

Coraid EtherDrive[®] VS Principles Of Operation Guide

Update 10/31/08

ABSTRACT

The Coraid VS network storage virtualization appliance provides a flexible way to allocate and use ATA-over-Ethernet storage appliances, such as the SR2461 or SR1521. Here we describe the limitations common in raw storage appliances and explain how these problems are addressed by using storage virtualization. Finally we illustrate how to use the VS to perform common operations.

Limitations Of Simple RAID Arrays

Network RAID appliances that combine several disk drives into a single high performance, reliable network storage target are a cost-effective way to provide storage.

The Coraid EtherDrive SR2461, for example, can combine twenty-four disk drives into a single large RAID target. It is common to assemble twenty-three 1TB disks in a RAID 5 configuration yielding 22TB of storage with one disk being used as a hot standby. If one of the disks in the RAID fails, the standby is drafted to be the failed disk's replacement.

Network RAID appliances have many advantages. Since the storage is no longer captive to the system to which it is directly attached, it can be shared by numerous servers on the network. The design of Ethernet permits a practically limitless amount of storage to be put onto a network.

RAID appliances do have some limitations, though. One is related to the size of storage created using RAID techniques. The RAID storage is a single large chunk which may not be the right size for a particular task. This large volume could be divided using client software to partition the device, but partition allocation is inherently static. A partition cannot be grown when the storage beyond its end is already being used. Making a partition smaller would leave a hole which may be of an awkward and unusable size.

Another problem is the issue of redundancy. One reason a Coraid SR is so affordable is that it is a non-redundant RAID controller. If the RAID controller fails, then the data is not available until the RAID controller is replaced. The VS can use multiple SR appliances to achieve redundancy in a modular way, so that storage remains available in the event of a single controller failure. The SR itself remains simple and affordable for those customers who do not need this redundancy.

The Network Storage Virtualization Solution

With the advent of the Coraid EtherDrive VS, AoE storage networks now have many of the advantages available in much more expensive traditional SAN technologies. The VS virtualization appliance creates virtual targets constructed out of available physical RAID unit storage. The exported virtual targets are called *logical units* (LUNs). The physical storage is added to the VS as *physical volumes* (PVs). The VS divides each physical volume into a collection of 4MB extents. All the extents are then added to a pool, called a *volume group* (VG). Multiple PVs can be added to a volume group, and there can be as many groups as needed. These groups can be used to keep storage segregated by administrative criteria. For example, different volume groups can be used to keep track of the resources used by various departments

within an organization, separating various 'clients' of the system. If desired, all the storage can simply be put into a single volume group.

The extents in these groups are the raw material used to build logical units. On the VS, a LUN is a virtual device that is the aggregate of extents drawn from a pool. VS LUNs are completely virtual. The real storage is on the physical volumes. There is no storage to speak of in the VS unit itself; all the storage is in the underlying targets. The VS creates a logical unit by allocating extents from a volume group. The LUN is then placed online to make it accessible from AoE client initiators.

Since the LUN is made of independent extents, it can be any size. A LUN can be grown and shrunk without moving any data. The underlying storage for a LUN can be spread over several physical volumes. Data can be migrated from one physical volume to another while being used by the LUN. By chunking the storage into extents and allocating a map of them for each LUN, virtualization provides all the flexibility needed.

The VS translates and forwards AoE requests. A client sends a read or write AoE request to the LUN, and the VS looks up in a table the real location of the storage. The request's logical block address and shelf/slot address are changed to reflect the location on the physical volume. The request is then sent to that physical volume for processing. The AoE response in most cases can return directly to the original requester avoiding having to pass back through the VS.

This leads to the question of where the VS keeps its information about the storage. That is "Where is the map of real storage kept?". Such information is called *metadata*. The VS's metadata is stored in extents allocated from the same pool as the data. When a physical volume is added to a volume group, extents are allocated from the physical volume to hold the extent table metadata. Each extent entry contains the LUN and the offset in that LUN for the extent the entry represents. Every extent in the physical volume has an extent entry. These entries are used to form the map of logical-to-physical translation mapping for a LUN and to track extent features. Tracking whether the extent has been written, for example, allows the VS to perform PV/LUN copying efficiently.

