developerWorks®

The NIM cheat sheet

Christian Pruett

January 07, 2013

Have you ever had to install a large number of IBM AIX servers from scratch? Do you need a quick and easy way to create and store operating system backups? Would you like to know how to perform AIX operating system migrations on live servers quickly? Well, here is your complete cheat sheet for setting up and using the Network Installation Manager (NIM) tool in AIX. Learn how to set up the server and client components, perform a basic installation, use NIM to create `mksysb` backups, and perform `alt_disk` migrations over the network.

If you have been an IBM® AIX® systems administrator for some time, you probably remember the old days of building out server after server by carrying around a stack of CDs under your arm and shuttling them between drives. It could be quite an endeavor to build out even a dozen IBM RS/6000® servers, taking several days depending on how many copies of the AIX operating system base media you had available. And that did not count any postinstallation customization or configuration required on every server. Plus, if you were restoring system images from a `mksysb` tape backup, there was the additional fun of making sure that the CDs had a version of the operating system on them so that any drivers that were not present on the tape media could be installed on the server, and then you had to make it bootable, too.

Frequently used acronyms

- **DNS:** Domain Name System
- **LPAR:** Logical partition
- **NFS:** Network file system
- **SAN:** Storage area network
- **SSH:** Secure Shell

Fortunately, with AIX version 4, IBM introduced the Network Installation Manager (NIM) feature, a versatile tool that handles remote installation and management of the operating system over a network. This tool, which comes as a part of the AIX operating system base media, transforms a server into a specialized host from which other servers can be built, patched, and customized. As the years have progressed, along with the development of newer versions of the AIX operating system and the evolution of hardware technology from IBM RS/6000 to IBM eServer™, IBM pSeries® systems, and now IBM Power System® servers, NIM has become a robust piece of software for systems management.

This article is a cheat sheet for setting up and configuring a basic NIM environment using industry-standard practices. It provides realworld tips and tricks to help you make a maintainable and well-designed system that can grow and expand with your environment. The article also walks you through using NIM to build out a new system, create a backup of it, and perform a migration upgrade.

Understanding NIM

NIM works by making the AIX operating system base media, associated Licensed Program Products (LPPs), scripts, and backups on a server available to other AIX servers across a network. It does this through specialized filesets that provide the code for configuring the server and common network protocols and services such as the Bootstrap Protocol (BOOTP), the Trivial File Transfer Protocol (TFTP), and NFS to make the data accessible.

The system that has the NIM software and associated files on it is referred to as the *NIM master;* all of the systems it manages are called *NIM clients.* The NIM master must be at the same operating system level or higher as the NIM clients to properly build and manage them. The NIM master can manage hundreds of NIM clients at once and run multiple tasks simultaneously on them.

For a NIM client to boot from the NIM master and be built, two things are necessary: an LPP source and a shared product-object tree (SPOT) image. The LPP source is essentially a copy of the filesets from the AIX operating system base media, which you can customize with other software packages. The SPOT is a bootable image that contains key files such as the AIX kernel and drivers. These NIM objects are accessed when a NIM client hits the NIM server through BOOTP, and then transfers the SPOT through TFTP. From there, the rest of the software from the LPP source is restored by NFS onto the NIM client, which saves all of the Object Data Manager information, and then sends back a confirmation signal to the NIM master and reboots the NIM client.

In the case of other software management (including upgrading the operating system, installing a fileset, or creating a `mksysb` backup), you perform this management through NFS and communication with each other either by `r` commands or with a special NIM Service Handler (NIMSH) subsystem for management.

NIM master considerations

There are four considerations for building a NIM master:

- Base media
- Proper location
- Sufficient disk space
- Network routing

First, to build the NIM master, you need a copy of the AIX operating system base media on CD, DVD, or some other storage device, because you need to install the NIM filesets off of the media to create the NIM master. You then use this media to help shape your initial LPP source.

Second, you need to determine the proper location or system to be your NIM master. Generally, I have found it best to have a separate system partition or logical partition (LPAR) to accomplish this task, because if you ever need to do some semi-intrusive work on the server, such as restart NFS or reboot the system, you do not want it to interfere with a production or customer environment. Plus, I like to use my NIM server as a bastion host from which I can manage the rest of my servers in other ways, such as keeping all of the SSH keys for the systems in one place. Fortunately, NIM is not a huge resource hog, and creating a NIM master with shared network and storage resources through virtual input and output, with a fraction of a processor and 1 or 2 GB of RAM, is usually more than enough to give it all the resources it needs.

