Oracle® VM Server for SPARC 3.2 Reference Manual

Part No: E48734
March 2015

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

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

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

Contents

System Administration

Name

ldm — command-line interface for the Logical Domains Manager

```
ldm or ldm --help [subcommand]
ldm -V
ldm add-domain -i file
ldm add-domain [cpu-arch=generic|native|migration-class1|sparc64-class1] [mac-addr=num]
  [hostid=num] [failure-policy=ignore|panic|reset|stop] [extended-mapin-space=off]
  [master=master-ldom1,...,master-ldom4] [max-cores=[num|unlimited]]
  [uuid=uuid] [threading=max-ipc] [shutdown-group=num] [rc-add-policy=[iov]]
  [perf-counters=counter-set] domain-name
ldm add-domain domain-name...
ldm set-domain -i file
ldm set-domain [cpu-arch=generic|native|migration-class1|sparc64-class1] [mac-addr=num]
  [hostid=num] [failure-policy=ignore|panic|reset|stop] [extended-mapin-space=[on|off]]
  [master=[master-ldom1,...,master-ldom4]] [max-cores=[num|unlimited]]
  [threading=[max-ipc|max-throughput]] [shutdown-group=num] [rc-add-policy=[iov]]
  [perf-counters=[counter-set]] domain-name
ldm remove-domain -a
ldm remove-domain domain-name...
ldm list-domain [-e] [-l] [-o format] [-p] [-S] [domain-name...]
ldm migrate-domain [-f] [-n] [-p filename] source-ldom [user@]target-host[:target-ldom]
ldm migrate-domain [-f] [-n] -c source-ldom target-host[:target-ldom]
ldm add-vcpu CPU-count domain-name
ldm add-vcpu -c core-count domain-name
ldm set-vcpu CPU-count domain-name
ldm set-vcpu -c core-count domain-name
ldm remove-vcpu CPU-count domain-name
ldm remove-vcpu -c core-count domain-name
ldm add-core num domain-name
ldm add-core cid=core-ID[,core-ID[,...]] domain-name
ldm set-core num domain-name
ldm set-core cid=[core-ID[,core-ID[,...]]] domain-name
ldm remove-core [-f] num domain-name
ldm remove-core cid=core-ID[,core-ID[,...]] domain-name
ldm remove-core -g resource-group [-n number-of-cores] domain-name
ldm add-crypto number domain-name
ldm set-crypto [-f] number domain-name
ldm remove-crypto [-f] number domain-name
ldm add-memory [--auto-adj] size[unit] domain-name
ldm add-memory mblock=PA-start:size[,PA-start:size[,...]] domain-name
ldm set-memory [--auto-adj] size[unit] domain-name
ldm set-memory mblock=[PA-start:size[,PA-start:size[,...]]] domain-name
ldm remove-memory [--auto-adj] size[unit] domain-name
ldm remove-memory mblock=PA-start:size[,PA-start:size[,...]] domain-name
ldm remove-memory -g resource-group [-s size[unit]] domain-name
ldm start-reconf domain-name
ldm cancel-reconf domain-name
ldm cancel-operation (migration | reconf | memdr) domain-name
ldm add-io (device | vf-name) domain-name
ldm add-io [iov=on|off] bus domain-name
ldm set-io name=value [name=value...] pf-name
ldm set-io iov=on|off bus
ldm set-io [mac-addr=num] [alt-mac-addrs=[auto|num1,[auto|num2,...]]]
  [pvid=[pvid]] [vid=[vid1,vid2,...]] [mtu=size]
  [name=value...] net-vf-name
ldm set-io name=[value...] ib-pf-name
ldm set-io [bw-percent=[value]] [port-wwn=value node-wwn=value] fc-vf-name
ldm remove-io [-n] (bus | device | vf-name) domain-name
ldm list-io [-l] [-p] [bus | device | pf-name]
ldm list-io -d pf-name
ldm add-vsw [-q] [default-vlan-id=VLAN-ID] [pvid=port-VLAN-ID] [vid=VLAN-ID1,VLAN-ID2,...]
  [linkprop=phys-state] [mac-addr=num] [net-dev=device] [mode=sc] [mtu=size]
  [id=switch-ID] [inter-vnet-link=on|off] vswitch-name domain-name
ldm set-vsw [-q] [pvid=port-VLAN-ID] [vid=VLAN-ID1,VLAN-ID2,...] [mac-addr=num]
  [net-dev=device] [linkprop=[phys-state]] [mode=[sc]] [mtu=size]
  [inter-vnet-link=[on|off]] vswitch-name
ldm remove-vsw [-f] vswitch-name
ldm add-vnet [mac-addr=num] [mode=hybrid] [pvid=port-VLAN-ID]
  [pvlan=secondary-vid,pvlan-type]
  [protection=protection-type[,protection-type],...]
  [allowed-ips=ipaddr[,ipaddr]...] [priority=high|medium|low] [cos=0-7]
  [allowed-dhcp-cids=[macaddr|hostname,macaddr|hostname,...]]
  [alt-mac-addrs=auto|num1[,auto|num2,...]] [vid=VLAN-ID1,VLAN-ID2,...]
  [linkprop=phys-state] [id=network-ID] [mtu=size]
  [maxbw=value] if-name vswitch-name domain-name
ldm set-vnet [mac-addr=num] [vswitch=vswitch-name] [mode=[hybrid]]
  [pvid=port-VLAN-ID] [pvlan=[secondary-vid,pvlan-type]]
  [protection=[protection-type[,protection-type],...]]
  [allowed-ips=[ipaddr[,ipaddr]...]] [priority=high|medium|low] [cos=0-7]
  [allowed-dhcp-cids=[macaddr|hostname,macaddr|hostname,...]]
  [alt-mac-addrs=auto|num1[,auto|num2,...]] [vid=VLAN-ID1,VLAN-ID2,...]
  [linkprop=[phys-state]] [mtu=size]
  [maxbw=[value]] if-name domain-name
ldm remove-vnet [-f] if-name domain-name
ldm add-vds service-name domain-name
ldm remove-vds [-f] service-name
ldm add-vdsdev [-f] [-q] [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 domain-name
ldm set-vdisk [timeout=seconds] [volume=volume-name@service-name] disk-name domain-name
ldm remove-vdisk [-f] disk-name domain-name
ldm add-vdpcs vdpcs-service-name domain-name
ldm remove-vdpcs [-f] vdpcs-service-name
ldm add-vdpcc vdpcc-name vdpcs-service-name domain-name
ldm remove-vdpcc [-f] vdpcc-name domain-name
ldm add-vcc port-range=x-y vcc-name domain-name
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]
  [log=[on|off]] domain-name
```

```
ldm create-vf -n number | max pf-name
ldm create-vf [mac-addr=num] [alt-mac-addrs=[auto|num1,[auto|num2,...]]]
  [pvid=pvid] [vid=vid1,vid2,...] [mtu=size]
  [name=value...] net-pf-name
ldm create-vf [name=value...] ib-pf-name
ldm create-vf [port-wwn=value node-wwn=value] [bw-percent=[value]] fc-pf-name
ldm destroy-vf vf-name
ldm destroy-vf -n number | max pf-name
ldm add-variable var-name=[value]... domain-name
ldm set-variable var-name=[value]... domain-name
ldm remove-variable var-name... domain-name
ldm list-variable [var-name...] domain-name
ldm start-domain (-a | -i file | domain-name...)
ldm stop-domain [[-f | -q] | [[-h | -r | -t sec] [-m msg]]] (-a | domain-name...)
ldm panic-domain domain-name
ldm bind-domain [-f] [-q] (-i file | domain-name)
ldm unbind-domain domain-name
ldm list-bindings [-e] [-p] [domain-name...]
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]) [domain-name...]
ldm list-devices [-a] [-p] [-S] [cmi] [core] [cpu] [crypto] [io] [memory]
ldm list-hvdump
ldm list-permits
ldm list-services [-e] [-p] [domain-name...]
ldm set-hvdump [hvdump=on|off] [hvdump-reboot=on|off]
ldm start-hvdump
ldm add-policy [enable=yes|no] [priority=value] [attack=value] [decay=value]
  [elastic-margin=value] [sample-rate=value] [tod-begin=hh:mm[:ss]]
  [tod-end=hh:mm[:ss]] [util-lower=percent] [util-upper=percent] [vcpu-min=value]
  [vcpu-max=value] name=policy-name domain-name...
ldm set-policy [enable=[yes|no]] [priority=[value]] [attack=[value]] [decay=[value]]
  [elastic-margin=[value]] [sample-rate=[value]] [tod-begin=[hh:mm:ss]]
  [tod-end=[hh:mm:ss]] [util-lower=[percent]] [util-upper=[percent]] [vcpu-min=[value]]
  [vcpu-max=[value]] name=policy-name domain-name...
ldm remove-policy [name=]policy-name... domain-name
ldm init-system [-frs] -i file
ldm list-netdev [-b] [-l] [-o net-device] [-p] [domain-name...]
ldm list-netstat [-o net-device] [-p] [-t interval [-c count]] [-u unit] [domain-name]
ldm list-dependencies [-l] [-p] [-r] [domain-name]
ldm list-rsrc-group [-a] [-d domain-name] [-l] [-o core|memory|io] [-p] [resource-group]
ldm add-cmi num domain-name
ldm add-cmi cmi_id=id[,id[,...]] domain-name
ldm set-cmi [-f] num domain-name
ldm set-cmi [-f] cmi_id=[id[,id[,...]]] domain-name
ldm remove-cmi num domain-name
ldm remove-cmi cmi_id=id[,id[,...]] domain-name
ldm grow-cmi vcpus=num cmi_id=id domain-name
ldm grow-cmi cores=num cmi_id=id domain-name
ldm shrink-cmi vcpus=num cmi_id=id domain-name
ldm shrink-cmi cores=num cmi_id=id domain-name
ldm evict-cmi vcpus=num cmi_id=id domain-name
ldm evict-cmi cores=num cmi_id=id domain-name
ldm list-cmi [-l] [-p] [cmi_id=id[,id[,...]]] [domain-name...]
ldm grow-socket vcpus=num socket_id=id domain-name
ldm grow-socket cores=num socket_id=id domain-name
ldm grow-socket memory=size[unit] socket_id=id domain-name
ldm shrink-socket vcpus=num socket_id=id domain-name
ldm shrink-socket cores=num socket_id=id domain-name
ldm shrink-socket memory=size[unit] socket_id=id domain-name
ldm list-socket [-l] [-p] [socket_id=id[,id[,...]]] [domain-name...]
```

The `ldm` command interacts with the Logical Domains Manager and is used to create and manage logical domains. The Logical Domains Manager runs on the control domain, which is the initial domain created by the service processor. For those platforms that have physical domains, the Logical Domains Manager runs only in the control domain of each physical domain. 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 access to a physical I/O device, such as a network card in a PCI EXPRESS (PCIe) controller or a single-root I/O virtualization (SR-IOV) virtual function. An I/O domain can own a PCIe root complex, or it can own a PCIe slot or on-board PCIe device by using the direct I/O feature and an SR-IOV virtual function by using the SR-IOV feature.

An I/O domain can share physical I/O devices with other domains in the form of virtual devices when the I/O domain is also used as a service domain.

Root domain

Has a PCIe root complex assigned to it. This domain owns the PCIe fabric and all connected devices, and provides all fabric-related services, such as fabric error handling. A root domain owns all of the SR-IOV physical functions from which you can create virtual functions and assign them to I/O domains. A root domain is also an I/O domain, as it owns and has direct access to physical I/O devices.

The number of root domains that you can have depends on your platform architecture. For example, if you are using a Sun SPARC Enterprise T5440 server from Oracle, you can have up to four root domains.

The default root domain is the `primary` domain. Starting with the Oracle VM Server for SPARC 3.1 release, you can use non-`primary` domains to act as root domains.

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 slave domain enacts a failure policy when the master domain fails. For instance, 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. When one or more of the master domains fail, the failure policy dictates the slave domain's behavior.

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 "Using Rights Profiles and Roles" in "Oracle VM Server for SPARC 3.2 Administration Guide ".

Subcommand	Description	Authorization
`add-`*resource*	Adds a resource to an existing logical domain. See RESOURCES for resource definitions.	`solaris.ldoms.write`
`add-domain`	Creates a logical domain.	`solaris.ldoms.write`
`add-policy`	Adds a resource management policy to an existing logical domain.	`solaris.ldoms.write`
`add-spconfig`	Adds a logical domain configuration to the service processor (SP).	`solaris.ldoms.write`
`add-variable`	Adds one or more variables to a logical domain.	`solaris.ldoms.write`
`bind-domain`	Binds resources to a created logical domain.	`solaris.ldoms.write`
`cancel-operation`	Cancels an operation, such as a delayed reconfiguration (`reconf`), memory dynamic reconfiguration removal (`memdr`), or domain migration (`migration`).	`solaris.ldoms.write`
`cancel-reconf`	Cancels a delayed reconfiguration operation on the primary domain.	`solaris.ldoms.write`
`create-vf`	Creates one or more virtual functions.	`solaris.ldoms.write`

Subcommand	Description	Authorization
`destroy-vf`	Destroys one or more virtual functions.	`solaris.ldoms.write`
`evict-cmi`	Removes virtual CPUs or virtual CPU cores that are associated with a specific CMI device from the logical domain that owns the device.	`solaris.ldoms.write`
`grow-cmi`	Adds virtual CPUs or virtual CPU cores that are associated with a specific CMI device to the logical domain that owns the device.	`solaris.ldoms.write`
`grow-socket`	Adds virtual CPUs, virtual CPU cores, or virtual memory that is associated with a specific CPU socket to an existing logical domain.	`solaris.ldoms.write`
`init-system`	Configures one or more guest domains, the control domain, or both, by using an existing configuration.	`solaris.ldoms.write`
`list-bindings`	Lists server bindings for logical domains.	`solaris.ldoms.read`
`list-cmi`	Lists devices for logical domains.	`solaris.ldoms.read`
`list-constraints`	Lists resource constraints for logical domains.	`solaris.ldoms.read`
`list-dependencies`	Lists dependencies.	`solaris.ldoms.read`
`list-devices`	Lists devices for logical domains.	`solaris.ldoms.read`
`list-domain`	Lists logical domains and their states.	`solaris.ldoms.read`
`list-hvdump`	Lists hypervisor data collection property values.	`solaris.ldoms.read`
`list-io`	Lists I/O devices for logical domains.	`solaris.ldoms.read`
`list-netdev`	Lists network devices for logical domains.	`solaris.ldoms.read`
`list-netstat`	Lists network device statistics for logical domains.	`solaris.ldoms.read`
`list-permits`	Lists CPU core activation information.	`solaris.ldoms.read`
`list-rsrc-group`	Lists resource group information.	`solaris.ldoms.read`
`list-services`	Lists services for logical domains.	`solaris.ldoms.read`
`list-socket`	Lists CPU socket information.	`solaris.ldoms.read`
`list-spconfig`	Lists configurations for logical domains.	`solaris.ldoms.read`
`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 Oracle 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-domain`	Deletes a logical domain.	`solaris.ldoms.write`
`remove-policy`	Removes a resource management policy from an existing logical domain.	`solaris.ldoms.write`
`remove-spconfig`	Removes a logical domain configuration from the service processor.	`solaris.ldoms.write`

Subcommand	Description	Authorization
`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 `cmi`, `core`, `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-domain`	Sets properties on a logical domain.	`solaris.ldoms.write`
`set-hvdump`	Sets property values for the hypervisor data collection process.	`solaris.ldoms.write`
`set-io`	Modifies a physical function or a virtual function.	`solaris.ldoms.write`
`set-policy`	Sets properties for a resource management policy to an existing logical domain.	`solaris.ldoms.write`
`set-spconfig`	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`
`shrink-cmi`	Removes virtual CPUs or virtual CPU cores that are associated with a specific CMI device from the logical domain that owns the device.	`solaris.ldoms.write`
`shrink-socket`	Removes virtual CPUs, virtual CPU cores, or virtual memory that is associated with a specific CPU socket from an existing logical domain.	`solaris.ldoms.write`
`start-domain`	Starts one or more logical domains.	`solaris.ldoms.write`
`start-hvdump`	Manually starts the hypervisor data collection process.	`solaris.ldoms.write`
`start-reconf`	Enters delayed reconfiguration mode on a root domain.	`solaris.ldoms.write`
`stop-domain`	Stops one or more running 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 resource types.

Aliases

This section includes tables that show the short form and long form of the `ldm` subcommand actions (verbs), resource names (nouns), and full subcommands.

The following table shows the short form and long form of subcommand actions.

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

The following table shows the short form and long form of resource names.

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

The following table shows the short form and long form of subcommands.

Short Form	Long Form
`bind`	`bind-domain`
`cancel-op`	`cancel-operation`
`create`	`add-domain`
`destroy`	`remove-domain`
`list`	`list-domain`

Short Form	Long Form
`migrate`	`migrate-domain`
`modify`	`set-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.

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 Domains

The `add-domain` 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.

The syntax for the `add-domain` subcommand is:

```
ldm add-domain -i file
ldm add-domain [cpu-arch=generic|native|migration-class1|sparc64-class1] [mac-addr=num]
  [hostid=num] [failure-policy=ignore|panic|reset|stop] [extended-mapin-space=off]
  [master=master-ldom1,...,master-ldom4] [max-cores=[num|unlimited]]
  [uuid=uuid] [threading=max-ipc] [shutdown-group=num] [rc-add-policy=[iov]]
  [perf-counters=counter-set] domain-name
ldm add-domain domain-name...
```

where:

- `-i` *file* specifies the XML configuration file to use in creating the logical domain.
- `cpu-arch=generic|native|migration-class1|sparc64-class1` specifies one of the following values:
  - `generic` configures a guest domain for a CPU-type-independent migration.
  - `native` configures a guest domain to migrate only between platforms that have the same CPU type. `native` is the default value.

  - `migration-class1` is a cross-CPU migration family for SPARC T4, SPARC T5, SPARC M5, and SPARC M6 platforms that supports hardware cryptography across these migrations so that there is a lower bound to the supported CPUs.

    This value is not compatible with UltraSPARC T2, UltraSPARC T2 Plus, or SPARC T3 platforms, or Fujitsu M10 servers.

  - `sparc64-class1` is a cross-CPU migration family for SPARC64 platforms. The `sparc64-class1` value is based on SPARC64 instructions, so it has a greater number of instructions than the `generic` value. Therefore, the `sparc64-class1` value does not have a performance impact compared to the `generic` value.

    This value is not compatible with UltraSPARC T2, UltraSPARC T2 Plus, SPARC T3, SPARC T4, SPARC T5, SPARC M5, or SPARC M6 platforms.

- `mac-addr=`*num* is the MAC address for this domain. The number must be in standard octet notation, for example, `80:00:33:55:22:66`.
- `hostid` specifies the host ID for a particular domain. If you do not specify a host ID, the Logical Domains Manager assigns a unique host ID to each domain.
- `failure-policy` specifies the 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 (similar to running the `ldm panic` command).
  - `reset` stops and restarts any slave domains when the master domain fails (similar to running the `ldm stop -f` command and then the `ldm start` command).
  - `stop` immediately stops any slave domains when the master domain fails (similar to running the `ldm stop -f` command).
- `extended-mapin-space=off` disables the extended mapin space for the specified domain. By default, the extended mapin space is enabled.
- `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.

- `rc-add-policy` specifies whether to enable or disable the direct I/O and SR-IOV I/O virtualization operations on any root complex that might be added to the specified domain. Valid values are `iov` and no value (`rc-add-policy=`). When `rc-add-policy=iov`, the direct I/O and SR-IOV features are enabled for a root complex that is being added. When `rc-add-policy=`, the `iov` property value is cleared to disable the I/O virtualization features for the root complex (unless you explicitly set `iov=on` by using the `add-io` command). The default value is no value.

- `perf-counters=`*counter-set* specifies the types of access to grant to the performance counter. If no `perf-counters` value is specified, the value is `htstrand`. You can specify the following values for the `perf-counters` property:

  `global` Grants the domain access to the global performance counters that its allocated resources can access. Only one domain at a time can have access to the global performance counters. You can specify this value alone or with either the `strand` or `htstrand` value.

  `strand` Grants the domain access to the strand performance counters that exist on the CPUs that are allocated to the domain. You cannot specify this value and the `htstrand` value together.

  `htstrand` Behaves the same as the `strand` value and enables instrumentation of hyperprivilege mode events on the CPUs that are allocated to the domain. You cannot specify this value and the `strand` value together.

  To disable all access to any of the performance counters, specify `perf-counters=`.