Using The Coraid EtherDrive VS

The first step when installing a VS is to set its two shelf addresses. The first address -- the admin shelf or simply *shelf* address -- is used to manage the appliance. The *shelf* address is used for flash based LUNs and for CEC. The second address -- the service shelf or *srvshelf* -- is used to export the virtual storage LUNs. Due to failure scenarios made possible by clustering the *shelf* and *srvshelf* addresses cannot be the same.

```
VS-1:-1> shelf 66
VS66:-1> srvshelf 63
VS66:63>
```

In the above example the shelf address is set to 66 and the service shelf address is set to 63. Notice that the prompt reflects both these numbers.

The `lsaoe` command displays the AoE targets on the network:

```
VS66:63> lsaoe
TARGET          LENGTH  STATE  PORT  ADDR
61.0            5001.078GB  V      0    00304833f6aa
61.0            5001.078GB  V      1    00304833f6ab
61.1            5001.078GB  V      0    00304833f6aa
61.1            5001.078GB  V      1    00304833f6ab
VS66:63>
```

Each target may appear more than once if there are multiple network interfaces connected on the VS as each interface can see the target. This provides multiple paths to the targets for higher reliability. For network redundancy, two switches can be installed with each interface of the VS plugged into each switch and each interface of target devices plugged into each switch. The VS will load balance across all available local interfaces and all available target addresses for a given shelf.slot.

The VS claims AoE targets on the network as physical volumes by writing to the AoE config string on the AoE target. The config string includes information permitting the VS to know if the AoE target is a PV, if it is owned by the VS, if the PV is a mirror element, the volume group the PV belongs to, and where the PV metadata for the PV is stored.

`lsaoe -c` displays the config strings set on all known AoE targets. The VS uses the config string of an AoE target to store VS configuration information. The format of the config string is not published as it is subject to change, but the first field is a magic string used to denote this target as a PV, and the second field denotes the `srvshelf` the target belongs to. As below, target 21.0 is claimed by the VS with `srvshelf 77`.

```
VS66:63> lsaoe -c
TARGET      LEN  CONFIG
21.1        54  'CoraidPV 77 21.1 -1.-1 0 1 238469 storage 1 53 21.1.0 '
61.0         0
61.1         0
VS66:63>
```

In this case and in all the cases to follow, see the *VS Command Reference Manual* for a complete description of command output and capability.

To create a volume group, use the `mkvg` command. A volume group has to be created before any physical volumes can be put into it.

```
VS66:63> mkvg accounting
VS66:63> lsvg
NAME                LENGTH          AVAILABLE    EXTSZ    PVS
accounting          0.000GB         0.000GB     4MB     00/00
VS66:63>
```

As shown, the volume group is empty. A group without any members won't survive a reboot since its existence is recorded on its constituent physical volumes.

```
VS66:63> mkpv 61.0 accounting
updating 61.0
VS66:63> lsvg
NAME                LENGTH          AVAILABLE    EXTSZ    PVS
accounting          5001.074GB      5001.065GB   4MB     01/01
VS66:63> mkpv 61.1 accounting
updating 61.1
updating 61.0
VS66:63> lsvg
NAME                LENGTH          AVAILABLE    EXTSZ    PVS
accounting          10002.148GB     10002.131GB 4MB     02/02
VS66:63> lspv
TARGET      VOLUME GROUP      LENGTH          AVAILABLE          NPE  MIRROR
61.0        accounting        5001.074GB      5001.065GB        1192349
61.1        accounting        5001.074GB      5001.065GB        1192349
VS66:63>
```

In the above example the `mkpv` command was used to add physical volumes to the 'accounting' volume group. The `lsvg` command displays 10TB in the volume group divided up into 4MB extents. The `lspv` command displays information about the added pvs. The `NPE` field is the number of extents in the physical volume. These extents are the raw material used to build logical units.

You may notice that the AVAILABLE amount for each physical volume is less than the LENGTH amount. This is due to extent allocation for holding the physical volume's metadata. The `lspv -a` command displays additional information about a physical volume, including the location of the metadata extents. The following shows that the allocated 12MB corresponds to three metadata extents allocated for the PV.

```
VS66:63> lspv -a 61.0
TARGET          VOLUME GROUP      LENGTH      AVAILABLE      NPE  MIRROR
61.0            accounting        5001.074GB   5001.065GB     1192349
LUNs:
metadata: 61.0.1 61.0.2
nvec=1192349 data=0 written=0 dirty=0
cow=0 changing=0 metadata=2 scanned=1
VS66:63>
```

