Logical Domains (LDoms) Manager 1.0.3 Man Page Guide

Sun Microsystems, Inc.
www.sun.com

Part No. 820-4896-10
May 2008, Revision A

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

Adobe PostScript™

Contents

Preface

Both novice users and those familiar with the SunOS operating system can use online man pages to obtain information about the system and its features. A man page is intended to answer concisely the question “What does it do?” In general, man pages comprise a reference manual. They are not intended to be a tutorial.

Overview

The following contains a brief description of each section in the man pages and the information it references:

- Section 1M describes, in alphabetical order, commands that are used chiefly for system maintenance and administration purposes.

Below is a generic format for man pages. The man pages of each manual section generally follow this order, but include only needed headings. See the `intro` pages for more information and detail about each section, and `man`(1) for more information about man pages in general.

NAME	This section gives the names of the commands or functions documented, followed by a brief description of what they do.
SYNOPSIS	This section shows the syntax of commands or functions. When a command or file does not exist in the standard path, its full path name is shown. Options and arguments are alphabetized, with single letter arguments first, and options with arguments next, unless a different argument order is required.

The following special characters are used in this section:

`[ ]`	Brackets. The option or argument enclosed in these brackets is optional. If the brackets are omitted, the argument must be specified.
…	Ellipses. Several values may be provided for the previous argument, or the previous argument can be specified multiple times, for example “filename...”.
|	Separator. Only one of the arguments separated by this character can be specified at one time.
`{ }`	Braces. The options and/or arguments enclosed within braces are interdependent, such that everything enclosed must be treated as a unit.

DESCRIPTION	This section defines the functionality and behavior of the service. Thus it describes concisely what the command does. It does not discuss OPTIONS or cite EXAMPLES. Interactive commands, subcommands, requests, macros, functions and such, are described under USAGE.
OPTIONS	This lists the command options with a concise summary of what each option does. The options are listed literally and in the order they appear in the SYNOPSIS section. Possible arguments to options are discussed under the option, and where appropriate, default values are supplied.
OPERANDS	This section lists the command operands and describes how they affect the actions of the command.
USAGE	This section lists special rules, features and commands that require in-depth explanations. The subsections listed below are used to explain built-in functionality: Commands Modifiers Variables Expressions Input Grammar

EXAMPLES	This section provides examples of usage or of how to use a command or function. Wherever possible a complete example including command line entry and machine response is shown. Whenever an example is given, the prompt is shown as `example%` or if the user must be superuser, `example#`. Examples are followed by explanations, variable substitution rules, or returned values. Most examples illustrate concepts from the SYNOPSIS, DESCRIPTION, OPTIONS and USAGE sections.
ATTRIBUTES	This section lists characteristics of commands, utilities, and device drivers by defining the attribute type and its corresponding value. See `attributes`(5) for more information.
SEE ALSO	This section lists references to other man pages, in-house documentation and outside publications.
NOTES	This section lists additional information that does not belong anywhere else on the page. It takes the form of an aside to the user, covering points of special interest. Critical information is never covered here.

NAME

Intro - Logical Domains (LDoms) Manager

DESCRIPTION

This guide contains reference information about the `ldm` command and its subcommands.

LIST OF COMMANDS

The following commands are supported:

ldm	Logical Domains (LDoms) Manager command-line interface (CLI)

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* | **mac-addr=***num* *ldom* | *ldom***...)**

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

ldm list-domain [-e] [-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-vsw [mac-addr=*num***] [net-dev=***device***] [mode=[sc]]** *vswitch_name ldom*

ldm set-vsw [mac-addr=*num***] [net-dev=***device***] [mode=[sc]]** *vswitch_name*

