



VDCF - Virtual Datacenter Cloud Framework for the Solaris™ Operating System

Quick Reference

Version 8.3
23 August 2022

Copyright © 2005-2022 JomaSoft GmbH
All rights reserved.



Table of Contents

1 Introduction.....	3
2 Quick Reference – VDCF Entry Edition.....	4
2.1 vdcfadm command.....	4
2.2 cpool command.....	5
2.3 nodecfg command.....	6
2.4 console command.....	7
2.5 config command.....	8
2.6 serverconfig command.....	9
2.7 routecfg command.....	11
2.8 build command.....	11
2.9 flash command.....	12
2.10 node command.....	13
2.11 ipsadm command.....	15
2.12 diskadm command.....	17
2.13 dataset command.....	19
2.14 patchadm command.....	21
2.15 vpkgadm command.....	22
2.16 vpool command.....	23
2.17 vserver command.....	24
2.18 zfsadm command.....	29
2.19 dependadm command.....	30
2.20 cdom command.....	30
2.21 gdom command.....	31
2.22 rcadm command (Resource Management).....	33
3 Quick Reference – VDCF Standard Edition.....	34
3.1 rcmon command (Resource Monitoring).....	34
3.2 hwmon command (Hardware Monitoring).....	35
3.3 hamon command (High Availability Monitoring).....	35
3.4 osmon command (Operating System Monitoring).....	36



1 Introduction

This documentation describes the Virtual Datacenter Cloud Framework (VDCF) for the Solaris Operating System, Version 8.3 and contains a reference of all CLI commands available.

See these other documents for further information:

<i>VDCF – Release Notes</i>	for details about new releases
<i>VDCF – Installation Solaris 11</i>	for information about installing VDCF
<i>VDCF – Administration Guide</i>	for information about VDCF Usage
<i>VDCF – Resource Management</i>	for information about VDCF Resource Management
<i>VDCF – Monitoring</i>	for information about VDCF Monitoring (HW, Resource, OS)

These and all other VDCF documents can be found at:

<https://www.jomasoft.ch/vdcf/#js-docu>

New and changed operations and arguments since VDCF 8.2 are marked **bold** and **green**.

2 Quick Reference – VDCF Entry Edition

2.1 vdcfadm command

vdcfadm -c show_log [follow] [tail=nn]	lists the content of the message log
vdcfadm -c show_audit [follow] [tail=nn]	lists the content of the audit log
vdcfadm -c clear_log [archive]	clears the message log
vdcfadm -c clear_audit [archive]	clears the audit log
vdcfadm -c show_version	shows the current VDCF version
vdcfadm -c show_config [output]	shows the actual configuration
vdcfadm -c statistics	show VDCF statistics
vdcfadm -c clear_locks	clears eventually hung locks
vdcfadm -c dump_db	create dump files of current database
vdcfadm -c load_db date=<dump date>	load/initialize database from dump files
vdcfadm -c show_node node=<nodename> all	show client pkg version on nodes
vdcfadm -c update_node node=<nodename> all cluster noncluster	update client pkg on nodes

2.2 cpool command

```
cpool -c show      [ name=<compute pool name> ]  
                  [ parsable [ header ] ]  
                  [ patches [ sru ] [ summary ] ]  
  
cpool -c create    name=<compute pool name>  
                  comment=<comment>  
                  [ default ]  
                  [ node=<node name list> ]  
  
cpool -c set_default name=<compute pool name>  
  
cpool -c assign    name=<compute pool name>  
                  [ node=<node name list> ]  
  
cpool -c rename    name=<compute pool name>  
                  newname=<new pool name>  
  
cpool -c modify    name=<compute pool name>  
                  comment=<comment>  
  
cpool -c remove    name=<compute pool name>  
                  [ force ]  
  
cpool -c check     name=<compute pool name> | all
```

The following format rules apply to the below listed parameters:

```
lists ::= < element,element,... >
```

2.3 nodecfg command

```
nodecfg -c discover      name=<node name>
                        [ hostname=<hostname> [ nonroot ] ]
                        [ proxy=<PROXY> ]
                        [ add ]

nodecfg -c show          [ name=<node name> [ allif ] ]
                        [ cpool=<compute pool name> ]
                        [ physical ]
                        [ all ]

nodecfg -c show_profile [ profile=<platform profile> ]

nodecfg -c create_profile name=<node name>      (interactive)
                        [ setspeed ]

nodecfg -c remove_profile profile=<platform profile>

nodecfg -c add           name=<node name>      (interactive)
                        profile=<platform profile>
                        [ setprobes ]

nodecfg -c add          name=<node name> noprofile

nodecfg -c modify       name=<node name>
                        [ addgroup=<config group list> ]
                        [ remgroup=<config group list> ]
                        [ interface=<network interface>
                          speed=<network speed> ]
                        [ location=<physical location> |
                          serial=<serial no> |
                          scheduler=<scheduler> | clear_scheduler |
                          hostid=<hostid> |
                          invno=<inventory no> |
                          datacenter=<datacenter> |
                          clear_linknames |
                          benchmark=<default|baseline|recommended|pci-dss|. >
                        ]
                        [ proxy=<PROXY> | clear_proxy ]
                        [ comment=<comment> ]

nodecfg -c modify_net   name=<node name>
                        interface=<interface or ipmp group name>
                        [ ipaddr=<ip or hostname> ]
                        [ netmask=<netmask> ]
                        [ nettype=<MNGT|PUBL|...> ]
                        [ standby | clear_standby ]

nodecfg -c remove      name=<node name>
```

The following format rules apply to the below listed parameters:

```
lists ::= < element,element,... >
```

2.4 console command

```
console -c add          node=<node name> (interactive)

console -c show         [ node=<node name> ]
                       [ all ]

console -c modify      node=<node name>
                       [ type=<console type> ]
                       [ user=<console user> ]
                       [ protocol=<protocol> ]
                       [ port=<port> ]
                       [ hostname=<hostname/IP> ]
                       [ proxy=<PROXY> | clear_proxy ]

console -c set_pwd     node=<node name>

console -c check       node=<node name> | all

console -c remove      node=<node name>
```

