Sun Fire™ and SPARC® Enterprise T1000 and T2000 Servers, Netra™ T2000 Servers, Netra CP3060 Blades, and Sun Blade™ T6300 Server Modules. This guide is intended for the system administrators on these servers who have a working knowledge of UNIX® systems and the Solaris™ Operating System (Solaris OS).

Before You Read This Document

If you do not have a working knowledge of UNIX commands and procedures and your Solaris Operating System, read the Solaris OS user and system administrator documentation provided with your system hardware, and consider UNIX system administration training.

How This Book Is Organized

This guide contains the following information:

Chapter 1 provides an overview of the Logical Domains Manager software.

Chapter 2 discusses the Solaris™ Security Toolkit, and how it can provide security for the Solaris OS in logical domains.

Chapter 3 provides detailed procedures for installing and enabling Logical Domains Manager software.

Chapter 4 provides detailed procedures for setting up services and logical domains.

Chapter 5 provides other information and procedures for executing common tasks in using Logical Domain Manager software to manage logical domains.

Appendix A provides the reference information, man page, for the Logical Domain Manager, `ldm`(1M).

Glossary is a list of LDoms-specific abbreviations, acronyms, and terms and their definitions.

Using UNIX Commands

This document might not contain information on basic UNIX commands and procedures such as shutting down the system, booting the system, and configuring devices. Refer to the following for this information:

- Software documentation that you received with your system
- Solaris Operating System documentation, which is at

  `http://docs.sun.com`

Shell Prompts

Shell	Prompt
C shell	*machine-name*`%`
C shell superuser	*machine-name*`#`
Bourne shell and Korn shell	`$`
Bourne shell and Korn shell superuser	`#`

Typographic Conventions

Typeface[*]	Meaning	Examples
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail.`

`AaBbCc123`	What you type, when contrasted with on-screen computer output	`%` **`su`**
		`Password:`

AaBbCc123	Book titles, new words or terms, words to be emphasized. Replace command-line variables with real names or values.	Read Chapter 6 in the *User's Guide.* These are called *class* options. To delete a file, type **`rm`** *filename*.

[*] The settings on your browser might differ from these settings.

Related Documentation

The *Logical Domains (LDoms) 1.0 Administration Guide* and *Release Notes* are available at:

`http://www.sun.com/products-n-solutions/hardware/docs/Software/enterprise_computing/systems_management/ldoms/ldoms1_0/index.html`

The *Beginners Guide to LDoms: Understanding and Deploying Logical Domains Software* can be found at the Sun BluePrints™ site at:

`http://www.sun.com/blueprints/0207/820-0832.html`

You can find documents relating to your server or your Solaris OS at:

`http://www.sun.com/documentation/`

Type the name of your server or your Solaris OS in the Search box to find the documents you need.

Application	Title	Part Number	Format	Location
Release notes	*Logical Domains (LDoms) 1.0 Release Notes*	819-6429-10	HTML PDF	Online
Solaris man pages for LDoms	Solaris 10 Reference Manual Collection: • `drd`(1M) man page • `vntsd`(1M) man page	N/A	HTML PDF	Online
LDoms man page	`ldm`(1M) man page	N/A	SGML	Sun Software Download site with LDoms code
Basics for Logical Domains software	*Beginners Guide to LDoms: Understanding and Deploying Logical Domains Software*	820-0832	PDF	Online
Solaris OS including installation, using JumpStart™, and using the Service Management Facility (SMF)	Solaris 10 Collection	N/A	HTML PDF	Online
Security	*Solaris Security Toolkit 4.2 Administration Guide*	819-1402-10	HTML PDF	Online
Security	*Solaris Security Toolkit 4.2 Reference Manual*	819-1503-10	HTML PDF	Online

Application	Title	Part Number	Format	Location
Security	*Solaris Security Toolkit 4.2 Release Notes*	819-1504-10	HTML PDF	Online
Security	*Solaris Security Toolkit 4.2 Man Page Guide*	819-1505-10	HTML PDF	Online
Sun Management Center compatible with Logical Domains Manager 1.0 supported servers	*Sun Management Center 3.6 Version 6Add-On Software Release Notes:* *For Sun Fire, Sun Blade, Netra, and Sun Ultra™ Systems*	820-1041-10	PDF	Online
SunVTS™	*SunVTS 6.3 User's Guide*	820-0080-10	HTML PDF	Online
ALOM CMT compatible with Logical Domains Manager 1.0 and Sun Fire and SPARC Enterprise T1000 and T2000 Servers	*Advanced Lights Out Management (ALOM) CMT v1.3 Guide*	819-7981-11	HTML PDF	Online
Sun™ Explorer 5.7 Data Collector	Sun Explorer User's Guide		HTML PDF	Online
Sun Fire and SPARC Enterprise Sun Fire T1000 Servers	*Sun Fire* and *SPARC Enterprise T1000 Server Administration Guide*		HTML PDF	Online
Sun Fire and SPARC Enterprise Sun Fire T2000 Servers	*Sun Fire* and *SPARC Enterprise T2000 Server Administration Guide*		HTML PDF	Online
Netra T2000 Servers	*Netra T2000 Server Administration Guide*	819-5837-10	PDF	Online
Netra CP3060 Blades	*Netra CP3060 Board Product Notes*	819-4966-12	PDF	Online
Sun Blade T6300 Server Modules	*Sun Blade T6300 Server Module Administration Guide* *Sun Blade T6300 Server Module Installation Guide* *Sun Blade T6300 Server Module Product Notes*	820-0277-10 820-0275-10 820--0278-10	PDF	Online

Documentation, Support, and Training

Sun Function	URL
Documentation	`http://www.sun.com/documentation/`
Support	`http://www.sun.com/support`
Training	`http://www.sun.com/training/`

Third-Party Web Sites

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

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can submit your comments by going to:

`http://www.sun.com/hwdocs/feedback`

Please include the title and part number of your document with your feedback:

Logical Domains (LDoms) 1.0 Administration Guide, part number 819-6428-11

CHAPTER 1

Overview of the Logical Domains Software

This chapter provides an overview of the Logical Domains software. All of the Solaris OS functionality necessary to use Sun's Logical Domains technology is in the Solaris 10 11/06 release. However, system firmware and the Logical Domains Manager are also required to use logical domains. Refer to "Required and Recommended Software" in the *Logical Domains (LDoms) 1.0 Release Notes* for specific details.

Hypervisor and Logical Domains

This section provides a brief overview of the SPARC hypervisor and the logical domains it supports.

The SPARC hypervisor is a small firmware layer that provides a stable virtualized machine architecture to which an operating system can be written. Sun servers using the hypervisor provide hardware features to support the hypervisor's control over a logical operating system's activities.

A logical domain is a discrete logical grouping with its own operating system, resources, and identity within a single computer system. Each logical domain can be created, destroyed, reconfigured, and rebooted independently, without requiring a power cycle of the server. You can run a variety of applications software in different logical domains and keep them independent for performance and security purposes.

Each logical domain is allowed to observe and interact with only those server resources made available to it by the hypervisor. Using the Logical Domains Manager, the system administrator specifies what the hypervisor should do through the control domain. Thus, the hypervisor enforces the partitioning of the resources of a server and provides limited subsets to multiple operating system environments. This is the fundamental mechanism for creating logical domains. The following diagram shows the hypervisor supporting two logical domains. It also shows the layers that make up the Logical Domains functionality:

- Applications, or user/services
- Kernel, or operating systems
- Firmware, or hypervisor
- Hardware, including CPU, memory, and I/O

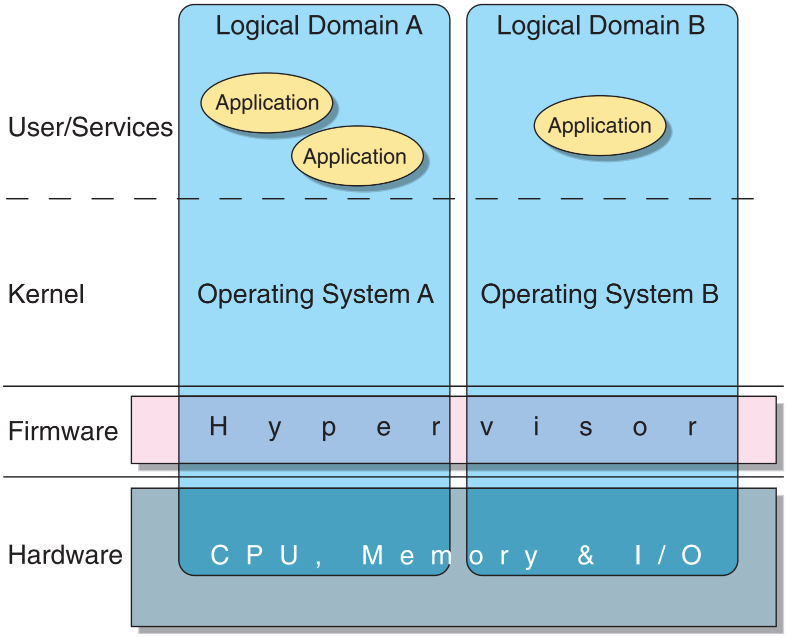

FIGURE 1-1 Hypervisor Supporting Two Logical Domains

The number and capabilities of each logical domain that a specific SPARC hypervisor supports are server-dependent features. The hypervisor can allocate subsets of the overall CPU, memory, and I/O resources of a server to a given logical domain. This enables support of multiple operating systems simultaneously, each within its own logical domain. Resources can be rearranged between separate logical domains with an arbitrary granularity. For example, memory is assignable to a logical domain with an 8-kilobyte granularity.

Each virtual machine can be managed as an entirely independent machine with its own resources, such as:

- Kernel, patches, and tuning parameters
- User accounts and administrators
- Disks

- Network interfaces, MAC addresses, and IP addresses

Each virtual machine can be stopped, started, and rebooted independently of each other without requiring a power cycle of the server.

The hypervisor software is responsible for maintaining the separation between logical domains. The hypervisor software also provides logical domain channels (LDCs), so that logical domains can communicate with each other. Using logical domain channels, domains can provide services to each other, such as networking or disk services.

The system controller monitors and runs the physical machine, but it does not manage the virtual machines. The Logical Domains Manager runs the virtual machines.

Logical Domains Manager

The Logical Domains Manager is used to create and manage logical domains. There can be only one Logical Domains Manager per server. The Logical Domains Manager maps logical domains to physical resources.

Roles for Logical Domains

All logical domains are the same except for the roles that you specify for them. There are multiple roles that logical domains can perform:

TABLE 1-1 Logical Domain Roles

Domain Role	Description
Control domain	Domain in which the Logical Domains Manager runs allowing you to create and manage other logical domains and allocate virtual resources to other domains. There can be only one control domain per server. The initial domain created when installing Logical Domains software is a control domain and is named `primary`.

TABLE 1-1 Logical Domain Roles *(Continued)*

Domain Role	Description
Service domain	Domain that provides virtual device services to other domains, such as a virtual switch, a virtual console concentrator, and a virtual disk server.
I/O domain	Domain which has direct ownership of and direct access to physical I/O devices, such as a network card in a PCI Express controller. Shares the devices to other domains in the form of virtual devices. You can have a maximum of two I/O domains, one of which also must be the control domain.
Guest domain	Domain that is managed by the control domain and uses services from the I/O and service domains.

If you have an existing system and already have an operating system and other software running on your server, that will be your control domain once you install the Logical Domains Manager. You might want to remove some of your applications from the control domain once it is set up, and balance the load of your applications throughout your domains to make the most efficient use of your system.

Command-Line Interface

The Logical Domains Manager provides a command-line interface (CLI) for the system administrator to create and configure logical domains.

To use the Logical Domains Manager CLI, you must have the Logical Domains Manager daemon, `ldmd`, running. The CLI is a single command, `ldm`(1M), with multiple subcommands. The `ldm`(1M) command and its subcommands are described in detail in Appendix A and the `ldm`(1M) man page.

To execute the `ldm` command, you must have the `/opt/SUNWldm/bin` directory in your UNIX $PATH variable. To access the `ldm`(1M) man page, add the directory path `/opt/SUNWldm/man` to the variable `$MANPATH`. Both are shown as follows:

```
# PATH=$PATH:/opt/SUNWldm/bin; export PATH (for Bourne or K shell)
# MANPATH=$MANPATH:/opt/SUNWldm/man; export MANPATH

% set PATH=($PATH /opt/SUNWldm/bin) (for C shell)
% set MANPATH=($MANPATH /opt/SUNWldm/man)
```

Virtual I/O

In a Logical Domains environment, an administrator can provision up to 32 domains on a Sun Fire or SPARC Enterprise T1000 or T2000 Server. Though each domain can be assigned dedicated CPUs and memory, the limited number of I/O buses and physical I/O slots in these systems makes it impossible to provide all domains exclusive access to the disk and network devices. Though some physical devices can be shared by splitting the PCI Express® (PCI-E) bus into two (see "Configuring Split PCI Express Bus to Use Multiple Logical Domains" on page 57), it is not sufficient to provide all domains exclusive device access. This lack of direct physical I/O device access is addressed by implementing a virtualized I/O model.

All logical domains with no direct I/O access are configured with virtual I/O devices that communicate with a service domain, which runs a service to export access to a physical device or its functions. In this client-server model, virtual I/O devices either communicate with each other or a service counterpart through interdomain communication channels called Logical Domain Channels (LDCs). In Logical Domains 1.0 software, the virtualized I/O functionality comprises support for virtual networking, storage, and consoles.

Virtual Network

The virtual network support is implemented using two components: the virtual network and virtual network switch device. The virtual network (`vnet`) device emulates an Ethernet device and communicates with other `vnet` devices in the system using a point-to-point channel. The virtual switch (`vsw`) device mainly functions as a [de]multiplexor of all the virtual network's incoming and outgoing packets. The `vsw` device interfaces directly with a physical network adapter on a service domain, and sends and receives packets on a virtual network's behalf. The vsw device also functions as a simple layer-2 switch and switches packets between the `vnet` devices connected to it within the system.

Virtual Storage

The virtual storage infrastructure enables logical domains to access block-level storage that is not directly assigned to them through a client-server model. It consists of two components: a virtual disk client (vdc) that exports as a block device interface; and a virtual disk service (vds) that processes disk requests on behalf of the virtual disk client and submits them to the physical storage residing on the service domain. Although the virtual disks appear as regular disks on the client domain, all disk operations are forwarded to the physical disk through the virtual disk service.

Virtual Consoles

In a Logical Domains environment, console I/O from all domains, except the `primary` domain, is redirected to a service domain running the virtual console concentrator (`vcc`) and virtual network terminal server (`vntsd`) services, instead of the systems controller. The virtual console concentrator service functions as a concentrator for all domains' console traffic, and interfaces with the virtual network terminal server daemon and exports access to each console through a UNIX socket.

Dynamic Reconfiguration

Dynamic reconfiguration (DR) is the ability to add or remove resources while the operating system is running. The Solaris 10 OS supports only the adding and removing of virtual CPUs (`vcpus`). Dynamic reconfiguration of memory and input/output is not supported in the Solaris 10 OS. To use the dynamic reconfiguration capability in the Logical Domains Manager CLI, you must have the Logical Domains dynamic reconfiguration daemon, `drd`(1M) running in the domain you want to change.

Delayed Reconfiguration

In contrast to dynamic reconfiguration operations that take place immediately, delayed reconfiguration operations take effect after the next reboot of the OS or stop and start of the logical domain if no OS is running. Any add or remove operations on active logical domains, except `add-vcpu`, `set-vcpu`, and `remove-vcpu` subcommands, are considered delayed reconfiguration operations. In addition, the `set-vswitch` subcommand on an active logical domain is considered a delayed reconfiguration operation.

When the Logical Domains Manager is first installed and enabled (or when the configuration is restored to `factory-default`), the LDoms Manager runs in the configuration mode. In this mode, reconfiguration requests are accepted and queued up, but are not acted upon. This allows a new configuration to be generated and stored to the SC without affecting the state of the running machine, and therefore, without being encumbered by any of the restrictions around things like delayed reconfiguration and reboot of I/O domains.

Once a delayed reconfiguration is in progress for a particular logical domain, any other reconfiguration requests for that logical domain are also deferred until the domain is rebooted or stopped and started. Also, when there is a delayed reconfiguration outstanding for one logical domain, reconfiguration requests for other logical domains are severely restricted and will fail with an appropriate error message.

Even though attempts to remove virtual I/O devices on an active logical domain will be handled as a delayed reconfiguration operation, some configuration change does occur immediately. This means the device will in fact stop functioning as soon as the associated Logical Domains Manager CLI operation is invoked.

The Logical Domains Manager subcommand `remove-reconf` cancels delayed reconfiguration operations. You can list delayed reconfiguration operations by using the `ldm list-domain` command. See Appendix A or the ldm(1M) man page for more information about how to use the delayed reconfiguration feature.

Note – You cannot use the `ldm remove-reconf` command if any other `ldm remove-*` commands have been issued on virtual I/O devices. The `ldm remove-reconf` command fails in these circumstances.

Persistent Configurations

The current configuration of a logical domain can be stored on the system controller (SC) using the Logical Domains Manager CLI commands. You can add a configuration, specify a configuration to be used, remove a configuration, and list the configurations on the system controller (see Appendix A). In addition, there is a ALOM CMT Version 1.3 command that enables you to select a configuration to boot (see "Using LDoms With ALOM CMT" on page 65).

CHAPTER 2

Security Considerations

The Solaris Security Toolkit software, informally known as the JumpStart™ Architecture and Security Scripts (JASS) toolkit, provides an automated, extensible, and scalable mechanism to build and maintain secure Solaris OS systems. The Solaris Security Toolkit provides security for devices critical to the management of your server, including the control domain in the Logical Domains Manager.

The Solaris Security Toolkit 4.2 software package, `SUNWjass`, provides the means to secure the Solaris Operating System on your control domain through the use of the `install-ldm` script by:

- Letting the Solaris Security Toolkit automatically harden your control domain by using the Logical Domains Manager install script (`install-ldm`) and the control driver specific to the Logical Domains Manager (`ldm_control-secure.driver`).
- Selecting an alternative driver when using the install script.
- Selecting no driver when using the install script and applying your own Solaris hardening.

The `SUNWjass` package is located with the Logical Domains (LDoms) Manager 1.0 software package, `SUNWldm`, at Sun's software download web site. You have the option to download and install the Solaris Security Toolkit 4.2 software package at the same time you download and install the Logical Domains Manager 1.0 software. The Solaris Security Toolkit 4.2 software package includes the required patches to enable the Solaris Security Toolkit software to work with the Logical Domains Manager. Once the software is installed, you can harden your system with Solaris Security Toolkit 4.2 software. Chapter 3 tells you how to install and configure the Solaris Security Toolkit, and harden your control domain.

Following are the security functions available to users of the Logical Domains Manager provided by the Solaris Security Toolkit:

- **Hardening** – Modifying Solaris OS configurations to improve a system's security using the Solaris Security Toolkit 4.2 software with required patches to enable the Solaris Security Toolkit to work with the Logical Domains Manager.

- **Authorization** – Setting up authorization using the Solaris OS Role-Based Access Control (RBAC) adapted for the Logical Domains Manager.
- **Auditing** – Using the Solaris OS Basic Security Module (BSM) adapted for the Logical Domains Manager to identify the source of security changes to the system to determine what was done, when it was done, by whom, and what was affected.
- **Compliance** – Determining if a system's configuration is in compliance with a predefined security profile using the Solaris Security Toolkit's auditing feature.

Solaris Security Toolkit and the Logical Domains Manager

Chapter 3 tells you how to install the Solaris Security Toolkit to make it work with the Logical Domains Manager. You would install the Solaris Security Toolkit on the control domain, which is where the Logical Domains Manager runs. You can also install the Solaris Security Toolkit on the other logical domains. The only difference would be that you would use the `ldm_control-secure.driver` to harden the control domain and you would use another driver, such as the `secure.driver`, to harden the other logical domains. This is because the `ldm_control-secure.driver` is specific to the control domain. The `ldm_control-secure.driver` is based on the `secure.driver` and has been customized and tested for use with the Logical Domains Manager. Refer to the *Solaris Security Toolkit 4.2 Reference Manual* for more information about the `secure.driver`.

Hardening

The driver (`ldm_control-secure.driver)` that Solaris Security Toolkit uses to harden the Solaris OS on the control domain is specifically tailored so that the Logical Domains Manager can run with the OS. The `ldm_control-secure.driver` is analogous to the `secure.driver` described in the *Solaris Security Toolkit 4.2 Reference Manual*.

The `ldm_control-secure.driver` provides a baseline configuration for the control domain of a system running the Logical Domains Manager software. It is intended to provide fewer system services than typical for a Solaris OS domain, reserving the control domain for Logical Domains Manager operations, rather than general usage.

The `install-ldm` script installs the Logical Domains Manager software if it is not already installed, and enables the software.

Following is a short summary of the other notable changes from `secure.driver`.

- The Telnet server is disabled from running. You can use Secure Shell (`ssh`) instead. You also can still use the Telnet client to access virtual consoles started by the Logical Domains virtual network terminal server daemon (`vntsd`). For example, if a virtual console is running that is listening to TCP port 5001 on the local system, you can access it as follows.

```
# telnet localhost 5001
```

  See "Enabling the Virtual Network Terminal Server Daemon" on page 44 for instructions on enabling `vntsd`. It is not automatically enabled.

