Veritas Volume Manager

VM daemons

vxconfigd	Maintains system configuration in the kernel & disk (private region). If the daemon is stopped it does not disable any configuration state loaded into the kernel, it only affects the ability to make configuration changes until vxconfigd is restarted. It can be in three states: Enable = Normal mode Disable = Most operations cannot be be used Booted = Normal startup while using boot disk group
vxrelocd	Monitors for failure events and relocates failed subdisks
vxconfigbackupd	Used to backup configuration chnages, the files created can be used with vxmake to restored lost groups.
vxnotify	Display veritas volume manager events used with the vxconfigd daemon
vxcached	used to administer a cache object that is assoicated with volumes that have one or more space-optimized snapshots. When the usage of a cached volume reaches the high watermark vxcached autom,atically grows the cache volume if required and configured
vxattachd	Monitors VxVM for disks being attached and reattaches a detached site if the disks belongs to that sire become accessible, the daemon monitors vxnotify command and waits for a failed disk, when the disk is attached vxattachd attempts to online the disk, if successful it then starts recovery using vxrecover
vxdbd	Handles comunication to and from the storage foundation product, it uses port 3233
Kernel Info	
Kernel States	The kernel can be in three states: Enabled - both private and public regions are accessible Disabled - no private or public regions are accessible Detached - only private regions are accessible

VM utilities

VxVM debug	vxconfigd -k -m enable -x <debug option> -x log log to /var/vxvm/vxconfigd.log -x logfile = <name> log to filename -x syslog log to syslog -x timestamp date and timestamp every entry -x tracefile=name log all possible tracing to file
vxiod	The vxiod utility starts, stops, or reports on VERITAS Volume Manager (VxVM) I/O daemons. An I/O daemon provides a process context for performing I/O in VxVM.Manage extended disk i/o & handles dirty regions, logging vxiod set <number> = set number of runnning viod daemon Note: when run on its own it displays # of vxiod daemons that are running.
vxdctl [option]	The vxdctl utility manages aspects of the state of the volume configuration daemon **vxconfigd** and also manages aspects of configuration for bootstrapping the rootdg disk group. mode = what mode the vxconfigd is running in enable = enable the vxconfigd daemon (reread the db) disable = disable the vxconfigd daemon stop = kill the vxconfigd daemon (Use 'vxconfigd -k -m enable' to start again) license [init] = print out license info or reread licenses support = display version and components list = display entries in /etc/vx/volboot

	init [dmp] = recreate /etc/vx/volboot Note: when in disabled or stopped mode no VX commands will be able to run
vxinstall	Install volume manger (use /etc/vx/disk to exclude any disks or controllers)
vxdbdctl	used to stop/start the vxdbd daemon, you can also use status to obtain the status of the daemon

Disk Regions

Disk Regions		
Private	This is were veritas holds the meta data regarding the disk. A copy of the configuration database is copied to each private region within the disk group. Veritas will try and keep 5 copies of the configuration database. See below for where the slice is kept Once the private region is created it cannot be changed, thus if it fills up then you have to reinitize the disk with a new private region length, see disks -> initialize disk	
Public	This is the area that will store the users data. See below for where the slice is kept	
Disk Layouts		
Sliced Disk layout	private and public region slices are on separate partitions, this type of disk is not suitable for moving between different O/S's but are suitable for boot partitions Can be converted to CDS	format=sliced
CDS (Cross-platform Data Sharing)	private and public regions are one slice , this type is suitable for moving between different O/S's but not suitable for boot partitions.	format=cdsdisk
Simple	Private and public are the same partition but continuous Can be converted to CDS	format=simple
None	No partitioning	n/a

VxVM Configuration Database

Public/Private partitions	`vxdisk list <disk> |egrep -i '^public|^private'`
DB size	`# the size of the configuration db` `vxdg list <group> | grep permlen`
DB location	`vxdisk list <disk> | grep -i config - db location`

File Locations

vxinstall has not be run	/etc/vx/reconfig.d/state.d/install-db
Host ID's	/etc/vx/volboot
backup config files (vxconfigbackupd)	/etc/vx/cbr/bk Note: you must have at least run vxconfigbackup once, otherwise the directory does not exist or you have specified another location.
delete or deported disk group config files	/etc/vx/dgcfg/deport

All commands logs	/var/adm/vx/cmdlog /var/adm/vx/veacmdlog
Licenses	/etc/vx/licenses/lic
Imported disk groups info	/var/vxvm/tempdb Note: to clear the tempdb file: vxconfigd -k -x cleartempdir - clear the /var/vxvm/tempdb
vxconfigd log file	/var/adm/vx/vxconfigd.log

