Migrating to AIX 6.1 with nimadm

Minimize AIX migration downtime with NIM Alternate Disk Migration

Chris Gibson

January 05, 2010

This article demonstrates how to migrate to AIX® 6.1 using the NIM Alternate Disk Migration tool. You'll learn how to minimize the downtime required when migrating to the latest release of the AIX operating system.

Introduction

Recently, I've been busy upgrading my entire AIX landscape from AIX 5.3 to AIX 6.1. My environment consists of close to 100 AIX LPARs. When tasked with such a challenge I always consider how I can best achieve this goal quickly, efficiently, and with minimal disruption to my customers.

The AIX OS provides the Network Installation Manager (NIM) to assist in administering and updating large numbers of AIX systems. A nice feature of this tool is the NIM Alternate Disk Migration (**nimadm**) facility. Using this tool, as you will soon see, allows you to perform your AIX migrations without the need for lengthy outages.

In this article I'll demonstrate the **nimadm** procedures we used to migrate our AIX systems. I'm going to assume that you are already very familiar with AIX and NIM. I'm also going to assume you already have a NIM master in your environment. If not, I recommend you review the documentation in the Related topics section first.

Overview

Over the years, I've migrated to several new releases of the AIX OS. To do this I would have typically used one of the conventional methods. These methods consisted of either A) migration using the AIX installation DVD or B) migration using NIM. Method A is still possible, even in virtualized environments via the use of File-Backed devices. And method B is also perfectly viable by network booting the client LPAR and performing the migration using a NIM.

The downside with both of these methods is that they both require significant downtime on the LPAR while the migration takes place. This downtime could be anywhere from 30-45 minutes to several hours, depending on the system. This can be a concern in environments with tight outage windows.

The **nimadm** utility offers several advantages over a conventional migration. For example, a system administrator can use**nimadm** to create a copy of a NIM client's rootvg (on a spare disk on the client, similar to a standard alternate disk install **alt_disk_install**) and migrate the disk to a newer version or release of AIX. All of this can be done without disruption to the client (there is no outage required to perform the migration). After the migration is finished, the only downtime required will be a scheduled reboot of the system.

Another advantage is that the actual migration process occurs on the NIM master, taking the load off the client LPAR. This reduces the processing overhead on the LPAR and minimizes the performance impact to the running applications.

For customers with a large number of AIX systems, it is also important to note that the **nimadm** tool supports migrating several clients at once.

To summarize, these are the benefits of **nimadm** over other migration methods:

- Reduced downtime for the client. The migration is executed while the system is up and running as normal. There is no disruption to any of the applications or services running on the client; therefore, the upgrade can be done at a time convenient to the administrator. At a later stage, a reboot can be scheduled in order to restart the system at the later level of AIX.
- The **nimadm** process is very flexible and it can be customized using some of the optional NIM customization resources, such as image_data, bosinst_data, pre/post_migration scripts, exclude_files, and so on.
- Quick recovery from migration failures. All changes are performed on the rootvg copy (altinst_rootvg). If there are any serious problems with the migration, the original rootvg is still available and the system has not been impacted. If a migration fails or terminates at any stage, **nimadm** is able to quickly recover from the event and clean up afterwards. There is little for the administrator to do except determine why the migration failed, rectify the situation, and attempt the **nimadm** process again. If the migration completed but issues are discovered after the reboot, then the administrator can back out easily by booting from the original rootvg disk.

Preparation

There are a few requirements that must be met before attempting to use **nimadm** to migrate to AIX 6.1. I'll mention just some of these here. I recommend that you review the online documentation for **nimadm** or the IBM NIM Redbook for more information (see the Related topics section at the end of this article).

- You must have a NIM master running AIX 6.1 or higher with the latest Technology Level or higher.
- The NIM master must have the **bos.alt_disk_install.rte** fileset installed in its own rootvg and in the SPOT that will be used for the migration. Both need to be at the same level. It is not necessary to install the alternate disk utilities on the client.

- The lpp_source and SPOT NIM resources that have been selected for the migration MUST match the AIX level to which you are migrating.
- The NIM master (as always) should be at the same or higher AIX level than the level you are migrating to on the client.
- The target client must be registered with the NIM master as a standalone NIM client.
- The NIM master must be able to execute remote commands on the client using rsh.
- Ensure the NIM client has a spare disk (not allocated to a volume group) large enough to contain a complete copy of its rootvg. If rootvg is mirrored, break the mirror and use one of the disks for the migration.
- Ensure the clients NIM master has a volume group (for example, nimadmvg) with enough free space to cater for a complete copy of the client's rootvg. If more than one AIX migration is occurring for multiple NIM clients, make sure there is capacity for a copy of each clients rootvg.