- The following finish scripts have been added. They enable the Logical Domains Manager to install and start. Some of these added scripts must be added to any customized drivers you make and some are optional. The scripts are marked as to whether they are required or optional.
  - `install-ldm.fin` – Installs the `SUNWldm` package. (*Required*)
  - `enable-ldmd.fin` – Enables the Logical Domains Manager daemon (`ldmd`) (*Required*)
  - `enable-ssh-root-login.fin` – Enables the superuser to directly log in through the Secure Shell (`ssh`). (*Optional*)
- The following files have changed. These changes are optional to make in any customized drivers you have and are marked as optional.
  - `/etc/ssh/sshd_config` – Root account access is allowed for the entire network. This file is not used in either driver. (*Optional*)
  - `/etc/ipf/ipf.conf` – UDP port 161 (SNMP) is opened. (*Optional*)
  - `/etc/host.allow` – The Secure Shell daemon (`sshd`) is open for the entire network, not just the local subnet. (*Optional*)
- The following finish scripts are disabled (commented out). You should comment out the `disable-rpc.fin script` in any customized driver you make. The other changes are optional. The scripts are marked as to whether they are required or optional.
  - `enable-ipfilter.fin` – IP Filter, a network packet filter, is not enabled. (*Optional*)
  - `disable-rpc.fin` – Leaves Remote Procedure Call (RPC) service enabled. The RPC service is used by many other system services, such as Network Information Service (NIS) and Network File System (NFS). (*Required*)
  - `disable-sma.fin` – Leaves the System Management Agent (NET-SNMP) enabled. (*Optional*)
  - `disable-ssh-root-login.fin` – `ssh` root login cannot be disabled.
  - `set-term-type.fin` – Unneeded legacy script. (*Optional*)

Authorization

Authorization for the Logical Domains Manager has two levels:

- Read – allows you to view, but not modify the configuration.
- Read and write – allows you to view and change the configuration.

The changes are not made to the Solaris OS, but are added to the authorization file by the package script `postinstall` when the Logical Domains Manager is installed. Similarly, the authorization entries are removed by the package script `preremove`.

The following table lists the `ldm` subcommands with the corresponding user authorization that is needed to perform the commands.

TABLE 2-1 `ldm` Subcommands and User Authorizations

ldm Subcommand[*]	**User Authorization**
`add-*`	`solaris.ldoms.write`
`bind-domain`	`solaris.ldoms.write`
`list`	`solaris.ldoms.read`
`list-*`	`solaris.ldoms.read`
`remove-*`	`solaris.ldoms.write`
`set-*`	`solaris.ldoms.write`
`start-domain`	`solaris.ldoms.write`
`stop-domain`	`solaris.ldoms.write`
`unbind-domain`	`solaris.ldoms.write`

[*] Refers to all the resources you can add, list, remove, or set.

Auditing

Auditing the Logical Domains Manager CLI commands is done with Solaris OS BSM Auditing. Refer to the Solaris 10 *System Administration Guide: Security Services* for detailed information about using Solaris OS BSM Auditing.

BSM Auditing is not enabled by default for the Logical Domains Manager; however, the infrastructure is provided. You can enable BSM Auditing in one of two ways:

- Run the `enable-bsm.fin` finish script in the Solaris Security Toolkit.
- Use the Solaris OS `bsmconv`(1M) command.

For further details about enabling, verifying, disabling, printing output, and rotating logs using BSM Auditing with the Logical Domains Manager, see "Enabling and Using BSM Auditing" on page 67.

Compliance

Solaris Security Toolkit does have its own auditing capabilities. The Solaris Security Toolkit software can automatically validate the security posture of any system running the Solaris OS by comparing it with a predefined security profile. Refer to "Auditing System Security" in the *Solaris Security Toolkit 4.2 Administration Guide* for more information about this compliance function.

CHAPTER 3

Installing and Enabling Software

This chapter describes how to install and enable Logical Domains Manager 1.0 software and other software on a control domain on the following servers:

- Sun Fire T1000 Server
- Sun Fire T2000 Server
- SPARC Enterprise T1000 Server
- SPARC Enterprise T2000 Server
- Netra T2000 Server
- Netra CP3060 Blade
- Sun Blade T6300 Server Module

The following topics are covered in this chapter:

- "Freshly Installing Software on the Control Domain" on page 15
- "Downloading Logical Domains Manager and Solaris Security Toolkit" on page 19
- "Installing Logical Domains Manager and Solaris Security Toolkit" on page 20
- "Enabling the Logical Domains Manager Daemon" on page 32
- "Creating Authorization and Profiles and Assigning Roles for User Accounts" on page 33

Freshly Installing Software on the Control Domain

The first domain that is created when the Logical Domains Manager software is installed is the control domain. That first domain is named `primary`, and you cannot change the name. The following major components are installed on the control domain.

- Solaris 10 11/06 OS. See "To Install the Solaris OS" on page 16.

- System firmware version 6.4.*x* for your server. See "To Upgrade System Firmware" on page 17.
- Logical Domain Manager 1.0 software. See "Installing Logical Domains Manager and Solaris Security Toolkit" on page 20.
- *(Optional)* Solaris Security Toolkit 4.2 software. See "Installing Logical Domains Manager and Solaris Security Toolkit" on page 20.

The Solaris OS and the system firmware must be installed on your server before you install the Logical Domains Manager. After the Solaris OS, the system firmware, and the Logical Domains Manager have been installed, the original domain becomes the control domain.

▼ To Install the Solaris OS

Install the Solaris 10 11/06 OS if it has not already been installed. Refer to the Solaris 10 OS installation guide for complete instructions. You can tailor your installation to the needs of your system.

Note – For logical domains, you can install the Solaris OS only to an entire disk or a file exported as a block device.

---

1. **Install the Solaris 10 11/06 OS, and do the following:**
   a. **Select the** `Entire Distribution` **for a regular installation or** `SUNWCall` **for a JumpStart installation.**
   b. **Select the** `English, C` **locale.**
2. **Install the following Solaris 10 11/06 patches:**
   - 124921-02, which contains updates to the Logical Domains 1.0 drivers and utilities. Logical Domains networking will be broken without this patch.
   - 125043-01, which contains updates to the console (`qcn`) drivers. This patch depends on kernel update (KU) 118833-36, so if this is not already updated on your system, you must install it also.

   You can find the patches at the SunSolve℠ site:

   `http://sunsolve.sun.com`

▼ To Upgrade System Firmware

You can find System Firmware 6.4.*x* at the SunSolve site:

`http://sunsolve.sun.com`

This procedure describes how to upgrade system firmware using the `flashupdate`(1M) command on your system controller. If you do not have access to a local FTP server, see "To Upgrade System Firmware Without an FTP Server" on page 18. If you want to update the system firmware from the control domain, refer to your system firmware release notes. Refer to the administration guides or product notes for the supported servers for more information about installing and updating system firmware for these servers.

1. **Shut down and power off the host server from either management port connected to the system controller: serial or network.**

```
# shutdown -i5 -g0 -y
ok #. (break to SC prompt)
sc> poweroff -yf
```

2. **Use the `flashupdate`(1M) command to upgrade the system firmware, depending on your server.**

```
sc> flashupdate -s IP-address -f
path/Sun_System_Firmware-6_4_x_build_nn-server-name.bin
username: your-userid
password: your-password
```

Where:

- *IP-address* is the IP address of your FTP server.
- *path* is the location in SunSolve or your own directory where you can obtain the system firmware image.
- *x* is the version number of System Firmware 6.4.
- *nn* is the number of the build that applies to this release.
- *server-name* is the name of your server. For example, the *server-name* for the Sun Fire T2000 server is `Sun_Fire_T2000`.

3. **Reset the system controller.**

```
sc> resetsc -y
```

4. **Power on and boot the host server.**

```
sc> poweron
sc> console -f
ok boot disk
```

▼ To Upgrade System Firmware Without an FTP Server

If you do not have access to a local FTP server to upload firmware to the system controller, you can use the `sysfwdownload` utility, which is provided with your system firmware upgrade package on the SunSolve site:

`http://sunsolve.sun.com`

Run the `sysfwdownload` utility from within the Solaris OS.

1. **Run the following commands within the Solaris OS:**

```
# cd firmware_location
# sysfwdownload system_firmware_file
```

2. **Shut down the Solaris OS instance:**

```
# shutdown -i5 -g0 -y
```

3. **Power off and update the firmware on the system controller:**

```
sc> poweroff -fy
sc> flashupdate -s 127.0.0.1
```

4. **Reset and power on the system controller:**

```
sc> resetsc -y
sc> poweron
```

▼ To Downgrade System Firmware

Once you have upgraded the system firmware for use with Logical Domains software, you can downgrade the firmware to the original non–Logical Domains firmware.

- **Run the `flashupdate`(1M) command and specify the path to the original non-Logical Domains firmware.**

Downloading Logical Domains Manager and Solaris Security Toolkit

▼ To Download the Logical Domains Manager and Solaris Security Toolkit

1. **Download the tar file (`LDoms_Manager_1.0.tar.gz`) containing the Logical Domains Manager package (`SUNWldm`), the Solaris Security Toolkit (`SUNWjass`) and installation script (`install-ldm`) from the Sun Software Download site at:**

   `http://www.sun.com/download/products.xml?id=462e6bd6`

2. **Unpack the tar file.**

```
$ gunzip -c LDoms_Manager_1.0.tar.gz | tar xvf -
```

The directory structure for the downloaded software is similar to the following:

CODE EXAMPLE 3-1 Directory Structure for Downloaded Logical Domains 1.0 Software

```
LDoms_Manager_1.0/
        Install/
                install-ldm
        LICENSE
        Patches/
                118833-36.zip
                125043-01.zip
                124921-02.tar.Z
        Product/
```

CODE EXAMPLE 3-1 Directory Structure for Downloaded Logical Domains 1.0 Software *(Continued)*

```
          SUNWldm.v/
          SUNWjass/
     README
```

Installing Logical Domains Manager and Solaris Security Toolkit

There are three methods of installing Logical Domains Manager and Solaris Security Toolkit software:

- Using the installation script to install the packages and patches. This automatically installs both the Logical Domains Manager and the Solaris Security Toolkit software. See "Using the Installation Script to Install the Logical Domains Manager 1.0 and Solaris Security Toolkit 4.2 Software" on page 20.
- Using JumpStart to install the packages. See "Using JumpStart to Install the Logical Domains Manager 1.0 and Solaris Security Toolkit 4.2 Software" on page 26.
- Installing each package manually. See "Installing Logical Domains Manager and Solaris Security Toolkit Software Manually" on page 29.

Using the Installation Script to Install the Logical Domains Manager 1.0 and Solaris Security Toolkit 4.2 Software

If you use the `install-ldm` installation script, you have three choices to specify how you want the script to run. Each choice is described in the procedures that follow.

- **Using the `install-ldm` script with no options – does the following automatically:**
  - Checks that the Solaris OS release is Solaris 10 11/06
  - Verifies that the package subdirectories `SUNWldm/` and `SUNWjass/` are present
  - Verifies that the prerequisite Solaris Logical Domains driver packages, `SUNWldomr` and `SUNWldomu`, are present
  - Verifies that the `SUNWldm` and `SUNWjass` packages have not been installed

---

Note – If the script does detect a previous version of `SUNWjass` during installation, you will need to remove it. You do *not* need to undo any previous hardening of your Solaris OS.

  - Installs the Logical Domains Manager 1.0 software (`SUNWldm` package)
  - Installs the Solaris Security Toolkit 4.2 software including required patches (`SUNWjass` package)
  - Verifies that all packages are installed
  - Enables the Logical Domains Manager daemon, `ldmd`
  - Hardens the Solaris OS on the control domain with the Solaris Security Toolkit `ldm_control-secure.driver` or one of the other drivers ending in `-secure.driver` that you select.
- **Using the** `install-ldm` **script with option** `-d` – allows you to specify a Solaris Security Toolkit driver other than a driver ending with `-secure.driver`. This option automatically performs all the functions listed in the preceding choice with the added option:
  - Hardens the Solaris OS on the control domain with the Solaris Security Toolkit customized driver that you specify; for example, the `server-secure-myname.driver`.
- **Using the** `install-ldm` **script with option** `-d` **and specifying** `none` – specifies that you do *not* want to harden the Solaris OS running on your control domain by using the Solaris Security Toolkit. This option automatically performs all the functions except hardening listed in the preceding choices. Bypassing the use of the Solaris Security Toolkit is not suggested and should only be done when you intend to harden your control domain using an alternate process.

▼ To Install Using the `install-ldm` Script With No Options

1. **Run the installation script with no options.**

   The installation script is part of the `SUNWldm` package and is in the `Install` subdirectory.

```
# Install/install-ldm
```

If the process is successful, you receive messages similar to the following examples.

CODE EXAMPLE 3-2 shows a successful run of the `install-ldm` script if you choose the following default security profile:

```
a) Hardened Solaris configuration for LDoms (recommended)
```

CODE EXAMPLE 3-2 Output From Hardened Solaris Configuration for LDoms

```
# Install/install-ldm
Welcome to the LDoms installer.

You are about to install the domain manager package that will enable
you to create, destroy and control other domains on your system. Given
the capabilities of the domain manager, you can now change the security
configuration of this Solaris instance using the Solaris Security
Toolkit.

Select a security profile from this list:

a) Hardened Solaris configuration for LDoms (recommended)
b) Standard Solaris configuration
c) Your custom-defined Solaris security configuration profile

Enter a, b, or c [a]: a
The changes made by selecting this option can be undone through the
Solaris Security Toolkit’s undo feature. This can be done with the
‘/opt/SUNWjass/bin/jass-execute -u’ command.

Installing LDoms and Solaris Security Toolkit packages.
pkgadd -n -d "/var/tmp/install/Product/Logical_Domain_Manager" -a pkg_admin
SUNWldm.v
Copyright 2006 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Installation of <SUNWldm> was successful.
pkgadd -n -d "/var/tmp/install/Product/Solaris_Security_Toolkit" -a pkg_admin
SUNWjass
Copyright 2005 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Installation of <SUNWjass> was successful.

Verifying that all packages are fully installed.  OK.
Enabling services: svc:/ldoms/ldmd:default
Running Solaris Security Toolkit 4.2.0 driver ldm_control-secure.driver.
Please wait. . .
/opt/SUNWjass/bin/jass-execute -q -d ldm_control-secure.driver
Executing driver, ldm_control-secure.driver
Solaris Security Toolkit hardening executed successfully; log file
/var/opt/SUNWjass/run/20070208142843/jass-install-log.txt.  It will not
take effect until the next reboot.  Before rebooting, make sure SSH or
the serial line is setup for use after the reboot.
```

CODE EXAMPLE 3-3 shows a successful run of the `install-ldm` script if you choose the following security profile:

`c) Your custom-defined Solaris security configuration profile`

The drivers that are displayed for you to choose are drivers ending with `-secure.driver`. If you write a customized driver that does not end with `-secure.driver`, you must specify your customized driver with the `install-ldm -d` option. (See "To Install Using the `install-ldm` Script With the `-d` Option" on page 24.)

CODE EXAMPLE 3-3 Output From Choosing Customized Configuration Profile

```
# Install/install-ldm
Welcome to the LDoms installer.

You are about to install the domain manager package that will enable
you to create, destroy and control other domains on your system. Given
the capabilities of the domain manager, you can now change the security
configuration of this Solaris instance using the Solaris Security
Toolkit.

Select a security profile from this list:

a) Hardened Solaris configuration for LDoms (recommended)
b) Standard Solaris configuration
c) Your custom-defined Solaris security configuration profile

Enter a, b, or c [a]: c
Choose a Solaris Security Toolkit .driver configuration profile from
this list
1) ldm_control-secure.driver
2) secure.driver
3) server-secure.driver
4) suncluster3x-secure.driver
5) sunfire_15k_sc-secure.driver

Enter a number 1 to 5: 2
The driver you selected may not perform all the LDoms-specific
operations specified in the LDoms Administration Guide.
Is this OK (yes/no)? [no] y
The changes made by selecting this option can be undone through the
Solaris Security Toolkit's undo feature. This can be done with the
'/opt/SUNWjass/bin/jass-execute -u' command.

Installing LDoms and Solaris Security Toolkit packages.
pkgadd -n -d "/var/tmp/install/Product/Logical_Domain_Manager" -a pkg_admin
SUNWldm.v
Copyright 2006 Sun Microsystems, Inc.  All rights reserved.
```

CODE EXAMPLE 3-3 Output From Choosing Customized Configuration Profile *(Continued)*

```
Use is subject to license terms.

Installation of <SUNWldm> was successful.
pkgadd -n -d "/var/tmp/install/Product/Solaris_Security_Toolkit" -a pkg_admin
SUNWjass
Copyright 2005 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Installation of <SUNWjass> was successful.

Verifying that all packages are fully installed.  OK.
Enabling services: svc:/ldoms/ldmd:default
Running Solaris Security Toolkit 4.2.0 driver secure.driver.
Please wait. . .
/opt/SUNWjass/bin/jass-execute -q -d secure.driver
Executing driver, secure.driver
Solaris Security Toolkit hardening executed successfully; log file
/var/opt/SUNWjass/run/20070102142843/jass-install-log.txt.  It will not
take effect until the next reboot.  Before rebooting, make sure SSH or
the serial line is setup for use after the reboot.
```

2. **Continue with** “Enabling the Logical Domains Manager Daemon” on page 32**.**

▼ To Install Using the `install-ldm` Script With the `-d` Option

1. **Run the installation script with the `-d` option to specify a Solaris Security Toolkit customized hardening driver; for example,** `server-secure-myname.driver`**.**

   The installation script is part of the `SUNWldm` package and is in the `Install` subdirectory.

   ```
   # Install/install-ldm -d server-secure-myname.driver
   ```

   If the process is successful, you receive messages similar to the following:

CODE EXAMPLE 3-4 Output From Successful Run of the `install-ldm -d` Script

```
# Install/install-ldm -d server-secure.driver
The driver you selected may not perform all the LDoms-specific
operations specified in the LDoms Administration Guide.
Installing LDoms and Solaris Security Toolkit packages.
```

CODE EXAMPLE 3-4 Output From Successful Run of the `install-ldm -d` Script *(Continued)*

```
pkgadd -n -d "/var/tmp/install/Product/Logical_Domain_Manager" -a pkg_admin
SUNWldm.v
Copyright 2006 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Installation of <SUNWldm> was successful.
pkgadd -n -d "/var/tmp/install/Product/Solaris_Security_Toolkit" -a pkg_admin
SUNWjass
Copyright 2005 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Installation of <SUNWjass> was successful.

Verifying that all packages are fully installed.  OK.
Enabling services: svc:/ldoms/ldmd:default
Running Solaris Security Toolkit 4.2.0 driver server-secure-myname.driver.
Please wait. . .
/opt/SUNWjass/bin/jass-execute -q -d server-secure-myname.driver
Executing driver, server-secure-myname.driver
Solaris Security Toolkit hardening executed successfully; log file
/var/opt/SUNWjass/run/20061114143128/jass-install-log.txt.  It will not
take effect until the next reboot.  Before rebooting, make sure SSH or
the serial line is setup for use after the reboot.
```

2. **Continue with "Enabling the Logical Domains Manager Daemon" on page 32.**

▼ To Install Using the `install-ldm` Script With the `-d none` Option

1. **Run the installation script with the `-d none` option to specify *not* to harden your system using a Solaris Security Toolkit driver.**

   The installation script is part of the `SUNWldm` package and is in the `Install` subdirectory.

   ```
   # Install/install-ldm -d none
   ```

   If the process is successful, you receive messages similar to the following:

CODE EXAMPLE 3-5 Output From Successful Run of the `install-ldm -d none` Script

```
# Install/install-ldm -d none
Installing LDoms and Solaris Security Toolkit packages.
pkgadd -n -d "/var/tmp/install/Product/Logical_Domain_Manager" -a pkg_admin
SUNWldm.v
Copyright 2006 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Installation of <SUNWldm> was successful.
pkgadd -n -d "/var/tmp/install/Product/Solaris_Security_Toolkit" -a pkg_admin
SUNWjass
Copyright 2005 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Installation of <SUNWjass> was successful.

Verifying that all packages are fully installed.  OK.
Enabling services: svc:/ldoms/ldmd:default
Solaris Security Toolkit was not applied. Bypassing the use of the
Solaris Security Toolkit is not recommended and should only be
performed when alternative hardening steps are to be taken.
```

2. **Continue with "Enabling the Logical Domains Manager Daemon" on page 32.**

Using JumpStart to Install the Logical Domains Manager 1.0 and Solaris Security Toolkit 4.2 Software

Refer to *JumpStart Technology: Effective Use in the Solaris Operating Environment* for complete information about using JumpStart.

▼ To Set Up a JumpStart Server

- If you have already set up a JumpStart server, proceed to "To Install Using JumpStart Software" on page 27 of this administration guide.
- If you have not already set up a JumpStart server, you must do so.

  Refer to the *Solaris 10 11/06 Installation Guide: Custom JumpStart and Advanced Installation* for complete information about this procedure. You can find this installation guide at:

  `http://docs.sun.com/app/doc/819-6397/`

1. **Refer to Chapter 3 "Preparing Custom JumpStart Installations (Tasks)" in the** *Solaris 10 11/06 Installation Guide: Custom JumpStart and Advanced Installation*, **and perform the following steps.**
   1. **Read the task map in "Task Map: Preparing Custom JumpStart Installations."**
   2. **Set up networked systems with the procedures in "Creating a Profile Server for Network Systems."**
   3. **Create the `rules` file with the procedure in "Creating the `rules` File."**
2. **Validate the `rules` file with the procedure in "Validating the `rules` File."**

   The Solaris Security Toolkit provides profiles and finish scripts. Refer to the *Solaris Security Toolkit 4.2 Reference Manual* for more information about profiles and finish scripts.

