Logical Domains 1.2 Reference Manual

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–7255–10
June 2009

090701@22510

Contents

REFERENCE

Logical Domains Commands

Name ldm– command-line interface for the Logical Domains Manager

Synopsis

```
ldm or ldm --help [subcommand]
ldm -V
ldm add-domain -i file
ldm add-domain [mac-addr=num] [hostid=num] [failure-policy=ignore|panic|reset|stop]
  [master=master-ldom1,...,master-ldom4] ldom
ldm add-domain ldom...
ldm set-domain -i file
ldm set-domain [failure-policy=ignore|panic|reset|stop]
  [master=[master-ldom1,...,master-ldom4]] ldom
ldm remove-domain -a
ldm remove-domain ldom...
ldm list-domain [-e] [-l] [-o format] [-p] [ldom...]
ldm migrate-domain [-n] source-ldom [user@]target-host[:target-ldom]
ldm add-vcpu number ldom
ldm set-vcpu number ldom
ldm remove-vcpu number ldom
ldm add-crypto number ldom
ldm set-crypto number ldom
ldm remove-crypto number ldom
ldm add-memory size [unit] ldom
ldm set-memory size [unit] ldom
ldm remove-memory size [unit] ldom
ldm cancel-operation (migration | reconf) ldom
ldm add-io [bypass=on] bus ldom
ldm remove-io bus ldom
ldm add-vsw [default-vlan-id=vlan-id] [pvid=port-vlan-id] [vid=vlan-id1,vlan-id2,...]
  [mac-addr=num] [net-dev=device] [mode=sc] [mtu=size] [id=switch-id] vswitch-name ldom
ldm set-vsw [pvid=port-vlan-id] [vid=vlan-id1,vlan-id2,...] [mac-addr=num] [net-dev=device]
  [mode=[sc]] [mtu=size] vswitch-name
ldm remove-vsw [-f] vswitch-name
ldm add-vnet [mac-addr=num] [mode=hybrid] [pvid=port-vlan-id] [vid=vlan-id1,vlan-id2,...]
  [id=network-id] [mtu=size] if-name vswitch-name ldom
ldm set-vnet [mac-addr=num] [vswitch=vswitch-name] [mode=[hybrid]] [pvid=port-vlan-id]
  [vid=vlan-id1,vlan-id2,...] [mtu=size] 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 [-f] [options={ro,slice,excl}] [mpgroup=mpgroup] backend
  volume-name@service-name
ldm set-vdsdev [-f] options=[{ro,slice,excl}] [mpgroup=mpgroup]
  volume-name@service-name
ldm remove-vdsdev [-f] volume-name@service-name
ldm add-vdisk [timeout=seconds] [id=disk-id] 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 [-f] vdpcs-service-name
```

```
ldm add-vdpcc vdpcc-name vdpcs-service-name ldom
ldm remove-vdpcc [-f] 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-spconfig config-name
ldm add-spconfig -r autosave-name [new-config-name]
ldm set-spconfig config-name
ldm set-spconfig factory-default
ldm remove-spconfig [-r] config-name
ldm list-spconfig [-r [autosave-name]]
ldm list-constraints ([-x] | [-e] [-p]) [ldom...]
ldm list-devices [-a] [-p] [cpu] [crypto] [memory] [io]
ldm list-services [-e] [-p] [ldom...]
```

Description The `ldm` command is referred to as the Logical Domains Manager and 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 service processor. The control domain is 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 use logical domains to run a variety of applications in different domains and keep them independent for security purposes.

All logical domains are the same and can be distinguished from one another based on the roles that you specify for them. The following are the 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 a

service 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 T1 processor, you can have a maximum of two I/O domains, one of which must also be the control domain.

Guest domain — Uses services from the I/O and service domains and is managed by the control domain.

You can use the Logical Domains Manager to establish dependency relationships between domains.

Master domain — A domain that has one or more domains that depend on it. A master domain specifies a failure policy to be enacted by its slave domains when the master domain fails. For instance, depending on the master domain's failure policy, a slave can be left as-is, panicked, rebooted, or stopped when the master domain fails.

Slave domain — A domain that depends on another domain. A domain can specify up to four master domains that dictate the failure policy to enact when one or more of the master domains fail.

Subcommand Summaries

Following are the supported subcommands along with a description and required authorization for each. For information about setting up authorization for user accounts, see "Creating Authorization and Profiles and Assigning Roles for User Accounts" in *Logical Domains 1.2 Administration Guide*.

Subcommand	Description	Authorization
`add-spconfig`	Adds a logical domain configuration to the service processor (SP).	`solaris.ldoms.write`
`add-domain`	Creates a logical domain.	`solaris.ldoms.write`
`add-`*resource*	Adds a resource to an existing logical domain. See RESOURCES for resource definitions.	`solaris.ldoms.write`
`bind-domain`	Binds resources to a created logical domain.	`solaris.ldoms.write`
`cancel-operation`	Cancels an operation, such as delayed reconfiguration (`reconf`) or domain migration (`migration`).	`solaris.ldoms.write`
`list-domain`	Lists logical domains and their states.	`solaris.ldoms.read`
`list-`*type*	Lists server resources, including bindings, constraints, devices, services, and configurations for logical domains.	`solaris.ldoms.read`

Subcommand	Description	Authorization
`list-variable`	Lists variables for logical domains.	`solaris.ldoms.read`
`migrate-domain`	Migrates a logical domain from one machine to another.	`solaris.ldoms.write`
`panic-domain`	Panics the Solaris OS on a specified logical domain.	`solaris.ldoms.write`
`remove-spconfig`	Removes a logical domain configuration from the service processor.	`solaris.ldoms.write`
`remove-domain`	Deletes a logical domain.	`solaris.ldoms.write`
`remove-`*resource*	Removes a resource from an existing logical domain. See RESOURCES for resource definitions.	`solaris.ldoms.write`
`remove-variable`	Removes one or more variables from an existing logical domain.	`solaris.ldoms.write`
`set-spconfig`	Specifies a logical domain configuration to use.	`solaris.ldoms.write`
`set-domain`	Sets properties on a 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 `crypto`. For a quantity change, the subcommand becomes a dynamic or a delayed reconfiguration 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-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 table shows the three kinds of aliases for `ldm` subcommands.