ldm remove-vsw [-f] *vswitch_name*

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

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

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

ldm add-vds *service_name ldom*

ldm remove-vds [-f] *service_name*

ldm add-vdsdev [options={ro,slice,excl}] *backend volume_name***@***service_name*

```
ldm set-vdsdev options=[{ro,slice,excl}]  volume_name@service_name

ldm remove-vdsdev  [-f]  volume_name@service_name

ldm add-vdisk  [timeout=seconds]  disk_name  volume_name@service_name  ldom

ldm set-vdisk [timeout=seconds]  [volume=volume_name@service_name]  disk_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-vcc  port-range=x-y  vcc_name  ldom

ldm set-vcc  port-range=x-y  vcc_name

ldm remove-vcc  [-f]  vcc_name

ldm set-vcons  [port=[port_num]]  [group=group]  [service=vcc_server]  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 panic-domain ldom

ldm bind-domain  (-i  file  |  ldom)

ldm unbind-domain  ldom

ldm list-bindings  [-e]  [-p]  [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] | [-e] [-p]) [*ldom*...]

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

ldm list-services [-e] [-p] [*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 when the I/O domain is also the control domain. The number of I/O domains you can have is dependent on your platform architecture. For example, if you are using a Sun UltraSPARC(R) T1 processor, 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) 1.0.3 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`
`panic-domain`	Panics the Solaris OS on a specified logical domain.	`solaris.ldoms.write`
`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

There are three kinds of aliases for ldm subcommands, each of which is listed in the tables following:

- Action aliases (verbs)
- Resource aliases (nouns)
- Subcommand shortcuts

TABLE 1 Action Aliases (Verbs)

Short Form	Long Form
`ls`	`list`
`rm`	`remove`

TABLE 2 Resource Aliases (Nouns)

Short Form	Long Form
`config`	`spconfig`
`dom`	`domain`
`mau`	`crypto`
`mem`	`memory`
`var`	`variable`
`vcc`	`vconscon`
`vcons`	`vconsole`
`vdpcc`	`ndpsldcc`
`vdpcs`	`ndpsldcs`
`vds`	`vdiskserver`
`vdsdev`	`vdiskserverdevice`
`vsw`	`vswitch`

TABLE 3 Subcommand Shortcuts

Short Form	Long Form
bind	bind-domain
cancel-reconf	remove-reconf
create	add-domain
destroy	remove-domain
panic	panic-domain
start	start-domain
stop	stop-domain
unbind	unbind-domain

Note – In the syntax and examples in the remainder of this man page, the short forms of the action and resource aliases 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`, `crypto`	Any LDoms-supported cryptographic unit on an LDoms-supported server. Currently, the two cryptographic units supported are the Modular Arithmetic Unit (MAU) and the Control Word Queue (CWQ).
`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.
`-e`	`--extended`	Generates an extended listing containing services and devices that are automatically set up; that is, not under your control.
`-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:

`bypass=on`	Turns on the I/O MMU bypass mode. Enable this bypass mode only if the respective I/O domain and I/O devices within that I/O domain are trusted by all guest domains.
`group=`	Specifies a group to which to attach a console. The group argument allows multiple consoles to be multiplexed onto the same TCP connection.
`mac-addr=`	Defines a MAC address. The number must be in standard octet notation; for example, 80:00:33:55:22:66.
`mode=`	Specify `mode=sc` to enable virtual networking support for prioritized processing of Solaris™ Cluster heartbeat packets in a Logical Domains environment. Omit this option when you are not running Solaris Cluster software in guest domains, because you could impact virtual network performance. Leave the `mode=` argument blank in the `set-vsw` subcommand to stop special processing of heartbeat packets.
`net-dev=`	Defines the path name of the actual network device.
`options=`	Specifies all or a subset of the following options for a specific virtual disk server device. Separate two or more options with commas and no spaces, such as `ro,slice,excl`. ■ `ro` - specifies read-only access ■ `slice` - exports a backend as a single slice disk ■ `excl` - specifies exclusive disk access Omit the `options=` argument or leave it blank in an `add-vdsdev` subcommand to have the default values of disk, not exclusive, and read/write. Leave the options= argument blank in the `set-vdsdev` subcommand to turn off any previous options specified.
`port=`	Specifies a specific port number or, left blank, lets the Logical Domains Manager set the port number.
`port-range=`	Defines a range of TCP ports.
`timeout=`	Defines the number of seconds a virtual disk should wait before timing out and sending an error message while trying to connect to a virtual disk server. Specify `0` to disable the timeout in the `set-vdisk` subcommand.

`service=`	Specifies the name of the existing virtual console concentrator that you want to handle the console connection.
`volume=`	Changes a volume name for a virtual disk.
`vswitch=`	Changes a virtual switch name for a virtual network.

FLAGS IN `list` SUBCOMMAND OUTPUT

Following are definitions of the flags in the `list` subcommand output:

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

The list flag values are position dependent. Following are the values that can appear in each of the five columns from left to right:

TABLE 4 List Flag Positions

Column 1	Column 2	Column 3	Column 4	Column 5
`s` or `-`	`n` or `t`	`d` or `-`	`c` or `-`	`v` or `-`

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 Domains

This subcommand adds one or more logical domain by specifying one or more logical domain names or by using an XML configuration file. You can also specify a MAC address for the domain, or let the Logical Domains Manager automatically assign the MAC address.

