xCAT 2.0 on AIX Alpha Release Cookbook

03/12/2008

1.1 Release Description

xCAT (Extreme Cluster Administration Tool) is a toolkit that provides support for the deployment and administration of cluster environments.

Earlier versions of xCAT have been used to deploy and manage many high end Linux clusters. The new version 2.0 of xCAT is a complete rewrite of xCAT that includes many architectural changes and functional enhancements.

This Cookbook describes the initial release (alpha) of xCAT version 2.0 support for an AIX cluster. It includes some basic support unique to AIX systems.

The intent of this early release is to give the xCAT user community an idea of how AIX will be supported and to get some initial feedback.

1.1.1 xCAT for AIX Support Summary

The initial support provided by xCAT for AIX clusters includes the following:

- A downloadable tar file containing the required Open Source Software (OSS).
- A downloadable tar file containing the xCAT for AIX software.
- Scripts to automatically install OSS and xCAT software. (Included in tar files.)
- xCAT cluster data stored in a relational database. (SQLite software is included in the OSS tar file.)
- Commands to manipulate the xCAT database tables directly. (**tabdump**, **tabrestore**, **tabedit**, **chtab**)
- Commands to manage xCAT data object definitions (**mkdef**, **chdef**, **lsdef**, **rmdef**).
- Hardware control commands for discovering hardware, gathering MAC addresses, VPD, and environmentals, power control, initiating a network boot, and LPAR creation/deletion. (**lsslp, rscan, rinv, rpower, rvitals, getmacs, rnetboot, lsvm, mkvm, chvm, rmvm**).
- Initial support for converting xCAT definitions into AIX/NIM definitions. (**xcat2nim**)
- Parallel remote shell and remote copy commands. (**psh, xdsh, xdcp, xdshbak**)
- Support for AIX 5.3 and higher releases.
- Support for POWER 5 and POWER 6 hardware.
- Notification infrastructure which lets users monitor xCAT database table changes. (**regnotif, unregnotif, startmon, stopmon**)
- "**man**" pages for all xCAT commands mentioned above.

1.1.2 Coming Soon

- xCAT service node support to improve scaling.
- Remote client access to the xCAT management node.
- Remote console support.
- Operating system image management support.
- Support for post-installation customization scripts.
- Support for AIX diskless/stateless nodes.
- Enhanced automation of NIM setup.
- Secondary adapter configuration support.
- Improved IBM HPC product integration.

1.1.3 Licensing

xCAT 2.0 is open source software with an EPL license. For license information visit:

http://www.opensource.org/licenses/eclipse-1.0.php

1.2 Installing xCAT 2.0 and prerequisite Software

1. Set up an AIX system to use as an xCAT Management Node

   - Follow AIX documentation and procedures to install and configure the base AIX operating system. (Typically by using the product media.)
   - Apply the latest software updates and fixes.
   - Install the latest versions of OpenSSL & OpenSSH from the AIX Expansion Pack. This software can also be downloaded from the following sites.
     OpenSSH:
     http://sourceforge.net/projects/openssh-aix

     OpenSSL:
     https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=aixbp

   Since these are **installp** file sets you should run **/usr/sbin/updtvpkg** to make sure that the RPM reflection of what was installed by **installp** is updated. This makes it possible for RPM packages with a dependency on OpenSSL to recognize that the dependency is satisfied.

2. Download and install the prerequisite Open Source Software (OSS)

   - Download the *dep_aix.tar.gz* tar file from http://sourceforge.net/projects/xcat/ and copy it to a convenient location on your xCAT management node.
   - Unwrap the tar file.
     gunzip dep_aix.*tar.gz*
     tar –xvf dep_aix.*tar*

   - Read the README file.
   - Run the **instoss** script (contained in the tar file) to install the OSS packages.

3. Download and install the xCAT software.

   - Download the *core_aix.tar.gz* tar file from http://sourceforge.net/projects/xcat/ and copy it to a convenient location on your xCAT management node.
   - Unwrap the xCAT tar file.
     gunzip core_aix.tar.gz
     tar –xvf core_aix.tar

   - Run the **instxcat** script (contained in the tar file) to install the xCAT software. The post processing provided by the xCAT packages will perform some basic xCAT configuration. (This includes initializing the SQLite database and starting **xcatd** daemon processes.)