Alias Type	Short Form	Long Form
Action alias (verb)	`ls`	`list`
Action alias (verb)	`rm`	`remove`
Resource alias (noun)	`config`	`spconfig`
Resource alias (noun)	`crypto`	`mau`
Resource alias (noun)	`dom`	`domain`
Resource alias (noun)	`mem`	`memory`
Resource alias (noun)	`var`	`variable`
Resource alias (noun)	`vcc`	`vconscon`
Resource alias (noun)	`vcons`	`vconsole`
Resource alias (noun)	`vdpcc`	`ndpsldcc`
Resource alias (noun)	`vdpcs`	`ndpsldcs`
Resource alias (noun)	`vds`	`vdiskserver`
Resource alias (noun)	`vdsdev`	`vdiskserverdevice`
Resource alias (noun)	`vsw`	`vswitch`
Subcommand shortcut	`bind`	`bind-domain`
Subcommand shortcut	`cancel-op`	`cancel-operation`
Subcommand shortcut	`create`	`add-domain`
Subcommand shortcut	`destroy`	`remove-domain`
Subcommand shortcut	`list`	`list-domain`
Subcommand shortcut	`migrate`	`migrate-domain`
Subcommand shortcut	`modify`	`set-domain`
Subcommand shortcut	`panic`	`panic-domain`
Subcommand shortcut	`start`	`start-domain`
Subcommand shortcut	`stop`	`stop-domain`
Subcommand shortcut	`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.
`crypto`	Any Logical Domains-supported cryptographic unit on a Logical Domains-supported server. Currently, the two cryptographic units supported are the Modular Arithmetic Unit (MAU) and the Control Word Queue (CWQ).
`mem`, `memory`	Default memory 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 CPU threads 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 name. 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 that allows you to export virtual disks to other logical domains.
`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 that 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.
`config`	Lists the logical domain configurations stored on the service processor.
`constraints`	Lists the constraints used to create a logical domain.
`devices`	Lists all free devices for the server.
`services`	Lists all services exported by a logical domain.

Options The following table describes the `ldm` command options. The short for of the option is followed by the long form, if applicable.

`-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.
`-n`	`--dry-run`	Makes a dry run of a migration to check to see if the migration will succeed. Does not actually migrate the domain.
`-o`	`--output`	Specifies one or more of the following formats for an `ldm list` command, depending on what you want to see: `console`, `cpu`, `crypto`, `disk`, `domain`, `memory`, `network`, `physio`, `serial`, and `status`. If you specify more than one format, delimit the items by a comma with no spaces.
`-p`	`--parseable`	Generates a machine-readable version of the output.
`-r`		Performs a manual configuration recovery.
`-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.
`ldm`	`--help`	Displays usage statements.

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.

`default-vlan-id=` Specifies the default virtual local area network (VLAN) to which a virtual network device or virtual switch needs to be a member, in tagged mode. The first VLAN ID (*vid1*) is reserved for the `default-vlan-id`.

`failure-policy=` Specifies the master domain's failure policy, which controls how slave domains behave when the master domain fails. This property is set on a master domain. The default value is `ignore`. Following are the valid property values:

- `ignore` ignores failures of the master domain (slave domains are unaffected).
- `panic` panics any slave domains when the master domain fails.
- `reset` resets any slave domains when the master domain fails.
- `stop` stops any slave domains when the master domain fails.

`group=` Specifies a group to which to attach a console. The group argument allows multiple consoles to be multiplexed onto the same TCP connection.

`hostid=` Specifies a number that uniquely identifies the physical machine. If you do not specify a host ID, the Logical Domains Manager assigns the last 24 bits of the MAC address.

`id=` Specifies an ID for a new virtual disk device, virtual network device, and virtual switch device, respectively.

`mac-addr=` Defines a MAC address. The number must be in standard octet notation, for example, 80:00:33:55:22:66.

`master=` Specifies the name of up to four master domains for a slave domain. This property is set on a slave domain. By default, there are no masters for the domain. The domain must already exist prior to an `ldm add-domain` operation.

Note – The Logical Domains Manager does not permit you to create domain relationships that result in a dependency cycle.

`mode=` **For `add-vsw` and `set-vsw` subcommands:**

Omit this option when you are not running Solaris Cluster software in guest domains because you could impact virtual network performance.

Otherwise, specify one of the following:

- Set `mode=sc` to enable virtual networking support for prioritized processing of Solaris Cluster heartbeat packets in a Logical Domains environment.
- Leave the `mode=` argument blank in the `set-vsw` subcommand to stop special processing of heartbeat packets.

For `add-vnet` and `set-vnet` subcommands:

Omit this option when you do not want to use NIU Hybrid I/O.

Otherwise, specify one of the following:

- Set `mode=hybrid` to request the system to use NIU Hybrid I/O if possible. If it is not possible, the system reverts to virtual I/O. See "Using NIU Hybrid I/O" in *Logical Domains 1.2 Administration Guide*.
- Leave the `mode=` argument blank in the `set-vnet` subcommand to disable NIU Hybrid I/O.

`mpgroup=` Defines the multipath group name for several virtual disk server devices (`vdsdev`). So, when a virtual disk cannot communicate with a virtual disk server device, a failover is initiated to another virtual disk server device in the multipath group.

`mtu=` Specifies the maximum transmission unit (MTU) of a virtual switch, virtual network devices that are bound to the virtual switch, or both. Valid values are in the range of 1500-16000. The `ldm` command issues an error if an invalid value is specified.