Third, the NIM master needs to have sufficient disk space to hold everything. Although a NIM master might not need many resources to serve up images, this is the one area where I have found it better to go a little overboard. A proper NIM master needs ample storage to hold all of the LPP source images for the various operating system levels you plan to deploy, their SPOT images, and any `mksysb` images you will use for backups or rapid deployments.

In general, it is wise to allocate at least 6 GB for each LPP source, 4 GB for each SPOT image, about 10 GB for a `mksysb` image of a system that has most of its data in non-rootvg volume groups, and about 25 GB for a `mksysb` image of a system that has both operating system and data within the rootvg volume group. This configuration provides a little extra space in case of emergencies and for copying and integrating newer technology levels (TLs), which I discuss later. I usually create a separate volume group to contain everything and make at least one file system for each major version and release, coupled with one file system for `mksysb` images. Table 1 provides an example of how to set up the necessary storage for a NIM master.

Table 1. Setting up storage for a NIM master

Directory	Size	Description
nimvg	500 GB	NIM volume group
/nim	1 GB	Catch-all file system for operating system delineation
/nim/5300	50 GB	File system for AIX 5.3 and associated TLs
/nim/5300/lpp_source_5300-09-05		Directory for the LPP source of AIX 5.3 TL 09 SP 05
/nim/5300/spot_5300-09-05/usr		Directory for the SPOT of AIX 5.3 TL09 SP 05
/nim/5300/TEMP		Temporary working directory for newer operating systems
/nim/6100	50 GB	File system for AIX 6.1 and associated TLs
/nim/6100/lpp_source_6100-04-03		Directory for the LPP source of AIX 6.1 TL 04 SP 03
/nim/6100/spot_6100-04-03/usr		Directory for the SPOT of AIX 6.1 TL 04 SP 03
/nim/6100/lpp_source_6100-06-05		Directory for the LPP source of AIX 6.1 TL 06 SP 05
/nim/6100/spot_6100-06-05/usr		Directory for the SPOT of AIX 6.1 TL 06 SP 05
/nim/6100/TEMP		Temporary working directory for newer operating systems

/nim/mksysb	100 GB	Directory for mksysb images of servers

Finally, consider the network routing you need to build and manage NIM clients in your infrastructure. Sometimes, this routing is as simple as a single class C subnet that is shared with the NIM master itself. Other times, it might be necessary to deploy to several disparate networks that are accessible on different interfaces. Speak with your network administrators, and gather information such as the primary gateway for servers on that network, the subnet mask, and IP addresses for your NIM clients. The most important thing to know is which device serves as your primary network interface and the subnet on which it resides. For example, you might use en0 for communicating on the 192.168.0.1/24 subnet, because it houses the majority of your NIM clients. You need to know information to give the network a name within NIM for identification as you build your NIM master (for example, ether_192_168_0).

Building the NIM master

The first step to setting up the NIM environment is to establish a NIM master to control everything. The following procedure guides you through building a server as a NIM master:

1. Build a server from the AIX operating system base media, or pick a server that already has all of the considerations just listed to act as the NIM master.
2. From the AIX operating system base media, install the bos.sysmgt.nim.master, bos.sysmgt.nim.client, and bos.sysmgt.nim.spot filesets.
   Select the option for installing requisite software to get any other needed filesets.
3. Create the storage structure to be used for containing all of the LPP sources, SPOTs, and `mksysb` images.
4. Run the `nimconfig` command to initialize the server as the NIM master.
5. Insert your primary network interface and primary network name (NETNAME) in the appropriate variables.
   For example, the following command initializes the NIM master for 64-bit kernels using the Ethernet interface en0 on the 192.168.0.1/24 subnet and does not require clients to be preregistered on the NIM master:

   ```
   nimconfig a netname=ether_192_168_0 a pif_name=en0
   a netboot_kernel=64 a cable_type=tp a client_reg=no
   ```
6. Confirm that the NIM master has been built by running the `lsnim` command and checking for the network definition.