- Setting the `threading` property specifies the workflow throughput of the domain. However, using this property is deprecated in favor of relying on the Critical Threads API, which is automatically enabled. See *Complete Power* (`http://www.oracle.com/us/products/servers-storage/servers/sparc-enterprise/eagle-investment-sparc-hardware-366143.pdf`).

  The following are valid values for the `threading` property:

  - `max-ipc`**.** Only one thread is active for each CPU core that is assigned to the domain, which maximizes the number of instructions per cycle. Selecting this mode requires that the domain is also configured with the whole-core constraint. See the `add-vcpu` and `set-vcpu` subcommand descriptions.
  - `max-throughput`**.** Activates all threads that are assigned to the domain, which maximizes throughput. This mode is used by default and is also selected if you do not specify any mode (`threading=`).

- `uuid=`*uuid* specifies the universally unique identifier (UUID) for the domain. *uuid* is a hexadecimal string, such as `12345678-1234-abcd-1234-123456789abc`, which consists of five hexadecimal numbers separated by dashes. Each number must have the specified number of hexadecimal digits: 8, 4, 4, 4, and 12, as follows:

  xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

- `max-cores=[`*num*`|unlimited]` specifies the maximum number of cores that are permitted to be assigned to a domain. If the value is `unlimited`, there is no constraint on the number of CPU cores that can be allocated.

- `shutdown-group=`*num* specifies the shutdown group number for a domain. This value is used by the SP on a Fujitsu M10 server when an ordered shutdown is performed.

  When the SP initiates an ordered shutdown, domains are shut down in descending order of their shutdown group number. That is, the domain with the highest number is shut down first, and the domain with the lowest number is shut down last. When more than one domain shares a shutdown group number, the domains shut down concurrently. If a master domain and a slave domain share a shutdown group number, the domains shut down concurrently even though a master-slave relationship exists. Therefore, when establishing a dependency relationship between a master domain and a slave domain, assign a different shutdown group number to each domain.

  Valid values are from `1` to `15`. The control domain's shutdown group number is zero (`0`) and cannot be changed. The default value for any other domain is `15`.

  For the new `shutdown-group` property values to take effect, you must use the `ldm add-spconfig` command to save the configuration to the SP.

  This property pertains only to the Fujitsu M10 platform.
- *domain-name* specifies the logical domain to be added.

Set Options for Domains

The `set-domain` subcommand enables you to modify *only* the `mac-addr`, `hostid`, `failure-policy`, `extended-mapin-space`, `master`, `max-cores`, and `threading` properties of each domain. You *cannot* use this command to update resource properties.

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.

The syntax for the `set-domain` subcommand is:

```
ldm set-domain -i file
ldm set-domain [cpu-arch=generic|native|migration-class1|sparc64-class1] [mac-addr=num]
  [hostid=num] [failure-policy=ignore|panic|reset|stop] [extended-mapin-space=[on|off]]
  [master=[master-ldom1,...,master-ldom4]] [max-cores=[num|unlimited]]
  [threading=[max-throughput|max-ipc]] [shutdown-group=num] [rc-add-policy=[iov]]
  [perf-counters=[counter-set]] domain-name
```

where:

- `-i` *file* specifies the XML configuration file to use in creating the logical domain.

  Only the `ldom_info` nodes specified in the XML file are parsed. Resource nodes, such as `vcpu`, `mau`, and `memory`, are ignored.
- `cpu-arch=generic|native|migration-class1|sparc64-class1` specifies one of the following values:
  - `generic` configures a guest domain for a CPU-type-independent migration.

    - `native` configures a guest domain to migrate only between platforms that have the same CPU type. `native` is the default value.
    - `migration-class1` is a cross-CPU migration family for SPARC T4, SPARC T5, SPARC M5, and SPARC M6 platforms that supports hardware cryptography across these migrations so that there is a lower bound to the supported CPUs.

      This value is not compatible with UltraSPARC T2, UltraSPARC T2 Plus, or SPARC T3 platforms, or Fujitsu M10 servers.
    - `sparc64-class1` is a cross-CPU migration family for SPARC64 platforms. The `sparc64-class1` value is based on SPARC64 instructions, so it has a greater number of instructions than the `generic` value. Therefore, the `sparc64-class1` value does not have a performance impact compared to the `generic` value.

      This value is not compatible with UltraSPARC T2, UltraSPARC T2 Plus, SPARC T3, SPARC T4, SPARC T5, SPARC M5, or SPARC M6 platforms.
- `mac-addr=`*num* is the MAC address for this domain. The number must be in standard octet notation, for example, `80:00:33:55:22:66`.
- `hostid` specifies the host ID for a particular domain. If you do not specify a host ID, the Logical Domains Manager assigns a unique host ID to each domain.
- `failure-policy` specifies the 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` stops and restarts any slave domains when the master domain fails.
    - `stop` stops any slave domains when the master domain fails.
- `extended-mapin-space` enables or disables the extended mapin space for the specified domain. By default, the `extended-mapin-space=on`, which is equivalent to setting `extended-mapin-space=`.
- `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.

- `rc-add-policy` specifies whether to enable or disable the direct I/O and SR-IOV I/O virtualization operations on any root complex that might be added to the specified domain. Valid values are `iov` and no value (`rc-add-policy=`). When `rc-add-policy=iov`, the direct I/O and SR-IOV features are enabled for a root complex that is being added. When `rc-add-policy=`, the `iov` property value is cleared to disable the I/O virtualization features for the root complex (unless you explicitly set `iov=on` by using the `add-io` command). The default value is no value.

- `perf-counters=`*counter-set* specifies the types of access to grant to the performance counter. You can specify the following values for the `perf-counters` property:

  `global` Grants the domain access to the global performance counters that its allocated resources can access. Only one domain at a time can have access to the global performance counters. You can specify this value alone or with either the `strand` or `htstrand` value.

  `strand` Grants the domain access to the strand performance counters that exist on the CPUs that are allocated to the domain. You cannot specify this value and the `htstrand` value together.

  `htstrand` Behaves the same as the `strand` value and enables instrumentation of hyperprivilege mode events on the CPUs that are allocated to the domain. You cannot specify this value and the `strand` value together.

  To disable all access to any of the performance counters, specify `perf-counters=`.

- Setting the `threading` property specifies the workflow throughput of the domain. However, using this property is deprecated in favor of relying on the Critical Threads API, which is automatically enabled. See *Complete Power* (`http://www.oracle.com/us/products/servers-storage/servers/sparc-enterprise/eagle-investment-sparc-hardware-366143.pdf`).

  The following are valid values for the `threading` property:

  - `max-ipc`**.** Only one thread is active for each CPU core that is assigned to the domain, which maximizes the number of instructions per cycle. Selecting this mode requires that the domain is also configured with the whole-core constraint. See the `add-vcpu` and `set-vcpu` subcommand descriptions.
  - `max-throughput`**.** Activates all threads that are assigned to the domain, which maximizes throughput. This mode is used by default and is also selected if you do not specify any mode (`threading=`).

- `max-cores=[`*num*`|unlimited]` specifies the maximum number of cores that are permitted to be assigned to a domain. If the value is `unlimited`, there is no constraint on the number of CPU cores that can be allocated.

- `shutdown-group=`*num* specifies the shutdown group number for a domain. This value is used by the SP on a Fujitsu M10 server when an ordered shutdown is performed.

  When the SP initiates an ordered shutdown, domains are shut down in descending order of their shutdown group number. That is, the domain with the highest number is shut down first, and the domain with the lowest number is shut down last. When more than one domain shares a shutdown group number, the domains shut down concurrently. If a master domain and a slave domain share a shutdown group number, the domains shut down concurrently even though a master-slave relationship exists. Therefore, when establishing a dependency relationship between a master domain and a slave domain, assign a different shutdown group number to each domain.

Valid values are from `1` to `15`. The control domain's shutdown group number is zero (`0`) and cannot be changed. The default value for any other domain is `15`.

For the new `shutdown-group` property values to take effect, you must use the `ldm add-spconfig` command to save the configuration to the SP.

This property pertains only to the Fujitsu M10 platform.

- *domain-name* specifies the name of the logical domain for which you want to set options.

Remove Domains

The `remove-domain` subcommand removes one or more logical domains.

```
ldm remove-domain -a
ldm remove-domain domain-name...
```

where:

- `-a` deletes all logical domains except the control domain.
- *domain-name* 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 Domains

The `migrate-domain` subcommand migrates a domain from one location to another.

```
ldm migrate-domain [-f] [-n] [-p filename] source-ldom [user@]target-host[:target-ldom]
ldm migrate-domain [-f] [-n] -c source-ldom target-host[:target-ldom]
```

where:

- `-f` attempts to force the migration of the domain.
- `-n` performs a dry run on the migration to determine whether it will succeed. It does not actually migrate the domain.
- `-p` *filename* reads the password needed on the target machine from the first line of *filename*. This option enables you to perform non-interactive migrations that do not require you to provide the target machine password at a prompt.

  If you plan to store passwords in this manner, ensure that the file permissions are set so that only the root owner or a privileged user can read or write the file (400 or 600).

  This option cannot be used with the `-c` option.
- `-c` uses SSL trusted certificates to perform a domain migration. This option cannot be used with the `-p` *filename* option. You cannot specify a user name if you use the `-c` option.

  To use this option, you must first ensure that certificates are installed and configured on the source and target machines. When the `-c` option is specified, the source machine does not prompt for a password. The migration request is rejected if the target certificate cannot be verified.

When the SSL trusted certificates are accessed successfully, they are cached for the lifetime of the `ldmd` instance. When changing or removing the certificates, you must restart the `ldmd` daemon to make the changes take effect.

- *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*).

Reconfiguration Operations

Logical Domains supports the following types of reconfiguration operations:

- **Dynamic reconfiguration operations.** Dynamic reconfiguration 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. If a dynamic reconfiguration cannot be performed on the control domain, initiate a delayed reconfiguration operation. Sometimes, the delayed reconfiguration is automatically initiated.
- **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. You manually enter delayed reconfiguration mode on the root domain by running the `ldm start-reconf primary` command. When you initiate a delayed reconfiguration on a non-`primary` root domain, you can only perform a limited set of I/O operations (`add-io`, `set-io`, `remove-io`, `create-vf`, and `destroy-vf`). Other domains must be stopped prior to modifying resources that cannot be dynamically configured.

See “Resource Reconfiguration” in “Oracle VM Server for SPARC 3.2 Administration Guide ” for more information about dynamic reconfiguration and delayed reconfiguration.

CPU Operations

You can allocate either CPU threads or CPU cores to a domain. To allocate CPU threads, use the `add-vcpu`, `set-vcpu`, and `remove-vcpu` subcommands. To allocate CPU cores, use the `add-core`, `set-core`, and `remove-core` subcommands.

Add CPU Threads

The `add-vcpu` subcommand adds the specified number of CPU threads or CPU cores to a logical domain. Note that a domain *cannot* be configured simultaneously with CPU cores and CPU threads. CPU core configurations and CPU thread configurations are mutually exclusive.

The syntax for the `add-vcpu` subcommand is:

```
ldm add-vcpu CPU-count domain-name
ldm add-vcpu -c core-count domain-name
```

where:

- `-c` is a deprecated option that performs the following discrete CPU operations:
  - Sets the allocation unit for the domain from threads to cores, if not already set, and adds the specified number of cores to the domain.
  - If the domain is inactive, sets a cap on the number of cores that can be allocated to the domain when it is bound or active. A cap is set on the `primary` domain *only* if the domain is in a delayed reconfiguration.

  If any allocation request results in more cores being assigned to a domain than is permitted by the cap, the command fails.

  This option configures hard partitioning on your Oracle VM Server for SPARC system. See “Configuring the System With Hard Partitions” in “Oracle VM Server for SPARC 3.2 Administration Guide ”.

  You can change the allocation unit from cores to threads and remove the cap. Make these changes by issuing an `add-vcpu`, `set-vcpu`, or `remove-vcpu` command without the `-c` option on an inactive domain or on the `primary` domain that is in delayed reconfiguration mode.

  Starting with the Oracle VM Server for SPARC 2.2 release, the CPU cap and the allocation of CPU cores is handled by separate commands. By using these commands you can independently allocate CPU cores, set a cap, or both. The allocation unit can be set to cores even when no cap is in place. However, running the system when no cap is in place is *not* acceptable for configuring hard partitioning on your Oracle VM Server for SPARC system.

  - Allocate the specified number of CPU cores to a domain by using the `add-core`, `set-core`, and `remove-core` subcommands.
  - Set the cap by using the `create-domain` or `set-domain` subcommand to specify the `max-cores` property value.
- When the `-c` option is *not* specified, *CPU-count* is the number of CPU threads to be added to the logical domain. When the `-c` option is specified, *core-count* is the number of CPU cores to be added to the logical domain.
- *domain-name* specifies the logical domain where the CPU threads are to be added.

Set CPU Threads

The `set-vcpu` subcommand specifies the number of CPU threads or CPU cores to be set in a logical domain. Note that a domain *cannot* be configured simultaneously with CPU cores and CPU threads. CPU core configurations and CPU thread configurations are mutually exclusive.

The syntax for the `set-vcpu` subcommand is:

```
ldm set-vcpu CPU-count domain-name
ldm set-vcpu -c core-count domain-name
```

where:

- -c is a deprecated option that performs the following discrete CPU operations:
  - Sets the allocation unit for the domain from threads to cores, if not already set, and sets the allocation to the specified number of cores.
  - If the domain is inactive, sets a cap on the number of cores that can be allocated to the domain when it is bound or active. A cap is set on the primary domain *only* if the domain is in a delayed reconfiguration.

  If any allocation request results in more cores being assigned to a domain than is permitted by the cap, the command fails.

  This option configures hard partitioning on your Oracle VM Server for SPARC system. See "Configuring the System With Hard Partitions" in "Oracle VM Server for SPARC 3.2 Administration Guide ".

  You can change the allocation unit from cores to threads and remove the cap. Make these changes by issuing an add-vcpu, set-vcpu, or remove-vcpu command without the -c option on an inactive domain or on the primary domain that is in delayed reconfiguration mode.

  Starting with the Oracle VM Server for SPARC 2.2 release, the CPU cap and the allocation of CPU cores is handled by separate commands. By using these commands you can independently allocate CPU cores, set a cap, or both. The allocation unit can be set to cores even when no cap is in place. However, running the system when no cap is in place is *not* acceptable for configuring hard partitioning on your Oracle VM Server for SPARC system.

  - Allocate the specified number of CPU cores to a domain by using the add-core, set-core, and remove-core subcommands.
  - Set the cap by using the create-domain or set-domain subcommand to specify the max-cores property value.
- When the -c option is *not* specified, *CPU-count* is the number of CPU threads to be set in a logical domain. When the -c option is specified, *core-count* is the number of CPU cores to be set in a logical domain.
- *domain-name* is the logical domain where the number of CPU threads are to be set.

Remove CPU Threads

The remove-vcpu subcommand removes the specified number of CPU threads or CPU cores from a logical domain. Note that a domain *cannot* be configured simultaneously with CPU cores and CPU threads. CPU core configurations and CPU thread configurations are mutually exclusive.

The syntax for the remove-vcpu subcommand is:

```
ldm remove-vcpu CPU-count domain-name
ldm remove-vcpu -c core-count domain-name
```

where:

- `-c` is a deprecated option that performs the following discrete CPU operations:
  - Sets the allocation unit for the domain from threads to cores, if not already set, and removes the specified number of cores from the domain.
  - If the domain is inactive, sets a cap on the number of cores that can be allocated to the domain when it is bound or active. A cap is set on the `primary` domain *only* if the domain is in a delayed reconfiguration.

  If any allocation request results in more cores being assigned to a domain than is permitted by the cap, the command fails.

  This option configures hard partitioning on your Oracle VM Server for SPARC system. See "Configuring the System With Hard Partitions" in "Oracle VM Server for SPARC 3.2 Administration Guide ".

  You can change the allocation unit from cores to threads and remove the cap. Make these changes by issuing an `add-vcpu`, `set-vcpu`, or `remove-vcpu` command without the `-c` option on an inactive domain or on the `primary` domain that is in delayed reconfiguration mode.

  Starting with the Oracle VM Server for SPARC 2.2 release, the CPU cap and the allocation of CPU cores is handled by separate commands. By using these commands you can independently allocate CPU cores, set a cap, or both. The allocation unit can be set to cores even when no cap is in place. However, running the system when no cap is in place is *not* acceptable for configuring hard partitioning on your Oracle VM Server for SPARC system.

  - Allocate the specified number of CPU cores to a domain by using the `add-core`, `set-core`, and `remove-core` subcommands.
  - Set the cap by using the `create-domain` or `set-domain` subcommand to specify the `max-cores` property value.
- When the `-c` option is *not* specified, *CPU-count* is the number of CPU threads to be removed from the logical domain. When the `-c` option is specified, *core-count* is the number of CPU cores to be removed from the logical domain.
- *domain-name* specifies the logical domain where the CPU threads are to be removed.

Add CPU Cores

The `add-core` subcommand adds the specified number of CPU cores to a domain. When you specify the number of CPU cores, the cores to be assigned are automatically selected. However, when you specify a *core-ID* value to the `cid` property, the specified cores are explicitly assigned.

The `cid` property should *only* be used by an administrator who is knowledgeable about the topology of the system to be configured. This advanced configuration feature enforces specific allocation rules and might affect the overall performance of the system.

Note that you cannot use the `ldm add-core` command to add named core resources to a domain that already uses automatically assigned (anonymous) core resources.

The syntax for the `add-core` subcommand is:

```
ldm add-core num domain-name
ldm add-core cid=core-ID[,core-ID[,...]] domain-name
```

where:

- *num* specifies the number of CPU cores to assign to a domain.
- `cid=`*core-ID*`[,...]` specifies one or more physical CPU cores to assign to a domain.
- *domain-name* specifies the domain to which the CPU cores are assigned.

Set CPU Cores

The `set-core` subcommand specifies the number of CPU cores to assign to a domain. When you specify the number of CPU cores, the cores to be assigned are automatically selected. However, when you specify a *core-ID* value to the `cid` property, the specified cores are explicitly assigned.

The syntax for the `set-core` subcommand is:

```
ldm set-core num domain-name
ldm set-core cid=[core-ID[,core-ID[,...]]] domain-name
```

where:

- *num* specifies the number of CPU cores to assign to a domain.
- `cid=`*core-ID*`[,...]` specifies one or more physical CPU cores to assign to a domain. `cid=` removes all named CPU cores.
- *domain-name* specifies the domain to which the CPU cores are assigned.

Remove CPU Cores

The `remove-core` subcommand specifies the number of CPU cores to remove from a domain. When you specify the number of CPU cores, the cores to be removed are automatically selected. However, when you specify a *core-ID* value to the `cid` property, the specified cores are explicitly removed.

When you specify a resource group by using the `-g` option, the cores that are selected for removal all come from that resource group.

The syntax for the `remove-core` subcommand is:

```
ldm remove-core [-f] num domain-name
ldm remove-core cid=[core-ID[,core-ID[,...]]] domain-name
ldm remove-core -g resource-group [-n number-of-cores] domain-name
```

where:

- `-f` attempts to force the removal of one or more cores from an active domain.
- `-g` *resource-group* specifies that the operation is performed on the resources in the specified resource group.

- `-n` *number-of-cores* specifies the number of cores to remove. If this option is not specified, all cores are removed from the specified resource group that belongs to the specified domain. This option can be used only when the `-g` option is specified.
- *num* specifies the number of CPU cores to remove from a domain.
- `cid=`*core-ID*`[,...]` specifies one or more physical CPU cores to remove from a domain.
- *domain-name* specifies the domain from which the CPU cores are removed.

Cryptographic Unit Operations

The cryptographic unit subcommands only pertain to SPARC platforms that have discrete cryptographic units. Newer platforms, starting with the SPARC T4 platform, have integrated cryptographic instructions and do not use discrete cryptographic units.

Add Cryptographic Units

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

The syntax for the `add-crypto` subcommand is:

```
ldm add-crypto number domain-name
```

where:

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

Set Cryptographic Units

The `set-crypto` subcommand specifies the number of cryptographic units to be set in a logical domain. If you want to remove all cryptographic units from an active domain, you must specify the `-f` option.

To remove the last cryptographic unit from the `primary` domain when domains are active, you must do one of the following:

- Use dynamic reconfiguration and specify the `-f` option
- Use delayed reconfiguration

The syntax for the `set-crypto` subcommand is:

```
ldm set-crypto [-f] number domain-name
```

where:

- `-f` forces the removal of the last cryptographic unit in the domain if *number* is `0`.