Local disk caching versus NFS

By default, the **nimadm** tool utilizes NFS for many of the tasks during the migration. This can be a problem on slower networks because NFS writes can be very expensive. To avoid using NFS, a **Local Disk Caching** option exists that can provide some performance advantages.

Local disk caching allows the NIM master to avoid having to use NFS to write to the client. This can be useful if the **nimadm** operation is not performing well due to an NFS write bottleneck.

If the Local Disk Caching function is invoked, then **nimadm** will create the client file systems in a volume group on the NIM master. It will then use streams (via rshd) to cache all of the data from the client to the file systems on the NIM master.

The advantages of local disk caching over NFS could be summarized as:

- Improved performance for **nimadm** operations on relatively slow networks.
- Improved performance for **nimadm** operations that are bottlenecked in NFS writes.
- Decreased CPU usage on the client.
- Client file systems not exported.
- Allows TCB enabled systems to be migrated with **nimadm**.

Some potential disadvantages of local disk caching are:

- Cache file systems take up space on the NIM master. You must have enough disk space in a volume group on the NIM master to host the client's rootvg file systems, plus some space for the migration of each client.
- Increased CPU usage on the NIM master.
- Increased I/O on the master. For best performance, use a volume group on the NIM master that does not contain the NIM resources being used for the AIX migration.

For performance reasons, we deploy Local Disk Caching with **nimadm** in our environment.

The **nimadm** command performs a migration in 12 phases. It is useful to have some knowledge of each phase before performing a migration.

1. The master issues the alt_disk_install command to the client, which makes a copy of the clients rootvg to the target disks. In this phase, the alternate root volume group (altinst_rootvg) is created.
2. The NIM master creates the cache file systems in the nimadmvg volume group. Some initial checks for the required migration disk space are performed.
3. The NIM master copies the NIM client's data to the cache file systems in nimadmvg. This data copy is done via rsh.
4. If a pre-migration script resource has been specified, it is executed at this time.
5. System configuration files are saved. Initial migration space is calculated and appropriate file system expansions are made. The bos image is restored and the device database is merged (similar to a conventional migration). All of the migration merge methods are executed, and some miscellaneous processing takes place.
6. All system filesets are migrated using installp. Any required RPM images are also installed during this phase.
7. If a post-migration script resource has been specified, it is executed at this time.
8. The bosboot command is run to create a client boot image, which is written to the client's alternate boot logical volume (alt_hd5).
9. All the migrated data is now copied from the NIM master's local cache file and synced to the client's alternate rootvg via rsh.
10. The NIM master cleans up and removes the local cache file systems.
11. The alt_disk_install command is called again to make the final adjustments and put altinst_rootvg to sleep. The bootlist is set to the target disk.
12. Cleanup is executed to end the migration.

If you are unable to meet the requirements for phases 1 to 10, then you should consider performing a conventional migration.

Before we move onto a **nimadm** example, I just want to add a few points for you to consider first.

- I recommended that you do not to make any changes to your system once the migration is underway, such as adding users, changing passwords, adding print queues, and the like. If possible, wait until the migration has finished and the system has been rebooted on the new version of AIX. If you must perform administration tasks prior to the reboot, you should take note of the changes and re-apply them to the system after it has been rebooted into AIX 6.1.
- We developed, tested, and verified our migration procedures several times before implementing them on our production systems. This allowed us time to verify that the steps were correct and that the AIX migrations would complete as expected. I recommend you do the same.
- If you have a multibos image in rootvg, remove it. AIX migrations are not supported with multibos enabled systems. Ensure all rootvg LVs are renamed to their legacy names. If necessary, create a new instance of rootvg and reboot the LPAR. For example:

```
# multibos –sXp
# multibos –sX
# shutdown –Fr
```

Confirm the legacy LV names are now in use that is, **not bos_.**

```
# lsvg -l rootvg | grep hd | grep open
hd6           paging     80      160     2    open/syncd    N/A
hd8           jfs2log    1       2       2    open/syncd    N/A
hd4           jfs2       1       2       2    open/syncd    /
hd2           jfs2       7       14      2    open/syncd    /usr
hd3           jfs2       16      32      2    open/syncd    /tmp
hd1           jfs2       1       2       2    open/syncd    /home
hd9var        jfs2       8       16      2    open/syncd    /var
hd7           sysdump    8       8       1    open/syncd    N/A
hd7a          sysdump    8       8       1    open/syncd    N/A
hd10opt       jfs2       8       16      2    open/syncd    /opt
```

Remove the old multibos instance.

```
# multibos -R
```

Migrating to AIX 6.1 using nimadm

Let's use **nimadm** now to migrate an AIX system. Ensure that you document the system and perform a mksysb before performing any maintenance activity. You know this already, right? But I have to say it!

