Transition from Solaris to AIX

Filesystem management and tools

Ken Milberg

March 06, 2007

So you've been the UNIX® guru on your team for years now and senior management has determined, without your input, to migrate to IBM from Sun Microsystems. Other than updating your resume, what do you need to do? Can you make the transition to AIX®? What does IBM offer as an alternative to either VERITAS or Solaris Volume Manager (SVM)? In this article, use the filesystem management and tools available to you on AIX to make your transition easier.

Introduction

The first thing you need to do is change your mindset. The typical Solaris administrator is used to performing all their tasks from the command line. While you might have an equivalent command-line operation for almost any Solaris command in AIX®, using the command line might not be the preferred method. For those of you familiar with admintool (eliminated entirely in Solaris 10), the equivalent in AIX is the System Management Interface Tool (SMIT). SMIT (or smitty) is not like the Solaris Management Console (SMC), which is a very pretty looking graphical tool meant to be run from the console or in X windows mode. While you can run SMIT from graphical user interface menu that is nice looking, most run it using the ASCII mode (using smitty), as it can be run from any machine and session.

AIX uses a rather unique system, the Object Data Manager (ODM), for managing devices and configuration files. While Solaris depends almost exclusively on editing text-based configuration files, AIX stores them in the ODM. You can picture the ODM as a database system built into the operating system. You also have to make a big adjustment as far as terminology is concerned. As a Solaris administrator, you are very familiar with partitioning your systems, creating replicas and running dozens of commands to mirror your logical volumes. On AIX, one command might replace dozens of Solaris commands.

Learn how a system administrator can better spend their time in other ways than by typing endless commands from the command line. For the purposes of this article, when comparing mirroring, you need to use Solaris Volume Manager (SVM) -- formerly referred to as Solstice DiskSuite -- as a comparison to the AIX Logical Volume Manager (LVM). Though not as robust as the VERITAS Volume Manager (used by many Sun Microsystems shops and also available for AIX
for that matter) it is available free with Solaris so, in that respect it, is closer to an apples-to-apples comparison. What you find is that as a practical matter, AIX LVM is really more like VERITAS, but without the premium price and third party issues. Later in this article, I'll compare SVM to AIX LVM, and explore the tasks of what is involved in doing a basic disk mirror between both environments. Recently, I had the opportunity to mirror disks on both Solaris 10 and AIX 5.3, and it is not an understatement that what took hours with Solaris took minutes with AIX.

Configuring disk -- Solaris

You are not a Solaris administrator unless you’ve spent hours partitioning your disk, usually with the `format` command. Volumes in SVM are built from slices (disk partitions). SVM provides you with the ability to manage large number of disks, and it also improves data availability. In order to manage physical disks, the SVM uses a virtual disk (metadevice), which is identical to a physical disk from an application point of view. SVM converts the I/O requests from a metadevice to I/O requests to the physical disks that are associated to it. Each metadevice is built from slices in the actual physical disk partitions. You can add slices from any physical disk to any metadevice, or choose not to. Make sure to partition the disks you plan on using as volumes if needed. Afterwards, you must then create your state database replicas. The SVM state database stores configuration and states information about volumes, including hot spares and disk sets. The database replicas ensure that the data in the state database is always valid. The SVM saves all information about structures of volumes on a disk area called the replica. The best analogy with AIX here is the Volume Group Descriptor Area (VGDA), Volume Group Status Area (VGSA), and logical volume control block (LVCB) on AIX.

Let’s start off by creating three state replicas (see Listing 1).

**Listing 1. Creating three state replicas**

```
# metadb -a -c 2 c1t3d0s1
```

When this is complete, identify slices that are available for use by the SVM. The process for mirroring the root (/) filesystem on a SPARC platform is similar to mirroring other filesystems; however, you need to run the `metaroot` command instead of manually editing the `/etc/vfstab` file. You must also record the path to the alternate boot device, so the system can boot if the submirror fails. I do not go into the details of formatting a disk in this article but, suffice to say, it is not for the weak-hearted. You really need to spend some time thinking about what you are doing and, if you mess something up, you can be in big trouble. You cannot create your mirrors until the formatting is complete. Let’s assume that you have partitioned the disk and are now ready to move forward.

First, let’s force the creation of the first mirrored volume (see Listing 2), which I’ve referred to as d1. This contains the root filesystem.