  The `-f` option is only necessary in the following situations:

  - When the guest domain is active

  - On the `primary` domain, but *only* if at least one active guest domain exists on the system
- *number* is the number of cryptographic units to be set in the logical domain.
- *domain-name* specifies the logical domain where the number of cryptographic units are to be set.

Remove Cryptographic Units

The `remove-crypto` subcommand removes the specified number of cryptographic units from a logical domain. If you want to remove all cryptographic units from an active domain, you must specify the `-f` option.

To remove the last cryptographic unit from the `primary` domain when domains are active, you must do one of the following:

- Use dynamic reconfiguration and specify the `-f` option
- Use delayed reconfiguration

The syntax for the `remove-crypto` subcommand is:

```
ldm remove-crypto [-f] number domain-name
```

where:

- `-f` forces the removal of the last cryptographic unit in the domain if *number* is equal to the number of cryptographic units in the domain.

  The `-f` option is only necessary in the following situations:

  - When the guest domain is active
  - On the `primary` domain, but *only* if at least one active guest domain exists on the system
- *number* is the number of `cryptographic` units to be removed from the logical domain.
- *domain-name* specifies the logical domain where the cryptographic units are to be removed.

Memory Operations

Add Memory

The `add-memory` subcommand adds the specified amount of memory to a domain. When you specify a memory block size, the memory block to be assigned is automatically selected. However, when you specify a *PA-start*:*size* value to the `mblock` property, the specified memory blocks are explicitly assigned.

The `mblock` property should *only* be used by an administrator who is knowledgeable about the topology of the system to be configured. This advanced configuration feature enforces specific allocation rules and might affect the overall performance of the system.

The syntax for the `add-memory` subcommand is:

```
ldm add-memory [--auto-adj] size[unit] domain-name
ldm add-memory mblock=PA-start:size[,PA-start:size[,...]] domain-name
```

where:

- `--auto-adj` specifies that the amount of memory to be added to an active domain is automatically 256Mbyte-aligned, which might increase the requested memory size. If the domain is inactive, bound, or in a delayed reconfiguration, this option automatically aligns the resulting size of the domain by rounding up to the next 256-Mbyte boundary.
- *size* is the size of memory in bytes to be set in the logical domain.

  If you want a different unit of measurement, specify *unit* as one of the following values using either uppercase or lowercase:

  - `G` for gigabytes
  - `K` for kilobytes
  - `M` for megabytes
- `mblock=`*PA-start*`:`*size* specifies one or more physical memory blocks to assign to a domain. *PA-start* specifies the starting physical address of the memory block in hexadecimal format. *size* is the size of the memory block, including a unit, to be assigned to the domain. Note that you cannot use this property to specify the physical addresses of DIMMs.
- *domain-name* specifies the logical domain where the memory is to be added.

Set Memory

The `set-memory` subcommand sets a specific amount of memory in a domain. Depending on the amount of memory specified, this subcommand is treated as an `add-memory` or `remove-memory` operation.

When you specify a memory block size, the memory block to be assigned is automatically selected. However, when you specify a *PA-start*`:`*size* value to the `mblock` property, the specified memory blocks are explicitly assigned.

The syntax for the `set-memory` subcommand is:

```
ldm set-memory [--auto-adj] size[unit] domain-name
ldm set-memory mblock=PA-start:size[,PA-start:size[,...]] domain-name
```

where:

- `--auto-adj` specifies that the amount of memory to be added to or removed from an active domain is automatically 256Mbyte-aligned, which might increase the requested memory size. If the domain is inactive, bound, or in a delayed reconfiguration, this option automatically aligns the resulting size of the domain by rounding up to the next 256-Mbyte boundary.
- *size* is the size of memory in bytes to be set in the logical domain.

  If you want a different unit of measurement, specify *unit* as one of the following values using either uppercase or lowercase:

  - `G` for gigabytes
  - `K` for kilobytes

    - `M` for megabytes
- `mblock=`*PA-start*`:`*size* specifies one or more physical memory blocks to assign to a domain. *PA-start* specifies the starting physical address of the memory block in hexadecimal format. *size* is the size of the memory block, including a unit, to be assigned to the domain. Note that you cannot use this property to specify the physical addresses of DIMMs.
- *domain-name* specifies the logical domain where the memory is to be modified.

Remove Memory

The `remove-memory` subcommand removes the specified amount of memory from a logical domain. When you specify a memory block size, the memory block to be removed is automatically selected. However, when you specify a *PA-start*`:`*size* value to the `mblock` property, the specified memory blocks are explicitly removed.

When you specify a resource group by using the `-g` option, the memory that is selected for removal all comes from that resource group.

The syntax for the `remove-memory` subcommand is:

```
ldm remove-memory [--auto-adj] size[unit] domain-name
ldm remove-memory mblock=PA-start:size[,PA-start:size[,...]] domain-name
ldm remove-memory -g resource-group [-s size[unit]] domain-name
```

where:

- `--auto-adj` specifies that the amount of memory to be removed from an active domain is automatically 256Mbyte-aligned, which might increase the requested memory size. If the domain is inactive, bound, or in a delayed reconfiguration, this option automatically aligns the resulting size of the domain by rounding up to the next 256-Mbyte boundary.
- *size* is the size of memory in bytes to be set in the logical domain.

  If you want a different unit of measurement, specify *unit* as one of the following values using either uppercase or lowercase:

    - `G` for gigabytes
    - `K` for kilobytes
    - `M` for megabytes
- `mblock=`*PA-start*`:`*size* specifies one or more physical memory blocks to remove from a domain. *PA-start* specifies the starting physical address of the memory block in hexadecimal format. *size* is the size of the memory block, including a unit, to be removed from the domain. Note that you cannot use this property to specify the physical addresses of DIMMs.
- `-g` *resource-group* specifies that the operation is performed on the resources in the specified resource group.
- `-s` *size*[*unit*] specifies the amount of memory to remove. If this option is not specified, the command attempts to remove all memory from the specified resource group that is bound to the specified domain. This option can be used only when the `-g` option is specified.
- *domain-name* specifies the logical domain where the memory is to be removed.

Enter Delayed Reconfiguration Mode

The `start-reconf` subcommand enables the domain to enter delayed reconfiguration mode. Only root domains support delayed reconfiguration.

Note - When a non-`primary` root domain is in a delayed reconfiguration, you can perform only the `add-io`, `set-io`, `remove-io`, `create-vf`, and `destroy-vf` operations.

The syntax for the `start-reconf` subcommand is:

```
ldm start-reconf domain-name
```

Cancel a Delayed Reconfiguration Operation

The `cancel-reconf` subcommand cancels a delayed reconfiguration. Only root domains support delayed reconfiguration.

The syntax for the `cancel-reconf` subcommand is:

```
ldm cancel-reconf domain-name
```

Cancel Operations

The `cancel-operation` subcommand cancels a delayed reconfiguration (`reconf`), memory dynamic reconfiguration removal (`memdr`), or domain migration (`migration`) for a logical domain. Only root domains support the `reconf` operation.

The syntax for the `cancel-operation` subcommand is:

```
ldm cancel-operation migration domain-name
ldm cancel-operation reconf domain-name
ldm cancel-operation memdr domain-name
```

Input/Output Devices

Add Input/Output Device

The `add-io` subcommand attempts to dynamically add a PCIe bus, device, or virtual function to the specified logical domain. If the domain does not support dynamic configuration, the command fails, and you must initiate a delayed reconfiguration or stop the domain before you can add the device.

If you add a root complex to the root domain when `iov=off`, you cannot successfully use the `create-vf`, `destroy-vf`, `add-io`, or `remove-io` subcommand to assign direct I/O and SR-IOV devices.

The syntax for the `add-io` subcommand is:

```
ldm add-io [iov=on|off] bus domain-name
ldm add-io (device | vf-name) domain-name
```

where:

- `iov=on|off` enables or disables I/O virtualization (direct I/O and SR-IOV) operations on the specified PCIe bus (root complex). When enabled, I/O virtualization is supported for devices in that bus. The `ldm add-io` command rebinds the specified PCIe bus to the root domain. The default value is `off`.

  Note that this command fails if the PCIe bus that you want to add is already bound to a domain.
- *bus*, *device*, and *vf-name* are a PCIe bus, a direct I/O-assignable device, and a PCIe SR-IOV virtual function, respectively. Although the operand can be specified as a device path or as a pseudonym, using the device pseudonym is recommended. The pseudonym is based on the ASCII label that is printed on the chassis to identify the corresponding I/O card slot and is platform specific.

  The following are examples of the pseudonyms that are associated with the device names:

  - **PCIe bus.** The `pci_0` pseudonym matches the `pci@400` device path.
  - **Direct I/O-assignable device.** The `PCIE1` pseudonym matches the `pci@400/pci@0/pci@c` device path.
  - **PCIe SR-IOV virtual function.** The `/SYS/MB/NET0/IOVNET.PF0.VF0` pseudonym matches the `pci@400/pci@2/pci@0/pci@6/network@0` device path.

  The specified guest domain must be in the inactive or bound state. If you specify the `primary` domain, this command initiates a delayed reconfiguration.
- *domain-name* specifies the logical domain where the bus or device is to be added.

Set a Property for a Virtual Function

The `set-io` subcommand modifies the current configuration of a virtual function by changing the property values or by passing new properties. This command can modify both the class-specific properties and the device-specific properties.

You can change most network class-specific properties without requiring a reboot of the root domain. However, to change the `mtu` and `mac-addresses` properties of a virtual function that is bound to a domain, you must first stop the domain or initiate a delayed reconfiguration on the root domain.

- All device-specific properties initiate a delayed reconfiguration so that those properties can be updated during the attach operation of the physical function device driver. As a result, the root domain must be rebooted.
- This command only succeeds when the physical function driver can successfully validate the resulting configuration.

The syntax for the `set-io` subcommand is:

```
ldm set-io name=value [name=value...] pf-name
ldm set-io iov=on|off bus
ldm set-io [mac-addr=num] [alt-mac-addrs=[auto|num1,[auto|num2,...]]]
  [pvid=[pvid]] [vid=[vid1,vid2,...]] [mtu=size] [name=value...] net-vf-name
ldm set-io name=[value...] ib-pf-name
ldm set-io [bw-percent=[value]] [port-wwn=value node-wwn=value] fc-vf-name
```

where:

- `alt-mac-addrs=auto|`*num1*`,[auto|`*num2*`,...]` is a comma-separated list of alternate MAC addresses. Valid values are numeric MAC addresses and the `auto` keyword, which can be used one or more times to request that the system generate an alternate MAC address. The `auto` keyword can be mixed with numeric MAC addresses. The numeric MAC address must be in standard octet notation, for example, `80:00:33:55:22:66`.

  You cannot change this property value on a virtual network device in a bound domain. You must first stop the domain or initiate a delayed reconfiguration on the root domain.

  You can assign one or more alternate MAC addresses to create one or more virtual NIC (VNICs) on this device. Each VNIC uses one alternate MAC address, so the number of MAC addresses assigned determines the number of VNICs that can be created on this device. If no alternate MAC addresses are specified, attempts to create VNICs on this device fail. For more information, see the Oracle Solaris 11 networking documentation and Chapter 11, "Using Virtual Networks," in "Oracle VM Server for SPARC 3.2 Administration Guide ".

- `iov=on|off` enables or disables I/O virtualization (direct I/O and SR-IOV) operations on the specified PCIe bus (root complex). When enabled, I/O virtualization is supported for devices in that bus. The default value is `off`.

  To modify the `iov` property value, the root complex must be bound to the domain and the domain must be in a delayed reconfiguration.

- `bw-percent=[`*value*`]` specifies the percentage of the bandwidth to be allocated to the Fibre Channel virtual function. Valid values are from 0 to 100. The total bandwidth value assigned to a Fibre Channel physical function's virtual functions cannot exceed 100. The default value is 0 so that the virtual function gets a fair share of the bandwidth that is not already reserved by other virtual functions that share the same physical function.

- `node-wwn=`*value* specifies the node world-wide name for the Fibre Channel virtual function. Valid values are non-zero. By default, this value is allocated automatically. If you manually specify this value, you must also specify a value for the `port-wwn` property.

  The IEEE format is a two-byte header followed by an embedded MAC-48 or EUI-48 address that contains the OUI. The first two bytes are either hexadecimal 10:00 or 2*x*:*xx* where *x* is vendor-specified) followed by the three-byte OUI and three-byte vendor-specified serial number.

- `port-wwn=`*value* specifies the port world-wide name for the Fibre Channel virtual function. Valid values are non-zero. By default, this value is allocated automatically. If you manually specify this value, you must also specify a value for the `node-wwn` property.

  The IEEE format is a two-byte header followed by an embedded MAC-48 or EUI-48 address that contains the OUI. The first two bytes are either hexadecimal 10:00 or 2*x*:*xx* where *x* is vendor-specified) followed by the three-byte OUI and three-byte vendor-specified serial number.

- *name*=*value* is the name-value pair of a property to set.
- *pf-name* is the name of the physical function.
- *bus* is the name of the PCIe bus.
- *net-vf-name* is the name of the network virtual function.
- *ib-pf-name* is the name of the InfiniBand physical function.
- *fc-vf-name* is the name of the Fibre Channel virtual function.

Set a Property for a Physical Function

The `set-io` subcommand modifies the physical function configuration. Only the physical function device-specific properties are supported. Any change to the properties causes a delayed reconfiguration because the properties are applied during the attach operation of the physical function device driver.

The property values must be an integer or a string. Run the `ldm list-io -d` command to determine the property value type and whether a particular property can be set.

Note that the `ldm set-io` command succeeds only when the physical function driver successfully validates the resulting configuration.

The syntax for the `set-io` subcommand is:

```
ldm set-io name=value [name=value...] pf-name
```

where:

- *name*=*value* is the name-value pair of a property to set.
- *pf-name* is the name of the physical function.

Remove Input/Output Device

The `remove-io` subcommand removes a PCIe bus, device, or virtual function from a specified domain.

The syntax for the `remove-io` subcommand is:

```
ldm remove-io [-n] (bus | device | vf-name) domain-name
```

where:

- `-n` performs a dry run of the command to determine whether it will succeed. It does not actually remove the I/O device.
- *bus*, *device*, and *vf-name* are a PCIe bus, a direct I/O-assignable device, and a PCIe SR-IOV virtual function, respectively. Although the operand can be specified as a device path or as a pseudonym, using the device pseudonym is recommended. The pseudonym is based on the ASCII label that is printed on the chassis to identify the corresponding I/O card slot and is platform specific.

The following are examples of the pseudonyms that are associated with the device names:

- **PCIe bus.** The `pci_0` pseudonym matches the `pci@400` device path.
- **Direct I/O-assignable device.** The `PCIE1` pseudonym matches the `pci@400/pci@0/pci@c` device path.
- **PCIe SR-IOV virtual function.** The `/SYS/MB/NET0/IOVNET.PF0.VF0` pseudonym matches the `pci@400/pci@2/pci@0/pci@6/network@0` device path.

The specified guest domain must be in the inactive or bound state. If you specify the `primary` domain, this command initiates a delayed reconfiguration.

- *domain-name* specifies the logical domain where the bus or device is to be removed.

Virtual Network Server

Add a Virtual Switch

The `add-vsw` subcommand adds a virtual switch to a specified logical domain.

The syntax for the `add-vsw` subcommand is:

```
ldm add-vsw [-q] [default-vlan-id=VLAN-ID] [pvid=port-VLAN-ID] [vid=VLAN-ID1,VLAN-ID2,...]
  [linkprop=phys-state] [mac-addr=num] [net-dev=device] [mode=sc] [mtu=size]
  [id=switch-ID] [inter-vnet-link=on|off] vswitch-name domain-name
```

where:

- `-q` disables the validation of the path to the network device that is specified by the `net-dev` property. This option enables the command to run more quickly, especially if the logical domain is not fully configured.
- `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 switch device needs to be a member, in untagged mode. This property also applies to the `set-vsw` subcommand. See "Using VLAN Tagging" in "Oracle VM Server for SPARC 3.2 Administration Guide ".
- `linkprop=phys-state` specifies whether the virtual device reports its link status based on the underlying physical network device. When `linkprop=phys-state` is specified on the command line, the virtual device link status reflects the physical link state. By default, the virtual device link status does not reflect the physical link state.
- `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" in "Oracle VM Server for SPARC 3.2 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 or aggregation over which this switch operates. The system validates that the path references an actual network device unless the `-q` option is specified.

  When setting this property on a path that includes VLANs, do *not* use the path name that has any VLAN tags.
- `mode=sc` enables virtual networking support for prioritized processing of Oracle Solaris Cluster heartbeat packets in a Logical Domains environment. Applications like Oracle Solaris Cluster need to ensure that high priority heartbeat packets are not dropped by congested virtual network and switch devices. This option prioritizes Oracle Solaris Cluster heartbeat frames and ensures that they are transferred in a reliable manner.

  You must set this option when running Oracle Solaris Cluster in a Logical Domains environment and using guest domains as Oracle Solaris Cluster nodes. Do *not* set this option when you are not running Oracle 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.
- `inter-vnet-link=on|off` specifies whether to assign a channel between each pair of virtual network devices that are connected to the same virtual switch. This behavior improves guest-to-guest performance. The default value is `on`.
- *vswitch-name* is the unique name of the switch that is to be exported as a service. Clients (network) can attach to this service.
- *domain-name* specifies the logical domain in which to add a virtual switch.

Set Options for a Virtual Switch

The `set-vsw` subcommand modifies the properties of a virtual switch that has already been added.

The syntax for the `set-vsw` subcommand is:

```
ldm set-vsw [-q] [pvid=port-VLAN-ID] [vid=VLAN-ID1,VLAN-ID2,...] [mac-addr=num]
  [net-dev=device] [linkprop=[phys-state]] [mode=[sc]] [mtu=size]
  [inter-vnet-link=[on|off]] vswitch-name
```

where:

- `-q` disables the validation of the path to the network device that is specified by the `net-dev` property. This option enables the command to run more quickly, especially if the logical domain is not fully configured.

- pvid=*port-VLAN-ID* specifies the VLAN to which the virtual switch device needs to be a member, in untagged mode. See "Using VLAN Tagging" in "Oracle VM Server for SPARC 3.2 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" in "Oracle VM Server for SPARC 3.2 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 or aggregation over which this switch operates. The system validates that the path references an actual network device unless the -q option is specified.

  When setting this property on a path that includes VLANs, do *not* use the path name that has any VLAN tags.

  Note that using the ldm set-vsw command to specify or update the net-dev property value causes the primary domain to enter a delayed reconfiguration.
- linkprop=phys-state specifies whether the virtual device reports its link status based on the underlying physical network device. When linkprop=phys-state is specified on the command line, the virtual device link status reflects the physical link state. By default, the virtual device link status does not reflect the physical link state. The default situation occurs when the linkprop property is unspecified or when you run the ldm set-vsw command with the linkprop= argument.
- mode=sc enables virtual networking support for prioritized processing of Oracle Solaris Cluster heartbeat packets in a Logical Domains environment. Applications like Oracle Solaris Cluster need to ensure that high priority heartbeat packets are not dropped by congested virtual network and switch devices. This option prioritizes Oracle 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 Oracle Solaris Cluster in a Logical Domains environment and using guest domains as Oracle Solaris Cluster nodes. Do *not* set this option when you are not running Oracle 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.
- inter-vnet-link=on|off specifies whether to assign a channel between each pair of virtual network devices that are connected to the same virtual switch. This behavior improves guest-to-guest performance. The default value is on.
- *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

The remove-vsw subcommand removes a virtual switch.

The syntax for the `remove-vsw` subcommand is:

```
ldm remove-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

The `add-vnet` subcommand adds a virtual network device to the specified logical domain.