2.5 config command

```
config -c add      type=<config type>
                  name=<name>
                  [ os=all|10|11 ]
                  [ platform=all|sparc|i386 ]
                  [ comment=<comment> ]
                  <args ...>           depending on type

config -c modify  type=<config type>
                  name=<name>
                  [ comment=<comment> ]
                  <args ...>           depending on type

config -c modify  type=<config type>
                  name=<name>
                  [ os=all|10|11 ]
                  [ platform=all|sparc|i386 ]
                  [ comment=<comment> ]

config -c rename  type=<config type>
                  name=<name>
                  newname=<new name>

config -c remove  type=<config type>
                  name=<name>

config -c show    [ type=<config type>
                  [ name=<name> ] ]
                  [ os=10|11 ]
                  [ platform=sparc|i386 ]
```

Supported configuration types are:
COMMAND, DEFAULTROUTE, DNS, FILE, NTP, PKG, ROUTE, SCRIPT, SCSI_VHCI, SERVICES

Type specific arguments:

```
type=COMMAND      command=<command with options>

type=DEFAULTROUTE ipaddr=<ip address of defaultrouter>

type=DNS          domain=<domain>
                  search=<search>
                  server=<server>

type=FILE         source=<file>
                  target=<directory or file>
                  owner=<fileowner>
                  mode=<filemode>

type=NTP          server=<serverlist>

type=PKG          pkgs=<pkg[,pkg,pkg]>
                  pkgdevice=<device>
                  [ options=<pkgadd options> ]

type=ROUTE        destination=<address[/prefix]>
                  gateway=<address>

type=SCRIPT       script=<script>

type=SCSI_VHCI    provider=<provider> productid=<productid>

type=SERVICES     [ enable=<servicelist> ]
                  [ disable=<servicelist> ]
```


2.6 serverconfig command

```
serverconfig -c list      default | all | groups | servers  
                        [ type=<config type> ]  
  
serverconfig -c show     default | group=<config group> |  
                        server=<node or vserver>  
                        [ type=<config type> ]  
  
serverconfig -c show_members  group=<config group>  
  
serverconfig -c add      type=<config type>  
                        name=<baseconfig name>  
                        [ server=<node or vserver> ]  
                        [ group=<group> ]  
                        [ section=<section> ]  
                        [ comment=<comment> ]  
  
serverconfig -c modify   type=<config type>  
                        name=<baseconfig name>  
                        section=<section>  
                        [ default | group=<config group> |  
                        server=<node or vserver> ]  
                        [ comment=<comment> ]  
  
serverconfig -c remove   type=<config type>  
                        name=<baseconfig name>  
                        default | group=<config group> |  
                        server=<node or vserver>  
  
serverconfig -c create_group  
                        supergroup=<group>  
                        subgroups=<group,group,...>  
                        [ comment=<comment> ]  
  
serverconfig -c modify_group  
                        supergroup=<group>  
                        subgroups=<group,group,...> |  
                        comment=<comment>  
  
serverconfig -c remove_group  
                        supergroup=<group>
```



```
serverconfig -c exec    command=<command>
                        server=<comma sep list> |
                        servergroup=<config group> |
                        serverfile=<abs. path to file> |
                        servertime=all|node|vserver|gdom|cdom
                        [ user=<user to run command> | root ]
                        [ parsable ]
                        [ quiet ]

serverconfig -c exec    type=<COMMAND|SCRIPT|FILE|PKG|SERVICES|DNS>
                        name=<config name>
                        server=<comma sep list> |
                        servergroup=<config group> |
                        serverfile=<abs. path to file> |
                        servertime=all|node|vserver|gdom|cdom

serverconfig -c exec    group=<config group>
                        server=<comma sep list> |
                        serverfile=<abs. path to file> |
                        servertime=all|node|vserver|gdom|cdom
```

Supported configuration types are:
COMMAND, DEFAULTROUTE, DNS, FILE, NTP, PKG, ROUTE, SCRIPT,
SCSI_VHCI, SERVICES

2.7 routecfg command

```
routecfg -c import      node=<node name> | all

routecfg -c verify     vserver=<vserver name> |
                       node=<node name> | all

routecfg -c show       [ node=<node name> [ full ] | vserver=<vserver name> ]
                       [ destination=<address[/prefix]> ]
                       [ gateway=<address> ]
                       [ parsable [ header ] ]

routecfg -c add         destination=<address[/prefix]> gateway=<address>
                       [ name=<route name> ]
                       node=<node name> | vserver=<vserver name>

routecfg -c remove     destination=<address[/prefix]> | destination=all |
                       name=<route name>
                       [ gateway=<address> ]
                       node=<node name> | vserver=<vserver name>

routecfg -c revert     node=<node name> | vserver=<vserver name>

routecfg -c commit     node=<node name> | vserver=<vserver name>

routecfg -c diff       server=<vserver1,node2>
```

2.8 build command

```
build -c enable_install  hostname=<hostname>
                        macaddr=<mac address>
                        netmask=<netmask>
                        architecture=<kernel architecture>
                        install_server=<solaris install image>
                        [ be_name=<boot environment name> ]
                        [ profile=<profile name> ]

build -c add_bootserver  install_server=<solaris install image>
                        boot_server=<bootserver directory>

build -c show_bootserver [ boot_server=<bootserver dir> | refresh ]

build -c remove_bootserver boot_server=<bootserver directory>

build -c create          version=<build version>
                        boot_server=<bootserver directory>
                        archive=<archive location>
                        [ architecture=<kernel architecture> ]

build -c update_archive  version=<build version>
                        archive=<archive location>
                        [ architecture=<kernel architecture> ]

build -c show            [ version=<build version> ]

build -c remove          version=<build version>
```