**Listing 2. Forcing the creation of the first mirrored volume**

```
# metainit -f d1 1 1 c0t0d0s0
d1: Concat/Stripe is setup
```

Then create the second concatenation, d2, from c0t1d0s0 (see Listing 3).
Listing 3. Creating the second concatenation

```bash
# metainit d2 1 1 c0t1d0s0
d2: Concat/Stripe is setup
```

The command in Listing 4 (using the `-m` flag) creates a one-way mirror, called d0, using the concatenation that contains root (/).

Listing 4. Using the `-m` flag to create a one-way mirror

```bash
# metainit d0 -m d1
d0: Mirror is setup
```

The command in Listing 5 (only used when mirroring root) automatically modifies the `/etc/vfstab` and `/etc/system` files, which allow the system to be booted under the control of SVM.

Listing 5. Booting the system under control of SVM

```bash
# metaroot d0
```

Running `lockfs` ensures that no changes are made prior to the reboot (see Listing 6).

Listing 6. Running lockfs

```bash
# lockfs -fa
```

Reboot the box by typing `# reboot`.

After the reboot, you need to issue the command in Listing 7, which attaches the second submirror to the system. This causes a mirror synchronization, which can take some time to complete (similar to mirrorvg in AIX).

Listing 7. Attaching the second submirror

```bash
# metattach d0 d2
d0: Submirror d2 is attached
```

In Solaris, the concepts of slices and partitions are extremely important to understand. With AIX, the concepts are completely different, as the disks themselves are entirely under the control of LVM. With Solaris, you don't need to use SVM or VERITAS. With AIX, LVM is always lurking. This is the fundamental difference between Solaris and AIX, from a filesystem standpoint. With AIX, filesystems management is always based on top of the logical volumes. In Solaris, the recognition of your disks is dependent on the hardware and available drives. The commands used to bring disks online include `devfsadm` and `cfgadm`. With AIX, you need to use `cfgmgr` and `smitty` to make your disks available if they are in a defined state.

Configuring disk -- AIX

How do you create a mirror? Because all disks are already under the control of the LVM, you don't need to partition nor define disk as being under a volume manager. You just need to determine where the name of the second disk is and issue one command. That is it. REALLY! I'm
oversimplifying things just a little, as one can choose to mirror only certain logical volumes in the root disk, if they choose to, with the `mklvcopy` command for each individual logical volume. There are also other commands you can use to assist in the process and post-mirror.

First, let's see what disks are in the root volume group (see Listing 8).

**Listing 8. Viewing disks in the root volume group**

```
# lsvg -p rootvg
rootvg:
PV_NAME           PV STATE          TOTAL PPs   FREE PPs    FREE DISTRIBUTION
hdisk0            active            546         186         00..00..00..77..109
```

Now let's see how many disks are on the system (see Listing 9).

**Listing 9. Verifying the number of disks on the system**

```
# lspv
hdisk0          00cf4f5d02a3c139                    rootvg          active
hdisk1          00cf4f5da1ca9500                    none
```

You need to bring hdisk1 into the root volume group (see Listing 10).

**Listing 10. Bringing hdisk1 into the root volume group**

```
# extendvg -f rootvg hdisk1
```

Make sure the disks are in an active state (see Listing 11).

**Listing 11. Ensuring that the disks are in an active state**

```
# lspv
hdisk0          00cf4f5d02a3c139                    rootvg          active
hdisk1          00cf4f5da1ca9500                    rootvg          active
```

After this is completed, you're ready to run the `mirror` command. This command mirrors all the logical volumes that exist on a given volume group (see Listing 12).

**Listing 12. Mirroring all the logical volumes that exist on a given volume group**

```
# /usr/sbin/mirrorvg rootvg hdisk0 hdisk1
0516-1124 mirrorvg: Quorum requirement turned off, reboot system for this
to take effect for rootvg.
0516-1126 mirrorvg: rootvg successfully mirrored, user should perform
bosboot of system to initialize boot records. Then, user must modify
bootlist to include: hdisk0 hdisk1.
```

Finally, create the boot image on the new disk and put it on the bootlist (see Listing 13).