We will migrate a system from AIX 5.3 to AIX 6.1. The NIM master in this environment is running AIX 6.1 TL3 SP2. Our NIM client name is aix1 (running AIX 5.3 TL7 SP5 and migrating to AIX 6.1 TL3 SP1) and the NIM masters name is nim1.

Ensure that you read the AIX 6.1 release notes and review the documented requirements such as the amount of free disk space required.

Prior to a migration, it is always a good idea to run the pre_migration script on the system to catch any issues that may prevent the migration from completing successfully. You can find this script on the AIX 6.1 installation media.

Run this script, review the output (in /home/pre_migration), and correct any issues that it reports before migrating.

```
#./pre_migration

All saved information can be found in: /home/pre_migration.090903105452

Checking size of boot logical volume (hd5).

Your rootvg has mirrored logical volumes (copies greater than 1)
Recommendation:  Break existing mirrors before migrating.

Listing software that will be removed from the system.

Listing configuration files that will not be merged.

Listing configuration files that will be merged.

Saving configuration files that will be merged.

Running lppchk commands. This may take awhile.

Please check /home/pre_migration.090903105452/software_file_existence_check
for possible errors.
```

```
Please check /home/pre_migration.090903105452/software_checksum_verification
for possible errors.

Please check /home/pre_migration.090903105452/tcbck.output for possible errors.

All saved information can be found in: /home/pre_migration.090903105452

It is recommended that you create a bootable system backup of your system
before migrating.
```

I always take a copy of the /etc/sendmail.cf and /etc/motd files before an AIX migration. These files will be replaced during the migration and you will need to edit them again and add your modifications.

Commit any applied filesets. You should also consider removing any ifixes that may hinder the migration.

If rootvg is mirrored, I break the mirror and reduce it to a single disk. This gives me a spare disk that can be used for the migration.

To allow **nimadm** to do its job, I must temporarily enable rshd on the client LPAR. I will disable it again after the migration.

```
 # chsubserver -a -v shell -p tcp6 -r inetd
 # refresh –s inetd
 # cd /
 # rm .rhosts
 # vi .rhosts
 +
 # chmod 600 .rhosts
```

On the NIM master, I can now 'rsh' to the client and run a command as root.

```
 # rsh aix1 whoami
 root
```

At this point I'm ready to migrate. The process will take around 30-45 minutes; all the while the applications on the LPAR will continue to function as normal.

On the NIM master, I have created a new volume group (VG) named nimadmvg. This VG has enough capacity to cater for a full copy of the NIM clients root volume group (rootvg). This VG will be empty until the migration is started.

Likewise, on the NIM client, I have a spare disk which has enough capacity for a full copy of its rootvg.

On the master (nim1):

```
 # lsvg -l nimadmvg
 nimadmvg:
 LV NAME  TYPE       LPs     PPs     PVs  LV STATE      MOUNT POINT
```

On the client (aix1):

```
 # lspv
 hdisk0 0000273ac30fdcfc rootvg          active
 hdisk1 000273ac30fdd6e  None
```

The fileset bos.alt_disk_install.rte fileset is installed on the NIM master:

```
# lslpp -l bos.alt_disk_install.rte
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  bos.alt_disk_install.rte   6.1.3.1  APPLIED    Alternate Disk Installation
                                                 Runtime
```

And it is also installed in the AIX 6.1 TL3 SP1 SPOT:

```
# nim -o showres 'spotaix61031'  | grep bos.alt_disk_install.rte
  bos.alt_disk_install.rte   6.1.3.1    C     F    Alternate Disk Installation
```

The **nimadm** command is executed from the NIM master.

```
 # nimadm -j nimadmvg -c aix1 -s spotaix61031 -l lppsourceaix61031 -d "hdisk1" –Y
```

Where:

- –j flag specifies the VG on the master which will be used for the migration
- -c is the client name
- –s is the SPOT name
- -l is the lpp_source name
- -d is the hdisk name for the alternate root volume group (altinst_rootvg)
- –Y agrees to the software license agreements for software that will be installed during the migration.