`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.
`pvid=`	Specifies the VLAN to which the virtual network device needs to be a member, in untagged mode.
`service=`	Specifies the name of the existing virtual console concentrator that you want to handle the console connection.
`timeout=`	Defines the number of seconds for establishing a connection between a virtual disk client (`vdc`) and a virtual disk server (`vds`). If there are multiple virtual disk (`vdisk`) paths, then the `vdc` can try to connect to a different `vds`, and the timeout ensures that a connection to any `vds` is established within the specified amount of time. Specify `0` to disable the timeout in the `set-vdisk` subcommand.
`vid=`	Specifies the VLAN to which a virtual network device or virtual switch needs to be a member, in tagged mode.
`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
- `e` Error
- `n` Normal
- `s` Column 1 – starting or stopping

  Column 6 – source domain
- `t` Column 2 – transition

  Column 6 – target domain
- `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 1 List Flag Positions

Column 1	Column 2	Column 3	Column 4	Column 5	Column 6
s or -	n or t	d or -	c or -	v or -	s, t, or e

Subcommand Usage

This section contains descriptions of every supported command-line interface (CLI) operation, that is, every subcommand and resource combination.

Add, Set, Remove, and Migrate Domains

Add Logical Domains

This subcommand adds one or more logical domains by specifying one or more logical domain names or by using an XML configuration file. You can also specify property values to customize the domain, such as the MAC address, the host ID, a list of master domains, and a failure policy. If you do not specify these property values, the Logical Domains Manager automatically assigns default values.

```
ldm add-dom -i file
ldm add-dom [mac-addr=num] [hostid=num] [failure-policy=ignore|panic|reset|stop]
  [master=master-ldom1,...,master-ldom4] ldom
ldm add-dom ldom...
```

where:

- `-i` *file* specifies the XML configuration file to use in creating the logical domain.
- `mac-addr=`*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.
- `hostid=`*num* is a number uniquely identifying the physical machine. If you do not specify a host ID, the Logical Domains Manager assigns the last 24 bits of the MAC address.
- `failure-policy` specifies the master domain's failure policy, which controls how slave domains behave when the master domain fails. This property is set on a master domain. The default value is `ignore`. Following are the valid property values:
  - `ignore` ignores failures of the master domain (slave domains are unaffected).
  - `panic` panics any slave domains when the master domain fails.
  - `reset` resets any slave domains when the master domain fails.
  - `stop` stops any slave domains when the master domain fails.
- `master` specifies the name of up to four master domains for a slave domain. This property is set on a slave domain. By default, there are no masters for the domain. The master domain must exist prior to an `ldm add-domain` operation.

  Note – The Logical Domains Manager does not permit you to create domain relationships that result in a dependency cycle.
- *ldom* specifies the logical domain to be added.

Set Options for Logical Domains

This subcommand enables you to modify the `failure-policy` and `master` properties of each domain.

Note – If the slave domain is bound, all of its specified master domains must also be bound prior to invoking the `ldm set-domain` command.

`ldm set-dom -i` *file*
`ldm set-dom [failure-policy=ignore|panic|reset|stop]`
`[master=[`*master-ldom1,...,master-ldom4***`]]`** *ldom*

where:

- `-i` *file* specifies the XML configuration file to use in creating the logical domain.
- `failure-policy` specifies the master domain's failure policy, which controls how slave domains behave when the master domain fails. This property is set on a master domain. The default value is `ignore`. Following are the valid property values:
  - `ignore` ignores failures of the master domain (slave domains are unaffected).
  - `panic` panics any slave domains when the master domain fails.
  - `reset` resets any slave domains when the master domain fails.
  - `stop` stops any slave domains when the master domain fails.
- `master` specifies the name of up to four master domains for a slave domain. This property is set on a slave domain. By default, there are no masters for the domain. The master domain must already exist prior to this operation.

  Note – The Logical Domains Manager does not permit you to create domain relationships that result in a dependency cycle.
- *ldom* specifies the name of the logical domain for which you want to set options.

Remove Logical Domains

This subcommand removes one or more logical domains.

`ldm rm-dom -a`
`ldm rm-dom` *ldom***`...`**

where:

- `-a` deletes all logical domains except the control domain.
- *ldom* specifies the logical domain to be deleted.

  In the event that the domain to be destroyed is specified as a master domain, references to this domain are removed from all slave domains.

Migrate Logical Domain

This subcommand migrates a domain from one location to another.

`ldm migrate [-n]` *source-ldom* **`[`***user***`@]`***target-host***`[:`***target-ldom***`]`**

where:

- `-n` performs a dry run on the migration to determine whether it will succeed. It does not actually migrate the domain.
- *source-ldom* is the logical domain that you want to migrate.
- *user* is the user name that is authorized to run the Logical Domains Manager on the target host. If no user name is specified, the name of the user running the command is used by default.
- *target-host* is the host where you want to place the *target-ldom*.
- *target-ldom* is the logical domain name to be used on the target machine. The default is to keep the domain name used on the source domain (*source-ldom*).

Reconfigure Operations

There are three types of reconfiguration operations:

- **Dynamic reconfiguration operations.** Dynamic reconfiguration (DR) is the ability to add, set, or remove resources to or from an active domain. The ability to perform dynamic reconfiguration of a particular resource type is dependent on having support in the particular version of the OS running in the logical domain. For the control domain, if a dynamic reconfiguration cannot be done, a delayed reconfiguration operation is done instead.
- **Delayed reconfiguration operations.** 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. Delayed reconfiguration operations can only be performed on the control domain. Other domains must be stopped prior to modifying resources that cannot be dynamically configured.
- **Configuration mode.** The Logical Domains Manager runs in configuration mode when you are using a Sun UltraSPARC T1 processor, and the system is in the `factory-default` configuration. In this mode, no reconfiguration operations take effect until after the configuration is saved to the service processor by using the `add-config` subcommand and until that configuration is instantiated by rebooting the control domain.

See Chapter 1, "Overview of the Logical Domains Software," in *Logical Domains 1.2 Administration Guide* for more information about dynamic reconfiguration and delayed reconfiguration.

CPU Operations

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.

Note – Do not remove all the virtual CPUs on a core from a logical domain if the cryptographic unit in that core is allocated to that logical domain.

Crypto Operations

Add Cryptographic Units

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

ldm add-crypto *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-crypto *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-crypto *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.

Memory Operations

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* 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` is gigabytes
  - `K` is kilobytes
  - `M` is 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* 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` is gigabytes
  - `K` is kilobytes
  - `M` is 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* 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` is gigabytes
  - `K` is kilobytes
  - `M` is megabytes
- *ldom* specifies the logical domain where memory is to be removed.