2.9 flash command

```
flash -c enable_install node=<node name>  
      [ version=<FLASH version> ]  
  
flash -c disable_install node=<node name>  
  
flash -c list_active [ node=<node name> ]
```

2.10 node command

```
node -c update      name=<node name> | cpool=<computepool name> | all
                   [ vlan [ clear ] ]

node -c inactivate  name=<node name>

node -c activate    name=<node name>
                   [ force ]

node -c show        name=<node name> [ verbose ] [ allif ]
                   [ nettype=<MNGT|PUBL|BACK> ]
                   [ allfs | datafs ]

node -c show        [ cpool=<computepool name> ]
                   [ s10 | s11 [ u1|u2|u3|u4 ] ]
                   [ physical ]
                   [ parsable [ header ] ]
                   [ upgraded | checked ]
                   [ cluster | noncluster ]
                   [ all ]

node -c install     name=<node name>
                   [ force ]
                   [ console | wait ]

node -c upgrade     name=<node list>
                   build=<build name>
                   [ trial-run | reboot [ force ] [ wait ] ]
                   [ cluster_version=<incorporation-version> ]
                   [ geo_version=<incorporation-version> ]

node -c upgrade_check name=<node list>
                   build=<build name> | clear
                   [ cluster_version=<incorporation-version> ]
                   [ geo_version=<incorporation-version> ]

node -c upgrade_failback name=<node name>
                   [ reboot [ wait ] ]
                   [ destroy ]

node -c upgrade_finish name=<node list>
                   [ keep ]

node -c console     name=<node name>
                   [ escape=<escape character> ]
                   [ user=<console user> ]

node -c remove      name=<node name>
                   [ force ]

node -c boot        name=<node list>

node -c reboot      name=<node list>
                   [ wait ]
                   [ force ]

node -c shutdown    name=<node list>
                   [ force ]
```



```
node -c evacuate    name=<node name>
                   [ upgrade ]
                   [ force ]
                   [ shutdown ]

node -c register    name=<node name>
                   [ build=<build name> ]

node -c import      name=<node name>
                   [ hostname=<hostname> ]
                   [ proxy=<PROXY> ]
                   [ nodeonly ]

node -c verify      name=<node name> [ update ] | all
                   [ nodediscover ]

node -c addfs       name=<node name>
                   mountpoint=</directory>
                   [ dataset=<dataset name> ]
                   [ size=<size> ]
                   [ options=<mount options> ]

node -c remfs       name=<node name>
                   mountpoint=</directory | all> |
                   dataset=<dataset name>

node -c growfs      name=<node name>
                   mountpoint=</directory>
                   [ size=<size> ]

node -c revert      name=<node name>
                   mountpoint=</directory | all>

node -c commit      name=<node name> fs
                   [ remove ]

node -c destroy     name=<node name>
                   [ shutdown ]

node -c assess      help

node -c assess      name=<node name> | all
                   [ benchmark=default|baseline|recommended|,pci-dss|... ]
                   [ vserver ]

node -c harden      help

node -c harden      name=<node name>
                   profile=<hardening profile>

node -c enable_install name=<node name>
                   [ build=<build name> | active ]
                   [ group_pkg=<pkg name> ]

node -c enable_install all active

node -c disable_install name=<node name>

node -c show_enabled [ name=<node name> ]
```



2.11 ipsadm command

```

ipsadm -c show_repo [ name=<local repository name> |
                    port=<local pkg server port no> |
                    local | oracle |
                    repository=<remote repository url>
                    [ firmware | [ ai-pkg ] [ groups ] ]
                    ]

ipsadm -c create_repo name=<repository name>
                    isofile=<absolute path to repository ISO files> |
                    dir=<absolute path to existing directory with
                        zipped SRU files>
                    [ port=<pkg server port no> ]
                    [ zpool=<zpool name> ]

ipsadm -c config_repo name=<repository name>
                    dir=<absolute path to existing repository
                        directory>
                    [ port=<pkg server port no> ]

ipsadm -c update_repo name=<repository name> |
                    port=<pkg server port no>
                    [ repository=<url of source repository> |
                    p5pfile=<absolute path to package archive file> |
                    isofile=<absolute path to repository ISO file> ]
                    [ all-versions ]
                    [ all-pkgs ]
                    [ trial-run ]
                    [ firmware ]

ipsadm -c update_repo name=<repository name> |
port=<pkg server port no>
next | sru=<s11.4 SRU to download>
[ repository=<url of source repository> ]
[ trial-run ]

ipsadm -c update_repo name=<repository name> |
                    port=<pkg server port no>
                    dir=<absolute path to existing directory with
                        zipped SRU files>

ipsadm -c rebuild_repo name=<repository name> |
                    port=<pkg server port no>

ipsadm -c remove_repo name=<repository name> |
                    port=<pkg server port no>

ipsadm -c show_service [ name=<install service name> ]

ipsadm -c create_service name=<install service name>
                    isofile=<install service ISO file>

ipsadm -c create_service name=<install service name>
                    patchlevel=<0.0.0.0.0>
                    [ platform=<sparc|i386> ]
                    [ repository=<url of source repository> ]

ipsadm -c remove_service name=<install service name>
                    [ force ]
  
```



```
ipsadm -c show_build      [ name=<build name> | u1|u2|u3|u4 ]
ipsadm -c create_build    name=<build name>
                          [ service=<install service name> ]
                          [ patchlevel=<version to install> |
                          archive=<url of unified archive> ]
                          [ repository=<url of source repository> ]
ipsadm -c modify_build    name=<build name>
                          aiiso=<AI iso file>
ipsadm -c remove_build    name=<build name>
                          [ force ]
ipsadm -c check_archive
```