We can now look at the *config string* on the physical volumes and see that our VS has claimed the storage.

```
VS66:63> lsaoe -c
TARGET      LEN  CONFIG
21.1        54  'CoraidPV 77 21.1 -1.-1 0 1 238469 storage 1 53 21.1.0 '
61.0        64  'CoraidPV 63 61.0 -1.-1 0 2 1192349 accounting 1 2 61.0.1 61.0.2 '
61.1        64  'CoraidPV 63 61.1 -1.-1 0 2 1192349 accounting 1 2 61.1.1 61.1.2 '
VS66:63>
```

Now, let's build a LUN for user Stan..

```
VS66:63> mklun 0 750g accounting
updating 61.1
updating 61.0
VS66:63> label 0 Stan
VS66:63> lslun
LUN          LENGTH  ONLINE      TARGET  LABEL
0            750.000GB  OFF         63.0   Stan
VS66:63> lsvg accounting
NAME          LENGTH      AVAILABLE  EXTSZ    PVS
accounting    10002.148GB  9252.127GB  4MB     02/02
VS66:63>
```

We have now made a 750GB logical unit out of extents from the accounting group and used the `label` command to textually note who this LUN belongs to. We then executed the `lslun` command to see it and then `lsvg` to see that blocks were allocated from the physical volumes.

We still can't access LUN 0 from client initiators. It must first be put online with the `online` command.

```
VS66:63> online 0
2008.03.24 08:09:28 Putting LUN 0 online
VS66:63> lslun
LUN          LENGTH  ONLINE      TARGET  LABEL
0            750.000GB  ON         63.0   Stan
VS66:63>
```

Putting a LUN online triggers an AoE broadcast message announcing the target, in this case shelf 63 and slot zero. There can be up to 255 LUNs in a single shelf address.

To tear everything down, reverse the above procedure using the associated remove commands.

```
VS66:63> offline 0
2008.03.24 08:09:48 Taking LUN 0 offline
VS66:63> rmlun 0
Are you sure you want to remove LUN 0? [n]: y
freeing LUN...
updating 61.1
updating 61.0
VS66:63> rmpv 61.1
updating 61.0
VS66:63> rmpv 61.0
VS66:63> rmvg accounting
VS66:63>
```

Notice that the LUN has to be offline before it can be removed. A physical volume may not be removed if it has allocated extents; all LUNs using a pv must be removed before a physical volume can be removed. We finish by removing the volume group.

It's as simple as that.

Network Setup And Traffic Translation

The VS appliance requires a network that supports jumbo frames of size 9000 bytes to communicate with physical volume storage. Initiators that access the virtual storage LUNs exported by the VS needn't use jumbo frames, but it is strongly recommended.

As previously discussed, the VS translates I/O traffic to its LUNs by locating the real location of the requested data on physical volume storage. In many cases, the VS can use a feature called *direct response* to forward the request on to the appropriate SR and have the SR respond directly to the initiator. This feature permits the VS to translate more traffic as it does not need to be the "middle man" for every request. Users must configure their AoE SAN so that the VS and all SR physical storage appliances are in the same Ethernet broadcast domain for this feature to function correctly. This network setup is also necessary for clustering to operate properly. For an example network configuration, please see *A Fault Tolerant VS Network Configuration* at the VS support page at coraid.com.

There are a few cases where the VS cannot use direct response. If a request spans extents, the VS must satisfy the request itself as the two extents could reside on different physical volumes. Additionally, I/O to a LUN that translates to a mirrored PV cannot be satisfied with direct response.

Physical Extent Allocation

When a LUN is created from a volume group, extents are allocated to the LUN from the constituent physical volumes in the volume group. The allocation method is round-robin; the LUN's extents are allocated evenly from every available PV in the volume group. A user may specify the PVs to back the LUN by adding them to the end of the `mk1un` command. If more than one PV is specified, round-robin allocation is performed over the specified PVs.

Mirroring Physical Volumes

If the applications using the storage require more reliability than provided by the single SR unit, then the SR units can be mirrored to protect both the disk RAIDs and the controllers. The following example shows how to mirror a physical volume using the VS.

```
VS66:63> mkpv 61.0 accounting
updating 61.0
VS66:63> lsvg accounting
NAME                LENGTH          AVAILABLE  EXTSZ    PVS
accounting          5001.074GB      5001.065GB  4MB     01/01
VS66:63> mirror 61.0 61.1
updating 61.0
2008.03.24 08:10:59 started rebuilding: 61.0 -> 61.1
VS66:63>
2008.03.24 08:10:59 mirror rebuild complete: 61.0 -> 61.1
updating 61.0
VS66:63> lspv
TARGET          VOLUME GROUP    LENGTH          AVAILABLE          NPE  MIRROR
61.0            accounting      5001.074GB      5001.065GB        1192349 61.1
VS66:63>
```

The 5TB 61.0 is now mirrored on target 61.1. The mirror doesn't appear as a volume in the pool; it only shows up as the mirror of the physical volume. In this case, 61.0 is called the primary and 61.1 is referred to as the mirror. LUNs created from extents of a mirrored volume will automatically be synchronously mirrored. The VS ensures synchronization by executing writes to both the primary and the mirror before responding to the AoE initiator. AoE requests for reads are always serviced from the primary.