Cancel Operations

This subcommand cancels either delayed reconfiguration (`reconf`) or domain migration (`migration`) operations for a logical domain.

ldm cancel-op migration *ldom*
ldm cancel-op reconf *ldom*

Note – The obsolete `remove-reconf` and `cancel-reconf` subcommands can be used as aliases for the `cancel-op reconf` subcommand.

Input/Output Devices

Add Input/Output Device

The subcommand in this example adds a PCI bus to a specified logical domain.

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

where:

- `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.

  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 is no longer 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

The 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 Server

Add a Virtual Switch

This subcommand adds a virtual switch to a specified logical domain.

`ldm add-vsw [default-vlan-id=`*vlan-id***`] [pvid=`***port-vlan-id***`]`**
`[vid=`*vlan-id1,vlan-id2,...***`] [mac-addr=`***num***`] [net-dev=`***device***`] [mode=sc] [mtu=`***size***`]`**
`[id=`*switch-id***`]`** *vswitch-name ldom*

where:

- `default-vlan-id=`*vlan-id* specifies the default VLAN to which a virtual switch and its associated virtual network devices belong to implicitly, in untagged mode. It serves as the default port VLAN ID (*pvid*) of the virtual switch and virtual network devices. Without this option, the default value of this property is `1`. Normally, you would not need to use this option. It is provided only as a way to change the default value of `1`.
- `pvid=`*port-vlan-id* specifies the VLAN to which the virtual network device needs to be a member, in untagged mode. This property also applies to the `set-vsw` subcommand. See "Using VLAN Tagging With Logical Domains Software" in *Logical Domains 1.2 Administration Guide*.

- `vid=`*vlan-id* specifies one or more VLANs to which a virtual network device or virtual switch needs to be a member, in tagged mode. This property also applies to the `set-vsw` subcommand. See "Using VLAN Tagging With Logical Domains Software" in *Logical Domains 1.2 Administration Guide* for more information.
- `mac-addr=`*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.
- `net-dev=`*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.

  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.
- `mtu=`*size* specifies the maximum transmission unit (MTU) of a virtual switch device. Valid values are in the range of 1500-16000.
- `id=`*switch-id* is the ID of a new virtual switch device. By default, ID values are generated automatically, so set this property if you need to match an existing device name in the OS.
- *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 [pvid=`*port-vlan-id***`] [vid=`***vlan-id1,vlan-id2,...***`] [mac-addr=`***num***`] [net-dev=`***device***`] [mode=[sc]] [mtu=`***size***`]`** *vswitch-name*

where:

- `pvid=`*port-vlan-id* specifies the VLAN to which the virtual network device needs to be a member, in untagged mode. See "Using VLAN Tagging With Logical Domains Software" in *Logical Domains 1.2 Administration Guide*.
- `vid=`*vlan-id* specifies one or more VLANs to which a virtual network device or virtual switch needs to be a member, in tagged mode. See "Using VLAN Tagging With Logical Domains Software" in *Logical Domains 1.2 Administration Guide*.
- `mac-addr=`*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.

- `net-dev=`*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.
- `mtu=`*size* specifies the maximum transmission unit (MTU) of a virtual switch device. Valid values are in the range of 1500-16000.
- *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***`] [mode=hybrid] [pvid=`***port-vlan-id***`] [vid=`***vlan-id1,vlan-id2,...***`] [id=`***network-id***`] [mtu=`***size***`]`** *if-name vswitch-name ldom*

where:

- `mac-addr=`*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`.
- `mode=hybrid` requests the system to use NIU Hybrid I/O on this `vnet` if possible. If it is not possible, the system reverts to virtual I/O. This hybrid mode is considered a delayed reconfiguration if set on an active `vnet` on a control domain. See "Using NIU Hybrid I/O" in *Logical Domains 1.2 Administration Guide*.
- `pvid=`*port-vlan-id* specifies the VLAN to which the virtual network device needs to be a member, in untagged mode. See "Using VLAN Tagging With Logical Domains Software" in *Logical Domains 1.2 Administration Guide*.

- `vid=`*vlan-id* specifies one or more VLANs to which a virtual network device needs to be a member, in tagged mode. See "Using VLAN Tagging With Logical Domains Software" in *Logical Domains 1.2 Administration Guide*.
- `mtu=`*size* specifies the maximum transmission unit (MTU) of a virtual network device. Valid values are in the range of 1500-16000.
- `id=`*network-id* is the ID of a new virtual network device. By default, ID values are generated automatically, so set this property if you need to match an existing device name in the OS.
- *if-name* is a unique interface name to the logical domain, which is 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***`] [mode=hybrid] [pvid=`***port-vlan-id***`] [vid=`***vlan-id1,vlan-id2,...***`] [mtu=`***size***`]`** *if-name ldom*

where:

- `mac-addr=`*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=`*vswitch-name* is the name of an existing network service (virtual switch) to which to connect.
- `mode=hybrid` enables NIU Hybrid I/O operations on this `vnet`. This option is considered a delayed reconfiguration if set on an active `vnet` on a control domain. Leave the `mode=` argument blank to disable NIU Hybrid I/O.
- `pvid=`*port-vlan-id* specifies the VLAN to which the virtual network device needs to be a member, in untagged mode. See "Using VLAN Tagging With Logical Domains Software" in *Logical Domains 1.2 Administration Guide*.
- `vid=`*vlan-id* specifies one or more VLANs to which a virtual network device needs to be a member, in tagged mode. See "Using VLAN Tagging With Logical Domains Software" in *Logical Domains 1.2 Administration Guide*.
- `mtu=`*size* specifies the maximum transmission unit (MTU) of a virtual network device. Valid values are in the range of 1500-16000.
- *if-name* is the unique interface name assigned to the virtual network device that 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* is the unique interface name assigned to the virtual network device that 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, which might 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. See Chapter 6, "Using Virtual Disks," in *Logical Domains 1.2 Administration Guide*.