Backup & Restore

Backup	/usr/lib/vxvm/bin/vxconfigbackup -l /var/vxvm/backups -l = location where to store backup
Restore (precommit/commit)	vxconfigrestore -p <group> # either one of the below after the precommit vxconfigrestore -d <group> vxconfigrestore -c <group> Note: -p = when you want to check that the restore is correct (use vxprint to check) -d = abort the precommit -c = commit the precommit

Disks

Initialize disk	## increase the private region size default 1024 vxdisksetup -i c2t0d0 privlen= 2048 ## change the default of a disk vxdisksetup -i <device> format=sliced - initialized a disk as a sliced disk Note: format can be either sliced, simple, cdsdisk or none (see above - Disk Regions)
Uninitialize disk	vxdiskunsetup -C c2t0d0
Disk Information	vxdisk -g <group> list <disk> vxdisk -s list
Resize a LUN	vxdisk -g <group> resize <disk> length=8G
Add a disk slice to volboot	vxdctl add disk <device> type=simple
Add a disk slice	vxdisk –f <device> type=simple
Add a disk	vxdiskadd c1t0d0 or c1 (all disk on controller) vxdisksetup -i <device>
Remove a disk totally from VM	vxdisk rm <device>
Remove a disk from a volume	vxdg -g <group> rmdisk <diskname>
Remove a disk slice from VM	vxdctl rm disk <device>
Clear any host ID flags	vxdisk clearimport <disk name>
Renaming a disk	vxedit -g <disk> rename <old disk name> <new disk name>
Move disk to different disk group	vxdg move <source dg> <target dg> <disk> Note: you must always have one disk in a disk group, you cannot move a disk that is used by a volume
Offline a disk	vxdisk offline <dev name> Note: disk must not be in a disk group
Online a disk	vxdisk online <dev name> Note: disk must have a private region otherwise you need to initialise the disk

Hot spare	`vxedit –g <group> set spare=on <disk>`
NoHotUse	`vxedit –g <group> set nohotuse=on <disk>`
Turn off failing flag	`vxedit -g <group> set failing=off <disk>`
Encapsulate a disk	`vxdisk define c0t0d0s0 type=nopriv`

Reattach disk (SAN)	`vxreattach [-br | -c <device>]`
	
	`-b = Background process`
	`-r = Recover volumes`
	`-c = Checks to see if reattach is possible`

Discover new disks	`vxdisk scandisks [new | fabric ]`
Disk Comment	`vxedit -g <group> set comment="......." <disk>`

Private region problem	`## Here i am fixing a "online altused" issue but search the web for different issues as there are a number of ways to the same thing, just remember the vxprivutil command`
	
	`## obtain the disks private region tag 15`
	`prtvtoc /dev/rdsk/c2t0d0s2`
	
	`## dd the content sto a file`
	`dd if=/dev/rdsk/c2t0d0s3 of=/tmp/c2t0d0s3_privreg`
	
	`## run the below script`
	`fix_script`
	
	`## if now errors reported above and in the file, then dd back to disk`
	`dd if=/tmp/c2t0d0s3_privreg.good of=/dev/rdsk/c2t0d0s3`
	
	`## repeat for any other bad disks`

Disk Group

Create a disk group	`vxdg init group <disk>=<device>` `vxdg init <group> <disk>=<device> cds=off - initialize a non-cds disk group`
Remove a group	`vxdg destroy <group>`
Add a disk to a group	`vxdg –g <group> adddisk <disk>=<device>`
Remove a disk from a group	`vxdg –g <group> rmdisk <disk>`

Replace failed disk	`vxdg -k -g <group> adddisk <disk>=<device>`
	
	`-k = forces vxvm to take media name of the failed disk & assign it to the new disk`

Import a group	`vxdg import <group>`
	`vxdg -n <new-group-name> import <old-group-name>`
	`vxdg -C import <group>`

Import group (clear any flags)	`vxdg import -C <group>`
	
	`-C - clears any exist host flags`

Deport a group	`vxdg deport <group>` `vxdg -n <new-group-name> deport <old_group_name>`
List no hot use on disk	`vxdg –g <group> nohotuse`
List spare space on disk	`vxdg –g <group> spare`
Display free space	`vxdg –g <group> free`
Backup disk group (vxvm 4.0)	`vxconfigbackup`
Restore disk group (vxvm 4.0)	`vxconfigrestore [-p|-d|-c]`
Diskgroup Version	`vxdg list <group> | grep -i version`