```
ldm add-vnet [mac-addr=num] [mode=hybrid] [pvid=port-VLAN-ID]
  [pvlan=secondary-vid,pvlan-type]
  [protection=protection-type[,protection-type],...]
  [allowed-ips=ipaddr[,ipaddr]...] [priority=high|medium|low] [cos=0-7]
  [allowed-dhcp-cids=[macaddr|hostname,macaddr|hostname,...]]
  [alt-mac-addrs=auto|num1[,auto|num2,...]] [vid=VLAN-ID1,VLAN-ID2,...]
  [linkprop=phys-state] [id=network-ID] [mtu=size]
  [maxbw=value] if-name vswitch-name domain-name
```

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`.
- `alt-mac-addrs=auto|`*num1*`,[auto|`*num2*`,...]` is a comma-separated list of alternate MAC addresses. Valid values are numeric MAC addresses and the `auto` keyword, which can be used one or more times to request that the system generate an alternate MAC address. The `auto` keyword can be mixed with numeric MAC addresses. The numeric MAC address must be in standard octet notation, for example, `80:00:33:55:22:66`.

  You can assign one or more alternate MAC addresses to create one or more virtual NIC (VNICs) on this device. Each VNIC uses one alternate MAC address, so the number of MAC addresses assigned determines the number of VNICs that can be created on this device. If no alternate MAC addresses are specified, attempts to create VNICs on this device fail. For more information, see the Oracle Solaris 11 networking documentation and Chapter 11, "Using Virtual Networks," in "Oracle VM Server for SPARC 3.2 Administration Guide ".
- `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.

  Note that the NIU Hybrid I/O feature is deprecated in favor of the SR-IOV feature. See Chapter 8, "Creating an I/O Domain by Using PCIe SR-IOV Virtual Functions," in "Oracle VM Server for SPARC 3.2 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" in "Oracle VM Server for SPARC 3.2 Administration Guide ".
- `pvlan=`*secondary-vid*`,`*pvlan-type* configures a private VLAN (PVLAN). A primary VLAN forwards traffic downstream to its secondary VLANs, which can be either isolated or community. You must also specify the `pvid` property. The `pvlan` property specifies a PVLAN's *secondary-vid*, which is a value from 1-4094, and a *pvlan-type*, which is one of the following values:
  - `isolated` – The ports that are associated with an isolated PVLAN are isolated from all of the peer virtual networks and Oracle Solaris virtual NICs on the back-end network device. The packets reach only the external network based on the values you specified for the PVLAN.
  - `community` – The ports that are associated with a community PVLAN can communicate with other ports that are in the same community PVLAN but are isolated from all other ports. The packets reach the external network based on the values you specified for the PVLAN.
- `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" in "Oracle VM Server for SPARC 3.2 Administration Guide ".
- `mtu=`*size* specifies the maximum transmission unit (MTU) of a virtual network device. Valid values are in the range of `1500-16000`.
- `linkprop=phys-state` specifies whether the virtual network device reports its link status based on the underlying physical network device. When `linkprop=phys-state` is specified on the command line, the virtual network device link status reflects the physical link state. By default, the virtual network device link status does not reflect the physical link state.
- `maxbw=`*value* specifies the maximum bandwidth limit for the specified port in megabits per second. This limit ensures that the bandwidth from the external network (specifically the traffic that is directed through the virtual switch) does not exceed the specified value. This bandwidth limit does not apply to the traffic on the inter-vnet links. You can set the bandwidth limit to any high value. The value is ignored when it is higher than the bandwidth supported by the network back-end device.
- `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.
- `allowed-dhcp-cids=`*macaddr*`|`*hostname*`,`*macaddr*`|`*hostname*`,...`

  Specifies a comma-separated list of MAC addresses or host names. *hostname* can be a host name or a fully qualified host name with a domain name. This name must begin with an alphabetic character. *macaddr* is the numeric MAC address in standard octet notation, for example, `80:00:33:55:22:66`. For more iinformation, see `dhcp_nospoof`.
- `allowed-ips=ipaddr[,`*ipaddr*`,...]`

  Specifies a comma-separated list of IP addresses. For more information, see `ip_nospoof`.
- `cos=0-7`

  Specifies the class of service (802.1p) priority that is associated with outbound packets on the link. When this property is set, all outbound packets on the link have a VLAN tag with its priority field set to this property value. Valid values are 0-7, where 7 is the highest class of service and 0 is the lowest class of service. The default value is 0.

- `priority=`*value*

  Specifies the relative priority of the link, which is used for packet processing scheduling within the system. Valid values are `high`, `medium`, and `low`. The default value is `medium`.

- `protection=`*protection-type*`[,`*protection-type*`]...]`

  Specifies the types of protection (*protection-type*) in the form of a bit-wise OR of the protection types. By default, no protection types are used. The following values are separated by commas:

  - `mac_nospoof` enables MAC address anti-spoofing. An outbound packet's source MAC address must match the link's configured MAC address. Non-matching packets are dropped. This value includes datalink MAC configuration protection.
  - `ip_nospoof` enables IP address anti-spoofing. This protection type works in conjunction with the `allowed-ips` link property, which specifies one or more IP addresses (IPv4 or IPv6). An outbound IP packet can pass if its source address is specified in the `allowed-ips` list. An outbound ARP packet can pass if its sender protocol address is in the `allowed-ips` list. This value includes IP address configuration protection.
  - `dhcp_nospoof` enables DHCP client ID (CID) and hardware address anti-spoofing. By default, this value enables anti-spoofing for the configured MAC address of the device port node. If the `allowed-dhcp-cids` property is specified, DHCP anti-spoofing is enabled for the DHCP client IDs for that node.
  - `restricted` enables packet restriction, which restricts outgoing packet types to only IPv4, IPv6, and ARP packets.

- *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 `remove-vnet` subcommands.
- *vswitch-name* is the name of an existing network service (virtual switch) to which to connect.
- *domain-name* specifies the logical domain to which to add the virtual network device.

Set Options for a Virtual Network Device

The `set-vnet` 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] [pvlan=[secondary-vid,pvlan-type]]
  [protection=[protection-type[,protection-type],...]]
  [allowed-ips=[ipaddr[,ipaddr]...]] [priority=[high|medium|low]] [cos=[0-7]]
  [allowed-dhcp-cids=[macaddr|hostname,macaddr|hostname,...]]
  [alt-mac-addrs=auto|num1[,auto|num2,...]] [vid=VLAN-ID1,VLAN-ID2,...]
  [linkprop=[phys-state]] [mtu=size]
```

`[maxbw=[`*value*`]]` *if-name domain-name*

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`.
- `alt-mac-addrs=auto|`*num1*`,[auto|`*num2*`,...]` is a comma-separated list of alternate MAC addresses. Valid values are numeric MAC addresses and the `auto` keyword, which can be used one or more times to request that the system generate an alternate MAC address. The `auto` keyword can be mixed with numeric MAC addresses. The numeric MAC address must be in standard octet notation, for example, `80:00:33:55:22:66`.

  You can assign one or more alternate MAC addresses to create one or more virtual NIC (VNICs) on this device. Each VNIC uses one alternate MAC address, so the number of MAC addresses assigned determines the number of VNICs that can be created on this device. If no alternate MAC addresses are specified, attempts to create VNICs on this device fail. For more information, see the Oracle Solaris 11 networking documentation and Chapter 11, "Using Virtual Networks," in "Oracle VM Server for SPARC 3.2 Administration Guide ".
- `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.

  Note that the NIU Hybrid I/O feature is deprecated in favor of the SR-IOV feature. See Chapter 8, "Creating an I/O Domain by Using PCIe SR-IOV Virtual Functions," in "Oracle VM Server for SPARC 3.2 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" in "Oracle VM Server for SPARC 3.2 Administration Guide ".
- `pvlan=`*secondary-vid*`,`*pvlan-type* configures a PVLAN. A PVLAN forwards traffic downstream to its secondary VLANs, which can be either isolated or community. You must have at least one `pvid` specified. The `pvlan` property specifies a PVLAN's *secondary-vid*, which is a value from 1-4094, and a *pvlan-type*, which is one of the following values:
  - `isolated` – The ports that are associated with an isolated PVLAN are isolated from all of the peer virtual networks and Oracle Solaris virtual NICs on the back-end network device. The packets reach only the external network based on the values you specified for the PVLAN.
  - `community` – The ports that are associated with a community PVLAN can communicate with other ports that are in the same community PVLAN but are isolated from all other ports. The packets reach the external network based on the values you specified for the PVLAN.
- `linkprop=phys-state` specifies whether the virtual device reports its link status based on the underlying physical network device. When `linkprop=phys-state` is specified on the

command line, the virtual device link status reflects the physical link state. By default, the virtual device link status does not reflect the physical link state. The default situation occurs when the `linkprop` property is unspecified or when you run the `ldm set-vnet` command with the `linkprop=` argument.

- `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" in "Oracle VM Server for SPARC 3.2 Administration Guide ".
- `mtu=`*size* specifies the maximum transmission unit (MTU) of a virtual network device. Valid values are in the range of `1500-16000`.
- `maxbw=`*value* specifies the maximum bandwidth limit for the specified port in megabits per second. This limit ensures that the bandwidth from the external network (specifically the traffic that is directed through the virtual switch) does not exceed the specified value. This bandwidth limit does not apply to the traffic on the inter-vnet links. You can set the bandwidth limit to any high value. The value is ignored when it is higher than the bandwidth supported by the network back-end device.
- `allowed-dhcp-cids=`*macaddr*`|`*hostname*`,`*macaddr*`|`*hostname*`,...`

  Specifies a comma-separated list of MAC addresses or host names. *hostname* can be a host name or a fully qualified host name with a domain name. This name must begin with an alphabetic character. *macaddr* is the numeric MAC address in standard octet notation, for example, `80:00:33:55:22:66`. For more information, see `dhcp_nospoof`.
- `allowed-ips=ipaddr[,`*ipaddr*`,...]`

  Specifies a comma-separated list of IP addresses. For more information, see `ip_nospoof`.
- `cos=0-7`

  Specifies the class of service (802.1p) priority that is associated with outbound packets on the link. When this property is set, all outbound packets on the link have a VLAN tag with its priority field set to this property value. Valid values are 0-7, where 7 is the highest class of service and 0 is the lowest class of service. The default value is 0.
- `priority=`*value*

  Specifies the relative priority of the link, which is used for packet processing scheduling within the system. Valid values are `high`, `medium`, and `low`. The default value is `medium`.
- `protection=`*protection-type*`[,`*protection-type*`]...]`

  Specifies the types of protection (*protection-type*) in the form of a bit-wise OR of the protection types. By default, no protection types are used. The following values are separated by commas:

  - `mac_nospoof` enables MAC address anti-spoofing. An outbound packet's source MAC address must match the link's configured MAC address. Non-matching packets are dropped. This value includes datalink MAC configuration protection.
  - `ip_nospoof` enables IP address anti-spoofing. This protection type works in conjunction with the `allowed-ips` link property, which specifies one or more IP addresses (IPv4 or IPv6). An outbound IP packet can pass if its source address is specified in the `allowed-`

ips list. An outbound ARP packet can pass if its sender protocol address is in the allowed-ips list. This value includes IP address configuration protection.

  - dhcp_nospoof enables DHCP client ID (CID) and hardware address anti-spoofing. By default, this value enables anti-spoofing for the configured MAC address of the device port node. If the allowed-dhcp-cids property is specified, DHCP anti-spoofing is enabled for the DHCP client IDs for that node.
  - restricted enables packet restriction, which restricts outgoing packet types to only IPv4, IPv6, and ARP packets.
- *if-name* is the unique interface name assigned to the virtual network device that you want to set.
- *domain-name* specifies the logical domain in which to modify the virtual network device.

Remove a Virtual Network Device

The remove-vnet subcommand removes a virtual network device from the specified logical domain.

The syntax for the remove-vnet subcommand is:

```
ldm remove-vnet [-f] if-name domain-name
```

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.
- *domain-name* specifies the logical domain from which to remove the virtual network device.

Virtual Disk – Service

Add a Virtual Disk Server

The add-vds subcommand adds a virtual disk server to the specified logical domain.

The syntax for the add-vds subcommand is:

```
ldm add-vds service-name domain-name
```

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.
- *domain-name* specifies the logical domain in which to add the virtual disk server.

Remove a Virtual Disk Server

The `remove-vds` subcommand removes a virtual disk server.

The syntax for the `remove-vds` subcommand is:

```
ldm remove-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

The `add-vdsdev` 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 10, “Using Virtual Disks,” in “Oracle VM Server for SPARC 3.2 Administration Guide ”.

The syntax for the `add-vdsdev` subcommand is:

```
ldm add-vdsdev [-f] [-q] [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.
- `-q` disables the validation of the virtual disk back end that is specified by the *backend* operand. This option enables the command to run more quickly, especially if the logical domain or the back end is not fully configured.
- `options=` are as follows:
  - `ro` – Specifies read-only access
  - `slice` – Exports a back end 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 back end can be a disk, a disk slice, a file, a volume (including ZFS, Solaris Volume Manager, or VxVM), or any disk pseudo device. The disk label can be SMI VTOC, EFI, or no label at all. A back end 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 back end is exported from the service domain. When adding a device, the *volume-name* must be paired with the *backend*. The system validates that the location specified by *backend* exists and can be used as a virtual disk back end unless the `-q` option is specified.
- *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

The `set-vdsdev` subcommand sets options for a virtual disk server. See the “Oracle VM Server for SPARC 3.2 Administration Guide ”.

The syntax for the `set-vdsdev` subcommand is:

```
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 back end 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

The `remove-vdsdev` subcommand removes a device from a virtual disk server.

The syntax for the `remove-vdsdev` subcommand is:

```
ldm remove-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 `remove-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

The `add-vdisk` 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.

When *disk-name* is an mpgroup disk, the `ldm add-vdisk` command does the following:

- Adds the virtual disk to the specified domain
- Selects *volume-name*@*service-name* as the first path to access the virtual disk

The syntax for the `add-vdisk` subcommand is:

```
ldm add-vdisk [timeout=seconds] [id=disk-ID] disk-name volume-name@service-name domain-name
```

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.
- *domain-name* specifies the logical domain in which to add the virtual disk.

Set Options for a Virtual Disk

The `set-visk` 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.

Except when used for mpgroup disks, this command can be used only when the domain is bound or inactive.

When *disk-name* is an mpgroup disk, you can use the `ldm set-vdisk` command to specify the first path to the virtual disk as the value of the `volume` property. The path that you specify as the selected path must already belong to the mpgroup.

Dynamic path selection is available when updated virtual disk drivers are running. To determine the version of the Oracle Solaris OS that contains these updated drivers, see "Oracle VM Server for SPARC 3.2 Administration Guide ".

Dynamic path selection occurs when the first path in an mpgroup disk is changed by using the `ldm set-vdisk` command to set the `volume` property to a value in the form *volume-name*@*service-name*. Only an active domain that supports dynamic path selection can switch to the selected path. If the updated drivers are not running, this path is selected when the Oracle Solaris OS reloads the disk instance or at the next domain reboot.

The syntax for the `set-vdisk` subcommand is:

```
ldm set-vdisk [timeout=seconds] [volume=volume-name@service-name] disk-name  domain-name
```

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.
- *domain-name* specifies the existing logical domain where the virtual disk was previously added.

Set Options for a Virtual Disk

The `remove-vdisk` subcommand removes a virtual disk from the specified logical domain.

The syntax for the `remove-vdisk` subcommand is:

```
ldm remove-vdisk [-f] disk-name domain-name
```

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.
- *domain-name* specifies the logical domain from which to remove the virtual disk.

Virtual Data Plane Channel – Service

Add a Virtual Data Plane Channel Service

The `add-vdpcs` 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.

The syntax for the `add-vdpcs` subcommand is:

```
ldm add-vdpcs vdpcs-service-name domain-name
```

where:

- *vdpcs-service-name* is the name of the virtual data plane channel service that is to be added.
- *domain-name* specifies the logical domain to which to add the virtual data plane channel service.

Remove a Virtual Data Plane Channel Service

The `remove-vdpcs` subcommand removes a virtual data plane channel service. This subcommand should only be used in a Netra Data Plane Software (NDPS) environment.

The syntax for the `remove-vdpcs` subcommand is:

```
ldm remove-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

The `add-vdpcc` 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.

The syntax for the `add-vdpcc` subcommand is:

`ldm add-vdpcc` *vdpcc-name vdpcs-service-name domain-name*

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.
- *domain-name* specifies the logical domain to which to add the virtual data plane channel client.

Remove a Virtual Data Plane Channel Client

The `remove-vdpcc` 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.

The syntax for the `remove-vdpcc` subcommand is:

`ldm remove-vdpcc [-f]` *vdpcc-name domain-name*

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.
- *domain-name* specifies the logical domain from which to remove the virtual data plane channel client.

Virtual Console

Add a Virtual Console Concentrator

The `add-vcc` subcommand adds a virtual console concentrator to the specified logical domain.

The syntax for the `add-vcc` subcommand is:

```
ldm add-vcc port-range=x-y vcc-name domain-name
```

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.
- *domain-name* specifies the logical domain to which to add the virtual console concentrator.

Set Options for a Virtual Console Concentrator

The `set-vcc` subcommand sets options for a specific virtual console concentrator.

The syntax for the `set-vcc` subcommand is:

```
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

The `remove-vcc` subcommand removes a virtual console concentrator from the specified logical domain.

The syntax for the `remove-vcc` subcommand is:

```
ldm remove-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

The `set-vcons` 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.

The syntax for the `set-vcons` subcommand is:

```
ldm set-vcons [port=[port-num]] [group=group] [service=vcc-server]
  [log=[on|off]] domain-name
```

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 Oracle 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.
- `log=[on|off]` enables or disables virtual console logging. Valid values are `on` to enable logging, `off` to disable logging, and a null value (`log=`) to reset to the default value. The default value is `on`.

  Log data is saved to a file called `/var/log/vntsd/`*domain-name*`/console-log` on the service domain that provides the virtual console concentrator service. Console log files are rotated by using the `logadm` command. See the `logadm`(1M) and `logadm.conf`(4) man pages.
- *domain-name* specifies the logical domain in which to set the virtual console.

  You can enable virtual console logging for any guest domain that runs the Oracle Solaris 10 OS or Oracle Solaris 11 OS. The service domain must run the Oracle Solaris 11.1 OS.

Physical Functions and Virtual Functions

Virtual Functions

The PCIe single-root I/O virtualization (SR-IOV) standard enables the efficient sharing of PCIe devices among I/O domains. This standard is implemented in the hardware to achieve near-native I/O performance. SR-IOV creates a number of virtual functions that are virtualized instances of the physical device or function. The virtual functions are directly assigned to I/O domains so that they can share the associated physical device and perform I/O without CPU and hypervisor overhead.

PCIe *physical functions* have complete access to the hardware and provide the SR-IOV capability to create, configure, and manage virtual functions. A PCIe component on the system board or a PCIe plug-in card can provide one or more physical functions. An Oracle Solaris driver interacts with the physical functions that provide access to the SR-IOV features.

PCIe *virtual functions* contain the resources that are necessary for data movement. An I/O domain that has a virtual function can access hardware and perform I/O directly by means of an Oracle Solaris virtual function driver. This behavior avoids the overhead and latency that is involved in the virtual I/O feature by removing any bottlenecks in the communication path between the applications that run in the I/O domain and the physical I/O device in the root domain.

Some of these commands require that you specify an identifier for a physical function or virtual function as follows:

```
pf-name ::= pf-pseudonym | pf-path
vf-name ::= vf-pseudonym | vf-path
```

Use the pseudonym form when referring to a corresponding device. This is the form of the name that is shown in the `NAME` column of the `ldm list-io` output. When you run the `ldm list-io -l` command, the path form of the name appears in the output. The `ldm list-io -p` output shows the pseudonym form as the value of the `alias=` token and the path form as the value of the `dev=` token.

Create a Virtual Function

The `create-vf` subcommand creates a virtual function from a specified physical function by incrementing the number of virtual functions in the specified physical function by one. The new virtual function is assigned the highest number in the sequence of virtual function numbers.

To dynamically create virtual functions, ensure that you set the `iov` property for the parent root complex.

Network class virtual functions *must* have a MAC address assigned, which is assigned by default. To override the default MAC address value, specify another value for the `mac-addr` property.

You can also set class-specific properties and device-specific properties when you create a virtual function. This command succeeds only when the physical function driver successfully validates the resulting configuration. By default, a new virtual function is not assigned to any domain. The virtual function can only be assigned (bound) to an I/O domain after the root domain is rebooted and the virtual function is instantiated in the hardware. Plan ahead by determining whether you want to create multiple virtual functions. If you do, create them one after the other to avoid performing multiple reboots.

The device-specific properties depend on the properties that are exported by the physical function driver. For more information, use the `ldm list-io -d` command. When the command is successful, you see a message about a delayed reconfiguration.

The syntax for the `create-vf` subcommand is:

```
ldm create-vf -n number | max pf-name
ldm create-vf [mac-addr=num] [alt-mac-addrs=[auto|num1,[auto|num2,...]]] [pvid=pvid]
  [vid=vid1,vid2,...] [mtu=size] [name=value...] net-pf-name
ldm create-vf [name=value...] ib-pf-name
ldm create-vf [port-wwn=value node-wwn=value] [bw-percent=[value]] fc-pf-name
```

where:

- `-n` creates *number* virtual functions. If you specify `max` instead of *number*, the maximum number of virtual functions are created for the specified physical function.
- `mac-addr=`*num* is the primary MAC address of the Ethernet virtual function
- `alt-mac-addrs=auto|`*num1*`,[auto|`*num2*`,...]` is a comma-separated list of alternate MAC addresses for the Ethernet virtual function. Valid values are numeric MAC addresses and the `auto` keyword, which can be used one or more times to request that the system generate an alternate MAC address. The `auto` keyword can be mixed with numeric MAC addresses. The numeric MAC address must be in standard octet notation, for example, `80:00:33:55:22:66`.

  You can assign one or more alternate MAC addresses to create one or more virtual NIC (VNICs) on this device. Each VNIC uses one alternate MAC address, so the number of MAC addresses assigned determines the number of VNICs that can be created on this device. If no alternate MAC addresses are specified, attempts to create VNICs on this device fail. For more information, see the Oracle Solaris 11 networking documentation and Chapter 11, “Using Virtual Networks,” in “Oracle VM Server for SPARC 3.2 Administration Guide ”.
- `pvid=`*port-VLAN-ID* is the port VLAN ID (no default value) for the Ethernet virtual function
- `vid=`*VLAN-ID1*`,`*VLAN-ID2*`...` is a comma-separated list of integer VLAN IDs for the Ethernet virtual function.
- `mtu=`*size* is the maximum transmission unit (in bytes) for the Ethernet virtual function.
- *name*=*value* is the name-value pair of a property to specify.
- `bw-percent=[`*value*`]` specifies the percentage of the bandwidth to be allocated to the Fibre Channel virtual function. Valid values are from 0 to 100. The total bandwidth value assigned to a Fibre Channel physical function's virtual functions cannot exceed 100. The default value is 0 so that the virtual function gets a fair share of the bandwidth that is not already reserved by other virtual functions that share the same physical function.
- `node-wwn=`*value* specifies the node world-wide name for the Fibre Channel virtual function. Valid values are non-zero. By default, this value is allocated automatically. If you manually specify this value, you must also specify a value for the `port-wwn` property.

  The IEEE format is a two-byte header followed by an embedded MAC-48 or EUI-48 address that contains the OUI. The first two bytes are either hexadecimal 10:00 or 2*x*:*xx* where *x* is vendor-specified) followed by the three-byte OUI and three-byte vendor-specified serial number.
- `port-wwn=`*value* specifies the port world-wide name for the Fibre Channel virtual function. Valid values are non-zero. By default, this value is allocated automatically. If you manually specify this value, you must also specify a value for the `node-wwn` property.

  The IEEE format is a two-byte header followed by an embedded MAC-48 or EUI-48 address that contains the OUI. The first two bytes are either hexadecimal 10:00 or 2*x*:*xx* where *x* is vendor-specified) followed by the three-byte OUI and three-byte vendor-specified serial number.
- *pf-name* is the name of the physical function.

- *net-pf-name* is the name of the network physical function.
- *ib-pf-name* is the name of the InfiniBand physical function.
- *fc-pf-name* is the name of the Fibre Channel physical function.

Destroy a Virtual Function

The `destroy-vf` subcommand destroys a virtual function from the specified physical function. This command succeeds *only* if the following are true:

- The specified virtual function is not currently assigned to any domain.
- The specified virtual function is the last virtual function in the corresponding physical function.
- The resulting configuration is successfully validated by the physical function driver.
- A successful operation triggers a delayed reconfiguration, as the change to the number of virtual functions can only be done as part of rebooting. See the `create-vf` subcommand for more information.

The syntax for the `destroy-vf` subcommand is:

```
ldm destroy-vf vf-name
ldm destroy-vf -n number | max pf-name
```

where:

- *vf-name* is the name of the virtual function.
- `-n` destroys *number* virtual functions. If you specify `max` instead of *number*, the maximum number of virtual functions are destroyed for the specified physical function.
- *pf-name* is the name of the physical function.

Variables

Add Variable

The `add-variable` subcommand adds one or more variables for a logical domain.

The syntax for the `add-variable` subcommand is:

```
ldm add-variable var-name=[value]... domain-name
```

where:

- *var-name*=*value* is the name-value pair of a variable to add. The value is optional.
- *domain-name* specifies the logical domain in which to add the variable.

Set Variable

The `set-variable` subcommand sets variables for a logical domain.

The syntax for the `set-variable` subcommand is:

```
ldm set-variable var-name=[value]... domain-name
```

where:

- *var-name*=*value* is the name-value pair of a variable to set. The value is optional.
- *domain-name* specifies the logical domain in which to set the variable.

Note - Leaving *value* blank, sets *var-name* to no value.

Remove Variable

The `remove-variable` subcommand removes a variable for a logical domain.

The syntax for the `remove-variable` subcommand is:

```
ldm remove-variable var-name... domain-name
```

where:

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

Other Operations

Start Domains

The `start-domain` subcommand starts one or more logical domains.

The syntax for the `start-domain` subcommand is:

```
ldm start-domain -a
ldm start-domain -i file
ldm start-domain domain-name...
```

where:

- `-a` starts all bound logical domains.
- `-i` *file* specifies an XML configuration file to use in starting the logical domain.
- *domain-name* specifies one or more logical domains to start.

Stop Domains

The `stop-domain` subcommand stops one or more running domains by doing one of the following:

- Sending a `shutdown` request to a domain if it runs the appropriate Logical Domains agent
- Sending a `uadmin` request to a domain if the Oracle Solaris OS is booted

By default, the command first attempts to use `shutdown` to stop the domain. However, if the appropriate Logical Domains agent is not available, the command uses `uadmin` to stop the domain. See the `shutdown`(1M) and `uadmin`(1M) man pages.

You can change this default behavior by setting the `ldmd/default_quick_stop` SMF property. See the `ldmd`(1M) on page 91 man page.

The syntax for the `stop-domain` subcommand is:

```
ldm stop-domain [[-f | -q] | [[-h | -r | -t sec] [-m msg]]] (-a | domain-name...)
```

where:

- `-a` stops all running logical domains except the control domain.
- `-f` attempts to force a running logical domain to stop. Use only if the domain cannot be stopped by any other means.
- `-h` uses *only* the `shutdown` command to halt the operating system and stop the domain. This option does not fall back to using the `uadmin` command.
- `-m` *msg* specifies the message to send to the domains to be shut down or rebooted.

  The *msg* string must be enclosed within quotation marks if the string contains white space.
- `-q` issues a quick stop of the specified domain by issuing a `uadmin` command.
- `-r` uses the `shutdown` command to stop and reboot the operating system.
- `-t` *sec* waits for the end of the domain shutdown sequence at least *sec* seconds before reissuing the command with the `-q` option to shut down any specified domains that are still running. The command is only reissued if the domain shutdown request does not complete in time. *sec* must be a value greater than `0`.

  Note that if the shutdown request cannot be performed for a particular domain, the command immediately falls back to the `-q` option for that domain.
- *domain-name* specifies one or more running logical domains to stop.

To perform a graceful Oracle Solaris shutdown on a domain that is not running the supporting Logical Domains agent version, perform a `shutdown` or `init` operation in the domain itself. See the `init`(1M) man page. To determine whether a domain is running a version of the Logical Domains agent that supports graceful shutdown, run the `ldm stop-domain -h` command, which only executes a graceful shutdown.

Panic Oracle Solaris OS

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

The syntax for the `panic-domain` subcommand is:

```
ldm panic-domain domain-name
```

domain-name specifies the logical domain to panic.

Provide Help Information

The `ldm --help` command 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

The `ldm --version` command provides version information.

```
ldm --version
ldm -V
```

Bind Resources to a Domain

The `bind-domain` subcommand binds, or attaches, configured resources to a logical domain.

The syntax for the `bind-domain` subcommand is:

```
ldm bind-domain [-f] [-q] -i file
ldm bind-domain [-f] [-q] domain-name
```

where:

- `-f` attempts to force the binding of the domain even if invalid network or disk back-end devices are detected.
- `-q` disables the validation of network or disk back-end devices so that the command runs more quickly.
- `-i` *file* specifies an XML configuration file to use in binding the logical domain.
- *domain-name* specifies the logical domain to which to bind resources.

Unbind Resources From a Domain

The `unbind-domain` subcommand releases resources bound to configured logical domains.

The syntax for the `unbind-domain` subcommand is:

```
ldm unbind-domain domain-name
```

domain-name specifies the logical domain from which to unbind resources.

Configure Operations

Add a Logical Domain Configuration

The `add-spconfig` 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 SP.

The syntax for the `add-spconfig` subcommand is:

```
ldm add-spconfig config-name
ldm add-spconfig -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.

  Updates the specified configuration based on the autosave information. Note that you must perform a power cycle after using this command to instantiate the updated configuration.
- *new-config-name* is the name of the logical domain configuration to add.

Set a Logical Domain Configuration

The `set-spconfig` subcommand enables you to specify the logical domain configuration to use at the next system power cycle. The configuration is stored on the SP.

The syntax for the `set-spconfig` subcommand is:

```
ldm set-spconfig 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:

```
primary# ldm set-spconfig factory-default
```

Remove a Logical Domain Configuration

The `remove-spconfig` subcommand removes a logical domain configuration that is stored on the SP, as well as any corresponding autosave configuration from the control domain.

The syntax for the `remove-spconfig` subcommand is:

```
ldm remove-spconfig [-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 Operations

Flags in list Subcommand Output

The following flags can be shown in the output for a domain (ldm list). If you use the long, parseable options (-l -p) for the command, the flags are spelled out; for example, flags=normal,control,vio-service. If not, you see the letter abbreviation; for example -n-cv-. The list flag values are position dependent. Following are the values that can appear in each of the six columns from left to right.

Column 1 – Starting or stopping domains

- s starting or stopping

Column 2 – Domain status

- n normal
- t transition
- d degraded domain that cannot be started due to missing resources

Column 3 – Reconfiguration status

- d delayed reconfiguration
- r memory dynamic reconfiguration

Column 4 – Control domain

- c control domain

Column 5 – Service domain

- v virtual I/O service domain

Column 6 – Migration status

- s source domain in a migration
- t target domain in a migration
- e error occurred during a migration

List Domains and States

The list-domain subcommand lists logical domains and their states. If you do not specify a logical domain, all logical domains are listed.

The syntax for the `list-domain` subcommand is:

```
ldm list-domain [-e] [-l] [-o format] [-p] [-S] [domain-name...]
```

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 each format by a comma with no spaces.
  - `cmi` – Output contains information about CMI devices, which includes the shared memory and the associated virtual CPUs and cores that are bound to the domain.
  - `console` – Output contains the virtual console (`vcons`) and virtual console concentrator (`vcc`) service.
  - `core` – Output contains information about cores, core ID and physical CPU set.
  - `cpu` – Output contains information about the CPU thread (`vcpu`), physical CPU (`pcpu`), and core ID (`cid`).
  - `crypto` – Cryptographic unit output contains the Modular Arithmetic Unit (`mau`) and any other 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 the variables (`var`), host ID (`hostid`), domain state, flags, universally unique identifier (UUID), software state, utilization percentage, normalized 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`).
  - `resmgmt` – Output contains DRM policy information, indicates which policy is currently running, and indicates whether the `whole-core`, `max-core`, and `threading` constraints are enabled.
  - `serial` – Output contains the virtual logical domain channel (`vldc`) service, virtual logical domain channel client (`vldcc`), virtual data plane channel client (`vdpcc`), and virtual data plane channel service (`vdpcs`).
  - `status` – Output contains the status of a migrating domain and a memory dynamic reconfiguration operation.

    You can use the `-o status` option to show the status of any migration operations or DR operations that are in progress. This information is derived from the flags in the `FLAGS` field. The `-o status` option does not relate to the `STATE` field.
- `-p` generates the list in a parseable, machine-readable format.

- `-S` generates status information about CPU-related and memory-related resources. Status values are `ok` to indicate that the resource is operating normally and `fail` to indicate that the resource is faulty.

  This status is only determined for CPU and memory resources on the Fujitsu M10 platform. On all other platforms, the status field is only shown in parseable output when the `-p` option is used. The status on these platforms is always shown as `status=NA`.
- *domain-name* is the name of the logical domain for which to list state information.

List Bindings for Domains

The `list-bindings` subcommand lists bindings for logical domains. If no logical domains are specified, all logical domains are listed.

If you specify the name of a domain, any alternate MAC addresses for a virtual network device are shown after the MAC address of the control domain. The following command shows the three alternate MAC addresses for `vnet1` on the `ldg1` domain:

```
primary# ldm list-bindings ldg1
...
NETWORK
NAME  SERVICE              ID DEVICE    MAC               MODE PVID VID MTU LINKPROP
vnet1 primary-vsw0@primary 0  network@0 00:14:4f:f8:0c:80      1        1500
                                        00:14:4f:fa:3a:f9
                                        00:14:4f:f9:06:ab
                                        00:14:4f:fb:3d:af

PEER                 MAC               MODE PVID VID MTU LINKPROP
primary-vsw0@primary 00:14:4f:fa:94:60      1        1500
vnet2@ldg2           00:14:4f:f9:38:d1      1        1500
vnet3@ldg3           00:14:4f:fa:60:27      1        1500
vnet4@ldg4           00:14:4f:f8:0f:41      1        1500
...
```

The following command shows the three alternate MAC addresses for `vnet1` on the `ldg1` domain in parseable output:

```
primary# ldm list-bindings -p ldg1
...
VNET|name=vnet1|dev=network@0|service=primary-vsw0@primary|mac-addr=00:14:4f:f8:0c:80
  |mode=|pvid=1|vid=|mtu=1500|linkprop=|id=0
|alt-mac-addr=00:14:4f:fa:3a:f9,00:14:4f:f9:06:ab,00:14:4f:fb:3d:af
|peer=primary-vsw0@primary|mac-addr=00:14:4f:fa:94:60|mode=|pvid=1|vid=|mtu=1500
|peer=vnet2@ldg2|mac-addr=00:14:4f:f9:38:d1|mode=|pvid=1|vid=|mtu=1500|linkprop=
|peer=vnet3@ldg3|mac-addr=00:14:4f:fa:60:27|mode=|pvid=1|vid=|mtu=1500|linkprop=
|peer=vnet4@ldg4|mac-addr=00:14:4f:f8:0f:41|mode=|pvid=1|vid=|mtu=1500|linkprop=
...
```

The `ldm list-bindings` command shows the following information about mpgroup disks:

- The `STATE` column shows one of the following states for each mpgroup path:
  - `active` indicates the current active path of the mpgroup

  - `standby` indicates that the path is not currently used
  - `unknown` indicates that the disk is unattached or in the midst of changing state, or that the specified domain does not run an OS that supports dynamic path selection
- The list of paths are shown in the order used for the driver to choose the active path (the first path listed is chosen first)
- The volume that is associated with the disk is the selected mpgroup path and is listed first

In this example, the selected path is `vol-ldg2@opath-ldg2`. The `ldm list-bindings` output shows that the active path is `vol-ldg1@opath-vds` instead of the selected path. This situation might occur if the selected path failed for some reason and the driver chose the second path from the list to be active. Even if your selected path becomes available in the meantime, the driver-chosen path continues as the active path. To make the first path active again, re-issue the `ldm set-vdisk` command to set the `volume` property to the name of the path you want, `vol-ldg1@opath-vds`.

```
primary# ldm list-bindings
DISK
    NAME       VOLUME                 TOUT ID DEVICE SERVER  MPGROUP
    disk       disk-ldg4@primary-vds0      0  disk@0 primary
    tdiskgroup vol-ldg2@opath-ldg2         1  disk@1 ldg2    testdiskgroup
        PORT MPGROUP VOLUME        MPGROUP SERVER STATE
        2    vol-ldg2@opath-ldg2   ldg2           standby
        0    vol-ldg1@opath-vds    ldg1           active
        1    vol-prim@primary-vds0 primary        standby
```

The syntax for the `list-bindings` subcommand is:

`ldm list-bindings [-e] [-p] [`*domain-name*`...]`

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.
- *domain-name* is the name of the logical domain for which you want binding information.

List Services for Domains

The `list-services` subcommand lists all the services exported by logical domains. If no logical domains are specified, all logical domains are listed.

The syntax for the `list-services` subcommand is:

`ldm list-services [-e] [-p] [`*domain-name*`...]`

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.
- *domain-name* is the name of the logical domain for which you want services information.

List Constraints for Domains

The list-constraints subcommand lists the constraints for the creation of one or more logical domains. If no logical domains are specified, all logical domains are listed.

Any resource that has been evacuated from the physical domain by a recovery mode operation has an asterisk (*) in front of its resource identifier.

The syntax for the list-constraints subcommand is:

```
ldm list-constraints [-x] [domain-name...]
ldm list-constraints [-e] [-p] [domain-name...]
```

where:

- -x writes the constraint output in XML format to the standard output (stdout) format. This output can be used as a backup.
- *domain-name* 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 CPU Core Activation Information

The list-permits subcommand lists CPU core activation information on the Fujitsu M10 platform. The PERMITS column shows the total number of CPU core activations that have been issued. This total includes all permanent CPU core activations and pay-per-use CPU core activations. A *permanent CPU core activation* is a permit for a resource that can be used for an unlimited amount of time. A *pay-per-use CPU core activation* is a permit for a resource that can be used for a limited amount of time. The number of issued permanent CPU core activations is shown in the PERMANENT column. The IN USE column shows the number of issued CPU core activations that are in use. The REST column shows the number of CPU core activations that are available for use.

The syntax for the list-permits subcommand is:

```
ldm list-permits
```

List Devices

The list-devices subcommand lists either free (unbound) resources or all server resources. The default is to list all free resources.

The syntax for the list-devices subcommand is:

```
ldm list-devices [-a] [-p] [-S] [cmi] [core] [cpu] [crypto] [io] [memory]
```

where:

- `-a` lists all server resources, bound and unbound.
- `-p` writes the constraint output in a parseable, machine-readable form.
- `-S` generates status information about CPU-related and memory-related resources. Status values are `ok` to indicate that the resource is operating normally and `fail` to indicate that the resource is faulty.

  This status is only determined for CPU and memory resources on the Fujitsu M10 platform. On all other platforms, the status field is only shown in parseable output when the `-p` option is used. The status on these platforms is always shown as `status=NA`.
- `cmi` lists information about CMI devices, which includes the shared memory and any unallocated virtual CPUs and cores that are associated with those devices.
- `core` lists information about cores, the core ID and physical CPU set, and specifies which CPUs in the core are still unallocated.
- `cpu` lists CPU thread and physical 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, a network, or direct I/O-assignable devices.

Note that resource IDs might have gaps in their numbering. The following example indicates that core 2 is unavailable or might have been disabled:

```
primary# ldm list-devices -a core
CORE
    ID      %FREE   CPUSET
    0       0      (0, 1, 2, 3, 4, 5, 6, 7)
    1       100    (8, 9, 10, 11, 12, 13, 14, 15)
    3       100    (24, 25, 26, 27, 28, 29, 30, 31)
    4       100    (32, 33, 34, 35, 36, 37, 38, 39)
    5       100    (40, 41, 42, 43, 44, 45, 46, 47)
    6       100    (48, 49, 50, 51, 52, 53, 54, 55)
```

List I/O Devices

The `list-io` subcommand lists the I/O devices that are configured on the system. The list of devices includes I/O buses (including NIUs) and direct I/O-assignable devices.

The output is divided into the following sections:

- **I/O bus information.** The `IO` column lists the device path of the bus or network device, and the `PSEUDONYM` column shows the associated pseudonym for the bus or network device. The `DOMAIN` column indicates the domain to which the device is currently bound.
- **Direct I/O-assignable devices.** The `PCIE` column lists the device path of the device, and the `PSEUDONYM` column shows the associated pseudonym for the device.

The STATUS column applies to slots that accept plug-in cards as well as to devices on a motherboard and can have one of the following values:

- UNK – The device in the slot has been detected by the firmware, but not by the OS.
- OCC – The device has been detected on the motherboard or is a PCIe card in a slot.
- IOV – The bus has been initialized to share its IOV resources.
- INV – The slot, virtual function, or physical function is in an invalid state and cannot be used.
- EMP – The slot is empty.

Slots that represent on-board devices always have the status of OCC. If the root domain does not support direct I/O, the slot status is UNK.

The syntax for the list-io subcommand is:

```
ldm list-io [-l] [-p] [bus | device | pf-name]
ldm list-io -d pf-name
```

Any resource that has been evacuated from the physical domain by a recovery mode operation has an asterisk (*) in front of its resource identifier.

where:

- -l lists information about subdevices that are hosted by direct I/O-assignable devices. Note that this output indicates which devices will be loaned with the direct I/O-assignable device to the receiving domain. The subdevice names *cannot* be used for command input.
- -p writes the output in a parseable, machine-readable form.
- -d *pf-name* lists information about the specified physical function.
- *bus*, *device*, and *pf-name* are the names of a PCIe bus, a direct I/O-assignable device, and a PCIe SR-IOV physical function, respectively.