`ldm add-vdsdev [-f] [options={ro,slice,excl}] [mpgroup=`*mpgroup***`]`** *backend volume-name*`@`*service-name*

where:

- `-f` attempts to force the creation of an additional virtual disk server when specifying a block device path that is already part of another virtual disk server. If specified, the `-f` option must be the first in the argument list.
- `options=` are as follows:
  - `ro` – Specifies read-only access
  - `slice` – Exports a backend as a single slice disk
  - `excl` – Specifies 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`.
- `mpgroup=`*mpgroup* is the disk multipath group name used for virtual disk failover support. You can assign the virtual disk several redundant paths in case the link to the virtual disk server device currently in use fails. To do this, you would group multiple virtual disk server devices (`vdsdev`) into one multipath group (`mpgroup`), all having the same `mpgroup` name. When a virtual disk is bound to any virtual disk server device in a multipath group, the virtual disk is bound to all the virtual disk server devices that belong to the `mpgroup`.
- *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 that 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 Device

This subcommand sets options for a virtual disk server. See the *Logical Domains 1.2 Administration Guide*.

`ldm set-vdsdev [-f] options=[{ro,slice,excl}] [mpgroup=`*mpgroup***`]`**
volume-name`@`*service-name*

where:

- `-f` removes the read-only restriction when multiple volumes in the same logical domain are sharing an identical block device path in read-only mode (`option=ro`). If specified, the `-f` option must be the first in the argument list.
- `options=` are as follows:
  - `ro` – Specifies read-only access
  - `slice` – Exports a backend as a single slice disk
  - `excl` – Specifies 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`.
- `mpgroup=`*mpgroup* is the disk multipath group name used for virtual disk failover support. You can assign the virtual disk several redundant paths in case the link to the virtual disk server device currently in use fails. To do this, you would group multiple virtual disk server devices (`vdsdev`) into one multipath group (`mpgroup`), all having the same `mpgroup` name. When a virtual disk is bound to any virtual disk server device in a multipath group, the virtual disk is bound to all the virtual disk server devices that belong to the `mpgroup`.
- *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 virtual disk server.

ldm add-vdisk [timeout=*seconds***] [id=***disk-id***]** *disk-name volume-name***@***service-name ldom*

where:

- `timeout=`*seconds* is the number of seconds for establishing a connection between a virtual disk client (`vdc`) and a virtual disk server (`vds`). If there are multiple virtual disk (`vdisk`) paths, then the `vdc` can try to connect to a different `vds`, and the timeout ensures that a connection to any `vds` is established within the specified amount of time.

  Omit the `timeout=` argument or set `timeout=0` to have the virtual disk wait indefinitely.
- `id=`*disk-id* is the ID of a new virtual disk device. By default, ID values are generated automatically, so set this property if you need to match an existing device name in the OS.
- *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 virtual disk server.

ldm set-vdisk [timeout=*seconds***] [volume=***volume-name***@***service-name***]** *disk-name ldom*

where:

- `timeout=`*seconds* is the number of seconds for establishing a connection between a virtual disk client (`vdc`) and a virtual disk server (`vds`). If there are multiple virtual disk (`vdisk`) paths, then the `vdc` can try to connect to a different `vds`, and the timeout ensures that a connection to any `vds` is established within the specified amount of time.

  Set `timeout=0` to disable the timeout.

  Do not specify a `timeout=` argument to have the virtual disk wait indefinitely.
- `volume=`*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 [-f]` *vdpcs-service-name*

where:

- `-f` attempts to force the removal of the virtual data plane channel service. The removal might fail.
- *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 [-f] *vdpcc-name ldom*

where:

- `-f` attempts to force the removal of the virtual data plane channel client. The removal might fail.
- *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:

- `port-range=`*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:

- `port-range=`*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, which might 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=`*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=`*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. When a group is specified, a service must also be specified.
- `service=`*vcc-server* is the name for the existing virtual console concentrator that should handle the console connection. A service must be specified when a group is specified.
- *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-value pair of a variable 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-value pair of a variable 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 no value.

Remove Variable

This subcommand removes a variable for a logical domain.

ldm rm-var *var-name... ldom*

where:

- *var-name* is the name of a variable to remove.
- *ldom* specifies the logical domain from which to remove the variable.

Other Operations

Start Logical Domains

This subcommand starts one or more logical domains.

```
ldm start -a
ldm start -i file
ldm start ldom...
```

where:

- `-a` starts 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 [-f] -a
ldm stop [-f] ldom...
```

where:

- `-f` attempts to force a running logical domain to stop. Use only if the domain cannot be stopped by any other means.
- `-a` stops 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` *ldom*

ldom specifies the logical domain to panic.

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***`]`**

subcommand specifies the `ldm` subcommand about which you want usage information.

Provide Version Information

This subcommand provides version information.

`ldm --version`
`ldm -V`

Bind Resources to a Logical Domain

This subcommand binds, or attaches, configured resources to a logical domain.

`ldm bind-dom -i` *file*
`ldm bind-dom` *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*

ldom specifies the logical domain from which to unbind resources.

Configure Operations

Add Logical Domain Configuration

This subcommand adds a logical domain configuration, either based on the currently active configuration or on a previously autosaved configuration. The configuration is stored on the service processor (SP).

`ldm add-config` *config-name*
`ldm add-config -r` *autosave-name* **`[`***new-config-name***`]`**

where:

- *config-name* is the name of the logical domain configuration to add.
- `-r` *autosave-name* applies the autosave configuration data to one of the following:
  - Configuration on the SP that has the same name
  - Newly created configuration, *new-config-name*, which does not exist on the SP

  If the target configuration does not exist on the SP, a configuration of that name is created and saved to the SP based on the contents of the corresponding autosave configuration. After the autosave configuration data is applied, those autosave files are deleted from the control domain. If *autosave-name* does not represent the currently selected configuration, or if *new-config-name* is specified, the state of the current configuration on the SP and any autosave files for it on the control domain are unaffected.

  To recover an autosave configuration that is known to be corrupted, you must specify `-r` *new-config-name*. You are not permitted to overwrite an existing configuration with one that is known to be corrupted.
- *new-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 SP.

`ldm set-config` *config-name*

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 that is stored on the SP, as well as any corresponding autosave configuration from the control domain.

`ldm rm-config [-r]` *config-name*

where:

- `-r` only removes autosave configurations from the control domain.
- *config-name* is the name of the logical domain configuration to remove.

List

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] [-o *format***] [-p][[***ldom***...]**

where:

- `-e` generates an extended listing containing services and devices that are automatically set up, that is, not under your control.
- `-l` generates a long listing.
- `-o` limits the output *format* to one or more of the following subsets. If you specify more than one format, delimit the items by a comma with no spaces.
  - `console` – Output contains the virtual console (`vcons`) and virtual console concentrator (`vcc`) service.
  - `cpu` – Output contains the virtual CPU (`vcpu`) and physical CPU (`pcpu`).
  - `crypto` – Cryptographic unit output contains the Modular Arithmetic Unit (`mau`) and any other LDoms-supported cryptographic unit, such as the Control Word Queue (CWQ).
  - `disk` – Output contains the virtual disk (`vdisk`) and virtual disk server (`vds`).
  - `domain` – Output contains variables (`var`), host ID (`hostid`), domain state, flags, software state, utilization percentage, a slave's master domains, and the master domain's failure policy.
  - `memory` – Output contains `memory`.
  - `network` – Output contains the media access control (`mac`) address , virtual network switch (`vsw`), and virtual network (`vnet`) device.
  - `physio` – Physical input/output contains the peripheral component interconnect (`pci`) and network interface unit (`niu`).
  - `serial` – Output contains the virtual logical domain channel (`vldc`) service, virtual logical domain channel client (`vldcc`), virtual data plane channel client (`vdpcc`), virtual data plane channel service (`vdpcs`).
  - `status` – Output contains the status of a migrating domain.
- `-p` generates the list in a parseable, machine-readable format.
- *ldom* is the name of the logical domain 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` generates an extended listing containing services and devices that are automatically set up, that is, not under your control.
- `-p` generates the list in a parseable, machine-readable format.

- *ldom* is the name of the logical domain 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` generates an extended listing containing services and devices that are automatically set up, that is, not under your control.
- `-p` generates the list in a parseable, machine-readable format.
- *ldom* is the name of the logical domain 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 no logical domains are specified, all logical domains are listed.

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

where:

- `-x` writes the constraint output in XML format to the standard output (`stdout`) format. This output can be used as a backup.
- *ldom* is the name of the logical domain for which you want to list constraints.
- `-e` generates an extended listing containing services and devices that are automatically set up, that is, not under your control.
- `-p` writes the constraint output in a parseable, machine-readable form.

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] [crypto] [memory] [io]
```

where:

- `-a` lists all server resources, bound and unbound.
- `-p` writes the constraint output in a parseable, machine-readable form.
- `cpu` lists only CPU resources.
- `crypto` 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.

In the power management column (PM) or field (pm=), yes means the virtual CPU is power managed, and no means the virtual CPU is powered on. It is assumed that 100 percent-free CPUs are power-managed by default.

List Logical Domain Configurations

This subcommand lists the logical domain configurations stored on the service processor.

ldm ls-config [-r [*autosave-name***]]**

-r [*autosave-name*] lists those configurations for which autosave files exist on the control domain. If *autosave-name* is specified, it only reports about *autosave-name*. The output also notes whether an autosave file is newer than the corresponding SP configuration.

Note – When a delayed reconfiguration is pending, the configuration changes are immediately autosaved. As a result, if you run the ldm ls-config -r command, the autosave configuration is shown as being newer than the current configuration.

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 the variable to list. If you do not specify any name, 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
VCC
    NAME         LDOM    PORT-RANGE
    primary-vcc0 primary 5000-5100
VSW
```

EXAMPLE 2 List Services *(Continued)*

```
    NAME         LDOM    MAC             NET-DEV   DEVICE     DEFAULT-VLAN-ID PVID VID MODE
    primary-vsw0 primary 00:14:4f:f9:68:d0 e1000g0 switch@0 1               1
VDS
    NAME         LDOM    VOLUME         OPTIONS      MPGROUP   DEVICE
    primary-vds0 primary
```

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 save the configuration on the service processor. You must reboot so the changes take effect.

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-crypto 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%
```

EXAMPLE 4 List Bindings *(Continued)*

```
    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 that you have the resources to create the desired guest domain configuration, add the guest domain, add the resources and devices that 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=vdisk1 ldg1
# ldm bind-dom ldg1
# ldm ls-constraints -x ldg1 > ldg1.xml
# ldm add-config ldg1_4cpu_512M
```

EXAMPLE 5 Create a Logical Domain *(Continued)*

```
# ldm start 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), 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 an inactive domain.

```
# ldm set-vcons group=ldg1 service=primary-vcc0 ldg2
```

If you use the `ldm ls -l` command after performing the `set-vcons` commands on all guest domains except the first, you can see that all domains are connected to the same port. See the `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 Control 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 a control

EXAMPLE 9 Cancel Delayed Reconfiguration Operations for a Control Domain *(Continued)*

domain. For example, you might do this so that you can perform other configuration commands on that domain or other domains. With this command, you can undo the delayed reconfiguration operation and do other configuration operations on this or other domains.

```
# ldm cancel-op reconf primary
```

EXAMPLE 10 Migrate a Domain

You can migrate a logical domain to another machine. This example shows a successful migration.

```
# ldm migrate ldg1 root@dt90-187:ldg
Target password:
```

EXAMPLE 11 List Configurations

The following examples show how to view the configurations. The first command shows the configurations that are stored on the SP. The second command shows the configurations on the SP as well as information about the autosave configurations on the control domain.