▼ To Install Using JumpStart Software

1. **Change to the directory where you have downloaded the Solaris Security Toolkit package (`SUNWjass`).**

```
# cd /path/to/download
```

2. **Install `SUNWjass` so that it creates the JumpStart (`jumpstart`) directory structure.**

```
# pkgadd -R /jumpstart -d . SUNWjass.v
```

3. **Use your text editor to modify the `/jumpstart/opt/SUNWjass/Sysidcfg/Solaris_10/sysidcfg` file to reflect your network environment.**
4. **Copy the `/jumpstart/opt/SUNWjass/Drivers/user.init.SAMPLE` to `/jumpstart/opt/SUNWjass/Drivers/user.init`**

```
# cp user.init.SAMPLE user.init
```

5. **Edit the `user.init` file to reflect your paths.**
6. **To install the Solaris Security Toolkit package (`SUNWjass`) onto the target system during a JumpStart install, you must place the package in the `JASS_PACKAGE_MOUNT` directory defined in your `user.init` file. For example:**

```
# cp -r /jumpstart/opt/SUNWjass /jumpstart/opt/SUNWjass/Packages
```

7. **To install the Logical Domains Manager package (`SUNWldm`) onto the target system during a JumpStart install, you must place the package from the download area in the `JASS_PACKAGE_MOUNT` directory defined in your `user.init` file. For example:**

```
# cp -r /path/to/SUNWldm /jumpstart/opt/SUNWjass/Packages
```

8. **If you experience problems with a multihomed JumpStart server, modify the two entries in the `user.init` file for `JASS_PACKAGE_MOUNT` and `JASS_PATCH_MOUNT` to the correct path to the `JASS_HOME_DIR/Patches` and `JASS_HOME_DIR/Packages` directories. Refer to the comments in the `user.init.SAMPLE` file for more information.**

9. **Use the `ldm_control-secure.driver` as the basic driver for the Logical Domains Manager control domain.**

   Refer to Chapter 4 in the *Solaris Security Toolkit 4.2 Reference Manual* for information about how to modify the driver for your use. The main driver in the Solaris Security Toolkit that is the counterpart to the `ldm_control-secure.driver` is the `secure.driver`.

10. **After completing the modifications to the `ldm_control-secure.driver`, make the correct entry in the `rules` file.**

    The entry should be similar to the following:

```
hostname imbulu - Profiles/oem.profile Drivers/ldm_control-secure-abc.driver
```

Note – If you undo hardening during a JumpStart install, you must run the following SMF commands to restart the Logical Domains Manager.

```
# svcadm enable svc:/ldoms/ldmd:default
```

Installing Logical Domains Manager and Solaris Security Toolkit Software Manually

Perform the following procedures to install the Logical Domains Manager and Solaris Security Toolkit Software manually:

- "To Download and Install the Logical Domains Manager (LDoms) 1.0 Software Manually" on page 29.
- "(Optional) To Download and Install the Solaris Security Toolkit 4.2 Software Manually" on page 29.
- "(Optional) To Harden the Control Domain Manually" on page 30.

▼ To Download and Install the Logical Domains Manager (LDoms) 1.0 Software Manually

1. **Download the Logical Domains Manager 1.0 software, the `SUNWldm` package, from the Sun Software Download site at:**

   http://www.sun.com/download/products.xml?id=45b14cf9

2. **Use the `pkgadd`(1M) command to install the `SUNWldm` package.**

```
# pkgadd -d . SUNWldm.v
```

3. **Answer `y` for yes to all questions in the interactive prompts.**

4. **Use the `pkginfo`(1) command to verify that the `SUNWldm` package for Logical Domains Manager 1.0 software is installed.**

   The revision (`REV`) information shown below is an example.

```
# pkginfo -l SUNWldm | grep VERSION
VERSION: 1.00,REV=2007.02.26.10.05
```

▼ (*Optional*) To Download and Install the Solaris Security Toolkit 4.2 Software Manually

Download and install the Solaris Security Toolkit software if you want to secure your system.

See Chapter 2 for information about security considerations when using Logical Domains Manager software. For further reference, you can find Solaris Security Toolkit 4.2 documentation at:

http://www.sun.com/products-n-solutions/hardware/docs/Software/enterprise_computing/systems_management/sst/index.html

If you want to harden your system, download and install the `SUNWjass` package. The required patches (122608-03 and 125672-01) are included in the `SUNWjass` package.

1. **Download the Solaris Security Toolkit 4.2 software, the `SUNWjass` package, from the Sun Software Download site at:**

   `http://www.sun.com/download/products.xml?id=45b14cf9`

2. **Use the `pkgadd`(1M) command to install the `SUNWjass` package.**

```
# pkgadd -d . SUNWjass
```

3. **Use the `pkginfo`(1) command to verify that the `SUNWjass` package for Solaris Security Toolkit 4.2 software is installed.**

```
# pkginfo -l SUNWjass | grep VERSION
VERSION: 4.2.0
```

▼ (*Optional*) To Harden the Control Domain Manually

Perform this procedure only if you have installed the Solaris Security Toolkit 4.2 package.

Note – When you use the Solaris Security Toolkit to harden the control domain, you disable many system services and place certain restrictions on network access. Refer to "Related Documentation" on page xviii in this book to find Solaris Security Toolkit 4.2 documentation for more information.

---

1. **Harden using the `ldm_control-secure.driver`.**

   You can use another driver to harden your system. You can also customize drivers to tune the security of your environment. Refer to the *Solaris Security Toolkit 4.2 Reference Manual* for more information about drivers and customizing them.

```
# /opt/SUNWjass/bin/jass-execute -d ldm_control-secure.driver
```

2. **Answer `y` for yes to all questions in the interactive prompts.**

3. **Use the `shutdown`(1M) command to shut down your server to reboot for the hardening to take place.**

```
# /usr/sbin/shutdown -y -g0 -i6
```

▼ To Validate Hardening

- **Check whether the Logical Domains hardening driver (`ldom_control-secure.driver`) applied hardening correctly.**

  If you want to check on another driver, substitute that driver's name in this command example.

```
# /opt/SUNWjass/bin/jass-execute -a ldom_control-secure.driver
```

▼ To Undo Hardening

1. **Undo the configuration changes applied by the Solaris Security Toolkit.**

```
# /opt/SUNWjass/bin/jass-execute -u
```

   The Solaris Security Toolkit asks you which hardening runs you want to undo.

2. **Select the hardening runs you want to undo.**

3. **Reboot the system so that the unhardened configuration takes place.**

```
# /usr/sbin/shutdown -y -g0 -i6
```

Note – If you undo hardening that was performed during a JumpStart installation, you must run the following SMF commands to restart the Logical Domains Manager and the Virtual Network Terminal Server Daemon.

```
# svcadm enable svc:/ldoms/ldmd:default
```

Enabling the Logical Domains Manager Daemon

The installation script `install-ldm` automatically enables the Logical Domains Manager Daemon (`ldmd`). If you have installed the Logical Domains Manager software manually, you must enable the Logical Domains Manager daemon, `ldmd`, which allows you to create, modify, and control the logical domains.

▼ To Enable the Logical Domains Manager Daemon

1. **Use the `svcadm`(1M) command to enable the Logical Domains Manager daemon, `ldmd`:**

```
# svcadm enable ldmd
```

2. **Use the `ldm list` command to verify that the Logical Domains Manager is running.**

   You receive a message similar to the following, which is for the `factory-default` configuration. Note that the `primary` domain is `active`, which means that the Logical Domains Manager is running.

```
# cd /opt/SUNWldm/bin
# /opt/SUNWldm/bin/ldm list
Name        State       Flags   Cons   VCPU   Memory   Util   Uptime
primary     active      ---c-   SP     32     3264M    0.3%   19d 9m
```

Creating Authorization and Profiles and Assigning Roles for User Accounts

You set up authorization and profiles and assign roles for user accounts using the Solaris OS Role-Based Access Control (RBAC) adapted for the Logical Domains Manager. Refer to the Solaris 10 System Administration Collection for more information about RBAC.

Authorization for the Logical Domains Manager has two levels:

- Read – allows you to view, but not modify the configuration.
- Read and write – allows you to view and change the configuration.

Following are the Logical Domains entries automatically added to the Solaris OS `/etc/security/auth_attr` file:

- `solaris.ldoms.:::LDom administration::`
- `solaris.ldoms.grant:::Delegate LDom configuration::`
- `solaris.ldoms.read:::View LDom configuration::`
- `solaris.ldoms.write:::Manage LDom configuration::`

Managing User Authorizations

▼ To Add an Authorization for a User

Use the following steps as necessary to add authorizations in the `/etc/security/auth_attr` file for Logical Domains Manager users. Because the superuser already has `solaris.*` authorization, the superuser already has permission for `solaris.ldoms.*` authorizations.

1. **Create a local user account for each user who needs authorization to use the `ldm`(1M) subcommands.**

Note – To add Logical Domains Manager authorization for a user, a local (non-LDAP) account must be created for that user. Refer to the Solaris 10 System Administrator Collection for details.

---

2. **Do one of the following depending on which `ldm`(1M) subcommands you want the user to be able to access.**

   See TABLE 2-1 for a list of `ldm`(1M) commands and their user authorizations.

- **Add a read-only authorization for a user using the** `usermod`**(1M) command.**

```
# usermod -A solaris.ldoms.read username
```

- **Add a read and write authorization for a user using the** `usermod`**(1M) command.**

```
# usermod -A solaris.ldoms.write username
```

▼ To Delete All Authorizations for a User

- **Delete all authorizations for a local user account (the only possible option).**

```
# usermod -A `` username
```

Managing User Profiles

The `SUNWldm` package adds two system-defined RBAC profiles in the `/etc/security/prof_attr` file for use in authorizing access to the Logical Domains Manager by non-superusers. The two LDoms-specific profiles are:

- `LDoms Review:::Review LDoms configuration:auths=solaris.ldoms.read`
- `LDoms Management:::Manage LDoms domains:auths=solaris.ldoms.*`

One of the preceding profiles can be assigned to a user account using the following procedure.

▼ To Add a Profile for a User

- **Add an administrative profile for a local user account; for example, LDoms Management.**

```
# usermod -P “LDoms Management” username
```

▼ To Delete All Profiles for a User

- **Delete all profiles for a local user account (the only possible option).**

```
# usermod -P `` username
```

Assigning Roles to Users

The advantage of using this procedure is that only a user who has been assigned a specific role can assume the role. In assuming a role, a password is required if the role is given a password. This provide two layers of security. If a user has not been assigned a role, then the user cannot assume the role (by doing the `su` *role_name* command) even if the user has the correct password.

▼ To Create a Role and Assign the Role to a User

1. **Create a role.**

```
# roleadd -A solaris.ldoms.read ldm_read
```

2. **Assign a password to the role.**

```
# passwd ldm_read
```

3. **Assign the role to a user; for example,** `user_1`**.**

```
# useradd -R ldm_read user_1
```

4. **Assign a password to the user (**`user_1`**).**

```
# passwd user_1
```

5. **Provide access to the user for** `ldm` **subcommands that have read authorization.**

```
# su ldm_read
```

6. **Type the user password if or when prompted.**

7. **Type the** `id` **command to show the user:**

```
# id
uid=nn(ldm_read) gid=nn(group_name)
```

CHAPTER 4

Setting Up Services and Logical Domains

This chapter describes the procedures necessary to set up default services, your control domain, and guest domains:

Creating Default Services

You must create the following virtual default services initially to be able to use them later:

- `vdiskserver` – virtual disk server
- `vswitch` – virtual switch service
- `vconscon` – virtual console concentrator service

▼ To Create Default Services

1. **Create a virtual disk server (`vds`) to allow importing virtual disks into a logical domain.**

   For example, the following command adds a virtual disk server (`primary-vds0`) to the control domain (`primary`).

   ```
   $ ldm add-vds primary-vds0 primary
   Notice: the LDom Manager is running in configuration mode. Any
   configuration changes made will only take effect after the machine
   configuration is downloaded to the system controller and the host
   is reset.
   ```

2. **Create a virtual console concentrator service (`vcc`) for use by the virtual network terminal server daemon (`vntsd`) and as a concentrator for all logical domain consoles.**

   For example, the following command would add a virtual console concentrator service (`primary-vcc0`) with a port range from 5000 to 5100 to the control domain (`primary`).

   ```
   $ ldm add-vcc port-range=5000-5100 primary-vcc0 primary
   Notice: the LDom Manager is running in configuration mode. Any
   configuration changes made will only take effect after the machine
   configuration is downloaded to the system controller and the host
   is reset.
   ```

3. **Create a virtual switch service (`vsw`) to enable networking between virtual network (`vnet`) devices in logical domains. Assign a GLDv3-compliant network adapter to the virtual switch if each of the logical domains needs to communicate outside the box through the virtual switch.**

   For example, the following command would add a virtual switch service (`primary-vsw0`) on network adapter driver `e1000g0` to the control domain (`primary`).

```
$ ldm add-vsw net-dev=e1000g0 primary-vsw0 primary
Notice: the LDom Manager is running in configuration mode. Any
configuration changes made will only take effect after the machine
configuration is downloaded to the system controller and the host
is reset.
```

   This command automatically allocates a MAC address to the virtual switch. If you want, you can specify your own MAC address as an option to the `ldm add-vsw` command. However, in that case, it is your responsibility to ensure that the MAC address specified does not conflict with an already existing MAC address.

   If the virtual switch being added replaces the underlying physical adapter as the primary network interface, it must be assigned the MAC address of the physical adapter, so that the Dynamic Host Configuration Protocol (DHCP) server assigns the domain the same IP address. See "Enabling Networking Between the Control/Service Domain and Other Domains" on page 42.

```
$ ldm add-vsw mac-addr=2:04:4f:fb:9f:0d net-dev=e1000g0 primary-vsw0 primary
Notice: the LDom Manager is running in configuration mode. Any configuration
changes made will only take effect after the machine configuration is downloaded
to the system controller and the host is reset.
```

4. **Verify the services have been created by using the `list-services` subcommand. Your output should look similar to this:**

```
$ ldm list-services primary
...
Vds:    primary-vds0
                 vdsdev: vol0 device=/export/home/solaris_10.disk
Vcc:    primary-vcc0
                 port-range=5000-5100
Vsw:    primary-vsw0
                 mac-addr=2:04:4f:fb:9f:0d
                 net-dev=e1000g0
                 mode=prog,promisc
...
```

Initial Configuration of the Control Domain

Initially, all system resources are allocated to the control domain. To allow the creation of other logical domains, you must release some of these resources.

▼ To Set Up the Control Domain

Note – The steps in this procedure contains examples of numbers of resources to set for your control domain. These numbers are examples only, and the values used may not be appropriate for your control domain.

---

1. **Assign cryptographic resources to the control domain.**

Note – If you have any cryptographic devices in the control domain, you cannot dynamically reconfigure CPUs. So if you are not using cryptographic devices, `set-mau` to `0`.

The following example would assign one cryptographic resource to the control domain, `primary`. This leaves the remainder of the cryptographic resources available to a guest domain.

```
$ ldm set-mau 1 primary
Notice: the LDom Manager is running in configuration mode. Any
configuration changes made will only take effect after the machine
configuration is downloaded to the system controller and the host
is reset.
```

2. **Assign virtual CPUs to the control domain.**

   For example, the following command would assign four virtual CPUs to the control domain, `primary`. This leaves the remainder of the virtual CPUs available to a guest domain.

   ```
   $ ldm set-vcpu 4 primary
   Notice: the LDom Manager is running in configuration mode. Any
   configuration changes made will only take effect after the machine
   configuration is downloaded to the system controller and the host
   is reset.
   ```

3. **Assign memory to the control domain.**

   For example, the following command would assign 1 gigabytes of memory to `the` control domain, `primary`. This leaves the remainder of the memory available to a guest domain.

   ```
   $ ldm set-memory 4G primary
   Notice: the LDom Manager is running in configuration mode. Any
   configuration changes made will only take effect after the machine
   configuration is downloaded to the system controller and the host
   is reset.
   ```

4. **Add a logical domain machine configuration to the system controller (SC).**

   For example, the following command would add a configuration called `initial`.

   ```
   $ ldm add-config initial
   ```

5. **Verify that the configuration is ready to be used at the next reboot.**

   ```
   $ ldm list-config
   factory-default [current]
   initial [next]
   ```

   This list subcommand shows that the `factory-default` configuration set is currently being used and the `initial` configuration set will be used once you reboot.

Rebooting to Use Logical Domains

You must reboot the control/service for the preceding changes to take effect and the resources to be released for other logical domains to use.

▼ To Reboot to Use Logical Domains

- **Shut down and reboot the** `primary` **domain, which is also the service domain in our examples.**

```
primary# shutdown -y -g0 -i6
```

Enabling Networking Between the Control/Service Domain and Other Domains

By default, networking between the control/service domain and other domains in the system is disabled. To enable this, the virtual switch device should be configured as a network device. The virtual switch can either replace the underlying physical device (`e1000g0` in this example) as the primary interface or be configured as an additional network interface in the domain.

Note – Perform the following configuration steps from the domain's console, as the procedure could temporarily disrupt network connectivity to the domain.

▼ To Configure the Virtual Switch as the Primary Interface

1. **Print out the addressing information for all interfaces.**

```
primary# ifconfig -a
```

2. **Plumb the virtual switch. In this example, `vsw0` is the virtual switch being configured.**

```
primary# ifconfig vsw0 plumb
```

3. **(*Optional*) To obtain the list of all virtual switch instances in a domain, you can list them by doing the following:**

```
primary# /usr/sbin/dladm show-link | grep vsw
vsw0            type: non-vlan  mtu: 1500       device: vsw0
```

4. **Unplumb the physical network device assigned to the virtual switch (`net-dev`), which is `e1000g0` in this example.**

```
primary# ifconfig e1000g0 down unplumb
```

5. **To migrate properties of the physical network device (`e1000g0`) to the virtual switch (`vsw0`) device, do one of the following:**
   - If networking is configured using a static IP address, reuse the IP address and netmask of `e1000g0` for `vsw0`.

```
primary# ifconfig vsw0 IP_of_e1000g0 netmask netmask_of_e1000g0 broadcast + up
```

   - If networking is configured using DHCP, enable DHCP for `vsw0`.

```
primary# ifconfig vsw0 dhcp start
```

6. **Make the required configuration file modifications to make this change permanent.**

```
primary# mv /etc/hostname.e1000g0 /etc/hostname.vsw0
primary# mv /etc/dhcp.e1000g0 /etc/dhcp.vsw0
```

Note – If necessary, you can also configure the virtual switch as well as the physical network device. In this case, plumb the virtual switch as in Step 2, and do not unplumb the physical device (skip Step 4). The virtual switch must then be configured with either a static IP address or obtain a dynamic IP address from a DHCP server.

Enabling the Virtual Network Terminal Server Daemon

You must enable the virtual network terminal server daemon (`vntsd`) to provide access to the virtual console of each logical domain. Refer to the Solaris 10 OS Reference Manual collection or the `vntsd`(1M) man page for information about how to use this daemon.

▼ To Enable the Virtual Network Terminal Server Daemon

Note – Be sure you have created the default service `vconscon` on the control domain before you enable `vntsd`. See "Creating Default Services" on page 37 for more information.

---

1. **Use the `svcadm`(1M) command to enable the virtual network terminal server daemon, `vntsd`(1M).**

```
# svcadm enable vntsd
```

2. **Use the `svcs`(1) command to verify that the `vntsd` is enabled.**

```
# svcs -l vntsd
fmri         svc:/ldoms/vntsd:default
name         virtual network terminal server
enabled      true
state        online
next_state   none
state_time   Sat Jan 27 03:14:18 2007
logfile      /var/svc/log/ldoms-vntsd:default.log
restarter    svc:/system/svc/restarter:default
contract_id  93
dependency   optional_all/error svc:/milestone/network (online)
dependency   optional_all/none svc:/system/system-log (online)
```

Creating and Starting a Guest Domain

The guest domain must run an operating system that understands both the sun4v platform and the virtual devices presented by the hypervisor. Currently, this is the Solaris 10 11/06 OS with required patches 124921-02 and 125043-01 (with KU 118833-36). Once you have created default services and reallocated resources from the control domain, you can create and start a guest domain.

▼ To Create and Start a Guest Domain

1. **Create a logical domain.**

   For example, the following command would create a guest domain named `ldg1`.

```
$ ldm add-domain ldg1
```

2. **Add CPUs to the guest domain.**

   For example, the following command would add four virtual CPUs to guest domain `ldg1`.

```
$ ldm add-vcpu 4 ldg1
```

3. **Add memory to the guest domain.**

   For example, the following command would add 512 megabytes of memory to guest domain `ldg1`.

   ```
   $ ldm add-memory 512m ldg1
   ```

4. **Add a virtual network device to the guest domain.**

   For example, the following command would add a virtual network device with these specifics to the guest domain `ldg1`:

   - `vnet1` is a unique interface name to the logical domain, assigned to this virtual network device instance for reference on subsequent `set-vnet` or `remove-vnet` subcommands.
   - `primary-vsw0` is the name of an existing network service (virtual switch) to which to connect.

   ```
   $ ldm add-vnet vnet1 primary-vsw0 ldg1
   ```

5. **Specify the device to be exported by the virtual disk server as a virtual disk to the guest domain.**

   You can export a physical disk, disk slice, volumes, or file as a block device. Exporting loopback (`lofi`) devices as block devices is not supported in this release of Logical Domains software. The following examples show a physical disk and a file.

   - **Physical Disk Example.** The first example adds a physical disk with these specifics.

   ```
   $ ldm add-vdsdev /dev/dsk/c0t0d0s2 vol1@primary-vds0
   ```

     Where:

     - `/dev/dsk/c0t0d0s2` is the path name of the actual physical device. When adding a device, the path name must be paired with the device name.
     - `vol1` is a unique name you must specify for the device being added to the virtual disk server. The device name must be unique to this virtual disk server instance, because this name is exported by this virtual disk server to the clients for adding. When adding a device, the device name must be paired with the path name of the actual device.
     - `primary-vds0` is the name of the virtual disk server to which to add this device.

- **File Example.** This second example is exporting a file as a block device.

```
$ ldm add-vdsdev path_to_file/filename vol1@primary-vds0
```

  Where:

  - *path_to_file/filename* is the path name of the actual file exported as a block device. When adding a device, the path name must be paired with the device name.
  - `vol1` is a unique name you must specify for the device being added to the virtual disk server. The device name must be unique to this virtual disk server instance, because this name is exported by this virtual disk server to the clients for adding. When adding a device, the device name must be paired with the path name of the actual device.
  - `primary-vds0` is the name of the virtual disk server to which to add this device.

6. **Add a virtual disk to the guest domain.**

   The following example adds a virtual disk to the guest domain `ldg1`.