```
ldm add-dom (-i file | mac-addr=num ldom | 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 you specify a MAC address, you only can specify one logical domain.
- `-i` *file* specifies the XML configuration file to use in creating the logical domain.
- *ldom* specifies the name of the logical domain to be added for which you specified a MAC address.
- *ldom*... specifies one or more logical domains to be added.

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 you are using a Sun UltraSPARC T1 processor, and it 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 Cryptographic Units

This subcommand specifies the number of cryptographic units to be added to a logical domain. Currently, the LDoms-supported cryptographic units on LDom-supported servers are the Modular Arithmetic Unit (MAU) and the Control Word Queue (CWQ).

```
ldm add-mau number ldom
```

Where:

- *number* is the number of cryptographic units to be added to the logical domain.
- *ldom* specifies the logical domain where the cryptographic units are to be added.

Set Cryptographic Units

This subcommand specifies the number of cryptographic units to be set in the logical domain.

```
ldm set-mau number ldom
```

Where:

- *number* is the number of cryptographic units to be set in the logical domain.
- *ldom* specifies the logical domain where the number of cryptographic units are to be set.

Remove Cryptographic Units

This subcommand removes the specified number of cryptographic units from a logical domain.

```
ldm rm-mau number ldom
```

Where:

- *number* is the number of cryptographic units to be removed from the logical domain.
- *ldom* specifies the logical domain where the cryptographic 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. Enable this bypass mode 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] [mode=sc | ] 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.
- `mode=sc` enables virtual networking support for prioritized processing of Solaris Cluster heartbeat packets in a Logical Domains environment. Applications like Solaris Cluster need to ensure that high priority heartbeat packets are not dropped by congested virtual network and switch devices. This option prioritizes Solaris Cluster heartbeat frames and ensures that they are transferred in a reliable manner.

  mode= (left blank) stops special processing of heartbeat packets.

  You must set this option when running Solaris Cluster in a Logical Domains environment and using guest domains as Solaris Cluster nodes. Do not set this option when you are not running Solaris Cluster software in guest domains, because you could impact virtual network performance.
- *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 Options for 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] [mode=[sc]] 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.
- `mode=sc` enables virtual networking support for prioritized processing of Solaris Cluster heartbeat packets in a Logical Domains environment. Applications like Solaris Cluster need to ensure that high priority heartbeat packets are not dropped by congested virtual network and switch devices. This option prioritizes Solaris Cluster heartbeat frames and ensures that they are transferred in a reliable manner.

  mode= (left blank) stops special processing of heartbeat packets.

  You must set this option when running Solaris Cluster in a Logical Domains environment and using guest domains as Solaris Cluster nodes. Do not set this option when you are not running Solaris Cluster software in guest domains, because you could impact virtual network performance.
- *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 Options for a Virtual Network Device

This subcommand sets options for a virtual network device in the specified logical domain.

```
ldm set-vnet [mac-addr=num] [vswitch=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 to connect.
- *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. Refer to the *Logical Domains (LDoms) 1.0.3 Administration Guide* for more information.