**Listing 13. Creating the boot image on the new disk**

```
# bosboot -ad /dev/hdisk1
bosboot: Boot image is 25166 512 byte blocks.
bootlist -m normal hdisk0 hdisk1
```
Listing 14 confirms that it is all there:

**Listing 14. Confirming that all of the volumes are on the boot image**

```
# lsvg -p rootvg
rootvg:
  PV_NAME           PV STATE          TOTAL PPs   FREE PPs    FREE DISTRIBUTION
  hdisk1            active            546         186         00..00..00..77..109
  hdisk0            active            546         186         00..00..00..77..109
```

You're done!

**Logical Volume Manager**

During the early stages of UNIX®, the management of physical disks was always a daunting task because of the restrictions on allocating the actual physical space. You'd have to define physical partitions, and each physical partition would actually be of a fixed size. Additionally, a physical disk could only have eight physical partitions, so the customer had to select the correct size of each partition before installing the system.

A major restriction of the physical partition was that each partition had to be contiguous; meaning you could not span multiple drives. What evolved from this messy picture was a new, flexible technique to manage the allocation, known as logical volumes. Here are some of the benefits:

- Logical volumes solve non-contiguous space problems.
- Logical volumes can span multiple disks.
- The size of logical volumes can dynamically increase.
- You can relocate and mirror them.

AIX and Solaris offer different methods for using logical volumes. With Solaris, it does not come ready out of the box. You need to determine which LVM to use, and decide whether or not to use SVM or VERITAS. While VERITAS offers the VERITAS Foundation suite for AIX 5L™, it is typically not used in AIX environments, as it is not necessary. VERITAS Volume Manager (VxVM) is a storage management subsystem that enables you to manage physical disks as logical devices called volumes. It is important to reiterate that this product is third-party software, and you will pay handsomely for it.

The LVM has been a feature of the AIX operating system since Version 3, and it is installed automatically with the operating system. *Table 1* helps lay out some of the differences between AIX and VERITAS, and it also compares the commands (for the complete table, see Resources). While the version numbers might be different, the concepts and commands remain the same.

**Table 1. LVM quick reference**

<table>
<thead>
<tr>
<th>Task</th>
<th>AIX 5L Version 5</th>
<th>Solaris 8/VxVM 3.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage structure</td>
<td>A disk is composed of physical partitions.</td>
<td>A disk is composed of partitions or slices.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A filesystem is placed onto a partition.</td>
</tr>
</tbody>
</table>
A physical volume is a physical disk the same thing as a disk. A volume group is composed of physical volumes. A volume group is divided into logical volumes. A filesystem is placed into a logical volume. A logical volume is extensible and can reside on more than one volume. A subdisk (somewhat similar to AIX physical partition) is composed of partitions or slices. A plex (similar to AIX logical partition) is composed of subdisks. A volume (similar to AIX logical volume) is composed of plexes. A VM disk is composed of subdisks. A disk group (similar to AIX volume group) is composed of VM disks.

| Run multiple tasks in a GUI environment | smit lvm  | /opt/VRTSvmsa/bin/vmsa |
| Move a logical volume to another logical volume | migratepv | vxassist move |
| Create a logical volume | mk lv | vxassist make |
| Extend a logical volume | extendlv | vxassist growto vxassist growby vxresize (recommended for filesystems) |
| Remove a logical volume | rmlv | vxassist remove vxedit rm |
| Create a volume group | mk vg | vxdg init |
| Remove a disk from a volume group | reducevg | vxdg -g dgname rmdisk |
| Add disks to a volume group | extendvg | vxdiskadd |
| Change logical volume settings | chlv | vxedite set |
| Display volume group information | lsvg | vxdg list |
| Display performance statistics for storage | lvmstat | vxstat |
| Manage volumes | chlv mk lv rmlv | vxvol |
| Add a copy to an existing volume | mklvcopy | vxassist -g dgname mirror |

Figure 1 illustrates how the physical disk relates to the actual logical partitions.

**Figure 1. AIX logical volume to physical volume map**

**Filesystems**

In this section, compare the filesystems of Solaris and AIX. Let's start with a look at the different filesystems available in Solaris.
Solaris

The primary filesystem for Solaris is the UNIX File System (UFS). It is very mature and, for most applications that run on Solaris, it is the filesystem of choice. It has roots in the Berkeley Fast File System (dating back to the 1980s). Some of the recent enhancements to it include having metadata logging to improve both reliability and performance. To mirror and use LVM without VERITAS, you would have to bring these filesystems under the control of SVM. Recent enhancements to SVM include support for multiterabyte volumes, cluster volume manager, and thousands of partitions per physical disk.