2.12 diskadm command

```
diskadm -c show [ all | free ] [ comment ]  
[ parsable [ header ] ]  
[ size=<size> ]  
  
diskadm -c show name=<GUID>  
[ parsable [ header ] ]  
  
diskadm -c show dataset=<dataset-name>  
[ parsable [ header ] ]  
  
diskadm -c show node=<node-name> [ all | free | inuse ]  
[ comment ]  
[ parsable [ header ] ]  
[ size=<size> ]  
  
diskadm -c show vserver=<vserver-name>  
[ comment ]  
[ parsable [ header ] ]  
[ size=<size> ]  
  
diskadm -c show tier=<storage tier> [ all | free ]  
[ comment ]  
[ parsable [ header ] ]  
[ size=<size> ]  
  
diskadm -c show location=<location|EMPTY> [ all | free ]  
[ comment ]  
[ parsable [ header ] ]  
[ size=<size> ]  
  
diskadm -c show box=<StorageBox|EMPTY> [ all | free ]  
[ comment ]  
[ parsable [ header ] ]  
[ size=<size> ]  
  
diskadm -c statistics  
[ date=<YYYY-MM-DD> ]  
[ server=<server-name>  
[ months=<no of months to show ] ]
```



```
diskadm -c register  node=<node-name> |
                    cpool=<compute pool name> |
                    all
                    [ methods=<method-list> ]
                    [ scan ]
                    [ full | new ]

diskadm -c mark      name=<GUID-list> foreign
                    [ comment=<"comment"> ]

diskadm -c mark      name=<GUID-list> usable

diskadm -c modify    name=<GUID-list>
                    [ comment=<"comment"> | remove_comment ]
                    [ tier=<storage tier> ]
                    [ location=<storage location> ]
                    [ box=<StorageBox> ]

diskadm -c deregister node=<node-name>
                    name=<GUID-list> | all

diskadm -c update    [ name=<GUID-list> ]

diskadm -c remove    name=<GUID-list>

diskadm -c label     name=<GUID-list>
                    [ node=<node-name> ]

diskadm -c init      name=<GUID-list>

diskadm -c verify  [ parsable [ header ] ]
```

The following format rules apply to the below listed parameters:
lists ::= < element,element,... >

2.13 dataset command

```
dataset -c create name=<dset name>
                vserver=<vServer name>
                [ type=<ZPOOL|DISKSET|RAW|VXVM> ]
                [ globalname ]
                [ delegated | swap | dump ]
                size=<size> [ newzvol [ zpool=<pool name> ] ] |
                layout=<layout description>

dataset -c create name=<dset name>
                node=<node name>
                [ type=<ZPOOL|DISKSET|VXVM> ]
                [ swap | dump ]
                size=<size> [ newzvol [ zpool=<pool name> ] ] |
                layout=<layout description>

dataset -c remove name=<dset name>
                [ force ]

dataset -c add name=<dset name>
                layout=<layout description>

dataset -c attach_mirror name=<dset name>
                layout=<layout description>
                [ force ]

dataset -c detach_mirror name=<dset name>
                mirror=<mirror> (i.e. 1st,2nd,3rd,4th)

dataset -c revert name=<dset name>

dataset -c commit name=<dset name>
                [ force ]
                [ upgrade ]

dataset -c show [ all | node=<node name> | vserver=<vServer name> ]
                [ parsable [ header ] ]

dataset -c show name=<dset name>
                [ verbose | parsable [ header ] ]

dataset -c detach name=<dset name>
                [ force ]

dataset -c attach name=<dset name>
                [ node=<node name> ]
                [ newname=<dset new name> ]
                [ force ]

dataset -c assign name=<dset name>
                vserver=<new vServer name>

dataset -c update name=<dset name>

dataset -c import node=<node name> | all

dataset -c verify name=<dset name> [ update ] |
                node=<node name> [ update ] |
                all [ update_size ]
                [ no_cdom_discover ]
```



```
dataset -c remdisk name=<dset name>  
                guides=<guid list> | layout=<layout description>
```

```
dataset -c addlog name=<dset name>  
                layout=<layout description>
```

```
dataset -c remlog name=<dset name>  
                guides=<guid list> | all
```

```
dataset -c replicate      name=<zpool name>  
                        [ target=<zpool name> ]  
                        [ destroy ]
```

```
dataset -c activate_replica name=<zpool name>
```

```
dataset -c cancel_replication name=<zpool name>
```

2.14 patchadm command

```
patchadm -c create_set      name=<patch-set name>
                             node=<node name> to=<date>

patchadm -c create_set      name=<patch-set name>
                             file=<patch order file>

patchadm -c create_set      name=<patch-set name>
                             platform=<platform>
                             [ from=<date> to=<date> ]

patchadm -c delete_set      name=<patch-set name>

patchadm -c modify_set      name=<patch-set name>
                             [ add    patches=<patch list> ]
                             [ delete patches=<patch list> ]

patchadm -c create_target   name=<target name>
                             desc=<description>
                             filter=<node-filter-spec>
                             [ patchset=<patch-set list> ]

patchadm -c delete_target   name=<target name list>

patchadm -c modify_target   name=<target name> [ rescan ]
                             [ filter=<node-filter-spec> ]
                             [ add    patchset=<patch-set list> ]
                             [ delete patchset=<patch-set list> ]
                             [ add    node=<node list> ]
                             [ delete node=<node list> ]

patchadm -c show            [ id=<patch-id> | verbose ]

patchadm -c show_set        [ name=<set name> ]

patchadm -c show_target     [ name=<target name> ]
                             [ verbose ]

patchadm -c show_node       node=<server name> | all
                             [ name=<set name> |
                               patchlevel ]

patchadm -c show_level      [ cpool=<compute pool name> ]

patchadm -c diff            server=<vserver1,node2> | node=<node name> | all
                             [ verbose ]

patchadm -c analyze         node=<node list> | all
                             [ localhost ] [ showonly ]

patchadm -c check           node=<node list> | all

patchadm -c download        [ id=<patch-id> ]

patchadm -c import          [ spool=<patch spool directory> ]
```



```
patchadm -c prepare      target=<patch target>
                        [ force ]

patchadm -c install      target=<patch target>
                        [ reboot ]
                        [ force ]

patchadm -c credentials  show |
                        set=oracle|proxy |
                        remove=oracle|proxy
```