Mirrors can be added to existing physical volumes that already have extents allocated and are being used by online LUNs. The metadata keeps track of whether the extents on a PV have been written. Note that on mirror creation in the example above the mirror is brought in sync. As we have not yet allocated any extents from 61.0, there is no work to be done to rebuild the mirror and the rebuild completes immediately. Had we created LUNs and been using them, however, all dirty extents would have been mirrored to 61.1. While a PV is being mirrored, the status of the background work can be displayed with the `wstat` command.

If a mirror fails, the VS will just use the primary. If the primary fails and the mirror is in sync, the mirror will be promoted to be the new primary. If the primary fails and the mirror is not yet in sync the mirror will be broken and the primary will be left to serve the requests it can as a best effort mechanism. In all cases the user will be notified by a log message that the physical volume is now unmirrored.

Mirrors can also be broken on purpose. The `unmirror` command will free the mirror from the primary. The mirror is not added to the volume group, but instead it is just freed from the mirror relation. Its *config string* will be cleared. Removing a mirrored primary from the volume group will cause the physical volume to be automatically unmirrored.

When Physical Volumes Go Missing

When the makeup of a volume group changes, a generation number is incremented and written out to each physical volume's *config string*. When the VS reboots, it checks these generation numbers to make sure that all the physical volumes are in sync with the group as a whole. If a physical volume is not available when the volume group changes in some way, the failed physical volume's generation number will not change. When the failed physical volume is put back on line, the VS will reject it due to its out-of-date generation number.

The out-of-date physical volume can be pulled back into the group using the `restore` command. `restore` updates the physical volume's generation number to the current VG generation number and loads the PV into the system. Loading the PV puts its extent descriptors back into the pool and creates any LUNs that were on the PV.

Users must use care when using the `restore` command. PV restoration can be a very dangerous action and should only be performed if the LUN configuration did not change in any way while the volume was offline. If LUNs were removed and recreated it is possible for extent numbers for a LUN to be duplicated on the old physical volume. Restoring a physical volume to the volume group in this scenario can lead to data corruption. Additionally, `restore` should not be used on an out of date pv that was in a mirrored configuration. The mirror will be brought in as an independent pv and extent duplication will occur. To remirror out of date PVs, use the `mirror` command.

LUN Copying

The VS supports LUN copying via the `copy` command. Simply specify the source and destination LUNs and the VS will perform the copy work in the background. As with PV mirroring, the `wstat` command displays the status of the background work.

LUN copying is very simple and does not ensure two LUNs remain in sync. The copy process merely reads every extent from the source LUN and writes it to the destination LUN. If either LUN is written during the copy, the LUNs may not be identical when the copy completes. It is recommended that users take both source and destination LUNs offline when using this feature.

Adding Currently Used Storage To The VS

AoE storage units with existing data can be added to the VS and used like any other physical storage by using a special `mklegacy` command. This process is special because it simultaneously adds the volume to a volume group and creates a new logical unit with the extents arranged one-to-one with the physical storage target. The first extent in the logical unit is the first extent in the physical volume, the second extent in the logical unit is the second extent in the physical volume, and so on. Accessing the LUN created from a legacy physical device is identical to accessing the original physical device.

Due to this approach, however, the metadata for the physical volume or the logical unit cannot be stored on the legacy physical volume. The additional metadata must be stored on another physical volume; free extents must be available in the volume group to create a legacy unit. Here is an example of creating a new volume group, adding a pv for metadata storage, and then adding the legacy volume.

```
VS66:63> mkvg existing
VS66:63> mkpv 61.0 existing
updating 61.0
VS66:63> mklegacy 1 61.1 existing
updating 61.1
updating 61.0
VS66:63> lspv
TARGET          VOLUME GROUP      LENGTH      AVAILABLE      NPE  MIRROR
61.0             existing          5001.074GB  5001.053GB     1192349
61.1             existing          5001.074GB  0.000GB       1192349
VS66:63> lslun
LUN          LENGTH  ONLINE  TARGET  LABEL
1           5001.074GB  OFF     63.1
```

We created a new volume group called *existing* and added a free physical volume 61.0. We then used the `mklegacy` command to simultaneously add 61.1 to the volume group and create a logical unit 1. Note that there are no available extents on 61.1 as they are all, by definition, taken up by the LUN.