Upgrade disk version	`vxdg upgrade <group> - upgrade to current version`
	`vxdg -T 50 upgrade <group> - upgrade to version 50`
	`vxdg -T 50 init <group> <disk>=<device> - creater new group @ version 50`

Boot/Default DG	vxdg bootdg vxdg defaultdg vxdctl defaultdg <group> - set defaultdg

Volume

Adding mirror to root	/etc/vx/bin/vxrootmir <alternate> create rootvol, swap vol,
Create a simple volume	vxassist make <volume> <size> <disk> vxassist -g <group> make <vol> <size> !ctrl:c2 - don't use controller 2 vxassist make <volume> <size> <disk> layout=[stripe-mirror|concat-mirror|mirror-concat|mirror- stripe-mirror = layered volume concat-mirror = layered volume mirror-concat = non-layered volume mirror-stripe = non-layered volume
Mirror a simple volume	vxassist mirror <volume> <disk> vxassist -g <group> remove mirror <vol> !disk01 - remove the disk01 mirror
Create a stripped volume	vxassist make <volume> <size> layout=stripe vxassist -g <group> -o ordered make <vol> <size> layout=stripe ncol=3 <disk1> <disk2> <disk3>
Create mirrored volume with log	vxassist make <volume> <size> layout=mirror, log nmirror=# nlog=# Note: for information about logging see logging section below
Create a raid volume	vxassist make <volume> <size> layout=raid5
Create a raw volume	vxassist -U <usage_type> make <volume> <size> alloc='<disk>' types: fsgen - filesystems gen - raw volumes raid - supports raid5 root - suuports root filesystems for booting swap - performs no recovery on startup relayout - used temporary for disk relayout operations
Remove a volume	vxedit –rf rm <volume> vxassist -g <group> remove volume <vol> Note: you must disable the volume first
Initializing a volume	vxvol init state <volume> [plex]state=clean,enable,active
Online Relayout	vxassist -g <group> relayout <vol> layout=stripe ncol=2 vxassist -g <group> relayout <vol> layout=stripe ncol=+1 vxassist -g <group> relayout <vol> layout=stripe ncol=-1 vxassist -g <group> relayout <vol> layout=stripe stripe=32k ncol=5 vxassist -g <group> relayout <vol> layout=raid5 stripeunit=32k ncol3 vxassist -g <group> convert <vol> layout=stripe-mirror # Display the relayout operation vxrelayout -g <group> [status|reverse|start] <vol> vxtask list
Remove a volume off a particular disk	## either make sure there is a another disk in the group, you can even specify it vxassist -g <group> move <vol> \!<disk> [<disk>]
Rename a volume	vxedit -g <group> rename <old_vol_name> <new_vol_name>
Starting a volume	vxvol start <volume>
Start a disabled volume	vxrecover –sb <volume> -s = start volume after recovery -b = background the recovery task
Disable a volume	vxvol -g <group> stop <volume>

Evacuate a volume	`vxevac -g <group> <from-disk> <to-disk>`
Maintenance mode	`vxvol maint <volume>`
Not clean	`vxmend mirror clean <plex>`
No kernel state	`vxplex att <vol_name> <plex>`
Extending a volume size	`vxresize <volume> <new length>` `vxresize -g <group> <volume> +100m - increase the volume by 100Mb`
Shrinking a volume size	`vxresize <volume> <new length>` `vxresize -g <group> <volume> -100m - decrease the volume by 100Mb`
add a DRL log to a volume	`vxassist addlog <volume>`
remove a DRL log from a volume	`vxassist remove log <volume>`
Extending log size	`vxvol set loglen = 2m <volume/log>`
Detering volume size	`vxassit -g <group> maxsize layout=mirror - the maximum size you can create a mirror` `vxassist -g <group> maxgrow <volume> - the maximum size the volume can grow too`
Recover a volume	`Vxmend fix clean <plex>`

Change volumes permissions	`vxedit -g <group> set owner=<user> group=<group> mode=<perms> <vol>`
	
	`# to display the permissions`
	`ls -l /dev/vx/rdsk/<group>`

Plexs

Creating a plex	`vxmake plex <plex> sd =<sub disk name>`

Remove a plex	`vxplex –o rm dis <plex>`
	`vxplex -g <group> dis <plex>`
	`vxedit -g <group> -rf rm <plex>`

Moving a plex	`vxplex mv <original plex> <new plex>`
Copying a plex	`vxplex cp <volume> <new plex>`
Attaching a plex	`vxplex att <volume> <plex>`
Detaching a plex	`vxplex det <plex>`
Offlining a plex	`vxmend off vol01-02`