   Where:

   - `vdisk1` is the name of the virtual disk.
   - `vol1` is the name of the existing virtual disk server device to which to connect.
   - `primary-vds0` is the name of the existing virtual disk server to which to connect.

```
$ ldm add-vdisk vdisk1 vol1@primary-vds0 ldg1
```

Note – The virtual disks are generic block devices that are backed by different types of physical devices, volumes, or files. A virtual disk is not synonymous with a SCSI disk and, therefore, excludes the target ID in the disk label. Virtual disks in a logical domain have the following format: `c`*N*`d`*N*`s`*N*, where `c`*N* is the virtual controller, `d`*N* is the virtual disk number, and `s`*N* is the slice.

---

7. **Set `auto-boot` and `boot-device` variables for the guest domain.**

   The first example command sets `auto-boot\?` to `true` for guest domain `ldg1`.

```
$ ldm set-var auto-boot\?=true ldg1
```

   The second example command sets `boot-device` to `vdisk` for the guest domain `ldg1`.

```
$ ldm set-var boot-device=vdisk ldg1
```

8. **Bind resources to the guest domain `ldg1` and then list the domain to verify that it is bound.**

```
$ ldm bind-domain ldg1
$ ldm list-domain ldg1
Name         State    Flags   Cons    VCPU  Memory  Util  Uptime
ldg1         bound    -----   5001    4     512m
```

9. **To find the console port of the guest domain, you can look at the output of the preceding `list-domain` subcommand.**

   You can see under the heading `Cons` that logical domain guest 1 (`ldg1`) has its console output bound to port `5001`.

10. **Start the guest domain `ldg1`.**

```
$ ldm start-domain ldg1
```

11. **Connect to the console of a guest domain. There are several ways you can do this.**

    - For example, you can connect directly to the console port on the local host:

```
$ ssh admin@controldom.domain
$ telnet localhost 5001
```

    - You can also connect to a guest console over a network if it is enabled in the `vntsd`(1M) SMF manifest. For example:

```
$ telnet host-name 5001
```

A Service Management Facility manifest is an XML file that describes a service. For more information about creating an SMF manifest, refer to the Solaris 10 Collection.

JumpStarting a Guest Domain

If you are JumpStarting a guest domain, you would use a normal JumpStart procedure with the following profile syntax changes from a regular Solaris OS JumpStart to a JumpStart specific to LDoms as shown in the following two examples.

Normal JumpStart Profile

```
filesys c1t1d0s0 free /
filesys c1t1d0s1 2048 swap
filesys c1t1d0s5 120 /spare1
filesys c1t1d0s6 120 /spare2
```

Virtual disk device names in a logical domain differ from physical disk device names in that they do not contain a target ID (`t`*N*) in the device name. Instead of the normal `c`*N*`t`*N*`d`*N*`s`*N* format, virtual disk device names are of the format `c`*N*`d`*N*`s`*N*, where `c`*N* is the virtual controller, `d`*N* is the virtual disk number, and `s`*N* is the slice. Modify your JumpStart profile to reflect this change as in the following profile example.

Actual Profile Used for a Logical Domain

```
filesys c0d0s0 free /
filesys c0d0s1 2048 swap
filesys c0d0s5 120 /spare1
filesys c0d0s6 120 /spare2
```

CHAPTER 5

Other Information and Tasks

This chapter contains the following information and tasks that you need to know about in using the Logical Domains Manager:

Stopping, Unbinding, and Deleting a Guest Logical Domain

This section describes how to stop, unbind, and remove a guest domain.

▼ To Stop, Unbind, and Remove a Guest Domain

1. **Stop the guest domain `ldg1` by using this command.**

```
$ ldm stop-domain ldg1
```

Note – The `stop-domain` subcommand sends a `shutdown`(1M) request to the logical domain if the Solaris OS is booted. If the domain cannot be stopped by any other means, use the `-f` option of the `stop-domain` subcommand to force the domain to stop.

---

2. **Release all the resources attached to (unbind) the guest domain `ldg1`.**

```
$ ldm unbind-domain ldg1
```

3. **Remove the guest domain `ldg1`.**

```
$ ldm remove-domain ldg1
```

Assigning MAC Addresses

In the Logical Domains Manager, you can manually assign MAC addresses to the virtual network (`vnet`) and the virtual switch (`vswitch`), or you can have the Logical Domains Manager automatically assign the MAC addresses.

The advantage to having the Logical Domains Manager assign the MAC addresses is that it utilizes the block of MAC addresses dedicated for use with logical domains. Also, the Logical Domains Manager detects and prevents MAC address collisions with other Logical Domains Manager instances on the same subnet. This means that the odds of having a MAC address collision with automatic allocation is quite small.

MAC address assignment for virtual network devices happens as soon as the virtual device (`vnet` or `vswitch`) is configured into a domain. In addition, the assignment is persistent until the device, or the logical domain itself, is removed.

CPU and Memory Address Mapping

The Solaris Fault Management Architecture (FMA) reports CPU errors in terms of physical CPU numbers and memory errors in terms of physical memory addresses.

If you want to determine within which logical domain an error occurred and the corresponding virtual CPU number or real memory address within the domain, then you must perform a mapping.

CPU Mapping

The domain and the virtual CPU number within the domain, which correspond to a given physical CPU number, can be determined as follows.

▼ To Determine the CPU Number

1. **Generate a long list for all domains.**

```
# ldm ls -l
```

2. **Look for the entry in the list's Vcpu: sections that has a pid field equal to the physical CPU number.**

   a. **If you find such an entry, the CPU is in the domain the entry is listed under, and the virtual CPU number within the domain is given by the entry's vid field.**

   b. **If you do not find such an entry, the CPU is not in any domain.**

Memory Mapping

The domain and the real memory address within the domain, which correspond to a given physical memory address (PA), can be determined as follows.

▼ To Determine the Real Memory Address

1. **Generate a long list for all domains.**

```
# ldm ls -l
```

2. **Look for the line in the list's Memory: sections where the PA falls within the inclusive range phys-addr to (phys-addr + size - 1): that is, phys-addr <= PA < (phys-addr + size - 1).**

   Here phys-addr and size refer to the values in the corresponding fields of the line.

   a. **If you find such an entry, the PA is in the domain the entry is listed under and the corresponding real address within the domain is given by real-addr + (PA - phys-addr).**

   b. **If you do not find such an entry, the PA is not in any domain.**

Example

Suppose you have a logical domain configuration as shown in CODE EXAMPLE 5-1, and you want to determine the domain and the virtual CPU corresponding to physical CPU number `5`, and the domain and the real address corresponding to physical address `0x7e816000`.

Looking through the `Vcpu:` entries in the list for the one with the `pid` field equal to `5`, you can find the following entry under logical domain `ldg1`:

```
vid     pid     util  strand
1       5        29%    100%
```

Hence, the physical CPU number `5` is in domain `ldg1` and within the domain it has virtual CPU number `1`.

Looking through the `Memory:` entries in the list, you can find the following entry under domain `ldg2`:

```
real-addr          phys-addr          size
0x8000000          0x78000000         1G
```

Where 0x78000000 <= 0x7e816000 < (0x78000000 + 0x40000000 - 1), that is, phys-addr < PA < (phys-addr + size - 1).

Hence, the PA is in domain `ldg2` and the corresponding real address is 0x8000000 + (0x7e816000 - 0x78000000) = 0xe816000.

CODE EXAMPLE 5-1 Long List of Logical Domain Configuration

```
# ldm ls -l
Name:   primary
State:  active
Flags:  normal,control,vio service
OS:     Solaris running
Util:   0.8%
Uptime: 2d 21h 9m
Vcpu:   4
        vid    pid    util strand
        0      0      1.2%   100%
        1      1      0.7%   100%
        2      2      0.7%   100%
        3      3      0.6%   100%
Mau:    1
        mau cpuset (0, 1, 2, 3)
Memory: 1G
        real-addr        phys-addr        size
        0x8000000        0x8000000        1G
Vars:   reboot-command=boot
IO:     pci@780 (bus_a)
        pci@7c0 (bus_b)
Vldc:   primary-vldc0   [num_clients=5]
Vldc:   primary-vldc3   [num_clients=7]
Vds:    primary-vds0    [num_clients=2]
                vdsdev: disk-ldg1        device=/opt/ldoms/testdisk.1
                vdsdev: disk-ldg2        device=/opt/ldoms/testdisk.2
Vcc:    primary-vcc0    [num_clients=2]
                port-range=5000-5100
Vsw:    primary-vsw0    [num_clients=2]
                mac-addr=0:14:4f:f8:7:26
```

CODE EXAMPLE 5-1 Long List of Logical Domain Configuration *(Continued)*

```
                 net-dev=e1000g0
                 mode=prog,promisc
Vcons:   SP
----------------------------------------------------------------------------
Name:    ldg1
State:   active
Flags:   normal
OS:      Solaris running
Util:    1.8%
Uptime:  10m
Vcpu:    2
         vid    pid    util strand
         0      4       29%   100%
         1      5       29%   100%
Memory:  768M
         real-addr        phys-addr        size
         0x8000000        0x48000000       768M
Vars:    auto-boot?=true
         boot-device=/virtual-devices@100/channel-devices@200/disk@0
Vnet:    net
         mac-addr=0:14:4f:f8:75:3
         service: primary-vsw0 @ primary
Vdisk:   vdisk-1 disk-ldg1@primary-vds0
         service: primary-vds0 @ primary
Vcons:   group1@primary-vcc0 [port:5000]
----------------------------------------------------------------------------
Name:    ldg2
State:   active
Flags:   normal
OS:      Solaris running
Util:    9.5%
Uptime:  3m
Vcpu:    3
         vid    pid    util strand
         0      6       35%   100%
         1      7       34%   100%
         2      8       35%   100%
Memory:  1G
         real-addr        phys-addr        size
         0x8000000        0x78000000       1G
Vars:    auto-boot?=true
         boot-device=/virtual-devices@100/channel-devices@200/disk@0
Vnet:    net
         mac-addr=0:14:4f:f8:50:74
         service: primary-vsw0 @ primary
Vdisk:   vdisk-2 disk-ldg2@primary-vds0
```

CODE EXAMPLE 5-1 Long List of Logical Domain Configuration *(Continued)*

```
        service: primary-vds0 @ primary
Vcons:  group2@primary-vcc0 [port:5001]
#
```

Configuring Split PCI Express Bus to Use Multiple Logical Domains

The PCI Express (PCI-E) bus on a supported server consists of two ports with various leaf devices attached to them. These are identified on a server with the names `pci@780 (bus_a)` and `pci@7c0 (bus_b)`. In a multidomain environment, the PCI-E bus can be programmed to assign each leaf to a separate domain using the Logical Domains Manager. Thus, you can enable more than one domain with direct access to physical devices instead of using I/O virtualization.

When the Logical Domains system is powered on, the control (`primary`) domain uses all the physical device resources, so the primary domain owns both the PCI-E bus leaves.

Caution – All internal disks on the supported servers are connected to a single leaf. If a control domain is booted from an internal disk, do not remove that leaf from the domain. Also, ensure that you are not removing the leaf with the primary network port. If you remove the wrong leaf from the control or service domain, that domain would not be able to access required devices and would become unusable. If the primary network port is on a different bus than the system disk, then move the network cable to an onboard network port and use the Logical Domains Manager to reconfigure the virtual switch (`vsw`) to reflect this change.

▼ To Create a Split PCI Configuration

The example shown here is for a Sun Fire T2000 Server. This procedure also can be used on a Sun Fire T1000 Server and a Netra T2000 Server. The instructions for different servers might vary slightly from these, but you can obtain the basic principles from the example. Mainly, you need to retain the leaf that has the boot disk and remove the other leaf from the primary domain and assign it to another domain.

1. **Verify that the `primary` domain owns both leaves of the PCI Express bus by using the following command:**

```
$ ldm list-bindings primary
...
     IO:      pci@780 (bus_a)
              pci@7c0 (bus_b)
...
```

2. **Determine the device path of the boot disk, which needs to be retained.**

```
primary# df /
/               (/dev/dsk/c1t0d0s0 ): 1309384 blocks   457028 files
```

3. **Determine the physical device to which the block device `c1t0d0s0` is linked.**

```
primary# ls -l /dev/dsk/c1t0d0s0
lrwxrwxrwx   1 root     root          65 Feb  2 17:19 /dev/dsk/c1t0d0s0 -> ../
../devices/pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/sd@0,0:a
```

In this example, the physical device for the boot disk for domain `primary` is under the leaf `pci@7c0`, which corresponds to our earlier listing of `bus_b`. This means that we can assign `bus_a` (`pci@780`) of the PCI-Express bus to another domain.

4. **Check `/etc/path_to_inst` to find the physical path of the onboard network ports.**

```
primary# grep e1000g /etc/path_to_inst
```

5. **Remove the leaf that does not contain the boot disk (`pci@780` in this example) from the `primary` domain.**

```
primary# ldm remove-io pci@780 primary
```

6. **Add this split PCI configuration (**`split-cfg` **in this example) to the system controller.**

```
primary# ldm add-config split-cfg
```

This configuration (`split-cfg`) is also set as the next configuration to be used after the reboot.

7. **Reboot the** `primary` **domain so that the change takes effect.**

```
primary# shutdown -i6 -g0 -y
```

8. **Add the leaf (**`pci@780` **in this example) to the domain (**`ldg1` **in this example) that needs direct access.**

```
primary# ldm add-io pci@780 ldg1
Notice: the LDom Manager is running in configuration mode. Any
configuration changes made will only take effect after the machine
configuration is downloaded to the system controller and the host
is reset.
```

If you have an Infiniband card, you might need to enable the bypass mode on the pci@780 bus. See "Enabling the I/O MMU Bypass Mode on a PCI Bus" on page 60 for a discussion of whether or not you need to enable the bypass mode.

9. **Reboot domain** `ldg1` **so that the change takes effect.**

All domains must be inactive for this reboot. If you are configuring this domain for the first time, the domain will be inactive.

```
ldg1# shutdown -i6 -g0 -y
```

10. **Confirm that the correct leaf is still assigned to the `primary` domain and the correct leaf is assigned to domain `ldg1`.**

```
primary# ldm list-bindings primary
Name:   primary
State:  active
Flags:  transition,control,vio service
OS:
Util:   0.3%
Uptime: 15m
Vcpu:   4
...
IO:     pci@7c0 (bus_b)
....
----------------------------------------------------------------
Name: ldg1
State:  active
Flags:  transition
OS:
Util:   100%
Uptime: 6m
Vcpu:   1
...
IO:     pci@780 (bus_a)
...
```

This output confirms that the PCI-E leaf `bus_b` and the devices below it are assigned to domain `primary`, and `bus_a` and its devices are assigned to `ldg1`.

Enabling the I/O MMU Bypass Mode on a PCI Bus

If you have an Infiniband Host Channel Adapter (HCA) card, you might need to turn the I/O memory management unit (MMU) bypass mode on. By default, Logical Domains software controls PCI-E transactions so that a given I/O device or PCI-E option can only access the physical memory assigned within the I/O domain. Any attempt to access memory of another guest domain is prevented by the I/O MMU. This provides a higher level of security between the I/O domain and all other domains. However, in the rare case where a PCI-E or PCI-X option card does not load or operate with the I/O MMU bypass mode off, this option allows you to turn the I/O MMU bypass mode on. However, if you turn the bypass mode on, there no longer is a hardware-enforced protection of memory accesses from the I/O domain.

The `bypass=on` option turns on the I/O MMU bypass mode. This bypass mode should be enabled only if the respective I/O domain and I/O devices within that I/O domain are trusted by all guest domains. This example turns on the bypass mode.

```
# ldm add-io bypass=on pci@780 ldg1
```

This example shows what the I/O information on a list for an unconfigured primary domain looks like:

```
...
IO:     pci@780 (bus_a)
                (in IO MMU bypass mode)
        pci@7c0 (bus_b)
                (in IO MMU bypass mode)
...
```

This example shows what the I/O information on a list for a configured primary domain looks like

```
...
IO:     pci@780 (bus_a)
        pci@7c0 (bus_b)
...
```

This example shows what the I/O information on a list for a guest domain with I/O MMU bypass mode enabled on `pci@7c0` (inactive) looks like

```
...
IO:     pci@7c0
                (IO MMU bypass mode requested)
...
```

This example shows what the I/O information on a list for a guest domain with I/O MMU bypass mode enabled on `pci@7c0` (bound) looks like

```
...
IO:     pci@7c0 (bus_b)
                (in IO MMU bypass mode)
```

Operating the Solaris OS With Logical Domains

This section describes the following changes in behavior in using the Solaris OS that occur once a configuration created by the Logical Domains Manager is instantiated; that is, domaining is enabled:

- "After a Solaris OS Shutdown" on page 62
- "After a Solaris OS Break Key Sequence (L1-A)" on page 62
- "After Halting or Rebooting the Primary Domain" on page 63
- "Some `format`(1M) Command Options Do Not Work With Virtual Disks" on page 63

After a Solaris OS Shutdown

If domaining is not enabled, the Solaris OS normally goes to the OpenBoot™ prompt after a `shutdown`(1M) command is issued. With domaining enabled, you receive the following prompt after shutdown:

```
r)eboot, o)k prompt, h)alt?
```

Type the letter that represents what you want the system to do after the shutdown.

After a Solaris OS Break Key Sequence (L1-A)

If domaining is not enabled, the Solaris OS normally goes to the OpenBoot prompt after a break key sequence (L1-A) is issued. With domaining enabled, you receive the following prompt after this type of break:

```
c)ontinue, s)ync, r)eboot, h)alt?
```

Type the letter that represents what you want the system to do after this type of break.

After Halting or Rebooting the Primary Domain

The following table shows the expected behavior of halting or rebooting the `primary` domain.

TABLE 5-1 Expected Behavior of Halting or Rebooting the Primary Domain

Action	Domaining Enabled?	Other Domain Configured?	Behavior
Halt	Disabled	N/A	Drops to the `ok` prompt
	Enabled	No	Powers off the host
	Enabled	Yes	Soft resets the host
Reboot	Disabled	N/A	Powers off and powers on the host
	Enabled	No	Powers off and powers on the host
	Enabled	Yes	Soft resets the host

When you type **`halt`** at the Solaris OS prompt, you receive the following prompt:

```
r)eboot, o)k prompt, h)alt?
```

If you select `h)alt` or `r)eboot`, the behavior in Table 5-1 applies. If you select the `o)k` prompt, the behavior is to reboot the domain and return the domain to the `ok` prompt. This is not the same as dropping to the `ok` prompt when domaining is disabled.

A soft reboot restarts the domain with the system controller or Virtual Blade System Controller (vBSC) involvement. This results in doing a soft reset of the PCI framework.

Some `format`(1M) Command Options Do Not Work With Virtual Disks

The Solaris OS `format`(1M) command does not work in a guest domain with virtual disks:

- Some subcommands, such as `label`, `verify`, or `inquiry` fail with virtual disks.
- The `format`(1M) command might display messages, such as:
  - `Inquiry failed`
  - `Disk unformatted`

  - `Current disk is unformatted`
  - `Drive type unknown`
- The format(1M) command crashes when you select a virtual disk that has an Extensible Firmware Interface (EFI) disk label.
- When running the `format`(1M) command in a guest domain, all virtual disks are seen as unformatted, even when they are correctly formatted and have a valid disk label.

For getting or setting the volume table of contents (VTOC) of a virtual disk, use the `prtvtoc`(1M) command and `fmthard`(1M) command instead of the `format`(1M) command. You also can use the `format`(1M) command from the service domain on the real disks.

Moving a Logical Domain From One Server to Another

You can move a logical domain that is not running from one server to another. Before you move the domain, if you set up the same domain on two servers, the domain will be easier to move. In fact you do not have to move the domain itself; you only have to unbind and stop the domain on one server and bind and start the domain on the other server.

During domain setup, do the following:

1. Create a domain with the same name on two servers; for example, create `domainA1` on `serverA` and `serverB`.
2. Add a virtual disk server device and a virtual disk to both servers. The virtual disk server opens the underlying device for export as part of the bind.
3. Bind the domain only on one server; for example, `serverA`. Leave the domain inactive on the other server.

When it comes time to move the domain, do the following:

1. Unbind and stop the domain on `serverA`.
2. Bind and start the domain on `serverB`.

Note – No resources are used until you bind the domain.

Using LDoms With ALOM CMT

The section describes information to be aware of in using Advanced Lights Out Manager (ALOM) Chip Multithreading (CMT) with the Logical Domains Manager. For more information about using the ALOM CMT software, refer to the *Advanced Lights Out Management (ALOM) CMT v1.3 Guide*.

Caution – The ALOM CMT documentation refers to only one domain, so you must be aware that the Logical Domains Manager is introducing multiple domains. If a logical domain is restarted, I/O services for guest domains might be unavailable until the control domain has restarted. This is because the control domain functions as a service domain in the Logical Domains Manager 1.0 software. Guest domains appear to freeze during the reboot process. Once the control domain has fully restarted, the guest domains resume normal operations. It is only necessary to shut down guest domains when power is going to be removed from the entire server. See "Stopping, Unbinding, and Deleting a Guest Logical Domain" on page 52 for details.

An additional option is available to the existing ALOM CMT command:

`bootmode [normal|reset_nvram|bootscript=strong|config="`*config-name*`"]`

The `config="`*config-name*`"` option enables you to set the configuration on the next power on to another configuration, including the `factory-default` shipping configuration.

You can invoke the command whether the host is powered on or off. It takes effect on the next host reset or power on.

▼ To Reset the Logical Domain Configuration to the Default or Another Configuration

- **Reset the logical domain configuration on the next power on to the default shipping configuration by executing this command in ALOM CMT software:**

```
sc> bootmode config="factory-default"
```

You also can select other configurations that have been created with the Logical Domains Manager using the `ldm add-config` command and stored on the system controller (SC). The name you specify in the Logical Domains Manager `ldm add-`

`config` command can be used to select that configuration with the ALOM CMT `bootmode` command. For example, assume you stored the configuration with the name `ldm-config1`:

```
sc> bootmode config="ldm-config1"
```

See Appendix A or the `ldm` man page for more information about the `ldm add-config` command.