```
ldm add-vdsdev [options={ro,slice,excl}] backend
volume_name@service_name
```

Where:

- `options=` are as follows:
  - `ro` - to specify read-only access
  - `slice` - to export a backend as a single slice disk
  - `excl` - to specify exclusive disk access

  Omit the `options=` argument to have the default values of disk, not exclusive, and read/write. If you add the options= argument, you must specify one or more of the options for a specific virtual disk server device. Separate two or more options with commas and no spaces, such as `ro,slice,excl`.
- *backend* is the location where data of a virtual disk are stored. The backend can be a disk, a disk slice, a file, a volume (including ZFS, SVM, or VxVM) or any disk pseudo device. The disk label can be SMI VTOC, EFI, or no label at all. A backend appears in a guest domain either as a full disk or as single slice disk, depending on whether the `slice` option is set when the backend is exported from the service domain. When adding a device, the *volume_name* must be paired with the *backend*.
- *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 *backend*.

- *service_name* is the name of the virtual disk server to which to add this device.

Set Options for a Virtual Disk Server

This subcommand sets options for a virtual disk server. Refer to the *Logical Domains (LDoms) 1.0.3 Administration Guide* for more information.

```
ldm set-vdsdev options=[{ro,slice,excl}] volume_name@service_name
```

Where:

- `options=` are as follows:
  - `ro` - to specify read-only access
  - `slice` - to export a backend as a single slice disk
  - `excl` - to specify exclusive disk access
- Leave the options= argument blank to turn off any previous options specified. You can specify all or a subset of the options for a specific virtual disk server device. Separate two or more options with commas and no spaces, such as `ro,slice,excl`.
- *volume_name* is the name of an existing volume exported by the service named by *service_name*.
- *service_name* is the name of the virtual disk server being modified.

Remove a Device From a Virtual Disk Server

This subcommand removes a device from a virtual disk server.

```
ldm rm-vdsdev [-f] volume_name@service_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.
- *service_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. An optional timeout property allows you to specify a timeout for a virtual disk if it cannot establish a connection with the vdisk server.

ldm add-vdisk [timeout=*seconds***]** *disk_name volume_name***@***service_name ldom*

Where:

- *seconds* is the number of seconds before the virtual disk times out when it cannot establish a connection to submit a request to the virtual disk server. If you specify a timeout period, the virtual disk returns an error when the timeout period expires.

  Omit the timeout= argument or set `timeout=0` to have the virtual disk wait indefinitely.
- *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.

Set Options for a Virtual Disk

This subcommand sets options for a virtual disk in the specified logical domain. An optional timeout property allows you to specify a timeout for a virtual disk if it cannot establish a connection with the vdisk server.

ldm set-vdisk [timeout=*seconds***] [volume=***volume_name***@***service_name***]**
disk_name ldom

Where:

- *seconds* is the number of seconds before the virtual disk times out when it cannot establish a connection to submit a request to the virtual disk server. If you specify a timeout period, the virtual disk returns an error when the timeout period expires.

  Set `timeout=0` to disable the timeout.

  Do not specify a `timeout=` argument to have the virtual disk wait indefinitely.
- *volume_name* is the name of the virtual disk server device to which to connect.
- *service_name* is the name of the virtual disk server to which to connect.
- *disk-name* is the name of the existing virtual disk.
- *ldom* specifies the existing logical domain where the virtual disk was previously added.

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 Options for a Virtual Console Concentrator

This subcommand sets options for 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 encompass all the ports assigned to clients of the concentrator.
- *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 Options for a Virtual Console

This subcommand sets a specific port number and group in the specified logical domain. You can also set the attached console's service. This subcommand can be used only when a domain is inactive.

```
ldm set-vcons [port=[port-num]] [group=group] [service=vcc_server]
ldom
```

Where:

- *port-num* is the specific port to use for this console. Leave the *port-num* blank to have the Logical Domains Manager automatically assign the port number.
- *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_server* is the name for the existing virtual console concentrator that should handle the console connection.
- *ldom* specifies the logical domain in which to set the virtual console concentrator.

VARIABLES

Add Variable

This subcommand adds one or more variables for a logical domain.

```
ldm add-var var_name[=value]... ldom
```

Where:

- *var_name=value*... is the name and value pair of one or more variables to add. The value is optional.
- *ldom* specifies the logical domain in which to add the variable.

Set Variable

This subcommand sets variables for a logical domain.

```
ldm set-var var_name[=value]... ldom
```

Where:

- *var_name=value*... is the name and value pair of one or more variables to set. The value is optional.
- *ldom* specifies the logical domain in which to set the variable.