The following format rules apply to the below listed parameters:

```
platform ::= < sparc | i386 >
date      ::= < YYYY-MM-DD >
lists     ::= < element,element,... >
filter    ::= node:<platform>
           node:<node list>
           build:<build-version list>
```

2.15 vpkgadm command

```
vpkgadm -c search      [ name=<name> ]
                       [ version=<version> ]
                       [ publisher=<publisher> ]
                       [ summary=<summary> ]
                       [ equal ]

vpkgadm -c show        server=<server name>

vpkgadm -c show        name=<name> [ version=<version> ] [ equal ] |
                       id=<pkg-id>

vpkgadm -c show_server name=<name> [ version=<version> ] [ equal ] |
                       id=<pkg-id>

vpkgadm -c diff        server=<server1,server2> [full]

vpkgadm -c analyze     node=<node list> | all
```

2.16 vpool command

```
vpool -c show      [ name=<vPool name> [ vservers | gdoms | nodes ] ]
                  [ user=<user name> ]
                  [ vserver=<vServer name> |
                  gdom=<Guest Domain name> |
                  node=<Physical node name> ]

vpool -c create    name=<vPool name>
                  comment=<"comment">
                  [ vserver=<vServer name list> |
                  gdom=<Guest Domain name list> |
                  node=<Physical node name list> ]
                  [ user=<user name list> ]

vpool -c create    name=<vPool name>
                  copyof=<source vPool name>
                  comment=<"comment">

vpool -c modify    name=<vPool name>
                  [ newname=<new vPool name> ]
                  [ comment=<comment> ]

vpool -c remove    name=<vPool name>
                  [ force ]

vpool -c add_user  name=<vPool name list>
                  user=<user name list>

vpool -c remove_user name=<vPool name list>
                  user=<user name list>

vpool -c add_vserver name=<vPool name list>
                  [ vserver=<vServer name list> |
                  cpool=<cPool name list> ]

vpool -c remove_vserver
                  name=<vPool name list>
                  [ vserver=<vServer name list> |
                  cpool=<cPool name list> ]

vpool -c add_gdom  name=<vPool name list>
                  [ gdom=<Guest Domain name list> |
                  cpool=<cPool name list> ]

vpool -c remove_gdom
                  name=<vPool name list>
                  [ gdom=<Guest Domain name list> |
                  cpool=<cPool name list> ]

vpool -c add_node  name=<vPool name list>
                  [ node=<Node name list> |
                  cpool=<cPool name list> ]

vpool -c remove_node
                  name=<vPool name list>
                  [ node=<Node name list> |
                  cpool=<cPool name list> ]
```

The following format rules apply to the below listed parameters:

```
lists ::= < element,element,... >
```