Accessing the `ldm`(1M) Man Page

The command line interface (CLI) to the Logical Domains Manager is the `ldm`(1M) command. The man page `ldm`(1M) is part of the `SUNWldm` package and is installed when the `SUNWldm` package is installed. See also Appendix A for the complete text of the `ldm`(1M) man page.

▼ To Access the `ldm`(1M) Man Page

● Add the directory path `/opt/SUNWldm/man` to the variable `$MANPATH`.

Restrictions on Entering Names in the CLI

The following section describes the restrictions on entering names in the Logical Domains Manager CLI.

File Names (*file*) and Variable Names (*var_name*)

- First character must be a letter, a number, or a forward slash (`/`).
- Subsequent letters must be letters, numbers, or punctuation.

Virtual Disk Server *file|device* and Virtual Switch *device* Names

- Must contain letters, numbers, or punctuation.

All Other Names

The remainder of the names, such as the logical domain name (*ldom*), service names (*vswitch_name*, *service_name*, *vdpcs_service_name*, and *vcc_name*), virtual network name (*if_name*), and virtual disk name (*disk_name*), must be in the following format:

- First character must be a letter or number.
- Subsequent characters must be letters, numbers, or any of the following characters: `'-_+#.:;~()'`

Enabling and Using BSM Auditing

The Logical Domains Manager uses the Solaris OS Basic Security Module (BSM) Auditing capability for auditing. BSM Auditing provides the means to examine the history of actions and events on your control domain to determine what happened. The history is kept in a log of what was done, when it was done, by whom, and what was affected.

If you want to use this auditing capability, this section describes how to enable, verify, disable, print output, and rotate audit logs. You can find further information about BSM Auditing in the Solaris 10 *System Administration Guide: Security Services*.

You can enable BSM Auditing in one of two ways. When you want to disable auditing, be sure you use the same method that you used in enabling. The two methods are:

- Use the `enable-bsm.fin` finish script in the Solaris Security Toolkit.

  The `enable-bsm.fin` script is not used by default by the `ldm_control-secure.driver`. You must enable the finish script in your chosen driver.

- Use the Solaris OS `bsmconv`(1M) command.

Here are the procedures for both methods.

▼ To Use the `enable-bsm.fin` Finish Script

1. **Copy the** `ldm_control-secure.driver` **to** *my-ldm.driver*, **where** *my-ldm.driver* **is the name for your copy of the** `ldm_control-secure.driver`.
2. **Copy the** `ldm_control-config.driver` **to** *my-ldm-config.driver*, **where** *my-ldm-config.driver* **is the name for your copy of the** `ldm_control-config.driver`.
3. **Copy the** `ldm_control-hardening.driver` **to** *my-ldm-hardening.driver*, **where** *my-ldm-hardening.driver* **is the name for your copy of the** `ldm_control-hardening.driver`.
4. **Edit** *my-ldm.driver* **to refer to the new configuration and hardening drivers,** *my-ldm-control.driver* **and** *my-ldm-hardening.driver*, **respectively.**
5. **Edit** *my-ldm-hardening.driver*, **and remove the pound sign (**#**) from the following line in the driver.**

```
# enable-bsm.fin
```

6. **Execute** *my-ldm.driver*.

```
# /opt/SUNWjass/bin/jass-execute -d my-ldm.driver
```

7. **Reboot the Solaris OS for auditing to take effect.**

▼ To Use the Solaris OS `bsmconv`(1M) Command

1. **Add** `vs` **in the** `flags:` **line of the** `/etc/security/audit_control` **file.**
2. **Run the** `bsmconv`**(1M) command.**

```
# /etc/security/bsmconv
```

For more information about this command, refer to the Solaris 10 Reference Manual Collection or the man page.

3. **Reboot the Solaris Operating System for auditing to take effect.**

▼ To Verify that BSM Auditing is Enabled

1. **Type:**

```
# auditconfig -getcond
```

2. **Check that** `audit condition = auditing` **appears in the output.**

▼ To Disable Auditing

You can disable auditing in one of two ways, depending on how you enabled it. See "Enabling and Using BSM Auditing" on page 67.

1. **Do one of the following:**
   - Undo the Solaris Security Toolkit hardening run which enabled BSM auditing.

```
# /opt/SUNWjass/bin/jass-execute -u
```

   - Use the Solaris OS `bsmunconv`(1M) command.

```
# /etc/security/bsmunconv
```

2. **Reboot the Solaris OS for the disabling of auditing to take effect.**

▼ To Print Audit Output

- **Use on of the following to print BSM audit output.**
  - Use the Solaris OS commands `auditreduce`(1M) and `praudit`(1M) to print audit output. For example:

```
# auditreduce -c vs | praudit
# auditreduce -c vs -a 20060502000000 | praudit
```

  - Use the Solaris OS `praudit -x` command to print XML output.

▼ To Rotate Audit Logs

- **Use the Solaris OS** `audit -n` **command to rotate audit logs.**

Using `ldm list` Subcommands

This section shows many of the `ldm list` subcommands and their output.

Syntax Usage for the ldm Subcommands

▼ To Show Syntax Usage for `ldm` Subcommands

- **To look at syntax usage for all** `ldm` **subcommands, do the following:**

CODE EXAMPLE 5-2 Syntax Usage for All `ldm` Subcommands

```
# ldm --help

Usage:
 ldm [--help] command [options] [properties] operands

Command(s) for each resource:

     bindings
         list-bindings <ldom>*

     services
         list-services <ldom>*

     constraints
         list-constraints (-x|--xml <ldom>) | ([-p|--parseable] <ldom>*)

     devices
         list-devices [-a] [cpu | mau | memory | io]

     domain      ( dom )
         add-domain -i <file> | --input <file> | <ldom>
         remove-domain -a|--all | <ldom> [<ldom>]*
         list-domain [-l | --long] [-p | --parseable ] <ldom>
         start-domain -a|--all | -i <file> | --input <file> | <ldom>  [<ldom>]*
         stop-domain [-f|--force] (-a|--all | <ldom> [<ldom>]*)
```

CODE EXAMPLE 5-2 Syntax Usage for All `ldm` Subcommands *(Continued)*

```
        bind-domain -i <file> | --input <file> | <ldom>
        unbind-domain <ldom>

    io
        add-io [bypass=on] <bus> <ldom>
        remove-io <bus> <ldom>

    mau
        add-mau <number> <ldom>
        set-mau <number> <ldom>
        remove-mau <number> <ldom>

    memory      ( mem )
        add-memory <number>[GMK] <ldom>
        set-memory <number>[GMK] <ldom>
        remove-memory <number>[GMK] <ldom>

    reconf
        remove-reconf <ldom>

    config      ( spconfig )
        add-spconfig <config_name>
        set-spconfig <config_name>
        remove-spconfig <config_name>
        list-spconfig

    variable    ( var )
        add-variable <var_name>=<value> <ldom>
        set-variable <var_name>=<value> <ldom>
        remove-variable <var_name> <ldom>
        list-variable [<var_name>*] <ldom>

    vconscon    ( vcc )
        add-vconscon port-range=<x>-<y> <vcc_name> <ldom>
        set-vconscon port-range=<x>-<y> <vcc_name>
        remove-vconscon [-f|--force] <vcc_name>

    vconsole    ( vcons )
        set-vconsole [<group>@]<vcc_name> <ldom>

    vcpu
        add-vcpu <number> <ldom>
        set-vcpu <number> <ldom>
        remove-vcpu <number> <ldom>

    vdisk
        add-vdisk <disk_name> <volume_name>@<service_name> <ldom>
```

CODE EXAMPLE 5-2 Syntax Usage for All `ldm` Subcommands *(Continued)*

```
          remove-vdisk [-f|--force] <disk_name> <ldom>

      vdiskserver ( vds )
          add-vdiskserver <service_name> <ldom>
          remove-vdiskserver [-f|--force] <service_name>

      vdpcc       ( ndpsldcc )
          add-vdpcc <vdpcc_name> <service_name> <ldom>
          remove-vdpcc [-f|--force] <vdpcc_name> <ldom>

      vdpcs       ( ndpsldcs )
          add-vdpcs <vdpcs_name> <ldom>
          remove-vdpcs [-f|--force] <vdpcs_name>

      vdiskserverdevice   ( vdsdev )
          add-vdiskserverdevice <file|device>  <volume_name>@<service_name>
          remove-vdiskserverdevice [-f|--force]  <volume_name>@<service_name>

      vnet
          add-vnet [mac-addr=<num>] <if_name> <vswitch_name> <ldom>
          set-vnet [mac-addr=<num>] [vswitch=<vswitch_name>] <if_name>  <ldom>
          remove-vnet [-f|--force] <if_name> <ldom>

      vswitch     ( vsw )
          add-vswitch [mac-addr=<num>] [net-dev=<device>]  <vswitch_name> <ldom>
          set-vswitch [mac-addr=<num>] [net-dev=<device>] <vswitch_name>
          remove-vswitch [-f|--force] <vswitch_name>

 Command aliases:
          Alias          Command
          -----          -------
          ls             list
          ls-*           list-*
          rm-*           remove-*
          *-dom          *-domain
          *-config       *-spconfig
          bind           bind-domain
          create         add-domain
          destroy        remove-domain
          start          start-domain
          stop           stop-domain
          unbind         unbind-domain
          cancel-reconf  remove-reconf
```

Flag Definitions in List Output

The following flags might be shown in the output for a domain:

- `c` = control domain
- `v` = virtual I/O service domain
- `d` = delayed reconfiguration
- `t` = transition
- `n` = normal
- `s` = starting or stopping

If you use the long (`-l`) option for the command, the flags are spelled out. If not, you see the letter abbreviation.

Examples of Various Lists

▼ To Show Software Versions

- **To show the current software versions installed, do the following and you receive a listing similar to the following:**

CODE EXAMPLE 5-3 Software Versions Installed

```
# ldm -V

Logical Domain Manager (v 1.0)
   Hypervisor control protocol v 1.0

System PROM:
   ResetConfig  v. 0.0.0  for LDoms 1.0 (full version information not yet
available)
   Hypervisor   v. 0.0.0  for LDoms 1.0 (full version information not yet
available)
   OpenBoot     v. 0.0.0  for LDoms 1.0 (full version information not yet
available)
```

▼ To Generate a Short List

- **To generate a short list for all domains, do the following:**

CODE EXAMPLE 5-4 Short List for All Domains

```
# ldm list
Name             State    Flags    Cons     VCPU  Memory    Util  Uptime
primary          active   -t-cv    SP       4     1G        0.5%  3d 21h 7m
ldg1             active   -t---    5000     8     1G         23%  2m
```

▼ To Generate a Long List

- **To generate a long list for all domains, do the following:**

CODE EXAMPLE 5-5 Long List for All Domains

```
# ldm list -l
Name:   primary
State:  active
Flags:  transition,control,vio service
OS:
Util:   0.4%
Uptime: 3d 21h 8m
Vcpu:   4
         vid    pid    util strand
         0      0      1.3%   100%
         1      1      0.4%   100%
         2      2      0.1%   100%
         3      3      0.1%   100%
Memory: 1G
         real-addr        phys-addr        size
         0x4000000        0x4000000        1G
Vars:   boot-device=/pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/ disk@0,0:a
         reboot-command=boot
IO:     pci@780 (bus_a)
         pci@7c0 (bus_b)
Vldc:   primary-vldc0   [num_clients=4]
Vldc:   primary-vldc3   [num_clients=7]
Vds:    primary-vds0    [num_clients=1]
                vdsdev: vol0    device=/export/home/solaris_10.disk
Vcc:    primary-vcc0    [num_clients=1]
                port-range=5000-5100
Vsw:    primary-vsw0    [num_clients=1]
                mac-addr=0:14:4f:fa:ff:fa
                net-dev=e1000g0
                mode=prog,promisc
Vcons:  SP
-----------------------------------------------------------------------------
```

CODE EXAMPLE 5-5 Long List for All Domains *(Continued)*

```
Name:   ldg1
State:  active
Flags:  transition
OS:
Util:   0.0%
Uptime: 2m
Vcpu:   8
         vid    pid    util strand
         0      4      1.4%   100%
         1      5      1.0%   100%
         2      6      1.0%   100%
         3      7      0.9%   100%
         4      8       77%   100
         5      9       78%   100%
         6      10      78%   100%
         7      11      79%   100%
Memory: 1G
         real-addr        phys-addr        size
         0x4000000        0x44000000       1G
Vars:   boot-device=/virtual-devices@100/channel-devices@200/disk@0:a
         nvramrc=devalias vnet0 /virtual-devices@100/channel-
          devices@200/network@0
         use-nvramrc?=true
Vdisk:  vdisk0  vol0@primary-vds
         service: primary-vds0 @ primary
Vnet:   vnet0
         mac-addr=0:14:4f:fa:f:5
         service: primary-vsw0 @ primary
Vcons:  ldg1@primary-vcc0 [port:5000]
```

▼ To Generate a Parseable, Machine-Readable List

- **To generate a parseable, machine-readable list of all domains, do the following:**

CODE EXAMPLE 5-6 Machine-Readable List

```
# ldm list -p
primary|active|-t-cv|SP|4|1073741824|0.5|335269
ldg1|active|-t---|5000|8|1073741824|0.2|128
```

▼ To Show the Status of a Domain

- **To look at the status of a domain, for example guest domain `ldg1`, do the following:**

CODE EXAMPLE 5-7 Guest Domain Status

```
# ldm list-domain ldg1
Name             State    Flags   Cons    VCPU  Memory   Util  Uptime
ldg1             active   -t---   5000    8     1G       0.3%  2m
```

▼ To List a Variable

- **To list a variable (for example, `boot-device`) for a domain (for example, `ldg1`), do the following:**

CODE EXAMPLE 5-8 Variable List for a Domain

```
# ldm list-variable boot-device ldg1
boot-device=/virtual-devices@100/channel-devices@200/disk@0:a
```

▼ To List Bindings

- **To list resources that are bound for a domain, for example `ldg1`, do the following:**

CODE EXAMPLE 5-9 Bindings List for a Domain

```
# ldm list-bindings ldg1
Name:   ldg1
State:  active
Flags:  transition
OS:
Util:   0.3%
Uptime: 5m
Vcpu:   8
         vid    pid    util strand
         0      4      0.3%   100%
         1      5      0.0%   100%
         2      6      2.4%   100%
         3      7      0.1%   100%
         4      8      0.0%   100%
         5      9      0.0%   100%
         6      10     1.0%   100%
```

CODE EXAMPLE 5-9 Bindings List for a Domain *(Continued)*

```
         7      11     0.4%   100%
Memory: 1G
         real-addr        phys-addr        size
         0x4000000        0x44000000       1G
Vars:   boot-device=/virtual-devices@100/channel-devices@200/disk@0:a
         nvramrc=devalias vnet0 /virtual-devices@100/channel-
          devices@200/network@0
         use-nvramrc?=true
Vldcc:  vldcc0  [Domain Services]
         service: primary-vldc0 @ primary
         [LDC: 0x0]
Vdisk:  vdisk0  vol0@primary-vds0
         service: primary-vds0 @ primary
         [LDC: 0x1]
Vnet:   vnet0
         mac-addr=0:14:4f:fa:f:55
         service: primary-vsw0 @ primary
         [LDC: 0x2]
Vcons:  [via LDC:3]
         ldg1@primary-vcc0 [port:5000]
```

▼ To List Configurations

- **To list logical domain configurations that have been stored on the SC, do the following:**

CODE EXAMPLE 5-10 Configurations List

```
# ldm list-config
factory-default [current]
initial [next]
```

The labels to the right of the configuration name mean the following:

- `current` - configuration currently being used
- `next` - configuration to be used at the next power cycle

▼ To List Devices

- **To list all server resources, bound and unbound, do the following:**

CODE EXAMPLE 5-11 List of All Server Resources

```
# ldm list-devices -a
vCPU:
         vCPUID  %FREE
         0       0%
         1       0%
         2       0%
         3       0%
         4       0%
         5       0%
         6       0%
         7       0%
         8       0%
         9       0%
         10      0%
         11      0%
         12      100%
         13      100%
         14      100%
         15      100%
         16      100%
         17      100%
         18      100%
         19      100%
         20      100%
         21      100
         22      100%
         23      100%
         24      100%
         25      100%
         26      100%
         27      100%
         28      100%
         29      100%
         30      100%
         31      100%

MAU:
         Free MA-Units:
         cpuset (0, 1, 2, 3)
         cpuset (4, 5, 6, 7)
         cpuset (8, 9, 10, 11)
         cpuset (12, 13, 14, 15)
```

CODE EXAMPLE 5-11 List of All Server Resources *(Continued)*

```
         cpuset (16, 17, 18, 19)
         cpuset (20, 21, 22, 23)
         cpuset (24, 25, 26, 27)
         cpuset (28, 29, 30, 31)

         Bound MA-Units:

Memory:
         Available mblocks:
         PADDR             SIZE
         0x84800000        6072M (0x17b800000)

         Bound mblocks:
         PADDR             SIZ
         0x0               512K (0x80000)
         0x80000           1536K (0x180000)
         0x4000000         1G (0x40000000)
         0x200000          62M (0x3e00000)
         0x84000000        8M (0x800000)
         0x44000000        1G (0x40000000)

         Total Memory 8G

I/O Devices:
         Free Devices:

         Bound Devices:
         pci@780 (bus_a)
         pci@7c0 (bus_b)
```

▼ To List Services

- **To list the services that are available, do the following:**

CODE EXAMPLE 5-12 Services List

```
# ldm list-services
Vldc:   primary-vldc0
Vldc:   primary-vldc3
Vds:    primary-vds0
                vdsdev: vol0   device=/export/home/solaris_10.disk
Vcc:    primary-vcc0
                port-range=5000-5100
Vsw:    primary-vsw0
                mac-addr=0:14:4f:fa:ff:fa
                net-dev=e1000g0
                mode=prog,promisc
```

Listing Constraints

To the Logical Domains Manager, constraints are one or more resources you want to have assigned to a particular domain. You either receive all the resources you ask to be added to a domain or you get none of them, depending upon the available resources. The `list-constraints` subcommand lists those resources you requested assigned to the domain.

▼ To List Constraints for All Domains

- **To list constraints for all domains, do the following:**

CODE EXAMPLE 5-13 Constraints List for All Domains

```
# ldm list-constraints
Name:   primary
Vcpu:   4
Memory: 1G
Vars:   boot-device=/pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a
        reboot-command=boot
IO:     pci@780
        pci@7c0
Vldc:   primary-vldc0
Vldc:   primary-vldc3
Vds:    primary-vds0
```

CODE EXAMPLE 5-13 Constraints List for All Domains *(Continued)*

```
                 vdsdev: vol0    device=/export/home/solaris_10.disk
Vcc:    primary-vcc0
                 port-range=5000-5100
Vsw:    primary-vsw0
                 mac-addr=0:14:4f:fa:ff:fa
                 net-dev=e1000g0
-----------------------------------------------------------------------------
Name:   ldg1
Vcpu:   8
Memory: 1G
Vars:   boot-device=/virtual-devices@100/channel-devices@200/disk@0:a
        nvramrc=devalias vnet0 /virtual-devices@100/channel-
         devices@200/network@0
use-nvramrc?=true
Vdisk:  vdisk0  vol0@primary-vds0
         service: primary-vds0
Vnet:   vnet0
         mac-addr=0:14:4f:fa:f:55
         service: primary-vsw0
```

▼ To List Constraints in XML Format

1. **To list constraints in XML format for a particular domain (`ldg1` in this example), do the following:**

CODE EXAMPLE 5-14 Constraints for a Domain in XML Format

```
# ldm list-constraints -x ldg1
<?xml version="1.0"?>
<LDM_interface version="1.0">
   <data version="1.0">
     <ldom_info>
       <ldom_name>ldg1</ldom_name>
     </ldom_info>
     <cpu>
       <number>8</number>
     </cpu>
     <memory>
       <size>1024M</size>
     </memory>
     <network>
       <vnet_name>vnet0</vnet_name>
       <service_name>primary-vsw0</service_name
     </network>
```

CODE EXAMPLE 5-14 Constraints for a Domain in XML Format *(Continued)*

```
    <disk>
       <vdisk_name>vdisk0</vdisk_name>
       <service_name>primary-vds0</service_name>
       <vol_name>vol0</vol_name>
    </disk>
    <var>
       <name>boot-device</name>
       <value>/virtual-devices@100/channel-devices@200/disk@0:a disk
        net</value
    </var>
    <var>
       <name>nvramrc</name>
       <value>devalias vnet0 /virtual-devices@100/channel-devices@200/
        network@0</value>
    </var>
    <var>
       <name>use-nvramrc?</name>
       <value>true</value>
    </var>
  </data>
</LDM_interface>
```

▼ To List Constraints in a Machine-Readable Format

- **To list constraints for all domains in a parseable format, do the following:**

CODE EXAMPLE 5-15 Constraints for All Domains in a Machine-Readable Format

```
# ldm list-constraints -p
Name:   primary
Vcpu:   4
Memory: 1073741824
Vars:
boot-device=/pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a
reboot-command=boot
IO:
pci@780
pci@7c0
Vldc:   primary-vldc0
Vldc:   primary-vldc3
Vds:    primary-vds0
                 vdsdev: vol0    device=/export/home/solaris_10.disk
Vcc:    primary-vcc0
                 port-range=5000-5100
```

CODE EXAMPLE 5-15 Constraints for All Domains in a Machine-Readable Format *(Continued)*

```
Vsw:    primary-vsw0
                 mac-addr=0:14:4f:fa:ff:fa
                 net-dev=e1000g0
-----------------------------------------------------------------------------
Name:   ldg1
Vcpu:   8
Memory: 1073741824
Vars:
boot-device=/virtual-devices@100/channel-devices@200/disk@0:a
nvramrc=devalias vnet0 /virtual-devices@100/channel-devices@200/network@0
use-nvramrc?=true
Vdisk:  vdisk0  vol0@primary-vds0
         service: primary-vds0
Vnet:   vnet0
         mac-addr=0:14:4f:fa:f:55
         service: primary-vsw0
```

Using Console Groups

The virtual network terminal server daemon, `vntsd`(1M), enables you to provide access for multiple domain consoles using a single TCP port. At the time of domain creation, the Logical Domains Manager assigns a unique TCP port to each console by creating a new default group for that domain's console. The TCP port is then assigned to the console group as opposed to the console itself. The console can be bound to an existing group using the `set-vcons` subcommand.