Establishing an LPP source and SPOT

After you have a NIM master, you need to set up an LPP source and build a SPOT image so that NIM clients can be booted off of the NIM master and have their operating systems installed. The following procedure allows you to use the same AIX operating system base media (used to build the NIM master) to set up these resources:

1. Make the operating system base media accessible to the server, and mount it, if necessary. Ensure that the operating system is greater than or equal to the version of the operating system on the media; an upgrade might be necessary.
2. Copy over all of the operating system media filesets into a temporary working directory.

Most of the time, these filesets can be found within the /usr/sys/inst.images/installp/ppc directory on the base media. Use the `bffcreate` command to accomplish this for each CD, DVD, or volume. For example, to put the contents from the /mnt directory in a temporary working directory for AIX 6.1, run the following command line:

```
bffcreate XSq d /mnt t /nim/6100/TEMP all
```

3. Use the `lppmgr` command to remove redundant filesets and extraneous language filesets. By running this command, it is possible to save several gigabytes of disk space for each LPP source. For example, to interactively keep only the en_US language filesets within the same directory as just mentioned, run the following command line:

   ```
   /usr/lib/instl/lppmgr d /nim/6100/TEMP pr k en_US
   ```

   Or, to interactively remove duplicate filesets within the same directory, run:

   ```
   /usr/lib/instl/lppmgr d/nim/6100/TEMP  burp
   ```
4. Create the LPP source by issuing the `nim` command to put the files from the temporary working directory into their final place.
   For example, to create an LPP source for AIX 6.1 TL 04 SP 03 named *lpp_source_6100-04-03,* run the following command line:

   ```
   nim o define t lpp_source
   a server=master a location=/nim/6100/lpp_source_6100-04-03
   a source=/nim/6100/TEMP lpp_source_6100-04-03
   ```
5. Create the SPOT image from the LPP source with the `nim` command.
   It is not necessary to specify a terminal directory, because this directory is automatically created in the process; in other words, you need only point to the underlying directory, because the SPOT directory is automatically generated with the SPOT name. Using the previous example, for instance, to create a SPOT image named *spot_6100-04-03,* run:

   ```
   nim o define t spot a server=master a location=/nim/6100
   a source=lpp_source_6100-04-03 a installp_flags=-aQg spot_6100-04-03
   ```
6. Use the `lsnim` command to confirm that the LPP source and SPOT image are in place.
7. Clean up any mounts and the temporary working directories, as needed.

Building a NIM client

After you have defined the LPP source and SPOT image, you are ready to build a new server as a NIM client. This process requires that you perform activities to define and prepare the NIM client as an object on the NIM master and to then boot the NIM client from the network. In this example, you build a new server named *client01* on the same subnet as the NIM master:

1. Enter the NIM client in the /etc/hosts file on the NIM master, or make sure that the intended hostname resolves through DNS.
2. Define the NIM client with the `nim` command, including the hostname, kernel type, network name, and server type.
   For example:

   ```
   nim -o define -t standalone -a platform=chrp
   -a netboot_kernel=64 -a if1="ether_192_168_0
   -a cable_type1=tp client01" client01
   ```
3. Initialize an LPP source installation by allocating the previously created LPP source and SPOT image to the NIM client.

   Doing so creates all the necessary underpinnings for BOOTP, TFTP, and NFS to share out the required resources to boot and install the operating system on the NIM client—for example:

   ```
   nim -o bos_inst -a spot=spot_6100-04-03
   -a lpp_source=lpp_source_6100-04-03
   -a no_client_boot=yes -a accept_licenses=yes client01
   ```

4. Boot the NIM client system into the System Management Services (SMS) menu by selecting the appropriate LPAR boot option or pressing the 1 or F1 key when the icon or word **keyboard** appears in the console:
   1. From the **Remote Initial Program Load (RIPL)** menu, navigate to the particular network adapter for the NIM installation.
   2. Enter the NIM client's IP address, the NIM master's IP address, the gateway, and the subnet mask that will be used.
   3. Ensure that the interface works by going back up a couple of menus and performing a Ping test.
5. Return to the main SMS window on the NIM client, and go into the boot options. Select the appropriate network interface as the boot device for a normal mode boot, and exit the SMS. At this point, the server should begin a packet count as the SPOT image is transferred to begin the boot process.
6. Select the appropriate console and language options. Then, in the main installation window, pick any other options for the installation.
   Generally, I find it unwise to accept the default installation option. At the very least, I confirm the hard disk on which the operating system will be installed.
7. Confirm the installation selection.