- Execute the system profile file to set the xCAT paths. This file was updated during the xCAT post install processing.
  . */etc/profile*

4. Verify the xCAT configuration.
   - Run the "*lsdef –h*" to check if the xCAT daemon is working. If you get a correct response then you should be ok.

   - Check to see if the initial xCAT definitions have been created. For example, you can run "*lsdef -t site -l*" to get a listing of the default site definition. You should see output similar to the following.

     Setting the name of the site definition to 'clustersite'.

     Object name: clustersite
     domain=abc.foo.com
     installdir=/install
     master=7.104.46.27
     rsh=/usr/bin/rsh
     rcp=/usr/bin/rcp
     xcatdport=3001
     xcatiport=3002

1.3 Syslog setup

The installation of xCAT will automatically setup syslog.conf with the following entries to log errors to the /var/log/localmessages file. Severe errors from commands and errors from the **xcatd** daemon will be logged in syslog. You can tailor the configuration but be sure to include the facility "local4" which is used by xCAT.

/etc/syslog.conf

*.debug /var/log/localmessages

*.crit /var/log/localmessages

1.4 xCAT 2.0 Commands Overview

The xCAT commands that are supported in this release are described in the following table. For more information about these commands you can run the command with the "-h" option (Ex. *xdsh –h*) to get a usage message. You can also view the **man** page for the command.

xCAT Command	Description
chtab	To add or update rows in a table. Allows you to add nodes, create groups, and add attributes to the xCAT tables.
chdef	Change xCAT data object definitions.
chvm	Changes HMC- and IVM-managed partition profiles.
getmacs	Collect node MAC addresses.
lsdef	Use this command to list xCAT data object definitions.
lsslp	Queries selected networked services information within the same subnet.
lsvm	Lists partition profile information for HMC- and IVM-managed nodes.
mkdef	Use this command to create xCAT data object definitions.
mkvm	Creates HMC- and IVM-managed partitions.
psh	Runs a command across a list of nodes or nodegroups in parallel
regnotif	Register a Perl module or a command that will get called when changes occur in desired xCAT database table.
rinv	Retrieves hardware configuration information for a single or range of nodes
rmdef	Use this command to remove xCAT data object definitions.
rmvm	Removes HMC- and IVM-managed partitions.
rnetboot	Will force an unattended network install for a range of nodes.
rpower	Boots, resets, powers on and off and queries nodes
rscan	Collects node information from hardware control point.
rvitals	Retrieves hardware vital information for a single or range of nodes.
startmon	Starts a monitoring plug-in module to monitor the xCAT cluster.
stopmon	Stops a monitoring plug-in module monitoring the xCAT cluster.
tabdump	Display Database table information for table requested. tabdump with no input will display a list of all valid table names.
tabedit	Edit a table. Must export EDITOR to define your editor.
tabrestore	Restore a table from the table.csv template or from a tabdump output file.
unregnotif	Unregistered a Perl module or a command that was watching for changes of desired xCAT database tables.
xdcp	Concurrently copies files to/from multiple nodes.
xdsh	Runs remote commands on multiple nodes.
xdshbak	Presents formatted output from the xdsh command.

1.5 Using xCAT database table commands.

The xCAT data that is used to manage a cluster is contained in a relational database. Different types of data are stored in different tables. You can manage this information directly using a set of table oriented commands provided by xCAT.

The following table describes the set of database tables that are currently defined by xCAT. Note that some of these tables are used to manage data that does not apply to AIX systems and some tables have been created for use by future xCAT support.

For more information on the xCAT database schema you can refer to the $XCATROOT/lib/perl5/site_perl/5.8.3/xCAT/Schema.pm file. (XCATROOT is an environment variable that is set during xCAT configuration. The default is "/opt/xcat".)