Note – Leaving *value* 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 one or more variables 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. The subcommand sends a `shutdown`(1M) request to the logical domain if the Solaris OS is booted.

```
ldm stop-dom [-f] (-a | ldom...)
```

Where:

- `-f` option attempts to force a running logical domain to stop. Use only if the domain cannot be stopped by any other means.
- `-a` option means stop all running logical domains except the control domain.
- *ldom...* specifies one or more running logical domains to stop.

Panic Solaris OS

This subcommand panics the Solaris OS on a specified logical domain, which provides a back trace and crash dump if you configure the Solaris OS to do that. The `dumpadm`(1M) command provides the means to configure the crash dump.

```
ldm panic-domain ldom
```

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). There is a limit of 8 configurations that you can add to the SC, not including the `factory-default` configuration.

```
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 [-e] [-l] [-p] [ldom...]
```

Where:

- `-e` means to generate an extended listing containing services and devices that are automatically set up; that is, not under your control.

- `-l` means to generate a long listing.
- `-p` means generate the list in a parseable, machine-readable format.
- *ldom*... is the name of 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 [-e] [-p] [ldom...]
```

Where:

- `-e` means to generate an extended listing containing services and devices that are automatically set up; that is, not under your control.
- `-p` means generate the list in a parseable, machine-readable format.
- *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 [-e] [-p] [ldom...]
```

Where:

- `-e` means to generate an extended listing containing services and devices that are automatically set up; that is, not under your control.
- `-p` means generate the list in a parseable, machine-readable format.
- *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 you specific nothing after the subcommand, all logical domains are listed.

```
ldm ls-constraints ([-x] | [-e] [-p]) [ldom...]
```

Where:

- `-x` means write the constraint output in XML format to the standard output (`stdout`) format. This output can be used as a backup.
- `-e` means to generate an extended listing containing services and devices that are automatically set up; that is, not under your control.
- `-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] [-p] [cpu] [mau] [memory] [io]
```

Where:

- `-a` lists all server resources, bound and unbound.
- `-p` means write the constraint output in a parseable, machine-readable form.
- `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. To list all variables for a domain, leave the *var_name* blank.

```
ldm ls-var [var_name...] ldom
```

Where:

- *var_name*... is the name of one or more variables to list. If you do not specify any name here, all variables will be listed for the domain.
- *ldom* is the name of the logical domain 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
    NAME              VOLUME          OPTIONS           DEVICE
    primary-vds0
```

```
VCC
    NAME             PORT-RANGE
    primary-vcc0     5000-5100

VSW
    NAME           MAC                 NET-DEV   DEVICE     MODE
    primary-vsw0   00:14:4f:f9:68:d0   e1000g0   switch@0   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 4G 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             STATE    FLAGS    CONS     VCPU  MEMORY  UTIL  UPTIME
primary          active   -t-cv             4     4G      12%   11m

MAC
    08:00:90:11:11:10

VCPU
    VID     PID     UTIL STRAND
    0       0        18%   100%
    1       1        13%   100%
    2       2       9.8%   100%
```

```
        3      3     5.4%   100%

    MEMORY
        RA                PA              SIZE
        0x4000000         0x4000000       4G

    IO
    DEVICE            PSEUDONYM        OPTIONS
    pci@780           bus_a
    pci@7c0           bus_b            bypass=on

    VCC
        NAME              PORT-RANGE
        primary-vcc0      5000-5100

    VSW
        NAME            MAC                   NET-DEV   DEVICE     MODE
        primary-vsw0    00:14:4f:f9:68:d0     e1000g0   switch@0   prog,promisc

    VDS
        NAME              VOLUME          OPTIONS           DEVICE
        primary-vds0
```

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), then you can use the `ldm set-vcons` command to attach all the other domains (second domain is `ldg2` in this example) to the same console port. Note that the set-vcons subcommand works only on a inactive domain.

```
# ldm set-vcons group=ldg1 service=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 bypass=on 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 `dumpadm`(1M) man page, the Solaris OS `vntsd`(1M) man page, the *Beginners Guide for LDoms: Understanding and Deploying Logical Domains*, and the *Logical Domains (LDoms) 1.0.3 Administration Guide*.