Sub-disks

Creating sub-disk	`vxmake sd <sub-disk> <disk>, offset, len`
Removing sub-disk	`vxedit rm <sub-disk>`
Moving sub-disk	`vxsd mv <old sub-disk> <new sub-disk>`
Associating with a plex	`vxmake plex <plex> sd=<sub-disk>, …` `i.e vxmake plex home-1 sd= disk02-01, disk02-00, disk02-02`
Dissociating	`vxsd dis <sub-disk>`
Splitting	`vxsd –s<size> split sd<new sub><newsub2>`
Joining	`vxsd join <sub-disk1><subdisk2><new subdisk>`
relocating a sub disk	`## either make sure there is a another disk in the group, you can even specify it` `vxassist -g <group> move <vol> \!<disk> [<disk>]`
relocating a whole disk sub disks	`vxprint -g rootdg -se 'sd_orig_dmname="disk02"'` `vxunreloc -g rootdg disk02`

Volume Manager Information

Disks	
Display all the physical disks	`vxdisk list` `vxdisk -o alldgs list`

Display detailed disk info	`vxdisk list <disk>`
Check for disk failures	`vxstat –g <group>–ff –d`
Disk Group	
Display group properties	`vxdg list`
Display detailed group info	`vxdg list <group>`
	`vxinfo -p -g <group>`
Volume	
Display volume info	`vxprint –Aht <vol>`
Display volume properties	`vxprint –vl`
Display unstartable volume	`vxinfo -g <group> <vol>`
Check for volume failures	`vxstat –g <group>–ff -v`
Plex	
Display plex properties	`vxprint –vp`
Check for plex failures	`vxstat –g <group>–ff -p`
Sub-Disks	
Display sub-disk properties	`vxprint –st`
Veritas Tasks	

Display tasks	`vxtask list`
	`vxtask monitor - continuously monitor`
	
	`States:`
	`r = running`
	`p = pause`
	`a = aborting`

Statistics and Tracing	

Iostats	`vxstat -g <group> -r –d <disks> - reset all stats on disk`
	`vxstat -g <group> -d - display stats`
	`vxstat -g <group> -i 1 -d <vol> - display stats every 1 sec intervals for volume`
	`vxstat -g <group> -i 10 -c 5 -d - display 5 sets @ 10 secs intervals`

Tracing	`vxtrace -d <filename> -o dev,disk <vol>` `vxtrace -f <filename> -o dev,disk <vol> | more`

Licensing

Add	`vxlicinst - versions greater than 3.5` `vxlicense -c - versions below 3.5`
View	`vxlicrep - versions greater than 3.5` `vxlicense -p - versions below 3.5`
Paths	`/etc/vx/licenses/lic - versions greater than 3.5` `/etc/vx/elm - versions below 3.5`
Reload new license	`vxdctl license init`

VEA

Start/Stop	`/etc/init.d/isisd [start|stop|restart]`

Status	`vxsvcctrl status`
	`vxsvc [-m|-k|-v]`
	
	`-m = status`
	`-k = kill`
	`-v = version`

Daemons	`/opt/VRTSob/bin/vxsvc` `/opt/VRTSob/bin/vxsvcctrl`

Start VEA GUI	/opt/VRTSob/bin/vea You must have X-windows running

Logging

Logging help in recovery and can speed it up dramatically, the main form of logging in veritas is the DRL (dirty region log) which performs the following

- log keeps track of changed regions
- if system fails only the changed regions of the volume are recovered

Add	vxassist -g <group> addlog <vol> logtype=drl vxassist -g <group> addlog <vol> - used for raid logs (no type)
Remove	vxassist -g <group> remove log <vol> [nlog=n] <vol>

Volume Read Policy

Policies can be used if you have slower disks within a volume and you wish to use the faster disks.

Round Robin	vxvol -g <group> rdpol round <vol>
Preffered Plex	vxvol -g <group> rdpol prefer <vol> <plex>
Selected Plex	vxvol -g <group> rdpol select <vol>

Storage Expert (vxse)

Veritas have created some scripots that can check the integrity of the vxvm setup i.e mirrored volumes, spares, etc. The scripts are based on rules and there are a number of differents rules veritas has set, look in the rules directory to see all of them.

Display Description	vxse_raid5log1 info
Check rules	vxse_raid5log1 -g <group> check
List spare rules	vxse_spares list
Run spare run	vxse_spares run Note: you need to run "/etc/init.d/isisd start" to start the necessary daemons first
VXSE Paths	
Rules	/opt/VRTS/vxse/vxvm
Default Parameters	/etc/default/vxse

VxDMP

see VxDMP for more information

Veritas links

Recovery features	http://prefetch.net/articles/veritasrecoveryfeatures.html