Table Name	Description
chain	Controls what operations are done (and in what order) when a node is discovered and deployed.
hosts	List of hosts, alias hostname, ip addresses. Used to update /etc/hosts.
ipmi	Lists information on the nodes IPMI interface – bmc, username, password
iscsi	Contains settings that control how to boot a node from an iSCSI disk.
mac	Lists a MAC address for each node.
monitoring	Lists the monitoring plug-in module names that are monitoring the xCAT cluster.
mp	This is the management processor network. Whereas the mpa.tab lists the adapter, this table lists devices that are networked off that adapter via daisy chained networks, or in the case of Blade Center, an internal network.
mpa	Lists the MPA, username and password for the nodes.
networks	Describes the networks in the cluster and info necessary to set up nodes on that network.
nodegroup	Contains information about xCAT node groups.
nodehm	Defines the hardware management method for each node.
nodelist	Defines all nodes and groups membership.
noderes	Installation resources for the node.
nodepos	Node physical location.
nodetype	Node install type information.
notification	Lists the Perl modules and commands that will get called for changes in certain xCAT database tables.
osimage	Contains information that describes a unique operating system image that may be deployed on a cluster node.
passwd	User names and passwords used by xCAT scripts.
policy	Table controls the policy for the execution of the xcat commands.
postscripts	The scripts that should be run on each node after installation or diskless boot
ppc	Contains HMC- and IVM-managed hardware component information – BPA, FSP, LPAR.
ppcdirect	Contains direct-attached FSP hardware information.
ppchcp	Contains HMC and IVM hardware information.
site	xCat cluster configuration information. (Holds global information for the cluster.)
switch	Lists switch interface(s) for the node.
vpd	Vital product data table. Holds machine serial number and model type.

To manage these tables directly, xCAT provides the **chtab, tabdump, tabrestore**, and **tabedit** commands.

The following are some basic examples of how to use the database table commands.

1. To edit the nodelist table using the **vi** editor.

export EDITOR=vi
tabedit nodelist

2. To see what tables exist in the xCAT database:

   tabdump

3. To display the contents of the site table:

   tabdump site

4. To back up all the xCAT tables.

   mkdir -p /tmp/xcatdb.backup

   for i in `tabdump`;do echo "Dumping $i..."; tabdump $i >/tmp/xcatdb.backup/$i.csv; done

5. To restore database tables that were dumped with tabdump:

   cd /tmp/xcatdb.backup

   *for i in *.csv;do echo "Restoring $i..."; tabrestore $i; done*

6. To add a new node “devnode01” to the “nodelist” table and assign it to the “all” and “compute” groups.

   chtab node=devnode01 nodelist.group=all,compute

7. Assign the “site” table “rsh” attribute to “/usr/bin/ssh”.

   chtab key=rsh site.value=/usr/bin/ssh

8. Delete the previously created node from the “nodelist” table.

   chtab –d node=devnode01 nodelist

1.6 Using xCAT object definition commands

In addition to managing the database tables directly, xCAT also supports the concept of data object definitions. Data objects are abstractions of the data that is stored in the xCAT database. This support provides a conceptually simpler implementation for managing cluster data, (especially data associated with a specific cluster node). It is also more consistent with other IBM systems management products such as Director, CSM, and AIX/NIM etc. The attributes and values defined in the data object definitions will still be stored in the database tables defined for xCAT 2.0. These data object definitions should not limit experienced xCAT customers from managing the specific tables directly, if they so desire. A new set of commands is provided to support the object definitions. These commands will automatically handle the storage in and retrieval from the correct tables.

The following data object types are currently supported.

- **site** - Cluster-wide information. All the data is stored in the *site* table.
- **node** - Information for a specific cluster node. The data for a node is stored in multiple tables in the database. The commands that are provided to manage these definitions automatically figure out which attributes are stored

in which table. It is therefore not necessary to keep track of a large number of table names and attribute locations.

- **network** - A description of a unique network. This data is stored in the *networks* table.
- **monitoring** - A description of a monitoring plugin. This data is stored in the *monitoring* table.
- **notification** - Defines the Perl modules and commands that will get called for changes in certain xCAT database tables. The data is stored in the *notification* table.
- **group** – Defines a set of nodes. A group definition can be used as the target set of nodes for a specific xCAT operation. It can also be used to define node attributes that are applied to all group members. The group data is stored in multiple tables in the database.

There are four basic xCAT commands that may be used to manage any of the data object definitions.

- **mkdef** – Make data object definitions.
- **chdef** - Change data object definitions.
- **lsdef** - List data object definitions.
- **rmdef** - Remove data object definitions.

The following are some examples of how to use the database object definition commands. For more information on using these commands refer to the **man** pages.