NIM begins pushing the operating system build. The duration of the installation can range from just a few minutes to more than an hour depending on the network speed, type, distance, and performance of the hardware. When the installation is complete, the server automatically reboots and comes back up to a login prompt.

Create a mksysb for NIM

After I have finished building a server, customizing it and tailoring it for the purpose for which it was built, I create a `mksysb` backup of the server by uing NIM. I do this for two reasons. First, in case of a disaster, I can quickly recover the operating system with all of its nuances, such as user IDs, file systems, software, and customizations, in place. Second, if I need to build several servers of similar nature at once, such as database servers, I can have the application teams sign off on the configuration of a single server, and then create a "gold image" through which all of the other systems can be installed.

To create a `mksysb` image of the new build, perform these steps:

1. Confirm that the server is a NIM client and defined on the NIM master.
   If the server has not been previously defined on the NIM master, use `smitty niminit fastpath` on the server that will be a NIM client, and enter the appropriate information for communicating and registering with the NIM master. This process creates the requisite /etc/niminfo file and starts the NIMSH subsystem, if needed.

2. On the NIM master, use the `nim` command to both define the `mksysb` image and simultaneously take a `mksysb` image of the NIM client.
   For example, to create a `mksysb` image of the server client01 and place the image in the /nim/mksysb file system, run:

   ```
   nim -o define -t mksysb -a server=master
   -a location=/nim/mksysb/client01_mksysb_GOLD -a mk_image=yes
   -a source=client01 client01_mksysb_GOLD
   ```
3. To use the `mksysb` image to build a new server, use the same `nim o bos_inst` command from the previous section, but include the option `a source=mksysb` and specify the image by the option `a mksysb=$IMAGE`.

Using NIM alternate-disk migrations

I have found AIX to be one of the best flavors of the UNIX® operating system in regard to upgrades. Although some forms of UNIX require a complete tear-down and reinstallation of the operating system if you want to upgrade it from one version and release to another, AIX has a terrific migration-installation option to carry the operating system along, such as going from AIX 5.3 to AIX 6.1. But with NIM, there is an even better way to upgrade a live server: the NIM alternate disk-migration function.

The NIM alternate-disk migration takes the native `alt_disk` function for creating a cloned disk and replicates the operating system onto it. Then, it applies the migration to that disk, all while the server remains up and running the entire time. All that is needed is a quick reboot, and the server comes up on the new disk at the higher operating system level. Here is the process for using this incredible tool:

1. On the NIM client, acquire a disk to contain the upgraded operating system.
   Generally, you can do this by breaking a root disk mirror or assigning a spare disk through a SAN. Typically, this disk must be at least the same size as the used space of the current rootvg, plus a few gigabytes more for the new code.
2. On the NIM master, use the `nimadm` command to make the alternate disk and perform the migration.
3. Specify the LPP source, SPOT image, and disk name to which the operating system will be cloned and upgraded.
   For example, to upgrade the server client01 to AIX 6.1 TL 06 SP 05 on hdisk1, run:

   ```
    nimadm H c client01 l lpp_source_6100-06-05 s spot_6100-06-05 d hdisk1 -Y
   ```
4. When the migration is complete, confirm that the server boots off of the target disk by running `bootlist om normal` and examining the output.
5. Reboot the server when appropriate with a `shutdown Fr` command.
6. When the server comes up, confirm the upgrade with the `oslevel` command.
7. If a problem, such as software incompatibility, warrants backing out the migration, set the server to boot off of the original root disk, and reboot one more time. The server will come back on the original, unaltered disk.
8. If no backout is needed, remirror the root disks when necessary.

Conclusion

Although this article barely touches the surface of the power available through NIM, it provides a basic walkthrough for setting up a simple NIM environment, complete with a NIM master, NIM clients, building operating systems, backing up servers, and upgrading them using the latest technology. Much more is available in the world of NIM, and I hope this article helps set you on the path for using it to manage your AIX servers.

Related topics

- Read the IBM Redbook, NIM from A to Z in AIX 5L for more information about NIM.
- Check out Network Installation Management in the AIX 7.1 IBM Publib Infocenter.
- Find helpful virtual input and output tips and tricks in the developerWorks article VIO cheat sheet (Christian Pruett, July 2010).
- Technology bookstore: Browse the technology bookstore for books on this and other technical topics.