2.17 vserver command

```
vserver -c create name=<vServer name>
                  node=<node name>
                  comment=<"comment">
                  [ type=<FULL|KERNEL|SOL10|SPARSE|SOL8|SOL9> ]
                  [ cpus=<dedicated virtual CPUs> | cores=<whole-cores> ]
                  [ ram=<capped memory in K,M,G,T> ]
                  [ sgroup=<server group> ]
                  [ vpool=<vPool name list> ]
                  [ priority=<integer, lower is more important> ]
                  [ category=<category name> ]
                  [ hostid=<hostid> ]

vserver -c remove name=<vServer name>
                  [ force ]

vserver -c destroy name=<vServer name>
                  [ shutdown ]

vserver -c adddisk name=<vServer name>
                  size=<size> | guids=<guid-list>

vserver -c attach_root_mirror name=<vServer name>
                  size=<size> | guid=<guid>

vserver -c remdisk name=<vServer name>
                  guids=<guid-list>

vserver -c addfs [ type=data ]
                  name=<vServer name>
                  [ dataset=<dataset name> ]
                  mountpoint=</directory>
                  [ size=<size> ]
                  [ options=<mount options> ]

vserver -c addfs type=root
                  name=<vServer name>
                  [ dataset=<dataset name> | local ]
                  [ size=<size> ]
                  [ options=<mount options> ]

vserver -c addfs type=lofs
                  name=<vServer name>
                  directory=</directory>
                  mountpoint=</directory>
                  [ options=<mount options> ]

vserver -c growfs name=<vServer name>
                  mountpoint=</directory | root>
                  [ size=<size> ]

vserver -c shrinkfs name=<vServer name>
                    mountpoint=</directory | root>
                    size=<size>
```




```
vserver -c mount name=<vServer name>
                mountpoint=</directory> |
                dataset=<dataset name>

vserver -c unmount name=<vServer name>
                mountpoint=</directory> |
                dataset=<dataset name>

vserver -c renamefs name=<vServer name>
                mountpoint=</directory>
                to=</newdirectory>
                [ keepzfs ]
                [ remount [ commit | reverttonerror | force ] ]

vserver -c clonefs name=<vServer name>
                mountpoint=</source directory>
                to=</target directory>
                [ snapshot=<existing source> ]
                [ tovserver=<target vServer> ]

vserver -c clonefs name=<vServer name>
                dataset=<source dataset>
                basedir=</target base directory>
                [ snapshot=<existing source> ]
                [ tovserver=<target vServer> ]

vserver -c clonefs name=<vServer name>
                filesystem=<zfs filesystem or snapshot>
                to=</target directory>
                [ tovserver=<target vServer> ]

vserver -c remfs name=<vServer name>
                mountpoint=</directory | root | all> |
                dataset=<dataset name>

vserver -c addnet name=<vServer name>
                type=<management|public|backup>
                ipaddr=<ip address | hostname>
                [ netmask=<network mask> ]
                [ vlan=<vid> ]
                [ stack=<shared|private|exclusive> ]
                [ probes=<test-ip | hostname,test-ip | hostname,...> ]

vserver -c remnet name=<vServer name>
                type=<management|public|backup|all> |
                ipaddr=<ip address | hostname>

vserver -c revert name=<vServer name>
                mountpoint=</directory | root | all>

vserver -c revert name=<vServer name> network

vserver -c revert name=<vServer name> resources

vserver -c revert name=<vServer name> rootdisk

vserver -c revert name=<vServer name> limitpriv
```



```
vserver -c modify  name=<vServer name>
                  [ comment=<comment> ]
                  [ cpus=<dedicated virtual CPUs> | cores=<whole-cores> ]
                  [ ram=<capped memory in K,M,G,T> ]
                  [ addgroup=<config group list> ]
                  [ remgroup=<config group list> ]
                  [ priority=<integer, lower is more important> ]
                  [ category=<category name> ]
                  [ hostid=<hostid> | clear_hostid ]
                  [ group_pkg=<pkg name> ]
                  [ build=<build name> ]
                  [ autoboot=<boolean> ]
                  [ locked=<boolean> ]
                  [ file-mac-profile=<profile name> ]
                  [ benchmark=default|baseline|recommended|pci-dss|... ]
                  [ limitpriv=<comma serperated list> | clear_limitpriv ]

vserver -c commit  name=<vServer name>
                  [ boot [ console ] ]
                  [ exec ]
                  [ remove ]
                  [ uninstall ]
                  [ preserve ]

vserver -c apply   name=<vServer name>
                  [ trial-run ]

vserver -c migrate name=<vServer list>
                  node=<new target node>
                  [ live | shutdown ]
                  [ upgrade [ full ] ]
                  [ noboot ]
                  [ nocheck ]
                  [ force ]

vserver -c migrate source=<source node>
                  node=<new target node>
                  [ shutdown ]
                  [ upgrade [ full ] ]
                  [ noboot ]
                  [ all ]
                  [ nocheck ]
                  [ force ]

vserver -c detach  name=<vServer list>
                  [ force ]
                  [ shutdown ]

vserver -c detach  node=<node name>
                  [ force ]
                  [ shutdown ]
```



```
vserver -c attach name=<vServer list>
[ node=<new target node> ]
[ nocheck ]
[ force ]
[ upgrade [ full ] ]
[ boot ]
[ zbe=<bootenv name> ]
[ ignore_upgrade ]

vserver -c reattach name=<vServer list> |
node=<node name> |
cdom=<control domain>
[ nocheck ]
[ force ]
[ upgrade [ full ] ]
[ boot ]

vserver -c show [ node=<node name> |
cpool=<compute pool name> |
cdom=<control domain> ]
[ all ]
[ s10 | s11 [ u1|u2|u3|u4 ] ]
[ active | all-states ]
[ parsable [ header ] ]
[ kernel ]

vserver -c show name=<vServer name>
[ verbose |
candidates [ full ] |
parsable [ header ] ]

vserver -c show_perf name=<vServer name>

vserver -c boot name=<vServer list> |
node=<node list> |
cdom=<control domain>

vserver -c reboot name=<vServer list> |
node=<node list> |
cdom=<control domain>

vserver -c shutdown name=<vServer list> |
node=<node list> |
cdom=<control domain>
[ halt ]

vserver -c console name=<vServer name>
[ escape=<escape char> |
history | follow | tail=<nn> ]

vserver -c import node=<node name>
[ vserver=<vServer name> ]
[ vpool=<vPool name list> ]

vserver -c import name=<vServer name>

vserver -c import all

vserver -c make_exclusive
name=<vServer name>
[ reboot ]
```



```
vserver -c assess help

vserver -c assess name=<vServer name> | all
                  [ benchmark=default|baseline|recommended|pci-dss|... ]

vserver -c harden help

vserver -c harden name=<vServer name>
                  profile=<hardening profile>

vserver -c verify name=<vserver name> [ update ] |
                  node=<node name> [ update ] |
                  all

vserver -c upgrade name=<vServer list>
                  build=<build name>
                  [ trial-run | reboot [ wait ] ]

vserver -c upgrade_check name=<vServer list>
                  build=<build name> | clear

vserver -c upgrade_failback name=<vServer name>
                  [ reboot [ wait ] ]
                  [ destroy ]

vserver -c upgrade_finish name=<vServer list>
                  [ keep ]
```

2.18 zfsadm command

```
zfsadm -c show      vserver=<vServer name>
                   [ snapshots | all ]

zfsadm -c snapshot vserver=<vServer name>
                   filesystem=<filesystem name> |
                   mountpoint=</directory>
                   snapshot=<snapshot name>
                   [ recursive | rec ]

zfsadm -c rollback vserver=<vServer name>
                   snapshot=<snapshot name>
                   [ filesystem=<filesystem name> |
                   mountpoint=</directory> ]
                   [ childs ]
                   [ recursive | rec ]
                   [ recursive_all | recall ]
                   [ force ]

zfsadm -c destroy  vserver=<vServer name>
                   snapshot=<snapshot name>
                   [ filesystem=<filesystem name> |
                   mountpoint=</directory> ]
                   [ recursive | rec ]
                   [ recursive_all | recall ]
                   [ force ]

zfsadm -c rename   vserver=<vServer name>
                   snapshot=<existing snapshot>
                   to=<new snapshot name>

zfsadm -c get      vserver=<vServer name>
                   filesystem=<filesystem name> |
                   mountpoint=</directory>
                   [ props=<property list> ]

zfsadm -c set      vserver=<vServer name>
                   filesystem=<filesystem name> |
                   mountpoint=</directory>
                   props=<property list>
```

The following format rules apply to the below listed parameters:

```
lists    ::= < element,element,... >
date     ..= <YYYY-MM-DD>
```