List Logical Domain Configurations

The list-spconfig subcommand lists the logical domain configurations stored on the SP.

The syntax for the list-spconfig subcommand is:

```
ldm list-spconfig [-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 on *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 list-spconfig -r command, the autosave configuration is shown as being newer than the current configuration.

List Variables

The `list-variable` subcommand lists one or more variables for a logical domain. To list all variables for a domain, leave the *var-name* blank.

The syntax for the `list-variable` subcommand is:

```
ldm list-variable [var-name...] domain-name
```

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.
- *domain-name* is the name of the logical domain for which to list one or more variables.

List Network Devices

The `list-netdev` subcommand lists the network devices that are configured on the system. The information provided about the device includes the following:

- `CLASS` – One of the following network device types:
  - `AGGR` – Network aggregation
  - `EOIB` – Ethernet over InfiniBand
  - `ESTUB` – Ethernet stub
  - `IPMP` – IP network multipathing group
  - `PART` – InfiniBand partition
  - `PHYS` – Physical network device
  - `VLAN` – Virtual local area network
  - `VNET` – Virtual network device
  - `VNIC` – Virtual network interface card
  - `VSW` – Virtual switch device
  - `VXLAN` – Virtual extended LAN
- `MEDIA` – Network media type, which can be `ETHER` for Ethernet or `IB` for InfiniBand
- `STATE` – State of the network device, which can be `up`, `down`, or `unknown`
- `SPEED` – Speed of the network device in megabits per second
- `OVER` – Physical device over which the network device is mapped
- `LOC` – Location of the network device

The syntax for the `list-netdev` subcommand is:

```
ldm list-netdev [-b] [-l] [-p] [-o net-device] [domain-name]
```

where:

- `-b` enables you to list only the valid virtual switch backend devices.
- `-l` lists information about network devices, virtual switch devices, and aggregations.
- `-p` writes the output in a parseable, machine-readable form.
- `-o` *net-device* lists information about the specified network device.
- *domain-name* specifies the logical domain for which to list network device information.

List Network Device Statistics

The `list-netstat` subcommand lists statistics for the network devices that are configured on the system. Statistical information is shown in the following fields:

- `IPACKETS` shows the inbound packets
- `OPACKETS` shows the outbound packets
- `RBYTES` shows the raw byte count
- `OBYTES` shows the transmitted (outbound) byte count

The syntax for the `list-netstat` subcommand is:

```
ldm list-netstat [-p] [-u unit] [-o net-device] [-t interval [-c count]] [domain-name]
```

where:

- `-c` *count* specifies the number of times to report statistics. A value of 0 reports statistics indefinitely. You must specify the time interval with the `-t` if you specify the `-c`.
- `-o` *net-device* lists information about the specified network device.
- `-p` writes the output in a parseable, machine-readable form.
- `-t` *interval* specifies an interval in seconds at which statistics are refreshed. The default value is one second.
- `-u` *unit* specifies the unit in which to show output. Valid values are:
  - `r` specifies raw bytes
  - `k` specifies kilobytes
  - `m` specifies megabytes
  - `g` specifies gigabytes
- *domain-name* specifies the logical domain for which to list network device information.

List Dependencies

The `list-dependencies` subcommand lists the dependencies within domains. When you specify no options, this command outputs a list of domains and the domains on which they depend.

The syntax for the `list-dependencies` subcommand is:

```
ldm list-dependencies [-l] [-p] [-r] [domain-name]
```

where:

- `-l` lists detailed information about dependencies.
- `-p` writes the output in a parseable, machine-readable form.
- `-r` shows dependents grouped by their dependencies.
- *domain-name* specifies the logical domain for which to list dependency information. If *domain-name* is not specified, dependency information is listed for all domains.

List Resource Groups

The `list-rsrc-group` subcommand shows information about a resource group. When you specify no options, this command produces a short listing for all resource groups in the system.

The syntax for the `list-rsrc-group` subcommand is:

```
ldm list-rsrc-group [-a] [-d domain-name] [-l] [-o core|memory|io] [-p] [resource-group]
```

where:

- `-a` lists information about all resources for each resource group. The output includes resources that are not bound to any domain.
- `-d` *domain-name* shows information only about the specified domain.
- `-l` lists detailed information about each resource group.
- `-o core|memory|io` lists information only about the specified resource type: core, memory, or I/O.
- `-p` writes the output in a parseable, machine-readable form.
- *resource-group* specifies the resource group.

List CMI Devices

The `list-cmi` subcommand shows information about the CMI devices that are configured on a Fujitsu M10 server. If no CMI devices are configured on the system, no output is shown.

The output is divided into the following sections:

- Allocation of CMI devices and the associated virtual CPUs and cores.
  - **Bound.** Shows domains that have CMI devices and the associated virtual CPUs and cores bound to those domains. These domains are targets for the `grow-cmi` and `shrink-cmi` subcommands.
  - **Tenant.** Shows domains that do not own any CMI devices and the associated virtual CPUs and cores bound to those domains. These domains are targets for the `evict-cmi` subcommand.

    - **Free.** Shows unallocated CMI devices and any unallocated virtual CPUs or cores that are associated with those devices.
- Message queue information.
- Shared memory information.

When you specify the `-l` option, additional information is shown for the virtual CPUs and cores that are associated with CMI devices. Furthermore, information about all virtual CPUs and cores that are associated with each CMI device is prepended to the output, unless *domain-name* is specified.

The syntax for the `list-cmi` subcommand is:

```
ldm list-cmi [-l] [-p] [cmi_id=id[,id[,...]]] [domain-name...]
```

where:

- `-l` lists physical CPU sets and core IDs.
- `-p` writes the output in a parseable, machine-readable form.
- `cmi_id=`*id*`[,...]` specifies one or more CMI devices for which to list information.
- *domain-name* specifies one or more logical domains for which to list information.

List CPU Sockets

The `list-socket` subcommand shows information about CPU sockets on a Fujitsu M10 server.

The output is divided into the following sections:

- Allocation of each CPU socket's virtual CPUs and cores.
    - **Tenant.** Shows logical domains and the virtual CPUs and cores bound to those domains.
    - **Free.** Shows any unallocated virtual CPUs or cores.
- Allocation of each CPU socket's memory.

When you specify the -l option, additional information is shown for vcpus and cores. Furthermore, information about all virtual CPUs and cores in each CPU socket is prepended to the output, unless *domain-name* is specified.

The syntax for the `list-socket` subcommand is:

```
ldm list-socket [-l] [-p] [socket_id=id[,id[,...]]] [domain-name...]
```

where:

- `-l` lists physical CPU sets and core IDs.
- `-p` writes the output in a parseable, machine-readable form.
- `socket_id=`*id*`[,...]` specifies one or more CPU sockets for which to list information.
- *domain-name* specifies one or more logical domains for which to list information.

Add, Set, and Remove Resource Management Policies

Add a Resource Management Policy

The `add-policy` subcommand enables you to add a resource management policy for one or more logical domains. A resource management policy consists of optional properties and their values.

The syntax for the `add-policy` subcommand is:

```
ldm add-policy [enable=yes|no] [priority=value] [attack=value] [decay=value]
  [elastic-margin=value] [sample-rate=value] [tod-begin=hh:mm[:ss]]
  [tod-end=hh:mm[:ss]] [util-lower=percent] [util-upper=percent] [vcpu-min=value]
  [vcpu-max=value] name=policy-name domain-name...
```

where:

- The properties are described in the Properties section.
- *domain-name* specifies the logical domain for which to add a resource management policy.

Modify a Resource Management Policy

The `set-policy` subcommand enables you to modify a resource management policy for one or more logical domains by specifying values for optional properties.

The syntax for the `set-policy` subcommand is:

```
ldm set-policy [enable=[yes|no]] [priority=[value]] [attack=[value]] [decay=[value]]
  [elastic-margin=[value]] [sample-rate=[value]] [tod-begin=[hh:mm:ss]]
  [tod-end=[hh:mm:ss]] [util-lower=[percent]] [util-upper=[percent]] [vcpu-min=[value]]
  [vcpu-max=[value]] name=policy-name domain-name...
```

where:

- The properties are described in the Properties section.
- *domain-name* specifies the logical domain for which to modify the resource management policy.

Remove a Resource Management Policy

The `remove-policy` subcommand enables you to remove a resource management policy from a logical domain by specifying one or more policy names.

The syntax for the `remove-policy` subcommand is:

```
ldm remove-policy [name=]policy-name... domain-name
```

where:

- The `name` property specifies the name of the resource management policy, *policy-name*.

- *domain-name* specifies the logical domain on which to remove the resource management policy.

Configure or Reconfigure a Domain From an XML File

The `init-system` subcommand enables you to use an existing configuration to configure one or more guest domains, the control domain, or both types of domains. The `ldm init-system` command takes an XML file (such as the output of `ldm list-constraints -x`) as input, configures the specified domains, and reboots the control domain. Run this command with the factory default configuration.

The syntax for the `init-system` subcommand is:

```
ldm init-system [-frs] -i file
```

where:

- `-i` *file* specifies the XML configuration file to use to create the logical domain.
- `-f` skips the factory-default configuration check and continues irrespective of what was already configured on the system.

  Use the `-f` option with caution. `ldm init-system` assumes that the system is in the factory-default configuration, and so *directly* applies the changes that are specified by the XML file. Using `-f` when the system is in a configuration other than the factory default will likely result in a system that is not configured as specified by the XML file. One or more changes might fail to be applied to the system depending on the combination of changes in the XML file and the initial configuration.
- `-r` reboots the system after configuration.
- `-s` restores only the virtual services configuration (`vds`, `vcc`, and `vsw`).

Collect Hypervisor Dump Data

The hypervisor dump data collection subcommands apply to the process that collects data from a hypervisor dump on the Fujitsu M10 platform only.

When a hypervisor abort event occurs, the contents of the hypervisor memory are preserved by the firmware, and the system is rebooted with the factory-default configuration. The `ldmd` daemon copies the preserved contents of hypervisor memory to a file on the control domain that is called `/var/opt/SUNWldm/hvdump.`*N*`.gz`. *N* is a number in the range `0-7`, inclusive. This file is a binary dump of the contents of hypervisor memory at the time the hypervisor abort occurred.

List Hypervisor Dump Data

The `list-hvdump` subcommand shows the values of the `hvdump` and `hvdump-reboot` properties that govern the hypervisor data collection process that can be used on the Fujitsu M10 platform.

```
ldm list-hvdump
```

Set Property Values for the Hypervisor Data Collection Process

The set-hvdump subcommand modifies the Fujitsu M10 hypervisor data collection properties. You can set properties that enable or disable the automatic hypervisor data collection process. You can also set properties that enable or disable an automatic reboot to restore the original configuration after collecting the data.

The syntax for the set-hvdump subcommand is:

```
ldm set-hvdump [hvdump=on|off] [hvdump-reboot=on|off]
```

where:

- hvdump=on|off enables or disables the hypervisor data collection process. The default value is on.
- hvdump-reboot=on|off enables or disables an automatic system reboot after the hypervisor data collection process completes. The default value is off.

Manually Start the Hypervisor Data Collection Process

The start-hvdump subcommand manually starts the Fujitsu M10 hypervisor data collection process if the automatic collection fails.

```
ldm start-hvdump
```

Perform CMI Operations

The CMI-related subcommands pertain only to the Fujitsu M10 platform.

In all of the CMI-related subcommands, specifying the number of CMI devices automatically selects the CMI resources to be assigned or removed. To explicitly assign or remove CMI resources, provide CMI ID values with the cmi_id property.

When you perform CMI-related operations, the domain must be inactive or, if it is the primary domain, the domain must be in a delayed reconfiguration.

Add CMI Devices

The add-cmi subcommand adds the specified number of CMI devices to a domain.

If you bind an inactive domain with a CMI constraint, unspecified virtual CPU, core, and memory constraints are automatically generated from the CMI constraint and available resources.

The syntax for the add-cmi subcommand is:

```
ldm add-cmi num domain-name
ldm add-cmi cmi_id=id[,id[,...]] domain-name
```

where:

- *num* specifies the number of CMI resources to assign to the domain.
- `cmi_id=`*id*`[,...]` specifies one or more CMI devices to add to a domain.
- *domain-name* specifies one or more logical domains to which to assign the CMI devices.

Modify CMI Devices

The `set-cmi` subcommand specifies the number of CMI devices to assign to a domain.

You can use the `-f` option to clear all existing virtual CPU, core, and memory constraints. When you bind an inactive domain with a CMI constraint, these constraints are then automatically generated from the CMI constraint and available resources.

The syntax for the `set-cmi` subcommand is:

```
ldm set-cmi [-f] num domain-name
ldm set-cmi [-f] cmi_id=id[,id[,...]] domain-name
```

where:

- `-f` clears all existing virtual CPU, core, and memory constraints.
- *num* specifies the number of CMI resources to assign to the domain.
- `cmi_id=`*id*`[,...]` specifies one or more CMI devices to add to a domain. `cmi_id=` removes all specified CMI devices.
- *domain-name* specifies the domain to which the CMI devices are assigned.

Remove CMI Devices

The `remove-cmi` subcommand specifies the number of CMI devices to remove from a domain.

The syntax for the `remove-cmi` subcommand is:

```
ldm remove-cmi [-f] num domain-name
ldm remove-cmi [-f] cmi_id=id[,id[,...]] domain-name
```

where:

- *num* specifies the number of CMI resources to remove from the domain.
- `cmi_id=`*id*`[,...]` specifies one or more CMI devices to remove from a domain.
- *domain-name* specifies the domain from which the CMI devices are removed.

Add CMI Device CPU Threads or CPU Cores

The `grow-cmi` subcommand enables you to add virtual CPUs or cores associated with a particular CMI device to a domain with one or more CMI resources. The specified CMI device must be assigned to the domain, and the domain must be bound or active.

The syntax for the `grow-cmi` subcommand is:

```
ldm grow-cmi vcpus=num cmi_id=id domain-name
ldm grow-cmi cores=num cmi_id=id domain-name
```

where:

- vcpus=*num* specifies the number of virtual CPUs to add to a domain.
- cores=*num* specifies the number of cores to add to a domain.
- cmi_id=*id* specifies a CMI device that is owned by the domain.
- *domain-name* specifies the domain to which the virtual CPUs or cores are added.

Remove CMI Device CPU Threads or CPU Cores

The shrink-cmi subcommand enables you to remove virtual CPUs or cores associated with a particular CMI device from a domain with one or more CMI resources. The specified CMI device must be assigned to the domain, and the domain must be bound or active.

The syntax for the shrink-cmi subcommand is:

```
ldm shrink-cmi vcpus=num cmi_id=id domain-name
ldm shrink-cmi cores=num cmi_id=id domain-name
```

where:

- vcpus=*num* specifies the number of virtual CPUs to remove from a domain.
- cores=*num* specifies the number of cores to remove from a domain.
- cmi_id=*id* specifies a CMI device that is owned by the domain.
- *domain-name* specifies the domain to which the virtual CPUs or cores are removed.

Evict CMI Device CPU Threads or CPU Cores

The evict-cmi subcommand enables you to remove virtual CPUs or cores associated with a particular CMI device from a bound or active domain that has no CMI devices assigned to it.

Run the list-cmi subcommand to determine the allocation of CMI devices and their associated virtual CPUs and cores.

The syntax for the evict-cmi subcommand is:

```
ldm evict-cmi vcpus=num cmi_id=id domain-name
ldm evict-cmi cores=num cmi_id=id domain-name
```

where:

- vcpus=*num* specifies the number of virtual CPUs to remove from a domain.
- cores=*num* specifies the number of cores to remove from a domain.
- cmi_id=*id* specifies a CMI device.
- *domain-name* specifies the domain from which the virtual CPUs or cores are removed.

Perform CPU Socket Operations

The CPU-socket-related commands pertain only to the Fujitsu M10 platform.

Add CPU Socket Threads, Cores, or Memory

The `grow-socket` subcommand enables you to add virtual CPUs, cores, or memory associated with a particular CPU socket to a domain. The domain must be bound or active.

If the domain has any CMI devices assigned to it, use the `grow-cmi` subcommand to add virtual CPUs or cores to the domain.

The syntax for the `grow-socket` subcommand is:

```
ldm grow-socket vcpus=num socket_id=id domain-name
ldm grow-socket cores=num socket_id=id domain-name
ldm grow-socket memory=size[unit] socket_id=id domain-name
```

where:

- `vcpus=`*num* specifies the number of virtual CPUs to add to a domain.
- `cores=`*num* specifies the number of cores to add to a domain.
- `memory=`*num* specifies the amount of memory to remove from a domain. The default amount is *size* in bytes. If you want a different unit of measurement, specify *unit* as one of the following values using either uppercase or lowercase:
  - `G` for gigabytes
  - `K` for kilobytes
  - `M` for megabytes
- `socket_id=`*id* specifies the CPU socket.
- *domain-name* specifies the domain to which the virtual CPUs, cores, or memory are added.

Remove CPU Socket Threads, Cores, or Memory

The `shrink-socket` subcommand enables you to remove virtual CPUs, cores, or memory associated with a particular CPU socket from a domain. The domain must be bound or active.

If the domain has any CMI devices assigned to it, use the `shrink-cmi` subcommand to remove virtual CPUs or cores from the domain.

The syntax for the `shrink-socket` subcommand is:

```
ldm shrink-socket vcpus=num socket_id=id domain-name
ldm shrink-socket cores=num socket_id=id domain-name
ldm shrink-socket memory=size[unit] socket_id=id domain-name
```

where:

- `vcpus=`*num* specifies the number of virtual CPUs to remove from a domain.
- `cores=`*num* specifies the number of cores to remove from a domain.
- `memory=`*num* specifies the amount of memory to remove from a domain. The default amount is *size* in bytes. If you want a different unit of measurement, specify *unit* as one of the following values using either uppercase or lowercase:

- `G` for gigabytes
  - `K` for kilobytes
  - `M` for megabytes
- `socket_id=`*id* specifies the CPU socket.
- *domain-name* specifies the domain from which the virtual CPUs, cores, or memory are removed.

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. The following example is for the Oracle Solaris 10 OS and you can obtain information for Oracle Solaris 11, see "How to Create Default Services" in "Oracle VM Server for SPARC 3.2 Administration Guide ":

```
primary# ldm add-vds primary-vds0 primary
primary# ldm add-vsw net-dev=net0 primary-vsw0 primary
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.

```
primary# ldm list-services primary
VCC
    NAME         LDOM    PORT-RANGE
    primary-vcc0 primary 5000-5100
VSW
    NAME         LDOM    MAC              NET-DEV   DEVICE     DEFAULT-VLAN-ID PVID VID MODE
    primary-vsw0 primary 00:14:4f:f9:68:d0 net0  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.

You must enable the virtual network terminal server daemon, `vntsd`(1M), to use consoles on the guest domains.