▼ To Use Console Groups

1. **Bind the consoles for the domains into one group.**

   The following example shows binding the console for three different domains (`ldg1`, `ldg2`, and `ldg3`) to the same console group (`group1`).

```
# ldm set-vcons group1@primary-vcc0 ldg1
# ldm set-vcons group1@primary-vcc0 ldg2
# ldm set-vcons group1@primary-vcc0 ldg3
```

2. **Connect to the associated TCP port (`localhost` at port `5000` in this example).**

```
# telnet localhost 5000
primary-vnts-group1: h, l, c{id}, n{name}, q:
```

You are prompted to select one of the domain consoles.

3. **List the domains within the group by selecting `l` (list).**

```
primary-vnts-group1: h, l, c{id}, n{name}, q: l
DOMAIN ID          DOMAIN NAME                  DOMAIN STATE
0                  ldg1                         online
1                  ldg2                         online
2                  ldg3                         online
```

Note – To re-assign the console to a different group or `vcc` instance, the domain must be unbound; that is, it has to be in the inactive state. Refer to the Solaris 10 OS `vntsd`(1M) man page for more information on configuring and using SMF to manage `vntsd` and using console groups.

Configuring Virtual Switch and Service Domain for NAT and Routing

The virtual switch (`vswitch`) is a layer-2 switch, that also can be used as a network device in the service domain. The virtual switch can be configured to act only as a switch between the virtual network (`vnet`) devices in the various logical domains but with no connectivity to a network outside the box through a physical device. In this mode, plumbing the `vswitch` as a network device and enabling IP routing in the service domain, enables virtual networks to communicate outside the box using the service domain as a router. This mode of operation is very essential to provide external connectivity to the domains when the physical network adapter is not GLDv3-compliant.

The advantages of this configuration are:

- The virtual switch does not need to use a physical device directly and can provide external connectivity even when the underlying device is not GLDv3-compliant.
- The configuration can take advantage of the IP routing and filtering capabilities of the Solaris OS.

▼ To Set Up the Virtual Switch to Provide External Connectivity to Domains

1. **Create a virtual switch with no associated physical device. If assigning an address, ensure that the virtual switch has an unique MAC address.**

```
ldm add-vsw [mac-addr=xx:xx:xx:xx:xx:xx] primary-vsw0 primary
```

2. **Plumb the virtual switch as a network device in addition to the physical network device being used by the domain.**

   See "To Configure the Virtual Switch as the Primary Interface" on page 43 for more information about plumbing the virtual switch.

3. **Configure the virtual switch device for DHCP, if needed.**

   See "To Configure the Virtual Switch as the Primary Interface" on page 43 for more information about configuring the virtual switch device for DHCP.

4. **Create the `/etc/dhcp.vsw` file, if needed.**

5. **Configure IP routing in the service domain, and set up required routing tables in all the domains.**

   For information about how to do this, refer to the section on "Packet Forwarding and Routing on IPv4 Networks" in Chapter 5 "Configuring TCP/IP Network Services and IPv4 Administration" in the *System Administration Guide: IP Services* in the Solaris Express System Administrator Collection.

Using ZFS With Virtual Disks

The following topics regarding using the Zettabyte File System (ZFS) with virtual disks on logical domains are described in this section:

- "Creating a Virtual Disk on Top of a ZFS Volume" on page 86
- "Using ZFS Over a Virtual Disk" on page 87
- "Using ZFS for Boot Disks" on page 89

Creating a Virtual Disk on Top of a ZFS Volume

The following procedure describes how to create a ZFS volume in a service domain and make that volume available to other domains as a virtual disk. In this example, the service domain is the same as the control domain and is named `primary`. The guest domain is named `ldg1` as an example. The prompts in each step show in which domain to run the command.

▼ To Create a Virtual Disk on Top of a ZFS Volume

1. **Create a ZFS storage pool (**`zpool`**).**

```
primary# zpool create -f tank1 c2t42d1
```

2. **Create a ZFS volume.**

```
primary# zfs create -V 100m tank1/myvol
```

3. **Verify that the zpool (**`tank1` **in this example) and ZFS volume (**`tank/myvol` **in this example) have been created.**

```
primary# zfs list
        NAME                     USED  AVAIL  REFER  MOUNTPOINT
        tank1                    100M  43.0G  24.5K  /tank1
        tank1/myvol             22.5K  43.1G  22.5K  -
```

4. **Configure a service exporting** `tank1/myvol` **as a virtual disk.**

```
primary# ldm add-vdsdev /dev/zvol/rdsk/tank1/myvol zvol@primary-vds0
```

5. **Add the exported disk to another domain (**`ldg1` **in this example).**

```
primary# ldm add-vdisk vzdisk zvol@primary-vds0 ldg2
```

6. **On the other domain (`ldg1` in this example), start the domain and ensure that the new virtual disk is visible (you might have to run the `devfsadm` command). In this example, the new disk appears as `/dev/rdsk/c2d2s0`.**

```
ldg1# newfs /dev/rdsk/c2d2s0
newfs: construct a new file system /dev/rdsk/c2d2s0: (y/n)? y
Warning: 4096 sector(s) in last cylinder unallocated
Warning: 4096 sector(s) in last cylinder unallocated
/dev/rdsk/c2d2s0: 204800 sectors in 34 cylinders of 48 tracks, 128
sectors
100.0MB in 3 cyl groups (14 c/g, 42.00MB/g, 20160 i/g) super-block
backups
(for fsck -F ufs -o b=#) at: 32, 86176, 172320,

ldg1# mount /dev/dsk/c2d2s0 /mnt

ldg1# df -h /mnt
Filesystem             size   used   avail capacity  Mounted on
/dev/dsk/c2d2s0         93M   1.0M     82M       2%  /mnt
```

Note – A ZFS volume is exported to a logical domain as a virtual disk slice. Therefore, it is not possible to either use the `format` command or install the Solaris OS to a `zvol`-backed virtual disk.

Using ZFS Over a Virtual Disk

The following procedure shows how to directly use ZFS from a domain on top of a virtual disk. You can create ZFS pools, file systems, and volumes over the top of virtual disks with the Solaris 10 OS `zpool`(1M) and `zfs`(1M) commands. Although the storage backend is different (virtual disks instead of physical disks), there is no change to the usage of ZFS.

Additionally, if you have an already existing ZFS file system, then you can export it from a service domain to use it in another domain.

In this example, the service domain is the same as the control domain and is named `primary`. The guest domain is named `ldg1` as an example. The prompts in each step show in which domain to run the command.

▼ To Use ZFS Over a Virtual Disk

1. **Create a `zpool` (`tank` in this example), and then verify that it has been created.**

```
primary# zpool create -f tank c2t42d0
primary# zpool list
NAME                     SIZE    USED   AVAIL    CAP    HEALTH   ALTROOT
tank                    43.8G    108K   43.7G    0%    ONLINE   -
```

2. **Create a ZFS file system (`tank/test` in this example), and then verify that it has been created. In this example, the file system is created on top of disk `c2t42d0` by running the following command on the service domain:**

```
primary# zfs create tank/test
primary# zfs list
NAME                       USED  AVAIL  REFER  MOUNTPOINT
tank                       106K  43.1G  25.5K  /tank
tank/test                 24.5K  43.1G  24.5K  /tank/test
```

3. **Export the ZFS pool (`tank` in this example).**

```
primary# zpool export tank
```

4. **Configure a service exporting the physical disk `c2t42d0s2` as a virtual disk.**

```
primary# ldm add-vdsdev /dev/rdsk/c2t42d0s2 volz@primary-vds0
```

5. **Add the exported disk to another domain (`ldg1` in this example).**

```
primary# ldm add-vdisk vdiskz volz@primary-vds0 ldg1
```

6. **On the other domain (`ldg1` in this example), start the domain and make sure the new virtual disk is visible (you might have to run the `devfsadm` command), and then import the ZFS pool.**

```
ldg1# zpool import tank
ldg1# zpool list
NAME              SIZE    USED    AVAIL   CAP   HEALTH   ALTROOT
tank            43.8G    214K   43.7G    0%   ONLINE   -

ldg1# zfs list
NAME                    USED  AVAIL  REFER  MOUNTPOINT
tank                    106K  43.1G  25.5K  /tank
tank/test              24.5K  43.1G  24.5K  /tank/test

ldg1# df -hl -F zfs
Filesystem              size   used  avail capacity  Mounted on
tank                     43G    25K    43G      1%    /tank
tank/test                43G    24K    43G      1%    /tank/test
```

The ZFS pool (`tank/test` in this example) is now imported and usable from domain `ldg1`.

Using ZFS for Boot Disks

A ZFS file system with a large file can be used as the virtual disks in logical domains.

Note – A ZFS file system requires more memory in the service domain. Take this into account when configuring the service domain.

ZFS enables:

- Cloning a file system quickly
- Using the clones to provision additional domains
- Net installing to disk on files and files within a ZFS file system

▼ To Use ZFS for Boot Disks

The following procedure can be used to create ZFS disks for logical domains, and also snapshot and clone them for other domains.

1. **On the `primary` domain, reserve a entire disk or slice for use as the storage for the ZFS pool. Step 2 uses slice 5 of a disk.**

2. **Create a ZFS pool; for example, `ldomspool`.**

```
# zpool create ldomspool /dev/dsk/c0t0d0s5
```

3. **Create a ZFS file system for the first domain (`ldg1` in this example).**

```
# zfs create ldomspool/ldg1
```

4. **Create a file to be the disk for this domain.**

```
# mkfile 1G /ldomspool/ldg1/bootdisk
```

5. **Specify the file as the device to use when creating the domain.**

```
# ldm add-vdsdev /ldomspool/ldg1/bootdisk vol1@primary-vds0
# ldm add-vdisk vdisk1 vol1@primary-vds0 ldg1
```

6. **Boot domain `ldg1` and net install to `vdisk1`. This file functions as a full disk and can have partitions; that is, separate partitions for `root`, `usr`, `home`, `dump`, and `swap`.**

7. **Once the installation is complete, snapshot the file system.**

```
# zfs snapshot ldomspool/ldg1@initial
```

Note – Doing the snapshot before the domain reboots does not save the domain state as part of the snapshot or any other clones created from the snapshot.

---

8. **Create additional clones from the snapshot and use it as the boot disk for other domains (ldg2 and ldg3 in this example).**

```
# zfs clone ldomspool/ldg1@initial ldomspool/ldg2
# zfs clone ldomspool/ldg1@initial ldomspool/ldg3
```

9. **Verify that everything was created successfully.**

```
# zfs list
   NAME                     USED  AVAIL  REFER  MOUNTPOINT
   ldomspool               1.07G  2.84G  28.5K  /ldomspool
   ldomspool/ldg1           1.03G  2.84G  1.00G  /ldomspool/ldg1
   ldomspool/ldg1@initial  23.0M      -  1.00G  -
   ldomspool/ldg2           23.2M  2.84G  1.00G  /ldomspool/ldg2
   ldomspool/ldg3           21.0M  2.84G  1.00G  /ldomspool/ldg3
```

Note – Ensure that the ZFS pool has enough space for the clones that are being created. ZFS uses copy-on-write and uses space from the pool only when the blocks in the clone are modified. Even after booting the domain, the clones only use a small percentage needed for the disk (since most of the OS binaries are the same as those in the initial snapshot).

Using Volume Managers in a Logical Domains Environment

The following topics are described in this section:

- "Using Virtual Disks on Top of Volume Managers" on page 91
- "Using Volume Managers on Top of Virtual Disks" on page 94

Using Virtual Disks on Top of Volume Managers

Any Zettabyte File System (ZFS), Solaris™ Volume Manager (SVM), or Veritas Volume Manager (VxVM) volume can be exported from a service domain to a guest domain as a virtual disk. The exported volume appears as a virtual disk with a single slice (`s0`) into the guest domain.

Note – The remainder of this discussion uses an SVM volume as an example. However, the discussion also applies to ZFS and VxVM volumes.

For example, if a service domain exports the SVM volume `/dev/md/dsk/d0` to `domain1` and `domain1` sees that virtual disk as `/dev/dsk/c0d2*`, then `domain1` only has a `s0` device; that is, `/dev/dsk/c0d2s0`.

The virtual disk in the guest domain (for example, `/dev/dsk/c0d2s0`) is directly mapped to the associated volume (for example, `/dev/md/dsk/d0`), and data stored onto the virtual disk from the guest domain are directly stored onto the associated volume with no extra metadata. So data stored on the virtual disk from the guest domain can also be directly accessed from the service domain through the associated volume.

Examples:

- If the SVM volume `d0` is exported from the `primary` domain to `domain1`, then the configuration of `domain1` requires some extra steps:

```
primary# metainit d0 3 1 c2t70d0s6 1 c2t80d0s6 1 c2t90d0s6
primary# ldm add-vdsdev /dev/md/dsk/d0 vol3@primary-vds0
primary# ldm add-vdisk vdisk3 vol3@primary-vds0 domain1
```

- After `domain1` has been bound and started, the exported volume appears as `/dev/dsk/c0d2s0`, for example, and you can use it:

```
domain1# newfs /dev/rdsk/c0d2s0
domain1# mount /dev/dsk/c0d2s0 /mnt
domain1# echo test-domain1 > /mnt/file
```

- After `domain1` has been stopped and unbound, data stored on the virtual disk from `domain1` can be directly accessed from the primary domain through SVM volume `d0`:

```
primary# mount /dev/md/dsk/d0 /mnt
primary# cat /mnt/file
test-domain1
```

Note – Such a virtual disk cannot be seen by the `format`(1M) command, cannot be partitioned, and cannot be used as an installation disk for the Solaris OS. See "Some `format`(1M) Command Options Do Not Work With Virtual Disks" on page 63 for more information about this topic.

Note on Using Virtual Disks on Top of SVM

When a RAID or mirror SVM volume is used as a virtual disk by another domain, and if there is a failure on one of the components of the SVM volume, then the recovery of the SVM volume using the `metareplace` command or using a hot spare does not start, and the `metastat` command sees the volume as resynchronizing, but the resynchronization does not progress.

For example, `/dev/md/dsk/d0` is a RAID SVM volume which is exported as a virtual disk to another domain and `d0` is configured with some hot-spare devices. If a component of `d0` fails, SVM replaces the failing component with a hot spare and resynchronizes the SVM volume, but the resynchronization does not start, and the volume is reported as resynchronizing, but the resynchronization does not progress:

```
# metastat d0
d0: RAID
    State: Resyncing
    Hot spare pool: hsp000
    Interlace: 32 blocks
    Size: 20097600 blocks (9.6 GB)
Original device:
    Size: 20100992 blocks (9.6 GB)
Device                                     Start Block  Dbase   State Reloc
c2t2d0s1                                           330  No       Okay  Yes
c4t12d0s1                                          330  No       Okay  Yes
/dev/dsk/c10t600C0FF0000000000015153295A4B100d0s1  330  No  Resyncing  Yes
```

In such a situation, the domain using the SVM volume as a virtual disk has to be stopped and unbound to complete the resynchronization. Then the SVM volume can be resynchronized using the `metasync` command.

```
# metasync d0
```

Note on Using Virtual Disks When VxVM Is Installed

When the Veritas Volume Manager (VxVM) is installed on your system, you have to ensure that Veritas Dynamic Multipathing (DMP) is not enabled on the physical disks or partitions you want to export as virtual disks. Otherwise, you receive an error in `/var/adm/messages` while binding a domain that uses such a disk:

```
vd_setup_vd():  ldi_open_by_name(/dev/dsk/c4t12d0s2) = errno 16
vds_add_vd():  Failed to add vdisk ID 0
```

You can check if Veritas DMP is enabled by checking multipathing information in the output of the command `vxdisk list`; for example:

```
# vxdisk list Disk_3
Device:     Disk_3
devicetag: Disk_3
type:       auto
info:       format=none
flags:      online ready private autoconfig invalid
pubpaths:   block=/dev/vx/dmp/Disk_3s2 char=/dev/vx/rdmp/Disk_3s2
guid:       -
udid:       SEAGATE%5FST336753LSUN36G%5FDISKS%5F3032333948303144304E0000
site:       -
Multipathing information:
numpaths:   1
c4t12d0s2   state=enabled
```

If Veritas DMP is enabled on a disk or a slice that you want to export as a virtual disk, then you must disable DMP using the `vxdmpadm` command. For example:

```
# vxdmpadm -f disable path=/dev/dsk/c4t12d0s2
```

Using Volume Managers on Top of Virtual Disks

This section describes the following situations in the Logical Domains environment:

- "Using ZFS on Top of Virtual Disks" on page 94
- "Using SVM on Top of Virtual Disks" on page 94
- "Using VxVM on Top of Virtual Disks" on page 95

Using ZFS on Top of Virtual Disks

Any virtual disk can be used with ZFS. A ZFS storage pool (`zpool`) can be imported in any domain that sees all the storage devices that are part of this `zpool`, regardless of whether the domain sees all these devices as virtual devices or real devices.

Using SVM on Top of Virtual Disks

Any virtual disk can be used in the SVM local disk set. For example, a virtual disk can be used for storing the SVM meta database (`metadb`) of the local disk set or for creating SVM volumes in the local disk set.

Currently, you can only use virtual disks with the local disk set, but not with any shared disk set (`metaset`). Virtual disks can not be added into a SVM shared disk set. Trying to add a virtual disk into a SVM shared disk set fails with an error similar to this:

```
# metaset -s test -a c2d2
metaset: domain1: test: failed to reserve any drives
```

Using VxVM on Top of Virtual Disks

VxVM does not currently work with virtual disks. The VxVM software can be installed into a domain having virtual disks but VxVM is unable to see any of the virtual disks available.

Configuring IPMP in a Logical Domains Environment

Internet Protocol Network Multipathing (IPMP) provides fault-tolerance and load balancing across multiple network interface cards. By using IPMP, you can configure one or more interfaces into an IP multipathing group. After configuring IPMP, the system automatically monitors the interfaces in the IPMP group for failure. If an interface in the group fails or is removed for maintenance, IPMP automatically migrates, or fails over, the failed interface's IP addresses. In a Logical Domains environment, either the physical or virtual network interfaces can be configured for failover using IPMP.

Configuring Virtual Network Devices into an IPMP Group in a Logical Domain

A logical domain can be configured for fault-tolerance by configuring its virtual network devices to an IPMP group. When setting up an IPMP group with virtual network devices, in a active-standby configuration, set up the group to use probe-based detection. Link-based detection and failover currently are not supported for virtual network devices in Logical Domains 1.0 software.

The following diagram shows two virtual networks (`vnet1` and `vnet2`) connected to separate virtual switch instances (`vsw0` and `vsw1`) in the service domain, which, in turn, use two different physical interfaces (`e1000g0` and `e1000g1`). In the event of a physical interface failure, the IP layer in `LDom_A` detects failure and loss of connectivity on the corresponding `vnet` through probe-based detection, and automatically fails over to the secondary `vnet` device.

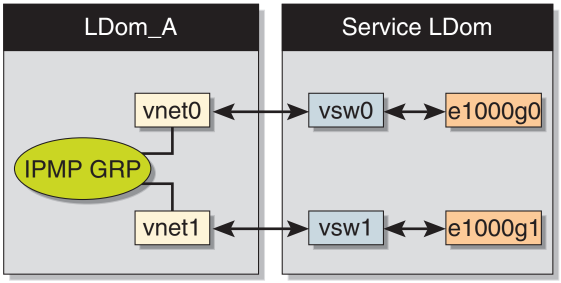

FIGURE 5-1 Two Virtual Networks Connected to Separate Virtual Switch Instances

Further reliability can be achieved in the logical domain by connecting each virtual network device (`vnet0` and `vnet1`) to virtual switch instances in different service domains (as shown in the following diagram). Two service domains (`Service_1` and `Service_2`) with virtual switch instances (`vsw1` and `vsw2`) can be set up using a split-PCI configuration. In this case, in addition to network hardware failure, `LDom_A` can detect virtual network failure and trigger a failover following a service domain crash or shutdown.

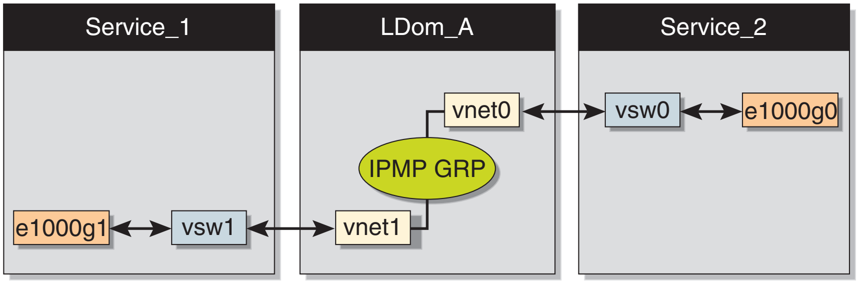

FIGURE 5-2 Each Virtual Network Device Connected to Different Service Domains

Refer to the Solaris 10 *System Administration Guide: IP Services* for more information about how to configure and use IPMP groups.

Configuring and Using IPMP in the Service Domain

Network failure detection and recovery can also be set up in a Logical Domains environment by configuring the physical interfaces in the service domain into a IPMP group. To do this, configure the virtual switch in the service domain as a network device, and configure the service domain itself to act as an IP router. (Refer to the Solaris 10 *System Administration Guide: IP Services* for information on setting up IP routing).

Once configured, the virtual switch sends all packets originating from virtual networks (and destined for an external machine), to its IP layer, instead of sending the packets directly via the physical device. In the event of a physical interface failure, the IP layer detects failure and automatically re-routes packets through the secondary interface.

Since the physical interfaces are directly being configured into a IPMP group, the group can be set up for either link-based or probe-based detection. The following diagram shows two network interfaces (`e1000g0` and `e1000g1`) configured as part of an IPMP group. The virtual switch instance (`vsw0`) has been plumbed as a network device to send packets to its IP layer.