2.19 dependadm command

```
dependadm -c show [ vserver=<name> ]

dependadm -c add      master=<vServer list>
                    slave=<vServer list>

dependadm -c remove  master=<vServer list>
                    slave=<vServer list>

dependadm -c remove  vserver=<vServer list>
```

The following format rules apply to the below listed parameters:

```
lists ::= < element,element,... >
```

2.20 cdom command

```
cdom -c create      name=<cdom name>
                    cpu=<virtual CPUs> | cores=<whole cores>
                    ram=<memory in K,M,G,T>
                    [ mau=<no of modular arithmetic units (MAU)> ]

cdom -c discover    name=<cdom name>

cdom -c show        [ s10 | s11 [ u1|u2|u3|u4 ] ]
                    [ cpool=<compute pool name> ]
                    [ parsable [ header ] ]
                    [ all ]

cdom -c show        name=<cdom name>
                    [ verbose ]

cdom -c modify      name=<cdom name>
                    [ cpu=<virtual CPUs> | cores=<whole cores> ]
                    [ ram=<memory in K,M,G,T> ]
                    [ mau=<no of modular arithmetic units (MAU)> ]

cdom -c commit      name=<cdom name>
                    [ reboot ]
                    [ force ]

cdom -c remove      name=<cdom name>
```

2.21 gdom command

```

gdom -c create      name=<guest domain name>
                   cdom=<control domain name>
                   cpus=<virtual CPUs> | cores=<whole cores>
                   | max-cores=<max cores>
                   ram=<memory in K,M,G,T>
                   comment=<"comment">
                   [ mau=<no of modular arithmetic units (MAU)> ]
                   [ vpool=<vPool name list> ]
                   [ profile=<partitioning profile> ]

gdom -c show       [ cdom=<control domain> |
                   cpool=<compute pool name> ]
                   [ all ]
                   [ s10 | s11 [ u1|u2|u3|u4 ] ]
                   [ active | all-states ]
                   [ parsable [ header ] ]

gdom -c show       name=<guest domain name>
                   [ [ verbose ] [ allfs | datafs ] |
                   candidates [ full | relevant |
                   cdom=<control domain> ]
                   [ noiocheck ] [ parsable ] ]

gdom -c modify     name=<guest domain name>
                   [ cpus=<virtual CPUs> | cores=<whole cores>
                   | max-cores=<max cores> ]
                   [ ram=<memory in K,M,G,T> ]
                   [ mau=<no of modular arithmetic units (MAU)> ]
                   [ profile=<partitioning profile> ]
                   [ autoboot=<boolean> ]
                   [ readonly=<boolean> ]
                   [ locked=location|true|false ]
                   [ livemigr=<boolean> ]
                   [ comment=<"comment"> ]

gdom -c revert     name=<guest domain name>
                   [ res | disk | net | all ]

gdom -c adddisk    name=<guest domain name>
                   type=<root|data>
                   size=<size> | guids=<guid-list>

gdom -c remdisk    name=<guest domain name>
                   guids=<guid-list>

gdom -c addnet     name=<guest domain name>
                   type=<management|public|backup>
                   ipaddr=<ip address | hostname>
                   netmask=<network mask>
                   [ vlan=<pVID> ]
                   [ ipmpgroup=<ipmp group name> ]
                   [ probes=<test-ip | hostname,test-ip | hostname,...> ]

gdom -c attach_root_mirror name=<guest domain name>
                   size=<size> | guid=<guid>

gdom -c remnet     name=<guest domain name>
                   [ type=<management|public|ackup> ]
                   [ ipaddr=<ip address | hostname> ]

gdom -c remove     name=<guest domain name>
                   [ force ]

```



```
gdom -c destroy      name=<guest domain name>
                    [ shutdown ]

gdom -c commit       name=<guest domain name>
                    [ install ]
                    [ remove ]

gdom -c install      name=<guest domain name>
                    [ console ]

gdom -c migrate      name=<guest domain name>
                    cdom=<target control domain>
                    live
                    [ noiocheck ]

gdom -c migrate      name=<guest domain name>
                    cdom=<target control domain>
                    [ shutdown ]
                    [ noboot ]
                    [ noiocheck ]
                    [ norescheck ]

gdom -c detach       name=<guest domain list>
                    [ force ]
                    [ shutdown ]

gdom -c detach       cdom=<control domain name>
                    [ force ]
                    [ shutdown ]

gdom -c attach       name=<guest domain name>
                    [ cdom=<new target control domain> ]
                    [ boot ]
                    [ noiocheck ]

gdom -c reattach     name=<guest domain name> |
                    cdom=<control domain name>
                    [ boot ]

gdom -c boot         name=<guest domain list> | cdom=<CDom name>

gdom -c reboot       name=<guest domain list> | cdom=<CDom name>
                    [ force ]
                    [ stop ]

gdom -c shutdown     name=<guest domain list>
                    [ force ]
                    [ stop ]

gdom -c shutdown     cdom=<control domain name>
                    [ iodom ]
                    [ force ]
                    [ stop ]

gdom -c console      name=<guest domain name>
                    [ history | follow | tail=<nn> ]
```

The following format rules apply to the below listed parameters:
lists ::= < element,element,... >

2.22 rcadm command (Resource Management)

```
rcadm -c show      [ name=<vServer or node name> ]
                  [ vservers=<vserver> ]
                  [ cpool=<compute pool> ]
                  [ all ]
                  [ all-states ]

rcadm -c show_perf { node | cpu }

rcadm -c statistics [ all-states ]

rcadm -c set       help

rcadm -c set       vservers=<vServer list>
                  <property>=<property value> ...
                  [ force ]

rcadm -c unset     vservers=<vServer list>
                  props=<property list>

rcadm -c revert    vservers=<vServer list>

rcadm -c remove    vservers=<vServer list>

rcadm -c clone     vservers=<vServer list>
                  template=<vServer>
                  [ force ]

rcadm -c commit    vservers=<vServer list>
                  [ push ]
                  [ force ]

rcadm -c convert_pool vservers=<vserver>
```