```
primary# ldm start-reconf primary
```

```
primary# ldm set-vcpu 8 primary
primary# ldm set-memory 8G primary
primary# ldm add-spconfig initial
primary# shutdown -y -g0 -i6
primary# 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.

```
primary# ldm list-bindings primary
NAME     STATE      FLAGS   CONS    VCPU  MEMORY   UTIL  NORM UPTIME
primary  active     -n-cv-  UART    8     16G      0.2%  0.2% 1d 18h 5m

UUID
    d8d2db22-21b9-e5e6-d635-92036c711e65

MAC
    00:21:28:c1:3f:3c

HOSTID
    0x8

CONTROL
    failure-policy=ignore
    extended-mapin-space=on
    cpu-arch=native
    rc-add-policy=
    shutdown-group=0

DEPENDENCY
    master=

CORE
    CID    CPUSET
    0      (0, 1, 2, 3, 4, 5, 6, 7)

VCPU
    VID    PID    CID    UTIL NORM STRAND
    0      0      0      0.4% 0.4%   100%
    1      1      0      0.2% 0.2%   100%
    2      2      0      0.1% 0.1%   100%
    3      3      0      0.1% 0.1%   100%
    4      4      0      0.2% 0.2%   100%
    5      5      0      0.5% 0.5%   100%
    6      6      0      0.2% 0.2%   100%
    7      7      0      1.2% 1.2%   100%

MEMORY
    RA               PA               SIZE
    0x20000000       0x20000000       8G
    0x400000000      0x400000000      8G
```

```
CONSTRAINT
    threading=max-throughput

VARIABLES
    pm_boot_policy=disabled=1;ttfc=0;ttmr=0;

IO
    DEVICE                           PSEUDONYM        OPTIONS
    pci@400                          pci_0
    niu@480                          niu_0
    pci@400/pci@1/pci@0/pci@8        /SYS/MB/RISER0/PCIE0
    pci@400/pci@2/pci@0/pci@8        /SYS/MB/RISER1/PCIE1
    pci@400/pci@1/pci@0/pci@6        /SYS/MB/RISER2/PCIE2
    pci@400/pci@2/pci@0/pci@c        /SYS/MB/RISER0/PCIE3
    pci@400/pci@1/pci@0/pci@0        /SYS/MB/RISER1/PCIE4
    pci@400/pci@2/pci@0/pci@a        /SYS/MB/RISER2/PCIE5
    pci@400/pci@1/pci@0/pci@4        /SYS/MB/SASHBA0
    pci@400/pci@2/pci@0/pci@4        /SYS/MB/SASHBA1
    pci@400/pci@2/pci@0/pci@6        /SYS/MB/NET0
    pci@400/pci@2/pci@0/pci@7        /SYS/MB/NET2

VCC
    NAME             PORT-RANGE
    primary-vcc0     5000-5100

VSW
    NAME             MAC               NET-DEV   ID   DEVICE   LINKPROP
    primary-vsw0     00:14:4f:fa:0b:57 net0      0    switch@0

    DEFAULT-VLAN-ID PVID VID                  MTU   MODE INTER-VNET-LINK
    1               1                         1500       on

VDS
    NAME             VOLUME         OPTIONS          MPGROUP        DEVICE
    primary-vds0

VCONS
    NAME             SERVICE                     PORT   LOGGING
    UART
```

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.

```
primary# ldm list-devices
primary# ldm add-domain ldg1
```

```
primary# ldm add-vcpu 8 ldg1
primary# ldm add-memory 8g ldg1
primary# ldm add-vnet vnet1 primary-vsw0 ldg1
primary# ldm add-vdsdev /dev/dsk/c0t1d0s2 vol1@primary-vds0
primary# ldm add-vdisk vdisk1 vol1@primary-vds0 ldg1
primary# ldm set-variable auto-boot\?=false ldg1
primary# ldm set-variable boot-device=vdisk1 ldg1
primary# ldm bind-domain ldg1
primary# ldm list-constraints -x ldg1 > ldg1.xml
primary# ldm add-spconfig ldg1_8cpu_1G
primary# ldm start-domain ldg1
primary# ldm list -l ldg1
primary# 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.

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

If you use the ldm list -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.

```
primary# ldm add-io pci@7c0 ldg1
```

EXAMPLE 8 Add Virtual Data Plane Channel Functionality for Netra Only

If your server has a Netra Data Plane Software (NDPS) environment, you might want to add virtual data plane channel functionality. First, you would add a virtual data plane channel service (primary-vdpcs0, for example) to the service domain, in this case, the primary domain.

```
primary# 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).

```
primary# ldm 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 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.

```
primary# ldm cancel-operation reconf primary
```

EXAMPLE 10 Migrate a Domain

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

```
primary# ldm migrate-domain 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.

```
primary# ldm list-spconfig
factory-default
3guests [current]
data1
reconfig_primary
split1
primary# ldm list-spconfig -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 performed a power cycle while in this state, the Logical Domains Manager would perform the `3guests` autosave based on the specified policy. The autosave 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 the `remove-spconfig` 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.

EXAMPLE 12 List I/O Devices

The following example lists the I/O devices on the system.

```
primary# ldm list-io
NAME                                 TYPE   BUS      DOMAIN   STATUS
----                                 ----   ---      ------   ------
pci_0                                BUS    pci_0    primary  IOV
niu_0                                NIU    niu_0    primary
/SYS/MB/RISER0/PCIE0                 PCIE   pci_0    primary  EMP
/SYS/MB/RISER1/PCIE1                 PCIE   pci_0    primary  EMP
/SYS/MB/RISER2/PCIE2                 PCIE   pci_0    primary  EMP
/SYS/MB/RISER0/PCIE3                 PCIE   pci_0    primary  OCC
/SYS/MB/RISER1/PCIE4                 PCIE   pci_0    primary  OCC
/SYS/MB/RISER2/PCIE5                 PCIE   pci_0    primary  EMP
/SYS/MB/SASHBA0                      PCIE   pci_0    primary  OCC
/SYS/MB/SASHBA1                      PCIE   pci_0    primary  OCC
/SYS/MB/NET0                         PCIE   pci_0    primary  OCC
/SYS/MB/NET2                         PCIE   pci_0    primary  OCC
/SYS/MB/RISER0/PCIE3/IOVIB.PF0       PF     pci_0    primary
/SYS/MB/RISER1/PCIE4/IOVIB.PF0       PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF0              PF     pci_0    primary
/SYS/MB/NET0/IOVNET.PF1              PF     pci_0    primary
/SYS/MB/NET2/IOVNET.PF0              PF     pci_0    primary
/SYS/MB/NET2/IOVNET.PF1              PF     pci_0    primary
/SYS/MB/RISER0/PCIE3/IOVIB.PF0.VF0   VF     pci_0    primary
/SYS/MB/RISER0/PCIE3/IOVIB.PF0.VF1   VF     pci_0    primary
/SYS/MB/RISER0/PCIE3/IOVIB.PF0.VF2   VF     pci_0    iodom1
/SYS/MB/RISER0/PCIE3/IOVIB.PF0.VF3   VF     pci_0    iodom1
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF0   VF     pci_0    primary
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF1   VF     pci_0    primary
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF2   VF     pci_0    iodom1
/SYS/MB/RISER1/PCIE4/IOVIB.PF0.VF3   VF     pci_0    iodom1
```

EXAMPLE 13 List CPU Core Activation Information

The following example shows information about the CPU core activations on a Fujitsu M10 server. The `PERMITS` column shows that 10 CPU core activations have been issued. This total includes all permanent and pay-per-use CPU core activations. The `PERMANENT` column shows that there are 10 permanent CPU core activations, which means that there are no issued pay-per-use CPU core activations. The `IN USE` column shows that only two of the CPU core activations are currently in use. The `REST` column shows that eight CPU core activations are available for use.

```
primary# ldm list-permits
```

```
CPU CORE
PERMITS (PERMANENT)   IN USE      REST
10      (10)          2           8
```

EXAMPLE 14 List Network Devices

The following example shows network device information for the `ldg1` domain.

```
primary# ldm list-netdev ldg1
DOMAIN
ldg1

NAME               CLASS    MEDIA    STATE    SPEED    OVER     LOC
----               -----    -----    -----    -----    ----     ---
net0               VNET     ETHER    up       0        --       primary-vsw0/vne t0_ldg1
net3               PHYS     ETHER    up       10000    --       /SYS/MB/RISER1/PCIE4
net4               VSW      ETHER    up       10000    --       ldg1-vsw1
net1               PHYS     ETHER    up       10000    --       /SYS/MB/RISER1/PCIE4
net5               VNET     ETHER    up       0        --       ldg1-vsw1/vnet1_ldg1
net6               VNET     ETHER    up       0        --       ldg1-vsw1/vnet2_ldg1
aggr2              AGGR     ETHER    unknown  0        net1,net3 --
ldoms-vsw0.vport3  VNIC     ETHER    unknown  0        --       ldg1-vsw1/vnet2_ldg1
ldoms-vsw0.vport2  VNIC     ETHER    unknown  0        --       ldg1-vsw1/vnet1_ldg1
ldoms-vsw0.vport1  VNIC     ETHER    unknown  0        --       ldg1-vsw1/vnet2_ldg3
ldoms-vsw0.vport0  VNIC     ETHER    unknown  0        --       ldg1-vsw1/vnet2_ldg2
```

The following example shows a detailed listing of network devices on the `ldg1` domain by specifying the `-l` option.

```
primary# ldm list-netdev -l ldg1
DOMAIN
ldg1

NAME               CLASS    MEDIA    STATE    SPEED    OVER     LOC
----               -----    -----    -----    -----    ----     ---
net0               VNET     ETHER    up       0        --       primary-vsw0/vnet0_ldg1
    [/virtual-devices@100/channel-devices@200/network@0]
    MTU       : 1500 [1500-1500]
    IPADDR    : 10.129.241.200/255.255.255.0
    MAC_ADDRS : 00:14:4f:fb:9c:df

net3               PHYS     ETHER    up       10000    --       /SYS/MB/RISER1/PCIE4
    [/pci@400/pci@1/pci@0/pci@0/network@0]
    MTU       : 1500 [576-15500]
    MAC_ADDRS : a0:36:9f:0a:c5:d2

net4               VSW      ETHER    up       10000    --       ldg1-vsw1
    [/virtual-devices@100/channel-devices@200/virtual-network-switch@0]
    MTU       : 1500 [1500-1500]
    IPADDR    : 192.168.1.2/255.255.255.0
    MAC_ADDRS : 00:14:4f:fb:61:6e

net1               PHYS     ETHER    up       10000    --       /SYS/MB/RISER1/PCIE4
```

```
    [/pci@400/pci@1/pci@0/pci@0/network@0,1]
    MTU       : 1500 [576-15500]
    MAC_ADDRS : a0:36:9f:0a:c5:d2

net5               VNET     ETHER    up       0        --       ldg1-vsw1/vnet1_ldg1
    [/virtual-devices@100/channel-devices@200/network@1]
    MTU       : 1500 [1500-1500]
    IPADDR    : 0.0.0.0  /255.0.0.0
              : fe80::214:4fff:fef8:5062/ffc0::
    MAC_ADDRS : 00:14:4f:f8:50:62

net6               VNET     ETHER    up       0        --       ldg1-vsw1/vnet2_ldg1
    [/virtual-devices@100/channel-devices@200/network@2]
    MTU       : 1500 [1500-1500]
    IPADDR    : 0.0.0.0  /255.0.0.0
              : fe80::214:4fff:fef8:af92/ffc0::
    MAC_ADDRS : 00:14:4f:f8:af:92

aggr2              AGGR     ETHER    unknown  0        net1,net3 --
    MODE      : TRUNK
    POLICY    : L2,L3
    LACP_MODE : ACTIVE
    MEMBER    : net1 [PORTSTATE = attached]
    MEMBER    : net3 [PORTSTATE = attached]
    MAC_ADDRS : a0:36:9f:0a:c5:d2

ldoms-vsw0.vport3  VNIC     ETHER    unknown  0        --       ldg1-vsw1/vnet2_ldg1
    MTU       : 1500 [576-1500]
    MAC_ADDRS : 00:14:4f:f8:af:92

ldoms-vsw0.vport2  VNIC     ETHER    unknown  0        --       ldg1-vsw1/vnet1_ldg1
    MTU       : 1500 [576-1500]
    MAC_ADDRS : 00:14:4f:f8:50:62

ldoms-vsw0.vport1  VNIC     ETHER    unknown  0        --       ldg1-vsw1/vnet2_ldg3
    MTU       : 1500 [576-1500]
    MAC_ADDRS : 00:14:4f:f9:d3:88

ldoms-vsw0.vport0  VNIC     ETHER    unknown  0        --       ldg1-vsw1/vnet2_ldg2
    MTU       : 1500 [576-1500]
    MAC_ADDRS : 00:14:4f:fa:47:f4
              : 00:14:4f:f9:65:b5
              : 00:14:4f:f9:60:3f
```

EXAMPLE 15 List Network Device Statistics

The following example shows the default network statistics for all the domains in the system.

```
primary# ldm list-netstat
DOMAIN
primary

NAME               IPACKETS     RBYTES       OPACKETS     OBYTES
```

```
----               --------     ------       --------     ------
net3               0            0            0            0
net0               2.72M        778.27M      76.32K       6.01M
net4               2.72M        778.27M      76.32K       6.01M
net6               2            140          1.30K        18.17K
net7               0            0            0            0
net2               0            0            0            0
net1               0            0            0            0
aggr1             0            0            0            0
ldoms-vsw0.vport0  935.40K      74.59M       13.15K       984.43K
ldoms-vsw0.vport1  933.26K      74.37M       11.42K       745.15K
ldoms-vsw0.vport2  933.24K      74.37M       11.46K       747.66K
ldoms-vsw1.vport1  202.26K      17.99M       179.75K      15.69M
ldoms-vsw1.vport0  202.37K      18.00M       189.00K      16.24M
------------------------------------------------------------------------------
DOMAIN
ldg1

NAME               IPACKETS     RBYTES       OPACKETS     OBYTES
----               --------     ------       --------     ------
net0               5.19K        421.57K      68           4.70K
net3               0            0            2.07K        256.93K
net4               0            0            4.37K        560.17K
net1               0            0            2.29K        303.24K
net5               149          31.19K       78           17.00K
net6               147          30.51K       78           17.29K
aggr2              0            0            0            0
ldoms-vsw0.vport3  162          31.69K       52           14.11K
ldoms-vsw0.vport2  163          31.74K       51           13.76K
ldoms-vsw0.vport1  176          42.99K       25           1.50K
ldoms-vsw0.vport0  158          40.19K       45           4.42K
------------------------------------------------------------------------------
DOMAIN
ldg2

NAME               IPACKETS     RBYTES       OPACKETS     OBYTES
----               --------     ------       --------     ------
net0               5.17K        418.90K      71           4.88K
net1               2.70K        201.67K      2.63K        187.01K
net2               132          36.40K       1.51K        95.07K
------------------------------------------------------------------------------
DOMAIN
ldg3

NAME               IPACKETS     RBYTES       OPACKETS     OBYTES
----               --------     ------       --------     ------
net0               5.16K        417.43K      72           4.90K
net1               2.80K        206.12K      2.67K        190.36K
net2               118          35.00K       1.46K        87.78K
```

EXAMPLE 16 List Dependencies

The following example shows detailed domain dependency information by specifying the `-l` option.

```
primary# ldm list-dependencies -l
DOMAIN         DEPENDENCY      TYPE      DEVICE
primary
svcdom
ldg0           primary         VDISK     primary-vds0/vdisk0
                               VNET      primary-vsw0/vnet0
               svcdom          VDISK     svcdom-vds0/vdisk1
                               VNET      svcdom-vsw0/vnet1
ldg1           primary         VDISK     primary-vds0/vdisk0
                               VNET      primary-vsw0/vnet0
                               IOV       /SYS/MB/NET0/IOVNET.PF0.VF0
               svcdom          VDISK     svcdom-vds0/vdisk1
                               VNET      svcdom-vsw0/vnet1
                               IOV       /SYS/MB/NET2/IOVNET.PF0.VF0
```

The following example shows shows detailed information about dependents grouped by their dependencies by specifying both the `-l` and `-r` options.

```
primary# ldm list-dependencies -r -l
DOMAIN         DEPENDENT       TYPE      DEVICE
primary        ldg0            VDISK     primary-vds0/vdisk0
                               VNET      primary-vsw0/vnet0
               ldg1            VDISK     primary-vds0/vdisk0
                               VNET      primary-vsw0/vnet0
                               IOV       /SYS/MB/NET0/IOVNET.PF0.VF0
svcdom         ldg0            VDISK     svcdom-vds0/vdisk1
                               VNET      svcdom-vsw0/vnet1
               ldg1            VDISK     svcdom-vds0/vdisk1
                               VNET      svcdom-vsw0/vnet1
                               IOV       /SYS/MB/NET2/IOVNET.PF0.VF0
```

EXAMPLE 17 List Resource Groups

The following example lists information about the contents of each resource group.

```
primary# ldm list-rsrc-group
NAME                                    CORE  MEMORY   IO
/SYS/CMU1                               12    512G     4
/SYS/CMU2                               12    512G     4
/SYS/CMU3                               12    512G     4
```

The following example lists detailed information about the contents of the `/SYS/CMU1` resource group.

```
primary# ldm list-rsrc-group -l /SYS/CMU1
NAME                                    CORE  MEMORY   IO
/SYS/CMU1                               12    512G     4
```

```
CORE
    BOUND            CID
    primary          (64,66,68,70,72,74,80,82,84,86,88,90)
MEMORY
    PA               SIZE             BOUND
    0x201ff0000000   256M             _sys_
    0x20000e400000   412M             _sys_
    0x200000000000   102M             _sys_
    0x200006600000   32M              _sys_
    0x200030000000   129792M          primary
    0x280000000000   128G             primary
IO
    DEVICE           PSEUDONYM        BOUND
    pci@500          pci_8            primary
    pci@540          pci_9            primary
    pci@580          pci_10           primary
    pci@5c0          pci_11           primary
```

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

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

Attribute Type	Attribute Value
Availability	SUNWldm
Interface Stability	Uncommitted

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

“Oracle VM Server for SPARC 3.2 Administration Guide ”

Name

ldmconfig — Oracle VM Server for SPARC Configuration Assistant

```
ldmconfig [-cdh]
```

The `ldmconfig` utility, the Oracle VM Server for SPARC Configuration Assistant, is a terminal-based application that streamlines the setup of systems that can run Oracle VM Server for SPARC. Only chip multithreaded-based (CMT) systems can be used to run Oracle VM Server for SPARC software.

Note that the `ldmconfig` command is supported *only* on Oracle Solaris 10 systems.

`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 Oracle Solaris OS media for valid packages

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

`-h` Displays usage message

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

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

Attribute Type	Attribute Value
Availability	`SUNWconfig`
Interface Stability	Uncommitted

`ldm`(1M) on page 9, `attributes`(5)

"Oracle VM Server for SPARC 3.2 Administration Guide "

Name

ldmd — Logical Domains Manager daemon

```
/opt/SUNWldm/bin/ldmd
```

The `ldmd` daemon is referred to as the Logical Domains Manager. It is the daemon program for the `ldm` command, which is used to create and manage logical domains. The `ldmd` daemon runs on the control domain, which is the initial domain created by the service processor (SP). For those platforms that have physical domains, the Logical Domains Manager runs only in the control domain of each physical domain. 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 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.

SMF Properties

You can use the `svccfg` command to modify the following properties:

`ldmd/autorecovery_policy`

Specifies the autorecovery policy. This property can have one of the following values:

- `autorecovery_policy=1` – Logs warning messages when an autosave configuration is newer than the corresponding running configuration. These messages are logged in the `ldmd` SMF log file. The user must manually perform any configuration recovery. This is the default policy.
- `autorecovery_policy=2` – Displays a notification message if an autosave configuration is newer than the corresponding running configuration. This notification message is printed in the output of any `ldm` command the first time an `ldm` command is issued after each restart of the Logical Domains Manager. The user must manually perform any configuration recovery.
- `autorecovery_policy=3` – Automatically updates the configuration if an autosave configuration is newer than the corresponding running configuration. This action overwrites the SP configuration that will be used during the next power cycle. This configuration is updated with the newer configuration that is saved on the control domain. This action does not impact the currently running configuration. It only impacts the configuration that will be used during the next power cycle. A message is also logged, which states that a newer configuration has been saved on the SP and that it will be booted the next time the system performs a power cycle. These messages are logged in the `ldmd` SMF log file.

`ldmd/autoreplacement_policy_cpu`

Specifies the CPU autoreplacement policy for the Fujitsu M10 platform. This property can have one of the following values:

- `autoreplacement_policy_cpu=1` – Enables the CPU autoreplacement process, which attempts to automatically replace faulty CPU resources. This is the default policy.
- `autoreplacement_policy_cpu=0` – Disables CPU autoreplacement process.

`ldmd/autoreplacement_retry_counter`

Specifies the maximum number of retries for the CPU autoreplacement process on the Fujitsu M10 platform. A value of `0` specifies that the number of retries are unlimited. The default value is `5` retries.

`ldmd/autoreplacement_retry_interval`

Specifies the interval in seconds between retries for the CPU autoreplacement process on the Fujitsu M10 platform. The minimum interval is `1` second. The default value is `300` seconds.

`ldmd/default_quick_stop`

Specifies which shutdown method to use. If `default_quick_stop` is set to `true` when the Logical Domains Manager is started, the next `ldm stop-domain` command uses the `-q` method unless overridden on the command line. If `default_quick_stop` is set to `false`, the `ldm stop-domain` command uses the `shutdown` command method, if available in the specified domain, or automatically falls back to the `-q` option. The default value is `false`.

`ldmd/hops`

Specifies the number of hops (or subnets) a MAC collision detection message is permitted to traverse before the message is dropped. Valid values are `0` (use default value), `1` (same subnet), `32` (same site), `64` (same region), `128` (same continent), and `255` (unrestricted). By default, `hops` is set to `1`, which means that multicast messages are only sent to other managers on the same subnet. When `hops` is set to `0`, `ldmd` uses the default value, which is `1`.

`ldmd/incoming_migration_enabled`

Enables a guest domain migration from another system to this system if `xmpp_enabled` is also set to `true`. The default value is `true`.

`ldmd/migration_authn_required`

Enforces the authentication of domain migrations to the system. The default behavior requires the user to be authenticated. This property does not affect migrations from the system. To permit a migration without specifying a password set `migration_authn_required` to `false` on the target machine.

This property is only used by the XML/XMPP interface and not by the `ldm migrate` command, which always requires a password to be given for a migration.

ldmd/pm_observability_enabled

Enables or disables the Power Management (PM) Observability Module at ldmd startup. When this module is enabled, you can run the ldmpower command to view power-consumption data. See the ldmpower(1M) on page 103 man page. The default value is true.

ldmd/outgoing_migration_enabled

Enables a guest domain migration from this system to another system if xmpp_enabled is also set to true. The default value is true.

ldmd/recovery_mode

Determines the action that is taken when recovery mode is requested by the SP. This property applies to only those systems that support recovery mode, such as SPARC T5, SPARC M5, and SPARC M6. The Fujitsu M10 platform ignores this property and behave as though the never value is specified. The following are the valid values:

- auto performs a recovery without intervention by the administrator.
- never specifies that no recovery is to be performed and that the system remains in the factory-default configuration. This is the default behavior.

ldmd/xmpp_enabled

Enables the ldmd XMPP server to listen for configuration requests from third-party management applications. Also, permits the ldmd daemon to communicate with the ldmd daemon on another system to coordinate a migration between the two systems. The default value is true.

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

Attribute Type	Attribute Value
Availability	SUNWldm
Interface Stability	Uncommitted

svcs(1), drd(1M), ldm(1M) on page 9, ldmad(1M), ldmpower(1M) on page 103, svcadm(1M), vntsd(1M), attributes(5), smf(5)

The ldmd service is managed by the Service Management Facility (SMF) and uses the svc:/ldoms/ldmd:default service identifier. See the smf(5) man page.

To have the changes made to these SMF properties take effect, you must refresh and restart the service.

Use the `svcadm` command to perform administrative actions on this service, such as enabling, disabling, refreshing, or requesting a restart. Use the `svcs` command to query the service's status.

For more information about the `ldmd` SMF properties, see the “Oracle VM Server for SPARC 3.2 Administration Guide ”.

Name

ldmp2v — command-line interface for the Oracle VM Server for SPARC Physical-to-Virtual (P2V) Conversion Tool