Now I can sit back and watch the migration take place. All migration activity is logged on the NIM master in the /var/adm/ras/alt_mig directory. For this migration, the log file name is aix1_alt_mig.log. Here's a sample of some of the output you can expect to see for each phase:

```
MASTER DATE: Mon Nov  9 14:29:09 EETDT 2009
CLIENT DATE: Mon Nov  9 14:29:09 EETDT 2009
NIMADM PARAMETERS: -j nimadmvg -c aix1 -s spotaix61031 -l lppsourceaix61031 -d hdisk1 -Y
Starting Alternate Disk Migration.

+-----------------------------------------------------------------------+
Executing nimadm phase 1.
+-----------------------------------------------------------------------+
Cloning altinst_rootvg on client, Phase 1.
Client alt_disk_install command: alt_disk_copy -j -i /ALT_MIG_IMD -M 6.1 -P1 -d "hdisk1"
Checking disk sizes.
Creating cloned rootvg volume group and associated logical volumes.
Creating logical volume alt_hd5.
Creating logical volume alt_hd6.
Creating logical volume alt_hd8.
Creating logical volume alt_hd4.
Creating logical volume alt_hd2.
Creating logical volume alt_hd9var.
Creating logical volume alt_hd3.
Creating logical volume alt_hd1.
Creating logical volume alt_hd10opt.
Creating logical volume alt_hd7.
Creating logical volume alt_local_lv.
Creating logical volume alt_varloglv.
Creating logical volume alt_nmonlv.
Creating logical volume alt_chksyslv.
Creating logical volume alt_hd71.
Creating logical volume alt_auditlv.
Creating logical volume alt_nsrlv.
Creating logical volume alt_hd11admin.
Creating /alt_inst/ file system.
Creating /alt_inst/admin file system.
Creating /alt_inst/home file system.
Creating /alt_inst/home/nmon file system.
Creating /alt_inst/nsr file system.
Creating /alt_inst/opt file system.
Creating /alt_inst/tmp file system.
Creating /alt_inst/usr file system.
Creating /alt_inst/usr/local file system.
Creating /alt_inst/usr/local/chksys file system.
Creating /alt_inst/var file system.
Creating /alt_inst/var/log file system.
Creating /alt_inst/var/log/audit file system.
Generating a list of files
for backup and restore into the alternate file system...
Phase 1 complete.

+-----------------------------------------------------------------------+
Executing nimadm phase 2.
+-----------------------------------------------------------------------+
Creating nimadm cache file systems on volume group nimadmvg.
Checking for initial required migration space.
Creating cache file system /aix1_alt/alt_inst
Creating cache file system /aix1_alt/alt_inst/admin
Creating cache file system /aix1_alt/alt_inst/home
Creating cache file system /aix1_alt/alt_inst/home/nmon
Creating cache file system /aix1_alt/alt_inst/nsr
Creating cache file system /aix1_alt/alt_inst/opt
Creating cache file system /aix1_alt/alt_inst/tmp
Creating cache file system /aix1_alt/alt_inst/usr
Creating cache file system /aix1_alt/alt_inst/usr/local
Creating cache file system /aix1_alt/alt_inst/usr/local/chksys
Creating cache file system /aix1_alt/alt_inst/var
Creating cache file system /aix1_alt/alt_inst/var/log
Creating cache file system /aix1_alt/alt_inst/var/log/audit

+-----------------------------------------------------------------------+
Executing nimadm phase 3.
+-----------------------------------------------------------------------+
Syncing client data to cache ...

+-----------------------------------------------------------------------+
Executing nimadm phase 4.
+-----------------------------------------------------------------------+
nimadm: There is no user customization script specified for this phase.

+-----------------------------------------------------------------------+
Executing nimadm phase 5.
+-----------------------------------------------------------------------+
Saving system configuration files.
Checking for initial required migration space.
Setting up for base operating system restore.
/aix1_alt/alt_inst
Restoring base operating system.
```

```
Merging system configuration files.
Running migration merge method: ODM_merge Config_Rules.
Running migration merge method: ODM_merge SRCextmeth.
Running migration merge method: ODM_merge SRCsubsys.
Running migration merge method: ODM_merge SWservAt.
Running migration merge method: ODM_merge pse.conf.
Running migration merge method: ODM_merge vfs.
Running migration merge method: ODM_merge xtiso.conf.
Running migration merge method: ODM_merge PdAtXtd.
Running migration merge method: ODM_merge PdDv.
Running migration merge method: convert_errnotify.
Running migration merge method: passwd_mig.
Running migration merge method: login_mig.
Running migration merge method: user_mrg.
Running migration merge method: secur_mig.
Running migration merge method: RoleMerge.
Running migration merge method: methods_mig.
Running migration merge method: mkusr_mig.
Running migration merge method: group_mig.
Running migration merge method: ldapcfg_mig.
Running migration merge method: ldapmap_mig.
Running migration merge method: convert_errlog.
Running migration merge method: ODM_merge GAI.
Running migration merge method: ODM_merge PdAt.
Running migration merge method: merge_smit_db.
Running migration merge method: ODM_merge fix.
Running migration merge method: merge_swvpds.
Running migration merge method: SysckMerge.

+-----------------------------------------------------------------------+
Executing nimadm phase 6.
+-----------------------------------------------------------------------+
Installing and migrating software.
Updating install utilities.
+-----------------------------------------------------------------------+
      Pre-installation Verification...
+-----------------------------------------------------------------------+
Verifying selections...done
Verifying requisites...done
Results...
```