```
# ldm ls-config
factory-default
3guests [current]
data1
reconfig_primary
split1
# ldm ls-config -r
3guests [newer]
data1 [newer]
reconfig_primary
split1
unit
```

Both the current `3guests` configuration and the `data1` configuration have autosaved changes that have not been saved to the SP. If the system is powercycled while in this state, the Logical Domains Manager would perform the `3guests` autosave recovery based on the specified policy. The autosave recovery action is taken for `3guests` because it is marked as current.

The `reconfig_primary` and `split1` autosave configurations are identical to the versions on the SP, not newer versions.

The `unit` configuration only exists as an autosave configuration on the control domain. There is no corresponding configuration for `unit` on the SP. This situation might occur if the configuration was lost from the SP. A configuration can be lost if the SP is replaced or if a problem occurred with the persistent version of the configuration on the SP. Note that using

EXAMPLE 11 List Configurations *(Continued)*

the `rm-config` command to explicitly remove a configuration also removes the autosave version on the control domain. As a result, no remnants of the configuration remain on either the control domain or on the SP.

Exit Status

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

Attributes

See the `attributes(5)` man page for a description of the following attributes:

Attribute Types	Attribute Values
Availability	`SUNWldm`
Interface Stability	Uncommitted

See Also

`dumpadm(1M)`, `ifconfig(1M)`, `shutdown(1M)`, `vntsd(1M)`, `attributes(5)`

Logical Domains 1.2 Administration Guide

Name ldmconfig– Logical Domains Configuration Assistant

Synopsis

```
ldmconfig [-cdh]
```

Description The `ldmconfig` utility, the Logical Domains Configuration Assistant, is a terminal-based application that streamlines the setup of systems that can run Sun Logical Domains. Only chip multithreaded-based (CMT) systems, which are also known as Sun Coolthreads Servers, can be used to run Logical Domains software.

`ldmconfig` inspects the system to provide the user with a default set of choices to generate a valid configuration. After gathering the setup property values, `ldmconfig` creates a configuration that is suitable for setting up a logical domain.

You can run the `ldmconfig` utility by means of a console connection, remote terminal emulator, or `ssh` session.

The Configuration Assistant uses the following options:

`-c` Checks Solaris OS media for valid packages

`-d` Specifies debug mode, which retains run and error logs after completion

`-h` Displays usage message

Exit Status The following exit values are returned:

0 Successful completion.

>0 An error occurred.

Attributes See the `attributes(5)` man page for a description of the following attributes:

Attribute Types	Attribute Values
Availability	`SUNWconfig`
Interface Stability	Uncommitted

See Also ldm(1M), `attributes(5)`

Logical Domains 1.2 Administration Guide

Name ldmp2v– command-line interface for the Logical Domains Physical-to-Virtual (P2V) Migration Tool

Synopsis

```
ldmp2v collect [-a ufsdump|flash|none] [-v] [-x exclude-fs [-x ...]] -d data-dir
ldmp2v prepare [-b zvol | file] [-c cpu] [-o keep-hostid] [-o keep-mac]
  [-m mountpoint:size [-m ...]] [-M memsize] [-p prefix] [-s] [-v]
  [-x no-auto-adjust-fs] [-x remove-unused-slices] -d data-dir domain
ldmp2v prepare -R guest-root  [-c cpu] [-o keep-hostid] [-o keep-mac]
  [-m mountpoint:size [-m ...]] [-M memsize] [-v] [-x no-auto-adjust-fs]
  [-x remove-unused-slices] -d data-dir domain
ldmp2v prepare -C domain
ldmp2v convert -i install-image -d data-dir [-v] domain
ldmp2v convert [-j] -n interface -d data-dir [-v] domain
```

Description Version 1.0 of the Logical Domains Physical-to-Virtual (P2V) Migration Tool automatically converts an existing physical system to a virtual system that runs in a logical domain on a chip multithreading (CMT) system. The source system can be any `sun4u` SPARC system that runs at least the Solaris 8 Operating System, or a non-Logical Domains `sun4v` system that runs the Solaris 10 OS.

The conversion from a physical system to a virtual system is performed in the following phases:

- **Collection phase.** Runs on the physical source system. `collect` creates a file system image of the source system based on the configuration information that it collects about the source system.
- **Preparation phase.** Runs on the control domain of the target system. `prepare` creates the logical domain on the target system based on the configuration information collected in the `collect` phase. The file system image is restored to one or more virtual disks. The image is modified to enable it to run as a logical domain.
- **Conversion phase.** Runs on the control domain of the target system. In the `convert` phase, the created logical domain is converted into a logical domain that runs the Solaris 10 OS by using the standard Solaris upgrade process.

The following sections describe how the conversion from a physical system to a virtual system is performed in phases.

Collection Phase `ldmp2v collect [-a ufsdump|flash|none] [-v] [-x exclude-fs [-x ...]] -d data-dir`

The `ldmp2v collect` command uses the following options:

`-a` *archive-method*	Specifies the archiving method to use. Valid values are `ufsdump`, `flash`, or `none`. The default is `ufsdump`.
`-d` *data-dir*	Specifies the per-system directory in which to store P2V files. For the collection phase, this directory must be writable by `root`. Any intermediate directories are created automatically.

`-v`	Uses verbose mode, which increases the verbosity of the messages that are issued by `ldmp2v`.
`-x` *exclude-fs*	Excludes the file system, *exclude-fs*, from the archive.

Preparation Phase