There are a couple of unexpected things about storage added to the system in this way. First, one may expect that storage added to the system in a special way will be special forever. It is not. After being created there is nothing special about LUN 1 or the physical volume 61.1. LUN 1, for example, can be shrunk, or grown. It is just an ordinary LUN and an ordinary physical volume. Freed extents will be available for other LUNs.

Legacy physical volumes can also be mirrored. The metadata for LUN 1 and for physical volume 61.1 is in this case on 61.0. To mirror PV 61.1, we need a physical volume that is larger than 61.1 and can hold this metadata at its end. For 4MB extents, this means we need one more extent for every 2.5TB of storage. It should also be noted that the `mkLegacy` command marks each extent on the legacy physical volume as written. In the event that the volume is mirrored, every extent will be copied to the mirror.

Lastly, all extents on a PV are defined by the extent size of the volume group the PV is created in. The extent size is reported in the output of the `lsvg` command under the `EXTSZ` column, and is 4MB by default. Since a PV is addressed using `EXTSZ` chunks, up to `EXTSZ-1` bytes at the end of the legacy volume may not be exported. It is recommended that users shrink their use of the legacy device by `EXTSZ` (4MB by default) before executing the `mkLegacy` command to ensure all user data is exported by the legacy LUN.

Updating The Firmware

The VS has a flash boot disk that contains two image areas, A and B. The `setboot` command without arguments will display the default boot image.

```
VS66:63> setboot
will boot from A
VS66:63>
```

In this case we are booting from the first area. To change the default boot image, the `setboot` command is used to specify the area to boot from.

```
VS66:63> setboot B
VS66:63> setboot
will boot from B
VS66:63>
```

When the system boots it displays a prompt providing the user the chance to change the booting image. **The boot area selected at boot time will not set the default boot area. Only the setboot command sets the default boot area.** If no choice is made in a few seconds, the default image location will be used.

To update the contents of the flash images, flash LUNs are created using the `mkLun` command. There is no reason to remove the LUN after update as flash LUNs are not persistent across reboots.

```
VS66:63> mkLun 15 flash A
VS66:63> online 15
2008.03.24 08:02:17 Putting LUN 15 online
VS66:63> lsLun 15
LUN          LENGTH  ONLINE  TARGET  LABEL
15           0.008GB  ON      66.15   flash area A
VS66:63>
```

Now copy the binary from Coraid into LUN 99 from an AoE initiator host on your network. For Linux users, the `coraid-update` command should be used. Other UNIX variants can use the `dd` command. The new firmware will be loaded on next boot.

Since there are two areas, it is easy to back out an update if things go wrong. You can reboot from B if the new code written into location A has some problem. Once the administrator is happy with the code, it should be copied to both areas to avoid future confusion.

Coraid Ethernet Console (CEC)

In addition to serial and KVM connections, the VS supports the Coraid Ethernet Console (CEC) protocol for obtaining a console connection using standard Ethernet frames. CEC is a simple lightweight communication mechanism for use on the AoE SAN and does not include any security or encryption mechanisms. CEC is enabled on all Ethernet interfaces and is currently not configurable per port.

When connecting from an initiator system each VS's CEC console is differentiated by the admin shelf address. The admin shelf address is set to -1 when shipped; CEC connections can be made to the -1 shelf address to initially configure the VS.

Please visit the VS support page at coraid.com to download the open source cec client program.

LUN Masking

Access to a LUN can be restricted to a specified set of initiator Ethernet MAC addresses using the `mask` command. An unset LUN mask permits access to the LUN from any initiator on the AoE SAN and is the default after LUN creation. If the LUN mask contains one or more MAC addresses, only those MAC addresses are permitted to access the LUN. LUN masks are persistent across reboot.

LUN masks can be set on SR LUNs used by the VS to permit only the VS to access the LUN. The initiator MAC does not need to be in the LUN mask for SR LUNs. A foolproof configuration would include properly set LUN masks on all AoE storage appliances to protect against accidentally writing to the wrong LUN from initiator systems.

Clustering

Starting with release 20081101 the VS supports cooperative access to PV storage from multiple VS appliances. Currently only two node configurations are tested and supported. VS clustering is built upon a single master, multiple slave design. All VS appliances in a cluster share the same service shelf address; each VS is differentiated in the cluster by its admin shelf address.