…output truncated….

```
install_all_updates: Generating list of updatable rpm packages.
install_all_updates: No updatable rpm packages found.

install_all_updates: Checking for recommended maintenance level 6100-03.
install_all_updates: Executing /usr/bin/oslevel -rf, Result = 6100-03
install_all_updates: Verification completed.
install_all_updates: Log file is /var/adm/ras/install_all_updates.log
install_all_updates: Result = SUCCESS
Restoring device ODM database.

+-----------------------------------------------------------------------+
Executing nimadm phase 7.
+-----------------------------------------------------------------------+
nimadm: There is no user customization script specified for this phase.

+-----------------------------------------------------------------------+
Executing nimadm phase 8.
+-----------------------------------------------------------------------+
Creating client boot image.
bosboot: Boot image is 40952 512 byte blocks.
Writing boot image to client's alternate boot disk hdisk1.

+-----------------------------------------------------------------------+
Executing nimadm phase 9.
+-----------------------------------------------------------------------+
Adjusting client file system sizes ...
Adjusting size for /
Adjusting size for /admin
Adjusting size for /home
Adjusting size for /home/nmon
Adjusting size for /nsr
Adjusting size for /opt
Adjusting size for /tmp
Adjusting size for /usr
Adjusting size for /usr/local
Adjusting size for /usr/local/chksys
Adjusting size for /var
Adjusting size for /var/log
Adjusting size for /var/log/audit
Syncing cache data to client ...

+-----------------------------------------------------------------------+
Executing nimadm phase 10.
+-----------------------------------------------------------------------+
Unmounting client mounts on the NIM master.
forced unmount of /aix1_alt/alt_inst/var/log/audit
forced unmount of /aix1_alt/alt_inst/var/log
forced unmount of /aix1_alt/alt_inst/var
forced unmount of /aix1_alt/alt_inst/usr/local/chksys
forced unmount of /aix1_alt/alt_inst/usr/local
forced unmount of /aix1_alt/alt_inst/usr
forced unmount of /aix1_alt/alt_inst/tmp
forced unmount of /aix1_alt/alt_inst/opt
forced unmount of /aix1_alt/alt_inst/nsr
forced unmount of /aix1_alt/alt_inst/home/nmon
forced unmount of /aix1_alt/alt_inst/home
forced unmount of /aix1_alt/alt_inst/admin
forced unmount of /aix1_alt/alt_inst
Removing nimadm cache file systems.
Removing cache file system /aix1_alt/alt_inst
Removing cache file system /aix1_alt/alt_inst/admin
Removing cache file system /aix1_alt/alt_inst/home
Removing cache file system /aix1_alt/alt_inst/home/nmon
Removing cache file system /aix1_alt/alt_inst/nsr
Removing cache file system /aix1_alt/alt_inst/opt
Removing cache file system /aix1_alt/alt_inst/tmp
Removing cache file system /aix1_alt/alt_inst/usr
Removing cache file system /aix1_alt/alt_inst/usr/local
Removing cache file system /aix1_alt/alt_inst/usr/local/chksys
Removing cache file system /aix1_alt/alt_inst/var
Removing cache file system /aix1_alt/alt_inst/var/log
Removing cache file system /aix1_alt/alt_inst/var/log/audit

+-----------------------------------------------------------------------+
Executing nimadm phase 11.
+-----------------------------------------------------------------------+
Cloning altinst_rootvg on client, Phase 3.
Client alt_disk_install command: alt_disk_copy -j -i /ALT_MIG_IMD -M 6.1 -P3 -d "hdisk1"
## Phase 3 ###################
Verifying altinst_rootvg...
Modifying ODM on cloned disk.
forced unmount of /alt_inst/var/log/audit
forced unmount of /alt_inst/var/log
forced unmount of /alt_inst/var
forced unmount of /alt_inst/usr/local/chksys
forced unmount of /alt_inst/usr/local
forced unmount of /alt_inst/usr
forced unmount of /alt_inst/tmp
forced unmount of /alt_inst/opt
forced unmount of /alt_inst/nsr
forced unmount of /alt_inst/home/nmon
forced unmount of /alt_inst/home
forced unmount of /alt_inst/admin
forced unmount of /alt_inst
Changing logical volume names in volume group descriptor area.
Fixing LV control blocks...
Fixing file system superblocks...
Bootlist is set to the boot disk: hdisk1 blv=hd5

+-----------------------------------------------------------------------+
Executing nimadm phase 12.
+-----------------------------------------------------------------------+
Cleaning up alt_disk_migration on the NIM master.
Cleaning up alt_disk_migration on client aix1.
```