```
ldmp2v prepare [-b zvol | file] [-c cpu] [-o keep-hostid] [-o keep-mac]
  [-m mountpoint:size [-m ...]] [-M memsize] [-p prefix] [-s] [-v]
  [-x no-auto-adjust-fs] [-x remove-unused-slices] -d data-dir domain
ldmp2v prepare -R guest-root  [-c cpu] [-o keep-hostid] [-o keep-mac]
  [-m mountpoint:size [-m ...]] [-M memsize] [-v] [-x no-auto-adjust-fs]
  [-x remove-unused-slices] -d data-dir domain
ldmp2v prepare -C domain
```

The `ldmp2v prepare` command uses the following operand and options:

domain	Specifies the logical domain on which to operate.
`-b` *backend-type*	Overrides the setting for `BACKEND_TYPE` in `/etc/ldmp2v.conf`. The virtual disks can be backed by ZFS volumes, `zvol`, or plain files, `file`.
`-c` *cpu*	Allocates the number of VCPUs to the logical domain. By default, `ldmp2v` allocates a VCPU for each CPU on the physical system.
`-C`	Cleans up the specified domain.
`-d` *data-dir*	Specifies the per-system directory where the files required for P2V are located.
`-m` *mountpoint*:*size*	Resizes the underlying slice and disk for the file system at *mountpoint*. The size is specified as *numunit*, where *unit* is `b` for blocks, `k` for kilobytes, `m` for megabytes, or `g` for gigabytes. You can specify this option more than one time. Using this option disables the automatic resizing of `/`, `/usr`, and `/var`.
`-M` *memsize*	Specifies the amount of memory in megabytes to allocate to the logical domain. By default, `ldmp2v` assigns the same amount that the physical system has.
`-o keep-hostid`	Transfers the host ID of the physical system to the logical domain. By default, the Logical Domains Manager assigns a new unique host ID.
`-o keep-mac`	Transfers the MAC addresses of the physical system to the logical domain. By default the Logical Domains Manager assigns a new unique MAC address.
`-p` *prefix*	Specifies the location where backend devices will be created. Denotes the ZFS dataset for the `zvol` backend, or a directory

relative to / for the `file` backend. This option overrides the `BACKEND_PREFIX` parameter in `/etc/ldmp2v.conf`.

`-R` *guest-root* — Selects non-automatic mode. The OS image modification steps are applied to the file system rooted at *guest-root*. Updates the `/etc/vfstab` of the logical domain to match the file system layout below *guest-root*.

`-s` — Creates sparse backend devices. This option overrides the `BACKEND_SPARSE` parameter in `/etc/ldmp2v.conf`.

`-v` — Uses verbose mode, which increases the verbosity of the messages that are issued by `ldmp2v`.

`-x no-auto-adjust-fs` — Prevents the automatic size adjustment of the `/`, `/usr`, and `/var` file systems to 10 Gbytes total. Use this option with care because the size of the existing file systems might not be sufficient to upgrade to a newer Solaris release.

You can manually resize file system sizes by using the `-m` option.

`-x remove-unused-slices` — Reduces the size of the virtual disk by not creating slices that do not hold a file system or a swap partition.

Conversion Phase

```
ldmp2v convert -i install-image -d data-dir [-v] domain
ldmp2v convert [-j] -n interface -d data-dir [-v] domain
```

The `ldmp2v convert` command uses the following options:

`-d` *data-dir* — Specifies the per-system directory where the files required for P2V are located.

`-i` *install-image* — Specifies the path to the Solaris 10 OS DVD ISO image to use for upgrade.

`-j` — Uses Custom JumpStart, which requires that a JumpStart server and JumpStart client are properly configured.

`-n` *interface* — Specifies the virtual network interface from which to boot when using a network install server.

`-v` — Uses verbose mode, which increases the verbosity of the messages issued by `ldmp2v`.

Caution – Before you begin the conversion phase, shut down the original physical system, as the logical domain uses the same IP addresses, and possibly also MAC addresses, as the physical system.

If any IP address of the physical system is active, the `ldmp2v convert` command exits with an error message.

Examples This section includes examples for the three phases.

EXAMPLE 1 Collection Phase Examples

The following examples show how you might use the `ldmp2v collect` command.

- **Sharing an NFS-mounted file system.** The following example shows the simplest way to perform the `collect` phase, where the source and target systems share an NFS-mounted file system.

```
# ldmp2v collect -d /home/dana/p2v/volumia
```

- **Not sharing an NFS-mounted file system.** When the source and target systems do not share an NFS-mounted file system, the file system image can be written to local storage and then later copied to the control domain. Since it is not possible to use `ufsdump` to exclude files, use the flash archiving method that is provided by `ldmp2v`. The flash tool automatically excludes the archive it creates.

```
# ldmp2v collect -d /home/dana/p2v/volumia -a flash
```

- **Skip file-system backup step.** If backups of the system are already available using a third-party backup tool such as NetBackup, you can skip the file system backup step by using the `none` archiving method. When you use this option, only the system configuration manifest is created.

```
# ldmp2v collect -d /home/dana/p2v/volumia -a none
```

Note – If the directory specified by `-d` is not shared by the source and target systems, copy the contents of that directory to the control domain. The directory contents must be copied to the control domain prior to beginning the preparation phase.

EXAMPLE 2 Preparation Phase Examples

The following examples show how you might use the `ldmp2v prepare` command.

- The following example creates a logical domain called `volumia` by using the defaults configured in `/etc/ldmp2v.conf` while keeping the MAC addresses of the physical system:

```
# ldmp2v prepare -d /home/dana/p2v/volumia -o keep-mac volumia
```

- The following shows how to completely remove a domain and its backend devices by using the `-C` option:

```
# ldmp2v prepare -C volumia
```

- The following shows how to resize one or more file systems during P2V by specifying the mount point and the new size with the `-m` option:

```
# ldmp2v prepare -d /home/dana/p2v/normaal -m /:8g normaal
```

EXAMPLE 3 Conversion Phase Examples

The following examples show how you might use the `ldmp2v convert` command.

- **Using a network install server.** The `ldmp2v convert` command boots the Logical Domain over the network by using the specified virtual network interface. You must run the `setup_install_server` and `add_install_client` scripts on the install server.

  Optionally, you can use the Custom JumpStart feature to perform a completely hands-off conversion.

  The following shows how to use a network install server to upgrade your system:

  ```
  # ldmp2v convert -n vnet0 -d /p2v/volumia volumia
  ```

  The following shows how to use Custom JumpStart to upgrade your system:

  ```
  # ldmp2v convert -j -n vnet0 -d /p2v/volumia volumia
  ```

- **Using an ISO image.** The `ldmp2v convert` command attaches the Solaris DVD ISO image to the logical domain and boots from it. To upgrade, answer all `sysid` prompts and select Upgrade.

  Note – The answers to the `sysid` questions are only used during the upgrade process, so you can select the simplest options (non-networked, no naming service, and so on). The system's original identity is preserved by the upgrade and takes effect on the reboot after the upgrade is complete. The time required to perform the upgrade depends on the Solaris cluster that is installed on the original system.

  ```
  # ldmp2v convert -i /tank/iso/s10s_u5.iso -d /home/dana/p2v/volumia volumia
  ```

Exit Status

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

Attributes

See the `attributes(5)` man page for a description of the following attributes:

Attribute Types	Attribute Values
Availability	`SUNWldmp2v`
Interface Stability	Uncommitted

See Also

ldm(1M), `attributes(5)`

Logical Domains 1.2 Administration Guide