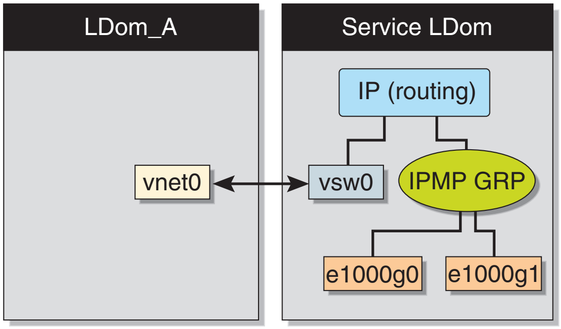

FIGURE 5-3 Two Network Interfaces Configured as Part of IPMP Group

APPENDIX **A**

Reference Section for the `ldm`(1M) Command

The command-line interface (CLI) to the Logical Domains Manager is the `ldm`(1M) command. To use this CLI, you must have the Logical Domains Manager daemon, `ldmd`, running. This appendix provides the `ldm`(1M) reference material found in the `ldm(1M)` man page that is included with the Logical Domains Manager software package (`SUNWldm`).

To access the man page, be sure you have added the directory path `/opt/SUNWldm/man` to the variable `$MANPATH`.

NAME

ldm - command-line interface for the Logical Domains Manager

SYNOPSIS

ldm *or* **ldm --help** **[** *subcommand***]**

ldm -V

ldm list [-l] [-p]

ldm add-domain (-i *file* **|** *ldom***)**

ldm remove-domain (-a | *ldom . . .***)**

ldm list-domain [-l] [-p] [*ldom***...]**

ldm add-vcpu *number ldom*

ldm set-vcpu *number ldom*

ldm remove-vcpu *number ldom*

ldm add-mau *number ldom*

ldm set-mau *number ldom*

ldm remove-mau *number ldom*

ldm add-memory *size*[*unit*] *ldom*

ldm set-memory *size*[*unit*] *ldom*

ldm remove-memory *size*[*unit*] *ldom*

ldm remove-reconf *ldom*

ldm add-io [bypass=on] *bus ldom*

ldm remove-io *bus ldom*

ldm add-vswitch [mac-addr=*num***] [net-dev=***device***]** *vswitch_name ldom*

ldm set-vswitch [mac-addr=*num***] [net-dev=***device***]** *vswitch_name*

ldm remove-vswitch [-f] *vswitch_name*

ldm add-vnet [mac-addr=*num***]** *if_name vswitch_name ldom*

ldm set-vnet [mac-addr=*num***] [***vswitch_name***]** *if_name ldom*

ldm remove-vnet [-f] *if_name ldom*

ldm add-vdiskserver *service_name ldom*

ldm remove-vdiskserver [-f] *service_name*

ldm add-vdiskserverdevice *file*|*device volume_name*@*service_name*

ldm remove-vdiskserverdevice [-f] *volume_name*@*vds_name*

ldm add-vdisk *disk_name volume_name*@*service_name ldom*

ldm remove-vdisk [-f] *disk_name ldom*

ldm add-vdpcs *vdpcs_service_name ldom*

ldm remove-vdpcs *vdpcs_service_name*

ldm add-vdpcc *vdpcc_name vdpcs_service_name ldom*

ldm remove-vdpcc *vdpcc_name ldom*

ldm add-vconscon port-range=*x***-***y vcc_name ldom*

ldm set-vconscon port-range=*x***-***y vcc_name*

ldm remove-vconscon [-f] *vcc_name*

ldm set-vconsole [group@]*vcc_name ldom*

ldm add-variable *var_name***=***value ldom*

ldm set-variable *var_name***=***value ldom*

ldm remove-variable *var_name ldom*

ldm list-variable *var_name***... [***ldom***...]**

ldm start-domain (-a | -i *file* **|** *ldom***...)**

ldm stop-domain [-f] (-a | *ldom***...)**

ldm bind-domain (-i *file* **|** *ldom***)**

ldm unbind-domain *ldom*

ldm list-bindings [*ldom***...]**

ldm add-config *config_name*

ldm set-config *config_name*

ldm set-config factory-default

ldm remove-config *config_name*

ldm list-config

ldm list-constraints (-x *ldom***) | ([-p] [***ldom***...])**

ldm list-devices [-a] [cpu | mau | memory | io]

ldm list-services [*ldom***...]**

DESCRIPTION

The Logical Domains Manager (`ldm`) is used to create and manage logical domains. There can be only one Logical Domains Manager per server. The Logical Domains Manager runs on the control domain, which is the initial domain created by the system controller (and named `primary`).

A logical domain is a discrete logical grouping with its own operating system, resources, and identity within a single computer system. Each logical domain can be created, destroyed, reconfigured, and rebooted independently, without requiring a power cycle of the server. You can run a variety of applications in different logical domains and keep them independent for security purposes.

All logical domains are the same except for the roles that you specify for them. There are several roles that logical domains can perform:

Control domain	Creates and manages other logical domains and services by communicating with the hypervisor.
Service domain	Provides services to other logical domains, such as a virtual network switch or a virtual disk service.
I/O domain	Has direct ownership of and direct access to physical I/O devices, such as a network card in a PCI Express controller. Shares the devices to other domains in the form of virtual devices. You can have a maximum of two I/O domains, one of which also must be the control domain.
Guest domain	Uses services from the I/O and service domains and is managed by the control domain.

SUBCOMMAND SUMMARIES

Following is a list of the supported subcommands with their descriptions and required authorization. For information about setting up authorization for user accounts, refer to “Creating Authorization and Profiles for User Accounts” in the *Logical Domains (LDoms) 1.0 Administration Guide*.

`add-`*resource*	Adds a resource to an existing logical domain. See RESOURCES for resource definitions.	`solaris.ldoms.write`
`add-config`	Adds a logical domain configuration to the system controller (SC).	`solaris.ldoms.write`
`add-domain`	Creates a logical domain.	`solaris.ldoms.write`
`bind-domain`	Binds resources to a created logical domain.	`solaris.ldoms.write`
`remove-reconf`	Cancels delayed reconfiguration operations for a logical domain.	`solaris.ldoms.write`
`remove-domain`	Deletes a logical domain.	`solaris.ldoms.write`

`list-`*type*	Lists server resources, including bindings, constraints, devices, services, and configurations for logical domains.	`solaris.ldoms.read`
`list-domain`	Lists logical domains and their states.	`solaris.ldoms.read`
`list-variable`	Lists variables for logical domains.	`solaris.ldoms.read`
`remove-`*resource*	Removes a resource from an existing logical domain. See RESOURCES for resource definitions.	`solaris.ldoms.write`
`remove-config`	Removes a logical domain configuration from the system controller.	`solaris.ldoms.write`
`remove-variable`	Removes one or more variables from an existing logical domain.	`solaris.ldoms.write`
`set-`*resource*	Specifies a resource for an existing logical domain. This can be either a property change or a quantity change. This represents a quantity change when applied to the resources `vcpu`, memory, or `mau`. For a quantity change, the subcommand becomes a dynamic reconfiguration (DR) operation where the quantity of the specified resource is assigned to the specified logical domain. If there are more resources assigned to the logical domain than are specified in this subcommand, some are removed. If there are fewer resources assigned to the logical domain than are specified in this subcommand, some are added. See RESOURCES for resource definitions.	`solaris.ldoms.write`
`set-config`	Specifies a logical domain configuration to use.	`solaris.ldoms.write`
`set-variable`	Sets one or more variables for an existing logical domain.	`solaris.ldoms.write`
`start`-domain	Starts one or more logical domains.	`solaris.ldoms.write`
`stop`-domain	Stops one or more running logical domains.	`solaris.ldoms.write`
`unbind`-domain	Unbinds or releases resources from a logical domain.	`solaris.ldoms.write`

Note – Not all subcommands are supported on all resources types.

ALIASES

The following aliases for the subcommands are supported. The short form is in the first column and the long form in the second column. The aliases apply regardless of the action performed. For example, because `dom` is an alias for domain, then `start-dom`, `stop-dom`, `bind-dom`, and the others are all valid.

`config`	`spconfig`
`dom`	`domain`
`ls`	`list`
`mem`	`memory`
`rm`	`remove`
`var`	`variable`
`vcc`	`vconscon`
`vcons`	`vconsole`
`vdpcc`	`ndpsldcc`
`vdpcs`	`ndpsldcs`
`vds`	`vdiskserver`
`vdsdev`	`vdiskserverdevice`
`vsw`	`vswitch`

Note – In the syntax and examples in the remainder of this man page, the short form of the subcommands are used.

RESOURCES

The following resources are supported:

`io`	I/O devices, such as internal disks and PCI-Express (PCI-E) controllers and their attached adapters and devices.
`mau`	Modular arithmetic unit, a cryptographic unit for a supported server.
`mem`, `memory`	Memory – default size in bytes, or specify gigabytes (`G`), kilobytes (`K`), or megabytes (`M`). Virtualized memory of the server that can be allocated to guest domains.
`vcc`, `vconscon`	Virtual console concentrator service with a specific range of TCP ports to assign to each guest domain at the time it is created.
`vcons`, `vconsole`	Virtual console for accessing system level messages. A connection is achieved by connecting to the `vconscon` service in the control domain at a specific port.

`vcpu`	Virtual CPUs represent each of the cores of a server. For example, an 8-core Sun Fire T2000 server has 32 virtual CPUs that can be allocated between the logical domains.
`vdisk`	Virtual disks are generic block devices backed by different types of physical devices, volumes, or files. A virtual disk is not synonymous with a SCSI disk and, therefore, excludes the target ID (`t`*N*) in the disk label. Virtual disks in a logical domain have the following format: `c`*N*`d`*N*`s`*N*, where `c`*N* is the virtual controller, `d`*N* is the virtual disk number, and `s`*N* is the slice.
`vds`, `vdiskserver`	Virtual disk server allows you to import virtual disks into a logical domain.
`vdsdev`, `vdiskserverdevice`	Device exported by the virtual disk server. The device can be an entire disk, a slice on a disk, a file, or a disk volume.
`vdpcc`	Virtual data plane channel client. Only of interest in a Netra Data Plane Software (NDPS) environment.
`vdpcs`	Virtual data plane channel service. Only of interest in a Netra Data Plane Software (NDPS) environment.
`vnet`	Virtual network device implements a virtual Ethernet device and communicates with other `vnet` devices in the system using the virtual network switch (`vsw`).
`vsw`, `vswitch`	Virtual network switch that connects the virtual network devices to the external network and also switches packets between them.

LIST TYPES

The following list types are supported:

`bindings`	Lists the resources bound to a logical domain.
`constraints`	Lists the constraints used to create a logical domain.
`devices`	Lists all free devices for the server.
`services`	Lists the services exported or consumed by a logical domain.
`config`	Lists the logical domain configurations stored on the system controller.

OPTIONS

The following options are supported. The short option is first, the long option is second, followed by the description of the option.

`ldm`	`--help`	Displays usage statements.
`-a`	`--all`	Operates on all of the operand types.
`-f`	`--force`	Attempts to force an operation.

`-i` *file*	`--input` *file*	Specifies the XML configuration file to use in creating a logical domain.
`-l`	`--long`	Generates a long listing.
`-p`	`--parseable`	Generates a machine-readable version of the output.
`-x`	`--xml`	Specifies that an XML file containing the constraints for the logical domain be written to standard output (`stdout`). Can be used as backup file.
`-V`	`--version`	Displays version information.

PROPERTIES

The following property types are supported:

`mac-addr=`	Defines a MAC address. The number must be in standard octet notation; for example, 80:00:33:55:22:66.
`net-dev=`	Defines the path name of the actual network device.
`port-range=`	Defines a range of TCP ports.

SUBCOMMAND USAGE

This section contains descriptions of every supported command line interface (CLI) operation; that is, every subcommand and resource combination.

ADD AND REMOVE DOMAINS

Add Logical Domain

This subcommand adds a logical domain by specifying a logical domain name or by using an XML configuration file.

`ldm add-dom (-i` *file* **|** *ldom***)**

Where:

- `-i` *file* specifies the XML configuration file to use in creating the logical domain.
- *ldom* specifies the name to use for the logical domain.

Remove Logical Domains

This subcommand removes one or more logical domains.

`ldm rm-dom (-a` **|** *ldom***...)**

Where:

- `-a` option means delete all logical domains except the control domain.
- *ldom*... specifies one or more logical domains to be deleted.

CPU AND MEMORY (Reconfiguration Operations)

There are three types of reconfiguration operations:

- Configuration mode – The Logical Domains Manager runs in configuration mode when the server is in the factory-default configuration. In this mode, no reconfiguration operations take effect until after the configuration is saved to the system controller using the `add-config` subcommand and until that configuration is instantiated by rebooting the control domain.
- Delayed reconfiguration operations – Any add or remove operations on active logical domains, except `add-vcpu`, `set-vcpu`, `rm-vcpu`, `add-vdsdev`, and `rm-vdsdev` subcommands, are considered delayed reconfiguration operations. In addition, the `set-vsw` subcommand on an active logical domain is considered a delayed reconfiguration operation. Delayed reconfiguration operations take effect after the next reboot of the OS or stop and start of the logical domain if no OS is running.
- Dynamic reconfiguration operations – Reconfiguration operations on domains in a bound or inactive state and `add-vcpu` and `rm-vcpu` operations on domains in an active state are considered dynamic reconfiguration operations. Dynamic reconfiguration operations take effect immediately.

Add Virtual CPUs

This subcommand adds the specified number of virtual CPUs to the logical domain.

```
ldm add-vcpu number ldom
```

Where:

- number is the number of virtual CPUs to be added to the logical domain.
- *ldom* specifies the logical domain where the virtual CPUs are to be added.

Set Virtual CPUs

This subcommand specifies the number of virtual CPUs to be set in a logical domain.

```
ldm set-vcpu number ldom
```

Where:

- *number* is the number of virtual CPUs to be set in a logical domain.
- *ldom* is the logical domain where the number of virtual CPUs are to be set.

Remove Virtual CPUs

This subcommand removes the specified number of virtual CPUs in the logical domain.

```
ldm rm-vcpu number ldom
```

Where:

- *number* is the number of virtual CPUs to be removed from the logical domain.
- *ldom* specifies the logical domain where the virtual CPUs are to be removed.

Add Modular Arithmetic Units

This subcommand specifies the number of modular arithmetic units (`mau`), or cryptographic units, to be added to a logical domain.

```
ldm add-mau number ldom
```

Where:

- *number* is the number of `mau` units to be added to the logical domain.
- *ldom* specifies the logical domain where the `mau` units are to be added.

Set Modular Arithmetic Units

This subcommand specifies the number of modular arithmetic units (`mau`) to be set in the logical domain.

```
ldm set-mau number ldom
```

Where:

- *number* is the number of `mau` units to be set in the logical domain.
- *ldom* specifies the logical domain where the number of `mau` units are to be set.

Remove Modular Arithmetic Units

This subcommand removes the specified number of modular arithmetic units (`mau`) from a logical domain.

```
ldm rm-mau number ldom
```

Where:

- *number* is the number of `mau` units to be removed from the logical domain.
- *ldom* specifies the logical domain where the `mau` units are to be removed.

Add Memory

This subcommand adds the specified quantity of memory to a logical domain.

```
ldm add-mem size[unit] ldom
```

Where:

- *size* is the size of memory to be added to a logical domain.
- *unit* (optional) is the unit of measurement. The default is bytes. If you want a different unit of measurement, specify one of the following (the *unit* is not case-sensitive).

`G`	gigabytes
`K`	kilobytes
`M`	megabytes

- *ldom* specifies the logical domain where the memory is to be added.

Set Memory

This subcommand sets a specific quantity of memory in a logical domain.

```
ldm set-mem size[unit] ldom
```

Where:

- *size* is the size of memory to be set in the logical domain.
- *unit* (optional) is the unit of measurement. The default is bytes. If you want a different unit of measurement, specify one of the following (the *unit* is not case-sensitive).

`G`	gigabytes
`K`	kilobytes
`M`	megabytes

- *ldom* specifies the logical domain where the memory is to be modified.

Remove Memory

This subcommand removes the specified quantity of memory from a logical domain.

```
ldm rm-mem size[unit] ldom
```

Where:

- *size* is the size of memory to be removed from the logical domain.
- *unit* (optional) is the unit of measurement. The default is bytes. If you want a different unit of measurement, specify one of the following (the *unit* is not case-sensitive).

`G`	gigabytes
`K`	kilobytes
`M`	megabytes

- *ldom* specifies the logical domain where memory is to be removed.

Remove Delayed Reconfiguration Operations

This subcommand in this example removes, or cancels, delayed reconfiguration operations for a logical domain.

```
ldm rm-reconf ldom
```

INPUT/OUTPUT DEVICES

Add Input/Output Device

This subcommand in this example adds a PCI bus to a specified logical domain.

```
ldm add-io [bypass=on] bus ldom
```

Where:

- `bypass=on` option turns on the I/O MMU bypass mode. This bypass mode should be enabled only if the respective I/O domain and I/O devices within that I/O domain are trusted by all guest domains.

Caution – By default, Logical Domains software controls PCI-E transactions so that a given I/O device or PCI-E option can only access the physical memory assigned within the I/O domain. Any attempt to access memory of another guest domain is prevented by the I/O MMU. This provides a higher level of security between the I/O domain and all other domains. However, in the rare case where a PCI-E or PCI-X option card does not load or operate with the I/O MMU bypass mode off, this option allows you to turn the I/O MMU bypass mode on. However, if you turn the bypass mode on, there no longer is a hardware-enforced protection of memory accesses from the I/O domain.

- *bus* is the requested PCI bus; for example, `pci@780` or `pci@7c0`.
- *ldom* specifies the logical domain where the PCI bus is to be added.

Remove Input/Output Device

This subcommand in this example removes a PCI bus from a specified logical domain.

```
ldm rm-io bus ldom
```

Where:

- *bus* is the requested PCI bus; for example, `pci@780` or `pci@7c0`.
- *ldom* specifies the logical domain where the PCI bus is to be removed.

VIRTUAL NETWORK - SERVICE

Add a Virtual Switch

This subcommand adds a virtual switch to a specified logical domain.

```
ldm add-vsw [mac-addr=num] [net-dev=device] vswitch_name ldom
```

Where:

- *num* is the MAC address to be used by this switch. The number must be in standard octet notation; for example, 80:00:33:55:22:66. If you do not specify a MAC address, the switch is automatically assigned an address from the range of public MAC addresses allocated to the Logical Domains Manager.
- *device* is the path to the network device over which this switch operates.
- *vswitch_name* is the unique name of the switch that is to be exported as a service. Clients (network) can attach to this service.
- *ldom* specifies the logical domain in which to add a virtual switch.

Set a Virtual Switch

This subcommand modifies the properties of a virtual switch that has already been added.

```
ldm set-vsw [mac-addr=num] [net-dev=device] vswitch_name
```

Where:

- num is the MAC address used by the switch. The number must be in standard octet notation; for example, 80:00:33:55:22:66.
- *device* is the path to the network device over which this switch operates.
- *vswitch_name* is the unique name of the switch that is to exported as a service. Clients (network) can be attached to this service.

Remove a Virtual Switch

This subcommand removes a virtual switch.

```
ldm rm-vsw [-f] vswitch_name
```

Where:

- `-f` attempts to force the removal of a virtual switch. The removal might fail.
- *vswitch_name* is the name of the switch that is to be removed as a service.

VIRTUAL NETWORK - CLIENT

Add a Virtual Network Device

This subcommand adds a virtual network device to the specified logical domain.

```
ldm add-vnet [mac-addr=num] if_name vswitch_name ldom
```

Where:

- num is the MAC address for this network device. The number must be in standard octet notation; for example, 80:00:33:55:22:66.

- *if_name*, interface name, is a unique name to the logical domain, assigned to this virtual network device instance for reference on subsequent `set-vnet` or `rm-vnet` subcommands.
- *vswitch_name* is the name of an existing network service (virtual switch) to which to connect.
- *ldom* specifies the logical domain to which to add the virtual network device.

Set a Virtual Network Device

This subcommand sets a virtual network device in the specified logical domain.

```
ldm set-vnet [mac-addr=num] [vswitch_name] if_name ldom
```

Where:

- num is the MAC address for this network device. The number must be in standard octet notation; for example, 80:00:33:55:22:66.
- *vswitch_name* is the name of an existing network service (virtual switch) to which the network device is connected.
- *if_name*, interface name, is the unique name assigned to the virtual network device you want to set.
- *ldom* specifies the logical domain in which to modify the virtual network device.

Remove a Virtual Network Device

This subcommand removes a virtual network device from the specified logical domain.

```
ldm rm-vnet [-f] if_name ldom
```

Where:

- `-f` attempts to force the removal of a virtual network device from a logical domain. The removal might fail.
- *if_name*, interface name, is the unique name assigned to the virtual network device you want to remove.
- *ldom* specifies the logical domain from which to remove the virtual network device.

VIRTUAL DISK - SERVICE

Add a Virtual Disk Server

This subcommand adds a virtual disk server to the specified logical domain.

```
ldm add-vds service_name ldom
```

Where:

- *service_name* is the service name for this instance of the virtual disk server. The *service_name* must be unique among all virtual disk server instances on the server.
- *ldom* specifies the logical domain in which to add the virtual disk server.

Remove a Virtual Disk Server

This subcommand removes a virtual disk server.

```
ldm rm-vds [-f] service_name
```

Where:

- `-f` attempts to force the removal of a virtual disk server. The removal might fail.
- *service_name* is the unique service name for this instance of the virtual disk server.

Caution – The `-f` option attempts to unbind all clients before removal, and could cause loss of disk data if writes are in progress.

Add a Device to a Virtual Disk Server

This subcommand adds a device to a virtual disk server. The device can be an entire disk, a slice on a disk, a file, or a disk volume.

```
ldm add-vdsdev file|device  volume_name@service_name
```

Where:

- *file* | *device* is the path name of either the actual physical device or the actual file exported as a block device. When adding a device, the *volume_name* must be paired with the *file* | *device*.
- *volume_name* is a unique name you must specify for the device being added to the virtual disk server. The *volume_name* must be unique for this virtual disk server instance, because this name is exported by this virtual disk server to the clients for adding. When adding a device, the *volume_name* must be paired with the *file* | *device*.
- *server_name* is the name of the virtual disk server to which to add this device.