Properties are defined as follows:

CPU_Shares/CPU_cap and CPUs/Importance Properties are mutually exclusive

'CPU_Shares'	Number of Base Units. If CPU_Shares are defined 'CPUs' and 'Importance' are not allowed and FSS based RM is activated
'CPU_cap'	CPU capping in Base Units. 'CPU_Shares' and 'CPU_cap' can be but must not be specified together. They indicate a guaranteed and a maximum CPU entitlement.
-- or --	
'CPUs'	Number of CPUs. Requires 'Importance'.
'Importance'	Relative importance of temp pool

General Properties with no additional dependencies

'RAM'	Physical RAM in K,M,G,T
'SWAP'	Virtual Memory in K,M,G,T
'Locked'	Maximum locked down Memory in K,M,G,T
'LWP'	Maximum number of LWPs
'PROC'	Maximum number of Processes
'MSG_ids'	Maximum number of Message Queues
'SEM_ids'	Maximum number of Semaphores
'SHM_ids'	Maximum number of Shared Memory Segments
'SHM_Size'	Maximum size of all Shared Memory Segments in K,M,G,T

Sizes are specified as n, n[bB], n[kK], n[mM], n[gG], n[tT] where n is megabytes. The minimum values are: 1048576b, 1024k, 1 or 1m, 1g, 1t. Properties are not case sensitive!

3 Quick Reference – VDCF Standard Edition

3.1 rcmon command (Resource Monitoring)

```
rcmon -c status      [ node ]

rcmon -c enable      aggregator | collector

rcmon -c enable      node=<node list> | node all

rcmon -c disable     aggregator | collector

rcmon -c disable     node=<node list> | node all

rcmon -c update      node=<node name> | node all

rcmon -c show        help

rcmon -c show        cpu | memory | memory_extended
                    hourly | daily | monthly | yearly
                    server=<server name>
                    [ verbose ]
                    [ gz_total | gzt ]

rcmon -c show        cpu | memory | memory_extended
                    from=<'time-spec'>
                    server=<server name>
                    [ to=<'time-spec'> ]
                    [ aggr=<aggr-spec> ]
                    [ verbose ]
                    [ gz_total | gzt ]

rcmon -c summary     [ node | vservers ]
                    [ cpool=<compute pool> ]
                    [ sortkey=server | cpu ]
                    [ asc ]
```

3.2 hwmon command (Hardware Monitoring)

```
hwmon -c enable
hwmon -c enable      node=<node name>
hwmon -c disable
hwmon -c disable     node=<node name>
hwmon -c status      [ verbose ]
hwmon -c show         [ node=<node name>
                       [ verbose ] [ full ] ]
hwmon -c show_power
hwmon -c update       all | node=<node name>
hwmon -c show_locator node=<node name>
hwmon -c set_locator  node=<node name>
hwmon -c clear_locator node=<node name>
hwmon -c clear_history node=<node name>
hwmon -c clear_state  node=<node name>
```

3.3 hamon command (High Availability Monitoring)

```
hamon -c status
hamon -c show      [ node=<node name> ]
hamon -c show      [ on|off|fault|maint|susp|all ]
hamon -c enable    daemon
hamon -c enable    node=<node name>
hamon -c disable   daemon
hamon -c disable   node=<node name>
hamon -c suspend   node=<node name>
hamon -c resume    node=<node name>
hamon -c clear     node=<node name>
```

3.4 osmon command (Operating System Monitoring)

```
osmon -c enable

osmon -c enable      report

osmon -c disable

osmon -c disable    report

osmon -c status

osmon -c show       [ summary ]
                   [ hwmon ]
                   [ email ]

osmon -c update     all | node=<node name>
                   [ dataset|fs|smf|disk|cores|net ]

osmon -c modify_fs  server=<server name>
                   mountpoint=<mountpoint>
                   warnover=<percent> | remove_warn

osmon -c modify_fs  server=<server name>
                   mountpoint=<mountpoint>
                   warnfree=<size in T,G,M> | remove_warn

osmon -c modify_dataset dataset=<dataset>
                   [ server=<server name> ]
                   warnover=<percent> | remove_warn

osmon -c modify_swap node=<node name>
                   warnover=<percent> [ virtualmem ] |
                   remove_warn [ all | virtualmem ]

osmon -c modify_disk node=<node name>
                   targetcount=<target path count>
                   guids=<guid list> | all
```



```
osmon -c show_dataset [ over=<percent> ]  
                    [ summary ]  
                    [ root | dataset ]  
  
osmon -c show_dataset warnover  
  
osmon -c show_fs     [ over=<percent> ]  
                    [ summary ]  
                    [ root ]  
                    [ parsable [ header ] ]  
  
osmon -c show_fs     warnover | warnfree  
  
osmon -c show_smf    [ state="state1,state2,state3" ]  
                    [ search=<smf name> ]  
                    [ server=<server name> ]  
                    [ summary ]  
  
osmon -c show_swap  [ over=<percent> ]  
                    [ summary ]  
  
osmon -c show_swap  warnover  
  
osmon -c show_disk  [ node=<node name> ]  
                    [ summary ]  
                    [ all ]  
  
osmon -c show_net   [ node=<node name> ]  
                    [ summary ]  
                    [ all ]  
  
osmon -c show_server server=<server name>  
                    [ all ]
```



```
osmon -c summary      [ server=<server name> ]  
                     [ dataset|fs|smf|swap|disk|net ]  
  
osmon -c assess      help  
  
osmon -c assess      node=<node name> | all  
                     [ benchmark=default|baseline|recommended|pci-dss|... ]  
                     [ all_vserver ]  
  
osmon -c assess      vserver=<vserver>  
                     [ benchmark=default|baseline|recommended|pci-dss|... ]  
  
osmon -c show_compliance  
                     [ server=<server name> ]  
                     [ parsable [ header ] ]
```