No article on the Solaris filesystem would be complete without mentioning Veritas; VxFS: the Extent-based logging filesystem from Veritas Corporation. It’s widely accepted that VxFs is more of an enterprise-ready filesystem than the UFS/SVM combo, and it scales much better. It's a very robust implementation of LVM, though it does have its issues. Regarding buffering of metadata, VxFS allows the user to specify that the metadata, which is usually written to the log on the platter, would not be synchronously written to disk. In this scenario, it’s in memory. The immediate acknowledgment back to the write routine optimizes performance, though there is one important flaw to this. The issue is that where a system crash occurs, the data would be lost, as it is in memory. Because the IBM Journaled File System (JFS) and the IBM Enhanced Journaled File System (JFS2) do not allow this caching, this clearly is a reliability issue to be mindful of.

Finally, let's discuss the Solaris Zettabyte File System (ZFS), which Sun references as the best filesystem on the planet. Does ZFS replace UFS, SVM, or VxFs? It depends. While it’s true that it does not require a separate volume manager, it still does not replace the volume manager used by either SVM or Veritas. You cannot use it for your root volume group. ZFS offers virtually unlimited data scalability and it is a 128-bit filesystem. It provides approximately 16 billion times more capacity than 32- and 64-bit filesystems. In other words, ZFS supports more of everything: storage, filesystems, snapshots, directory entries, and so forth. It really is an innovative ground-up redesign of the traditional UNIX filesystem. It uses end checksumming to protect against data corruption and a simple administrative model. ZFS also completely eliminates the need for volume managers, instead using pooled storage where many filesystems can share a pool. I like the concept but, unfortunately, there is no tool to convert from UFS/SVM or VxFS to ZFS. While ZFS might be the future, not a lot of people are presently using it and it has a long way to go.

Every filesystem in Solaris corresponds to a slice or to a logical volume (if you are using Veritas). In order to create a filesystem on a previously defined slice, you need to use the newfs command. The newfs command is a front-end to the mkfs program, which created UFS filesystems on disk partitions. You can increase the size of a filesystem by using the growfs command, among other commands.

AIX

IBM introduced JFS with the initial release of AIX 3.1. In May of 2001, IBM introduced JFS2. Both filesystem types link their file and directory data to the structure used by the AIX LVM for storage and retrieval. JFS2 is optimized for a 64-bit environment. JFS2 is architected for filesystems up to four petabytes, but it has currently only been tested up to 16 terabyte-sized filesystems. Also,
file sizes are limited to 16 terabytes. The number of inodes that can be created in a filesystem is dynamic and it is only limited by the amount of free space in the filesystem. JFS2 supports buffered I/O, synchronous I/O (the file is opened with O_SYNC or O_DSYNC flags), kernel asynchronous I/O (through the use of the Async I/O system calls), Direct I/O (on a per-file basis if the file is opened with O_DIRECT, or on a per-filesystem basis when the filesystem is mounted with the dio mount option), and Concurrent I/O (on a per-file basis if the file is opened with O_CIO or when the filesystem is mounted with the CIO mount option). With AIX, you can use either JFS or JFS2, as they are both linked to the LVM. Both are journalized and no third party filesystems are necessary. In AIX 5L Version 5.1, every filesystem corresponds to a logical volume. In order to create a journaled filesystem, you need to use the smit fastpath: `smitty crfs` or `crfs` from the command line. To increase the size of a filesystem, use the `chfs` command, in addition to using SMIT.

**Conclusion**

Having worked with many administrators who have transitioned to AIX from Solaris, the general rule is that it is a lot easier to transition from Solaris to AIX than the reverse. As a Sun administrator, you should already be very familiar with command-line tools. You will find SMIT refreshing and appreciate the integration of the AIX filesystem with their LVM. There are fewer choices to make, fewer files to edit, and fewer mistakes to make. This freedom allows you to be a more proactive systems administrator. Good luck with the transition!

© Copyright IBM Corporation 2007

**Trademarks**
(www.ibm.com/developerworks/ibm/trademarks/)