Remove a Device From a Virtual Disk Server

This subcommand removes a device from a virtual disk server.

```
ldm rm-vdsdev [-f] volume_name@vds_name
```

Where:

- `-f` attempts to force the removal of the virtual disk server device. The removal might fail.
- *volume_name* is the unique name for the device being removed from the virtual disk server.
- *vds_name* is the name of the virtual disk server from which to remove this device.

Caution – Without the `-f` option, the `rm-vdsdev` subcommand does not allow a virtual disk server device to be removed if the device is busy. Using the `-f` option can cause data loss for open files.

VIRTUAL DISK - CLIENT

Add a Virtual Disk

This subcommand adds a virtual disk to the specified logical domain.

```
ldm add-vdisk disk_name volume_name@service_name ldom
```

Where:

- *disk-name* is the name of the virtual disk.
- *volume_name* is the name of the existing virtual disk server device to which to connect.
- *service_name* is the name of the existing virtual disk server to which to connect.
- *ldom* specifies the logical domain in which to add the virtual disk.

Remove a Virtual Disk

This subcommand removes a virtual disk from the specified logical domain.

```
ldm rm-vdisk [-f] disk_name ldom
```

Where:

- `-f` attempts to force the removal of the virtual disk. The removal might fail.
- *disk_name* is the name of the virtual disk to be removed.
- *ldom* specifies the logical domain from which to remove the virtual disk.

VIRTUAL DATA PLANE CHANNEL - SERVICE

Add a Virtual Data Plane Channel Service

This subcommand adds a virtual data plane channel service to the specified logical domain. This subcommand should only be used in a Netra Data Plane Software (NDPS) environment.

```
ldm add-vdpcs vdpcs_service_name ldom
```

Where:

- *vdpcs_service_name* is the name of the virtual data plane channel service that is to be added.
- *ldom* specifies the logical domain to which to add the virtual data plane channel service.

Remove a Virtual Data Plane Channel Service

This subcommand removes a virtual data plane channel service. This subcommand should only be used in a Netra Data Plane Software (NDPS) environment.

```
ldm rm-vdpcs vdpcs_service_name
```

Where:

- *vdpcs_service_name* is the name of the virtual data plane channel service that is to be removed.

VIRTUAL DATA PLANE CHANNEL - CLIENT

Add a Virtual Data Plane Channel Client

This subcommand adds a virtual data plane channel client to the specified logical domain. This subcommand should only be used in a Netra Data Plane Software (NDPS) environment.

```
ldm add-vdpcc vdpcc_name vdpcs_service_name ldom
```

Where:

- *vdpcc_name* is the unique name of the virtual data plane channel service client.
- *vdpcs_service_name* is the name of the virtual data plane channel service to which to connect this client.
- *ldom* specifies the logical domain to which to add the virtual data plane channel client.

Remove a Virtual Data Plane Channel Client

This subcommand removes a virtual data plane channel client from the specified logical domain. This subcommand should only be used in a Netra Data Plane Software (NDPS) environment.

```
ldm rm-vdpcc vdpcc_name ldom
```

Where:

- *vdpcc_name* is the unique name assigned to the virtual data plane channel client that is to be removed.
- *ldom* specifies the logical domain from which to remove the virtual data plane channel client.

VIRTUAL CONSOLE

Add a Virtual Console Concentrator

This subcommand adds a virtual console concentrator to the specified logical domain.

```
ldm add-vcc port-range=x-y vcc_name ldom
```

Where:

- *x-y* is the range of TCP ports to be used by the virtual console concentrator for console connections.
- *vcc_name* is the name of the virtual console concentrator that is to be added.
- *ldom* specifies the logical domain to which to add the virtual console concentrator.

Set a Virtual Console Concentrator

This subcommand sets a specific virtual console concentrator.

```
ldm set-vcc port-range=x-y vcc_name
```

Where:

- *x-y* is the range of TCP ports to be used by the virtual console concentrator for console connections. Any modified port range must be a superset of the previous range.
- *vcc_name* is the name of the virtual console concentrator that is to be set.

Remove a Virtual Console Concentrator

This subcommand removes a virtual console concentrator from the specified logical domain.

```
ldm rm-vcc [-f] vcc_name
```

Where:

- `-f` attempts to force the removal of the virtual console concentrator. The removal might fail.
- *vcc_name* is the name of the virtual console concentrator that is to be removed.

Caution – The `-f` option attempts to unbind all clients before removal, and could cause loss of data if writes are in progress.

Set a Virtual Console

This subcommand sets the attached console's service or group in the specified logical domain.

```
ldm set-vcons [group@]vcc_name ldom
```

Where:

- *group* is the new group to which to attach this console. The group argument allows multiple consoles to be multiplexed onto the same TCP connection. Refer to the Solaris OS `vntsd`(1M) man page for more information about this concept.
- *vcc_name* is the name you specify for the new existing virtual console concentrator to handle the console connection.
- *ldom* specifies the logical domain in which to set the virtual console concentrator.

VARIABLES

Add Variable

This subcommand adds a variable for a logical domain.

```
ldm add-var var_name=value ldom
```

Where:

- *var_name=value* is the name and value pair of the variable to add.
- *ldom* specifies the logical domain in which to add the variable.

Set Variable

This subcommand sets a variable for a logical domain.

```
ldm set-var var_name=value ldom
```

Where:

- *var_name=value* is the name and value pair of the variable to set.
- *ldom* specifies the logical domain in which to set the variable.

Note – Leaving *var_name* blank, sets *var_name* to `NULL`.

Remove Variable

This subcommand removes a variable for a logical domain.

```
ldm rm-var var_name ldom
```

Where:

- *var_name* is the name of the variable to remove.
- *ldom* specifies the logical domain from which to remove the variable.

OPERATIONS

Start Logical Domains

This subcommand starts one or more logical domains.

```
ldm start-dom (-a | -i file | ldom...)
```

Where:

- `-a` means start all bound logical domains.
- `-i` *file* specifies an XML configuration file to use in starting the logical domain.
- *ldom*... specifies one or more logical domains to start.

Stop Logical Domains

This subcommand stops one or more running logical domains.

```
ldm stop-dom [-f] (-a | ldom...)
```

Where:

- `-f` option attempts to force a running logical domain to stop.
- `-a` option means stop all running logical domains except the control domain.
- *ldom*... specifies one or more running logical domains to stop.

Provide Help Information

This subcommand provides usage for all subcommands or the subcommand that you specify. You can also use the `ldm` command alone to provide usage for all subcommands.

```
ldm --help [subcommand]
```

Provide Version Information

This subcommand provides version information.

```
ldm (--version | -V)
```

Bind Resources to a Logical Domain

This subcommand binds, or attaches, configured resources to a logical domain.

```
ldm bind-dom (-i file | ldom)
```

Where:

- `-i` *file* specifies an XML configuration file to use in binding the logical domain.
- *ldom* specifies the logical domain to which to bind resources.

Unbind Resources From a Logical Domain

This subcommand releases resources bound to configured logical domains.

```
ldm unbind-dom ldom
```

Where:

- *ldom* specifies the logical domain from which to unbind resources.

LOGICAL DOMAIN CONFIGURATIONS

Add Logical Domain Configuration

This subcommand adds a logical domain configuration. The configuration is stored on the system controller (SC).

```
ldm add-config config_name
```

Where:

- *config_name* is the name of the logical domain configuration to add.

Set Logical Domain Configuration

This subcommand enables you to specify a logical domain configuration to use. The configuration is stored on the system controller (SC).

```
ldm set-config config_name
```

Where:

- *config_name* is the name of the logical domain configuration to use.

The default configuration name is `factory-default`. To specify the default configuration, use the following:

```
ldm set-config factory-default
```

Remove Logical Domain Configuration

This subcommand removes a logical domain configuration. The configuration is stored on the system controller (SC).

```
ldm rm-config config_name
```

Where:

- *config_name* is the name of the logical domain configuration to remove.

LISTS

List Logical Domains and States

This subcommand lists logical domains and their states. If you do not specify a logical domain, all logical domains are listed.

```
ldm ls-dom [-l] [-p] [ldom...]
```

Where:

- `-l` means to generate a long listing.
- `-p` means generate the list in a parseable, machine-readable format.
- *ldom*... is the name one or more logical domains for which to list state information.

List Bindings for Logical Domains

This subcommand lists bindings for logical domains. If no logical domains are specified, all logical domains are listed.

```
ldm ls-bindings [ldom...]
```

Where:

- *ldom*... is the name of one or more logical domains for which you want binding information.

List Services for Logical Domains

This subcommand lists all the services exported by logical domains. If no logical domains are specified, all logical domains are listed.

```
ldm ls-services [ldom...]
```

Where:

- *ldom*... is the name of one or more logical domains for which you want services information.

List Constraints for Logical Domains

This subcommand lists the constraints for the creation of one or more logical domains. If the `-x` option is specified, only the one specified logical domain is listed. If you specific nothing after the subcommand, all logical domains are listed.

```
ldm ls-constraints (-x ldom) | ([-p] [ldom...])
```

Where:

- `-x` means write the constraint output in XML format to the named logical domain. This can be used as a backup.
- `-p` means write the constraint output in a parseable, machine-readable form.

- *ldom*... is the name of one or more logical domains for which you want to list constraints.

List Devices

This subcommand lists free (unbound) resources or all server resources. The default is to list all free resources.

```
ldm ls-devices [-a] [cpu| mau| memory| io]
```

Where:

- `-a` lists all server resources, bound and unbound.
- `cpu` lists only CPU resources.
- `mau` lists only the modular arithmetic unit resources.
- `memory` lists only memory resources.
- `io` lists only input/output resources, such as a PCI bus or a network.

List Logical Domain Configurations

This subcommand lists the logical domain configurations stored on the system controller.

```
ldm ls-config
```

List Variables

This subcommand lists one or more variables for a logical domain.

```
ldm ls-var var_name... [ldom...]
```

Where:

- *var_name*... is the name of one or more variables to list.
- *ldom*... is the name of one or more logical domains for which to list one or more variables.

EXAMPLES

EXAMPLE 1 Create Default Services

Set up the three default services, virtual disk server, virtual switch, and virtual console concentrator, so that you can export those services to the guest domains.

```
# ldm add-vds primary-vds0 primary
# ldm add-vsw net-dev=e1000g0 primary-vsw0 primary
# ldm add-vcc port-range=5000-5100 primary-vcc0 primary
```

EXAMPLE 2 List Services

You can list services to ensure they have been created correctly or to see what services you have available.

```
#ldm ls-services primary
...
Vds:    primary-vds0
Vcc:    primary-vcc0
                port-range=5000-5100
Vsw:    primary-vsw0
                mac-addr=0:14:4f:f9:68:d0
                net-dev=e1000g0
                mode=prog,promisc
```

EXAMPLE 3 Set Up the Control Domain Initially

The control domain, named `primary`, is the initial domain that is present when you install the Logical Domains Manager. The control domain has a full complement of resources, and those resources depend on what server you have. Set only those resources you want the control domain to keep, so that you can allocate the remaining resources to the guest domains. Then you can save the configuration on the system controller.

You must reboot so the changes take place. Until this first reboot, the Logical Domains Manager is running in configuration mode. See CPU AND MEMORY (Reconfiguration Operations) for more details about the configuration mode.

If you want to enable networking between the control domain and the other domains, you must plumb the virtual switch on the control domain. You must enable the virtual network terminal server daemon, `vntsd`(1M), to use consoles on the guest domains.

```
# ldm set-mau 1 primary
# ldm set-vcpu 4 primary
# ldm set-mem 1G primary
# ldm add-config initial
# shutdown -y -g0 -i6
# ifconfig -a
# ifconfig vsw0 plumb
# ifconfig e1000g0 down unplumb
# ifconfig vsw0 IP_of_e1000g0 netmask netmask_of_e1000g0 broadcast + up
# svcadm enable vntsd
```

EXAMPLE 4 List Bindings

You can list bindings to see if the control domain has the resources you specified, or what resources are bound to any domain.

```
# ldm ls-bindings primary
------------------------------------------------------------------
Name:   primary
State:  active
Flags:  transition,control,vio service
OS:
Util:   12%
Uptime: 11m
Vcpu:   4
        vid    pid    util strand
        0      0       18%   100%
        1      1       13%   100%
        2      2      9.8%   100%
        3      3      5.4%   100%
Mau:    1
Memory: 4G
        real-addr        phys-addr        size
        0x4000000        0x4000000        4G
Vars:   reboot-command=boot
IO:     pci@780 (bus_a)
        pci@7c0 (bus_b)
......
```

EXAMPLE 5 Create a Logical Domain

Ensure you have the resources to create the desired guest domain configuration, add the guest domain, add the resources and devices you want the domain to have, set boot parameters to tell the system how to behave on startup, bind the resources to the domain, and save the guest domain configuration in an XML file for backup. You also might want to save the primary and guest domain configurations on the SC. Then you can start the domain, find the TCP port of the domain, and connect to it through the default virtual console service.

```
# ldm ls-devices
# ldm add-dom ldg1
# ldm add-vcpu 4 ldg1
# ldm add-mem 512m ldg1
# ldm add-vnet vnet1 primary-vsw0 ldg1
# ldm add-vdsdev /dev/dsk/c0t1d0s2 vol1@primary-vds0
# ldm add-vdisk vdisk1 vol1@primary-vds0 ldg1
# ldm set-var auto-boot\?=false ldg1
# ldm set-var boot-device=vdisk ldg1
# ldm bind-dom ldg1
# ldm ls-constraints -x ldg1 > ldg1.xml
# ldm add-config ldg1_4cpu_512M
# ldm start-dom ldg1
# ldm ls -l ldg1
# telnet localhost 5000
```

EXAMPLE 6 Use One Terminal for Many Guest Domains

Normally, each guest domain you create has its own TCP port and console. Once you have created the first guest domain (`ldg1` in this example) and bound it, then you can use the `ldm set-vcons` command after you bind each of the other guest domains (second domain is `ldg2` in this example) to attach all the domains to the same console port.

```
# ldm set-vcons ldg1@primary-vcc0 ldg2
```

If you do the `ldm ls -l` command after performing the `set-vcons` commands on all guest domains but the first, you can see that all domains are connected to the same port. Refer to the Solaris 10 OS `vntsd`(1M) man page for more information about using consoles.

EXAMPLE 7 Add a Virtual PCI Bus to a Logical Domain

I/O domains are a type of service domain that have direct ownership of and direct access to physical I/O devices. The I/O domain then provides the service to the guest domain in the form of a virtual I/O device. This example shows how to add a virtual PCI bus to a logical domain.

```
# ldm add-io pci@7c0 ldg1
```

EXAMPLE 8 Add Virtual Data Plane Channel Functionality for Netra Only

If your server has a Netra Data Plane Software (NDPS) environment, you might want to add virtual data plane channel functionality. First, you would add a virtual data plane channel service (`primary-vdpcs0` for example) to the service domain; in this case, the `primary` domain.

```
# ldm add-vdpcs primary-vdpcs0 primary
```

Now that you have added the service to the service domain (`primary`), you can add the virtual data plane channel client (`vdpcc1`) to a guest domain (`ldg1`).

```
# add-vdpcc vdpcc1 primary-vdpcs0 ldg1
```

EXAMPLE 9 Cancel Delayed Reconfiguration Operations for a Domain

A delayed reconfiguration operation blocks configuration operations on all other domains. There might be times when you want to cancel delayed configuration operations for one domain; for example, so you can perform other configuration commands on that domain or other domains. For another example, you might have attempted to add some memory to a domain (`ldg1`) and the Logical Domains Manager invoked delayed reconfiguration because the domain was not stopped. With this command, you can undo the delayed reconfiguration operation, stop the domain, and add memory again.

```
# ldm rm-reconf ldg1
```

ATTRIBUTES

Refer to the Solaris OS `attributes`(5) man page for a description of the following attributes:

Attribute Types	Attribute Values
Availability	`SUNWldm`
Interface Stability	Uncommitted

REFER ALSO

Refer also to the Solaris OS `vntsd`(1M) man page, the *Beginners Guide for LDoms: Understanding and Deploying Logical Domains*, and the *Logical Domains (LDoms) 1.0 Administration Guide*.

Glossary

This list defines terminology, abbreviations, and acronyms in the Logical Domains 1.0 documentation.

A

ALOM CMT	Advanced Lights Out Manager Chip Multithreading, which runs on the system controller and allows you to monitor and control your CMT server.

B

`bge`	Broadcom Gigabit Ethernet driver on Broadcom BCM57*xx* devices
BSM	Basic Security Module

C

CLI	command-line interface
`config`	name of logical domain configuration saved on the system controller
CMT	Chip Multithreading

constraints	To the Logical Domains Manager, constraints are one or more resources you want to have assigned to a particular domain. You either receive all the resources you ask to be added to a domain or you get none of them, depending upon the available resources.
control domain	domain that creates and manages other logical domains and services
CPU	central processing unit

D

DHCP	Dynamic Host Configuration Protocol
DMP	Dynamic Multipathing (Veritas)
DR	dynamic reconfiguration
`drd`**(1M)**	dynamic reconfiguration daemon for Logical Domains Manager (Solaris 10 OS)

E

`e1000g`	driver for Intel PRO/1000 Gigabit family of network interface controllers
EFI	Extensible Firmware Interface

F

FMA	Fault Management Architecture
`fmd`**(1M)**	fault management daemon (Solaris 10 OS)
FTP	File Transfer Protocol

G

guest domain	Uses services from the I/O and service domains and is managed by the control domain.
GLDv3	Generic LAN Driver version 3.

H

HDD	hard disk drive
hypervisor	firmware layer interposed between the operating system and the hardware layer

I

`io`	I/O devices, such as internal disks and PCI-Express (PCI-E) controllers and their attached adapters and devices
I/O domain	domain that has direct ownership of and direct access to physical I/O devices and that shares those devices to other logical domains in the form of virtual devices.
`ioctl`	input/output control call
IP	Internet Protocol
IPMP	Internet Protocol Network Multipathing

K

kaio	kernel asynchronous input/output
KB	kilobyte
KU	kernel update

L

LAN	local-area network
LDAP	Lightweight Directory Access Protocol
LDC	logical domain channel
`ldm`**(1M)**	Logical Domain Manager utility
`ldmd`	Logical Domains Manager daemon
logical domain	discrete logical grouping with its own operating system, resources, and identity within a single computer system
Logical Domains (LDoms) Manager	provides a CLI to create and manage logical domains and allocate resources to domains

M

MAC	media access control address, which LDoms can automatically assign or you can assign manually
`mau`	modular arithmetic unit, the cryptographic device for supported servers
MB	megabyte
MD	machine description in the server database
`mem`**,** `memory`	memory unit - default size in bytes, or specify gigabytes (`G`), kilobytes (`K`), or megabytes (`M`). Virtualized memory of the server that can be allocated to guest domains.
MMF	multimode fiber
MMU	memory management unit
mtu	maximum transmission unit

N

NAT	Network Address Translation
NDPS	Netra Data Plane Software
`ndpsldcc`	Netra Data Plane Software Logical Domain Channel Client. *See also* `vdpcc`.
`ndpsldcs`	Netra Data Plane Software Logical Domain Channel Service. *See also* `vdpcs`.
NFS	Network File System
NIS	Network Information Services
NTS	network terminal server
NVRAM	non-volatile random-access memory
`nxge`	driver for Sun x8 Express 1/10G Ethernet Adapter

O

OS	operating system

P

PCI	peripheral component interconnect bus
PCI-E	PCI Express bus
PCI-X	PCI Extended bus

R

RAID	Redundant Array of Inexpensive Disks
RBAC	Role-Based Access Control

RPC	Remote Procedure Call

S

SC	system controller, same as system processor
SCSI	Small Computer System Interface
service domain	logical domain that provides devices, such as virtual switches, virtual console connectors, and virtual disk servers to other logical domains
SMA	System Management Agent
SMF	Service Management Facility of Solaris 10 OS
SNMP	Simple Network Management Protocol
SP	system processor, same as system controller
SSH	Secure Shell
`ssh`**(1)**	Secure Shell command
`sshd`**(1M)**	Secure Shell daemon
SunVTS	Sun Validation Test Suite
SVM	Solaris Volume Manager

T

TCP	Transmission Control Protocol

U

UDP	User Diagram Protocol
USB	Universal Serial Bus

V

vBSC	Virtual Blade System Controller
`vcc`, `vconscon`	virtual console concentrator service with a specific port range to assign to the guest domains
`vcons`, `vconsole`	virtual console for accessing system level messages. A connection is achieved by connecting to `vconscon` service in the control domain at a specific port.
`vcpu`	virtual central processing unit. Each of the cores of a server are represented as virtual CPUs. For example, an 8-core Sun Fire T2000 Server has 32 virtual CPUs that can be allocated between the logical domains.
`vdpcc`	virtual data plane channel client in an NDPS environment
`vdpcs`	virtual data plane channel service in an NDPS environment
`vdisk`	virtual disks are generic block devices backed by different types of physical devices, volumes, or files.
`vds, vdiskserver`	virtual disk server allows you to import virtual disks into a logical domain.

`vdsdev,`
`vdiskserverdevice`
	virtual disk server device is exported by the virtual disk server. The device can be an entire disk, a slice on a disk, a file, or a disk volume.

`vnet`	virtual network device implements a virtual Ethernet device and communicates with other `vnet` devices in the system using the virtual network switch (`vswitch`).
`vntsd`**(1M)**	virtual network terminal server daemon for Logical Domains consoles (Solaris 10 OS)
`vsw`, `vswitch`	virtual network switch that connects the virtual network devices to the external network and also switches packets between them.
VTOC	volume table of contents
VxVM	Veritas Volume Manager

W

WAN	wide-area network

X

XML	Extensible Markup Language

Z

ZFS	Zettabyte File System (Solaris 10 OS)
`zpool`	ZFS storage pool