Pre-cluster firmware releases are **not compatible** with cluster releases. Pre-cluster releases assume they are the master of their `srvshelf` address in any network configuration. Rebooting a slave cluster node into a pre-cluster release may lead to corruption as metadata updates on physical storage will be uncoordinated between multiple VS appliances. **It is highly recommended that users updating to cluster releases update both A and B images on all VS appliances to avoid accidentally entering this state.**

To determine the master, each VS in the cluster participates in an election. The VS with the highest admin shelf address at the time of election is elected as the master. Once an election is completed the VS in the cluster continuously communicate to maintain cluster state. An election occurs on first boot, but a reelection may occur at any time during operation. The following events may cause a reelection to take place:

```
new cluster node introduced to network
slave node loses connection to master node
```

The master/slave status of all cluster nodes after election can be displayed with the `clstat` command.

In a VS cluster the master node is responsible for all metadata update operations as well as conducting all "work" (mirror rebuild, LUN copy, e.g.). Slave nodes forward all requests that alter metadata to the master. Examples of metadata altering situations include:

```
writes to a LUN that cause a PV extent to be marked "dirty"
making/removing a PV/LUN
online/offline/shrink/grow of a LUN
```

Slave nodes communicate with the master node using Ethernet Link (EL) connections. EL is a lightweight, reliable transport protocol designed by Coraid to operate on standard Ethernet frames. EL borrows heavily from the IL protocol designed by Bell Labs. After an election is completed, each slave node makes an EL connection to the master node. This connection is used to transfer metadata information and to remotely execute commands on the master. Slave nodes do not connect to each other. When a reelection is triggered, all EL connections are closed. The status of EL connections on a VS can be displayed with the `elstat` command.

Once up and running, each VS in the cluster presents the same set of LUNs at the same `srvshelf` address. Initiator systems see multiple targets on the network for the LUNs and can send I/O requests round-robin across them to utilize all VS appliances simultaneously. A VS cluster configuration is fault tolerant in this way; should a VS fail, the initiator systems can continue accessing the storage by any other VS in the cluster.

There are a few details related to clustering worth mentioning explicitly. The `srvshelf` and shelf addresses cannot be the same. This protects certain failure scenarios where flash LUNs collide with storage LUNs. Users updating to a cluster release from a prior release that permitted this will have their admin shelf addresses set to -1.

Each VS in the cluster independently sends syslog messages to its configured syslog host; syslog configuration must be set on each VS in the cluster.

To maintain proper state, a VS may not probe the network for storage until an election completes. Many commands (`lslun`, `lspv`, etc) will not function while an election is in progress. In conjunction with this if a reelection occurs while running, a VS must forget about all storage on the network. This includes stopping any background work that the master might be performing. To mitigate the cost of reelections, a master may participate in a reelection without dropping its storage provided the other election participants all have lower admin shelf addresses. In this way the master may continue work (rebuilding mirrors, eg) in situations where the master would not change.

Configuration of a clustered VS setup is straightforward; simply set both VS appliances to the same `srvshelf` address and give each a unique admin shelf address. The VS with the higher shelf address will be the master.

The following example shows two VS appliances configured with `srvshelf 63`. One VS uses admin shelf 65 and another uses admin shelf 66. To display an election we will start with the output both appliances display when rebooted.

First, on the master:

```
VS Release Thu Oct 30 16:25:37 EDT 2008
BIOS build 11/02/07
2,060,211,264 bytes
maxpe=68,307,462
cluster: waiting for election completion
Console password unset. Access granted.
VS66:63> cluster: awake and I'm a master
```

Next, the slave:

```
VS Release Thu Oct 30 16:25:37 EDT 2008
BIOS build 11/02/07
2,059,885,952 bytes
maxpe=68,296,618
cluster: waiting for election completion
Console password unset. Access granted.
VS65:63>
2008.10.30 16:22:01 leader changed from -1 to 66
cluster: awake and I'm a slave
VS65:63> clstat
SHELF  ROLE  RELEASE
66      M   20081030
65      S   20081030
VS65:63>
```

Conclusion

This has been an overview of the principles underlying the design and operation of the VS network storage virtualization appliance. The *VS Command Reference Manual* should be studied for a full education of the commands used to manage the appliance. It is our sincere hope that we have created a simple yet flexible and powerful appliance that will enable our customers to more effectively use their network storage.