After the migration is complete, I confirm that the bootlist is set to the **nst_rootvg** disk.

```
 # lspv | grep rootvg
 hdisk0 0000273ac30fdcfc rootvg          active
 hdisk1 000273ac30fdd6e  altinst_rootvg  active


 # bootlist -m normal -o
 hdisk1 blv=hd5
```

At an agreed time, I reboot the LPAR and confirm that the system is now running AIX 6.1.

```
 # shutdown –Fr

 ; system reboots here…

 # oslevel –s
 6100-03-01-0921


 # instfix -i | grep AIX
     All filesets for 6.1.0.0_AIX_ML were found.
     All filesets for 6100-00_AIX_ML were found.
     All filesets for 6100-01_AIX_ML were found.
     All filesets for 6100-02_AIX_ML were found.
     All filesets for 6100-03_AIX_ML were found.
```

At this point, I would perform some general AIX system health checks to ensure that the system is configured and running as I'd expect. There is also a post_migration script that you can run to verify the migration. You can find this script in /usr/lpp/bos, after the migration.

You may want to consider upgrading other software such as openssl, openssh, lsof, etc at this stage.

The rsh daemon can now be disabled after the migration.

```
 # chsubserver -d -v shell -p tcp6 -r inetd
 # refresh –s inetd
 # cd /
 # rm .rhosts
 # ln -s /dev/null .rhosts
```

With the migration finished, the applications are started and the application support team verify that everything is functioning as expected. I also take a mksysb and document the system configuration after the migration.

Once we are all satisfied that the migration has completed successfully, we then return rootvg to a mirrored disk configuration.

```
# lspv | grep old_rootvg
hdisk0  000071da26fe3bd0      old_rootvg
# alt_rootvg_op -X old_rootvg
# extendvg –f rootvg hdisk0
# mirrorvg rootvg hdisk0
# bosboot -a -d /dev/hdisk0
# bosboot -a -d /dev/hdisk1
# bootlist -m normal hdisk0 hdisk1
# bootlist -m normal -o
hdisk0 blv=hd5
hdisk1 blv=hd5
```

If there was an issue with the migration, I could easily back out to the previous release of AIX. Instead of re-mirroring rootvg (above), we would change the boot list to point at the previous rootvg disk (old_rootvg) and reboot the LPAR.

```
# lspv | grep old_rootvg
hdisk0  000071da26fe3bd0      old_rootvg
# bootlist -m normal hdisk0
# bootlist -m normal –o
hdisk0 blv=hd5
# shutdown –Fr
```

This is much simpler and faster than restoring a mksysb image (via NIM, tape, or DVD), as you would with a conventional migration method.

Conclusion

By using **nimadm**, we were able to reduce the overall downtime required when migrating our systems to AIX 6.1. We were also given a convenient way to back out a migration, had it been necessary to do so. I hope this provides you with some ideas on how to best migrate your systems to AIX 6.1, when the time comes.

Related topics

- nimadm
- The NIM from A to Z in AIX 5L Redbook helps you understand the benefits of implementing a Network Installation Manager (NIM) environment.
- Release Notes Index for AIX Version 6.1 and Expansion Pack
- BOS pre- and post-migration checks
- Migrating to AIX 6.1
- Checking modifications to configuration files