April 2003

Prepared by:
Network Storage Solutions
Online Storage Division
Hewlett Packard Company

Configuring Persistent Binding with the HP StorageWorks msa1000 on Linux Operating Systems

Abstract: *This technical note outlines step-by-step procedures for deploying a workaround to emulate Persistent LUNs on the HP msa1000 Storage array with Linux Operating Systems. Persistent binding is the mechanism to create a continuous logical route from a storage device object in the host to a volume in storage array across the fabric.*

Contents

Notice

Understanding Persistent Binding

Persistent binding is a mechanism to create a continuous logical route from a storage device object in the host to a volume in storage array across the fabric. Without a persistent binding mechanism, the host cannot maintain persistent logical routing of the communication from a storage device object across the fabric to a storage array volume. If the physical configuration of the switch is changed (for example, the cable is swapped or the host is rebooted), the logical route becomes inconsistent, causing possible data corruption.

Summary

Persistent binding from a hardware perspective is a way of permanently assigning SCSI target IDs to the same Fibre channel LUNs every time they are discovered, even if the device's ID on the Fabric change. Some HBA drivers have this capability built-in, and some do not; therefore they rely on software for persistent binding. From a software perspective the device files that get associated with the Fibre channel LUNs can be symbolically linked to the same secondary device file based on the LUN information to ensure persistence upon discovery even if the device's ID on the Fabric change.

Issue

In the case of the qLogic HBA driver it will always present LUNs to the host in numerical order. A Linux host will attach Fibre channel LUNs to device files in the order in which they are received, it does not attempt to attach them to the same files as before. In a perfect world LUNs 1, 2, and 3 will get attached to device files */dev/sda, sdb*, and *sdc* respectively. The world is not perfect however, so if for some reason we change the configuration and delete LUN 2 or assign it to a different host using selective storage presentation (SSP) LUN 3 will attach to */dev/sdb* the next time it is discovered, either via a reboot or reloading the driver module. From a hardware perspective LUN 3 may still be LUN 3, it may not get changed and the driver will still present it as LUN 3 to the host but because it is the "second" device Linux will attach it to */dev/sdb*. This will effectively change the location that LUN 3 is being mounted to - it will now be mounted to the location that LUN 2 was formally mounted. This clearly is not desirable behavior and would adversely effect any applications using these volumes. If this was a configuration with a large number of LUNs, all LUNs numerically greater than 2 would be affected in this manner. This similar behavior can be seen when adding LUNs in configurations involving cascaded storage systems. Additional LUNs created on the Primary storage system will cause all LUNs on the cascaded storage array to "move up" the device file list of a Linux host.

Configuration

Single qLogic HBA, Linux OS, HP msa1000 with 3 Logical Volumes (Any fibre attached storage is applicable); ext2 filesystem on the LUNs.
Appended `/etc/fstab` entries:

```
/dev/sda1 /lun01    ext2 defaults  1 2
/dev/sdb1 /lun02    ext2 defaults  1 2
/dev/sdc1 /lun03    ext2 defaults  1 2
```

Solution

In the above configuration, if we were to remove LUN 2 we would see the behavior as described earlier. After a reboot, LUN 3 would no longer be mounted to `/lun03`, instead it would be mounted to `/lun02`. To resolve this issue we could make the system mount LUN 3 to `/lun03` if we mounted by filesystem label instead of a hard device file.

Assume we just completed setting up the configuration described. The next task to do is assign a label to the ext2 filesystem that resides on each device using the *`e2label`* command.

```
# e2label /dev/sda1 LUN-1
# e2label /dev/sdb1 LUN-2
# e2label /dev/sdc1 LUN-3
```

Now instead of using device file entries in */etc/fstab* we will use the labels we just created.

```
LABEL=LUN-1    /lun01    ext2 defaults  1 2
LABEL=LUN-2    /lun02    ext2 defaults  1 2
LABEL=LUN-3    /lun03    ext2 defaults  1 2
```

In doing this we ensure that LUN 3 gets mounted to */lun03* every time, even if we delete LUN 2, and even if we delete LUN 1 & 2. Keep in mind the device file for LUN 3 will still change as we add/remove LUNs 1 & 2. This particular workaround/solution does not make the device file persistent to the Fibre Channel LUNs. We are merely using a feature of the ext2 filesystem to consistently mount the same LUNs to the same mount points if the device files change.