```
ldmp2v collect [-a flash|none] [-O "flarcreate-options"] [-v] [-x mount-point [-x ...]]
  -d data-dir
ldmp2v prepare [-b zvol|file|disk] [-B backend:volume:vdisk [-B ...]] [-c cpu]
  [-m mount-point:size [-m ...]] [-M memsize] [-o keep-hostid] [-o keep-mac] [-p prefix]
  [-s] [-v] [-x no-auto-adjust-fs] [-x remove-unused-slices] -d data-dir domain-name
ldmp2v prepare -R guest-root  [-c cpu] [-M memsize] [-o keep-hostid] [-o keep-mac]
  [-v] -d data-dir domain-name
ldmp2v prepare -C domain-name
ldmp2v convert -i install-image -d data-dir [-v] [-x skip-ping-test] domain-name
ldmp2v convert [-j] -n interface -d data-dir [-v] [-x skip-ping-test] domain-name
```

The Oracle VM Server for SPARC Physical-to-Virtual (P2V) Conversion Tool automatically converts an existing physical system to a virtual system that runs the Oracle Solaris 10 OS 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, Solaris 9, or Oracle Solaris 10 OS, or a non-Logical Domains `sun4v` system that runs the Oracle Solaris 10 OS. An image of the source system is converted into an Oracle VM Server for SPARC domain, upgrading to the current Oracle Solaris 10 version during the process, if necessary.

Note - The `ldmp2v` command does not support any SPARC system that runs the Oracle Solaris 10 OS with a ZFS root or the Oracle Solaris 11 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 flash|none] [-O "flarcreate-options"] [-v] [-x mount-point [-x ...]]
  -d data-dir
```

The `ldmp2v collect` command uses the following options:

`-a flash|none` Specifies the archiving method to use. Valid values are `flash` or `none`. The default is `flash`.

`-O` "*flarcreate-options*" Specifies a quoted list of options to pass to the `flarcreate` command. The only `flarcreate` options permitted are `-c` and `-x`. The `-c` option compresses the archive, and the `-x` option excludes files or directories from the archive. You can specify more than one `flarcreate` option. The `-O` option can only be used when you use `-a flash` to specify the flash archive method.

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

`-x` *mount-point* Excludes the file system, mounted on *mount-point*, from the archive.

`-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.

Preparation Phase

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

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

domain-name Specifies the logical domain on which to operate.

`-b zvol|file|disk` Specifies the back-end type to use. The virtual disks can be backed by ZFS volumes, `zvol`, plain files, `file`, or physical disks or volume manager volumes, `disk`. This option overrides the setting for `BACKEND_TYPE` in `/etc/ldmp2v.conf`.

`-B` *backend*:*volume*:*vdisk* Specifies the name of the back-end device and, optionally, the name of the volume and virtual disk to create. If the *volume* or *vdisk* value is omitted, a default name is used. You can omit values by specifying the colon character (`:`) for each value to omit. For example, these are valid uses of the `-B` option: `-B ::vdisk01` and `-B :volume001`.

This option is required for the `disk` back end and should at least specify a back-end device, such as `/dev/dsk/c0t2d0s2` or `/dev/md/dsk/d100`. For the `disk` back end, specify one `-B` option for each disk that is present in the manifest for the physical system.

For the `zvol` and `file` back ends, you can use *backend* to specify a file or ZFS dataset that `ldmp2v` should create for the virtual disk. For example, `-B data/ldom1/disk0`. Use the `-B` option to specify the back-end name and override the default name. The default name is generated by the `-p` option, or by the `BACKEND_PREFIX` setting in `/etc/ldmp2v.config` and the domain name.

`-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` *mount-point*:*size*

Resizes the underlying slice and disk for the file system at *mount-point*. The size is specified as *numunit*. *num* is the amount of space and *unit* is `b` for blocks, `k` for Kbytes, `m` for Mbytes, or `g` for Gbytes. You can specify this option more than one time. This option disables the automatic resizing of `/`, `/usr`, and `/var`. If *mount-point* is `swap`, the first configured swap device is resized to *size*.

`-M` *memsize*

Specifies the amount of memory to allocate to the logical domain. The memory size is specified as *numunit*, where *num* is the amount of memory and *unit* is one of the following:

- `m` or `M` represents Mbytes
- `g` or `G` represents Gbytes

If *unit* is not specified, the unit is Mbytes.

By default, the `ldmp2v` command allocates the same amount of memory that is in the physical system to the logical domain. If required, the memory size specified by the `-M` option is adjusted upward to 1 Gbyte to satisfy the minimum memory size for a guest domain.

`-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 device.

Conversion Phase

```
ldmp2v convert -i install-image -d data-dir [-v] [-x skip-ping-test] domain-name
ldmp2v convert [-j] -n interface -d data-dir [-v] [-x skip-ping-test] domain-name
```

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`.

`-x skip-ping-test`

Skips the `ping` test that is performed to determine whether the IP addresses of the source system are up. Use this option *only* if you are certain that no duplicate IP addresses will exist, such as when the original system is not active.

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.

This section includes examples for the three phases.

EXAMPLE 18 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. 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.

- **Exclude a file or directory from the flash archive.** When using the flash archiving method, you can exclude a file or directory from the archive by passing options to the `flarcreate` command. This capability requires that you have at least the following patch revisions installed on the source system:
  - **Solaris 8 OS:** Patch ID 109318-34
  - **Solaris 9 OS:** Patch ID 113434-06

  ```
  # ldmp2v collect -d /home/dana/p2v/volumia -a flash
    -O "-x /path/to/file -x /some/dir"
  ```

EXAMPLE 19 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 a file system and the swap device during P2V by using the `-m` option:

  ```
  # ldmp2v prepare -d /home/dana/p2v/normaal -m /:8g -m swap:4g normaal
  ```
- The following shows how to use Solaris Volume Manager metadevices `d100` and `d101` as back-end devices for the guest domain and to set the name of the first virtual disk to `vdisk100`:

  ```
  # ldmp2v prepare -b disk -B /dev/md/dsk/d100::vdisk100 -B /dev/md/dsk/d101
    -d /p2v/volumia volumia
  ```
- The following shows how to use ZFS volumes with non-default ZFS volume names:

  ```
  # ldmp2v prepare -b zvol -B tank/ldom1/zvol1 -B tank/ldom1/zvol2 -d /p2v/volumia
    volumia
  ```

EXAMPLE 20 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
```

The following exit values are returned:

0	Successful completion.
>0	An error occurred.

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

Attribute Type	Attribute Value
Availability	`SUNWldmp2v`
Interface Stability	Uncommitted

`ldm`(1M) on page 9, `attributes`(5)

“Oracle VM Server for SPARC 3.2 Administration Guide ”

Name

ldmpower — show per-domain, power-consumption information

```
ldmpower [-ehiprstvx | -o hours | -m minutes] [-c resource] [-l ldom[,ldom[,...]]]
  [interval [count]]
```

The `ldmpower` command shows the power-consumption data in watts for domains. By default, it shows the processor power that is consumed by each running domain. You can use options to view memory power consumption and the power consumption of the entire system divided among the running domains. The per-domain system power consumption is extrapolated from the per-domain processor and memory power consumption.

When specified with no options, the `ldmpower` command shows the average power consumption during the last 15, 30, and 60 seconds. The command can also show the most recent power-consumption data. For a longer history, the command can show up to 60 one-minute averages covering the last hour and 336 one-hour averages covering the last 14 days.

To run this command as a non-privileged user, you must be assigned the `LDoms Power Mgmt Observability` rights profile. If you already have been assigned the `LDoms Management` or `LDoms Review` rights profile, you automatically have permission to run the `ldmpower` command.

This command is not supported on UltraSPARC T2, UltraSPARC T2 Plus, and Fujitsu M10 platform.

The `ldmpower` command uses the following options and operands:

`-c` *resource*

Shows per-domain power consumption for the specified resource type, *resource*. Valid values are `processors` and `memory`. The long version of this option is `--component`.

`-e`

Shows the minimum and maximum power-consumption values since data recording began. The long version of this option is `--extremes`.

`-h`

Shows descriptions of the `ldmpower` command options. The alternate short version of this option is `-?` and the long version of this option is `--help`.

`-i`

Shows instantaneous power-consumption data. The long versions of this option are `--instant` and `--instantaneous`.

`-l` *ldom*

Shows power-consumption data for one or more specified domains. Domain names are separated by commas. The long version of this option is `--list`.

Note that less data is shown for domains that have been booted for a shorter amount of time than the requested period.

`-m` *minutes*
: Shows up to one hour of average power-consumption data in fixed one-minute intervals. The long version of this option is `--minutes`.

Note that less data is shown if the Logical Domains daemon has been running for a shorter amount of time than the requested period.

`-o` *hours*
: Shows up to 14 days of average power-consumption data in fixed one-hour intervals. The long version of this option is `--hours`.

Note that less data is shown if the Logical Domains daemon has been running for a shorter amount of time than the requested period.

`-p`
: Shows the overall power consumption for the entire system, which includes processors, memory, and fans. The long version of this option is `--platform`.

`-r`
: Shows the rolling average power-consumption history for the past 15, 30, and 60 seconds. The long version of this option is `--rolling`.

`-s`
: Suppresses the output header. The long version of this option is `--suppress`.

`-t`
: Shows time stamps in the output. The long version of this option is `--timestamps`.

`-v`
: Shows version data. The long version of this option is `--version`.

`-x`
: Shows extrapolated power-consumption averages. These averages reflect the percentage of overall system watts that are being consumed per domain. The per-domain system power consumption is extrapolated from the per-domain processor and memory power consumption. This option can be used with any of the `-o`, `-i`, `-m`, and `-r` options. The long version of this option is `--extrapolate`.

interval
: Reports output once each *interval* seconds.

count
: Shows *count* reports. If you specify the count, you *must* first specify the interval.

EXAMPLE 21 Viewing Processor Power-Consumption Data

The following command shows the 15-second, 30-second, and 60-second rolling average processor power-consumption data for all domains.

```
# ldmpower
Processor Power Consumption in Watts
DOMAIN  15_SEC_AVG  30_SEC_AVG  60_SEC_AVG
primary 75          84          86
gdom1   47          24          19
gdom2   10          24          26
```

EXAMPLE 22 Viewing Power-Consumption Data for the Entire System

The following commands show the overall instantaneous system power consumption in watts:

- The following command shows the instantaneous power-consumption data for the system, processors, memory, and fans. The `-t` option includes time stamps in the output.

  ```
  # ldmpower -tp
  Overall Instant System Power Consumption in Watts
  System          2012.08.23 23:13:33     738
  Processors      2012.08.23 23:13:33     295
  Memory          2012.08.23 23:13:33     138
  Fans            2012.08.23 23:13:33     28
  ```

- The following command shows the instantaneous power-consumption data for the system, processors, memory, and fans:

  ```
  # ldmpower -p
  Overall Instant System Power Consumption in Watts
  System          738
  Processors      295
  Memory          138
  Fans            25
  ```

EXAMPLE 23 Viewing Memory and Processor Power-Consumption Data

The following commands show the per-domain power consumption in watts for memory, processors, or both:

- The following command shows the power-consumption data for memory:

  ```
  # ldmpower -c memory
  Memory Power Consumption in Watts
  DOMAIN  15_SEC_AVG 30_SEC_AVG 60_SEC_AVG
  primary 138        138        138
  ldg1    19         19         19
  ldg2    19         19         19
  ```

- The following command shows the rolling power-consumption data for memory and processors:

  ```
  # ldmpower -c memory -c processors
  ```

```
Processor Power Consumption in Watts
DOMAIN  15_SEC_AVG 30_SEC_AVG 60_SEC_AVG
primary 63         61         60
ldg1    9          10         10
ldg2    9          9          10

Memory Power Consumption in Watts
DOMAIN  15_SEC_AVG 30_SEC_AVG 60_SEC_AVG
primary 138        138        138
ldg1    19         19         19
ldg2    19         19         19
```

- The following command shows the instantaneous power-consumption data for memory and processors:

```
# ldmpower -c memory -c processors -i
Processor Power Consumption in Watts
DOMAIN  INSTANT
primary 292
ldg1    10
ldg2    10

Memory Power Consumption in Watts
DOMAIN  INSTANT
primary 138
ldg1    19
ldg2    19
```

EXAMPLE 24 Viewing Processor Power-Consumption Data for Specified Domains

The following command shows the instantaneous processor power-consumption data for the gdom2 and gdom5 domains. The -i option shows the instantaneous power-consumption data, and the -t option shows time stamps in the output. The -l option arguments specify that only information about the gdom2 and gdom5 domains appear in the output. The first operand, 10, indicates that power-consumption data is reported in 10-second intervals. The second operand, 5, is the number of iterations for which the data is output.

```
# ldmpower -itl gdom2, gdom5 10 5
Processor Power Consumption in Watts
DOMAIN          TIMESTAMP               INSTANT
gdom2           2013.05.17 11:14:45     13
gdom5           2013.05.17 11:14:45     24

gdom2           2013.05.17 11:14:55     18
gdom5           2013.05.17 11:14:55     26

gdom2           2013.05.17 11:15:05     9
gdom5           2013.05.17 11:15:05     16
```

```
gdom2          2013.05.17 11:15:15     15
gdom5          2013.05.17 11:15:15     19

gdom2          2013.05.17 11:15:25     12
gdom5          2013.05.17 11:15:25     18
```

EXAMPLE 25 Viewing Power-Consumption Data for a Specified Amount of Time

The following commands show power-consumption data for specified amounts of time:

- The following command shows the average power-consumption data for the last 12 hours for all domains. The `-e` option shows the minimum and maximum values since data recording began, and the `-t` option shows time stamps in the output. The `-o` option argument specifies the number of hours to show the average hourly power-consumption data. Data is shown at one-hour intervals, starting from the last requested hourly calculation.

  ```
  # ldmpower -eto 12
  Per domain MINIMUM and MAXIMUM power consumption ever recorded:
  primary        2013.05.17 08:53:06     3               Min Processors
  primary        2013.05.17 08:40:44     273             Max Processors
  gdom1          2013.05.17 09:56:35     2               Min Processors
  gdom1          2013.05.17 08:53:06     134             Max Processors
  gdom2          2013.05.17 10:31:55     2               Min Processors
  gdom2          2013.05.17 08:56:35     139             Max Processors

  primary        2013.05.17 08:53:06     99              Min Memory
  primary        2013.05.17 08:40:44     182             Max Memory
  gdom1          2013.05.17 09:56:35     13              Min Memory
  gdom1          2013.05.17 08:53:06     20              Max Memory
  gdom2          2013.05.17 10:31:55     65              Min Memory
  gdom2          2013.05.17 08:56:35     66              Max Memory

  Processor Power Consumption in Watts
  12 hour's worth of data starting from 2013.05.16 23:17:02
  DOMAIN         TIMESTAMP               1 HOUR AVG
  primary        2013.05.17 09:37:35     112
  gdom1          2013.05.17 09:37:35     15
  gdom2          2013.05.17 09:37:35     26

  primary        2013.05.17 10:37:35     96
  gdom1          2013.05.17 10:37:35     12
  gdom2          2013.05.17 10:37:35     21

  primary        2013.05.17 11:37:35     85
  gdom1          2013.05.17 11:37:35     11
  gdom2          2013.05.17 11:37:35     23
  ...
  ```

- The following command shows the average power-consumption data for the last 30 minutes for all domains. The -e option shows the minimum and maximum values since data recording began, and the -t option shows time stamps in the output. The -m option argument specifies the number of minutes to show the average power-consumption data. Data is shown at one-minute intervals.

  ```
  # ldmpower -etm 30
  Per domain MINIMUM and MAXIMUM power consumption ever recorded:

  primary        2013.05.17 06:59:55     151             Min Processors
  primary        2013.05.17 06:56:20     682             Max Processors
  gdom1          2013.05.17 06:56:25     36              Min Processors
  gdom1          2013.05.17 07:02:10     318             Max Processors
  gdom2          2013.05.17 06:56:25     42              Min Processors
  gdom2          2013.05.17 07:05:45     417             Max Processors

  primary        2013.05.17 06:59:55     137             Min Memory
  primary        2013.05.17 06:56:20     138             Max Memory
  gdom1          2013.05.17 06:56:25     59              Min Memory
  gdom1          2013.05.17 07:02:10     60              Max Memory
  gdom2          2013.05.17 06:56:25     59              Min Memory
  gdom2          2013.05.17 07:05:45     60              Max Memory

  30 minute's worth of data starting from 2012.08.29 12:05:33:

  DOMAIN         TIMESTAMP               1 MIN AVG
  primary        2013.05.17 12:05:33     574
  gdom1          2013.05.17 12:05:33     268
  gdom2          2013.05.17 12:05:33     386

  primary        2013.05.17 12:06:33     555
  gdom1          2013.05.17 12:06:33     250
  gdom2          2013.05.17 12:06:33     364

  primary        2013.05.17 12:07:33     498
  gdom1          2013.05.17 12:07:33     276
  gdom2          2013.05.17 12:07:33     364
  ...
  ```

EXAMPLE 26 Viewing Extrapolated Power-Consumption Data for All Domains

The following command shows extrapolated power-consumption data for all the domains: primary, gdom1, and gdom2.

```
# ldmpower -x
System Power Consumption in Watts
```

```
DOMAIN          15_SEC_AVG      30_SEC_AVG      60_SEC_AVG
primary         585/57.47%      701/68.96%      712/70.22%
gdom1           132/12.97%      94/9.31%        94/9.30%
gdom2           298/29.27%      218/21.47%      205/20.22%
```

EXAMPLE 27 Viewing Power-Consumption Data for All Resources

The following command shows the system-wide rolling power averages for all resources: system, processors, memory, and fans.

By default, rolling power averages are shown, so this output matches the output of the `ldmpower -rp` command.

```
# ldmpower -p
Resource Power Consumption in Watts
TYPE            15_SEC_AVG      30_SEC_AVG      60_SEC_AVG
System          1016            1016            1016
Processors      270             270             269
Memory          179             179             179
Fans            202             202             203
```

EXAMPLE 28 Viewing Instant Power-Consumption Data for All Resources

The following command shows the system-wide instant power averages for all resources: system, processors, memory, and fans.

```
# ldmpower -ip
Resource Power Consumption in Watts
TYPE            INSTANT
System          1012
Processors      270
Memory          179
Fans            203
```

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

Attribute Type	Attribute Value
Availability	SUNWldm

`ldmd`(1M) on page 91, `attributes`(5)