1) To view the list of supported object definition types you can issue any of the commands with the "-h" option. Along with the usage you will also see a list of supported object types.
   lsdef –h

2) To get a description of the attributes that can be defined for each object type you can issue the **lsdef** command with the "-t <object type>" option.
   lsdef –h –t node

3) To get a list of all the objects currently defined.
   lsdef –a

4) To get the details of a specific node definition.
   lsdef -t node –l –o node01

5) To create a very simple node definition.
   mkdef –t node –o node02 groups="all,aix"

6) To create a node group containing all nodes that have a "nodetype" attribute set to "osi".
   mkdef -t group -o osinodes -w nodetype= osi

7) To change the site definition.
   chdef -t site -o clustersite rsh=/bin/rsh rcp=/bin/rcp installdir=/xcatinstall

8) To remove all node and group definitions.
   rmdef –t node,group

9) To remove the group called hmcnodes.
   rmdef -t group -o hmcnodes

In addition to the standard command line input and output the **mkdef**, **chdef**, and **lsdef** commands support the use of a stanza file format for the input and output of information. Input to a command can be read from a stanza file and the output of a command can be written to a stanza file. A stanza file contains one or more stanzas that provide information for individual object definitions. For example:

1. To create a set of definitions using information contained in a stanza file.
   cat mystanzafile | mkdef -z

2. To write all node definitions to a stanza file.
   lsdef –t node -l -z > nodestanzafile

The stanza file support also provides an easy way to backup and restore the cluster data.

For more information on the use of stanza files see the **xcatstanzafile man** page.

1.7 Using xCAT hardware commands.

1.7.1 Hardware discovery

The following commands can be used to gather information about cluster hardware. See the **man** pages for additional details.

1. **rinv** - Retrieves hardware configuration information for a single or range of nodes and groups.

   For example:
   rinv node5 all

   node5: Machine Type/Model: 9117-MMA
   node5: Serial Number: 10F6F3D
   node5:: I/O Bus Information
   node5: U789D.001.DQDMLNV-P1-T3:512:RAID Controller
   node5: U789D.001.DQDMLNV-P1-T1:512:Universal Serial Bus UHC Spec

node5: U789D.001.DQDMLNV-P1-C7:512:Empty slot
node5: U789D.001.DQDMLNV-P1-C4:513:PCI 10/100/1000Mbps Ethernet UTP 2-port
node5: U789D.001.DQDMLNV-P1-C5:514:Empty slot
node5: U789D.001.DQDMLNV-P1-C1:516:Empty slot
node5: U789D.001.DQDMLNV-P1-C2:517:Empty slot
node5: U789D.001.DQDMLNV-P1-C3:518:Empty slot
node5: U789D.001.DQDMLNV-P1-C6:519:Empty slot
node5: Machine Configuration Info
node5: Number of Processors: 1
node5: Total Memory (MB): 2048

2. **rvitals** - Retrieves hardware vital information for a single or range of nodes and groups.

   For example:

   rvitals node5 all

   node5: Frame Voltage (Vab): 201V
   node5: Frame Voltage (Vbc): 203V
   node5: Frame Voltage (Vca): 202V
   node5: Frame Current (Ia): 19A
   node5: Frame Current (Ib): 19A
   node5: Frame Current (Ic): 20A
   node5: System Temperature: 33 C (91.4 F)
   node5: Running

3. **lsslp** - Queries selected networked services information within the same subnet. If the HMC/IVM that you are interested in discovering is on the same subnet as your Management Node, you can run the **lsslp** to discover and add this hardware to the xCAT database.

   Note that the dependent programs **slp_query** and **libslp_client.so** are compiled modules required to perform SLP broadcasts. These modules can be obtained by posting a request to the xCAT mailing list (please specify the target O/S in the request).

   For example:

   lsslp -s HMC

   device type-model serial-number ip-addresses hostname

HMC 7310CR2 103F55A 1.1.1.115 2.2.2.164 3.3.3.102 hmc01
HMC 7310CR2 105369A 3.3.3.103 2.2.2.103 1.1.1.163 hmc02
HMC 7310CR3 KPHHK24 3.3.3.154 2.2.2.110 1.1.1.154 hmc03

4. **rscan** - Collects node information from one or more hardware control points.

   For example:

   rscan hmc01

   type name id type-model serial- number address

   hmc hmc01 7310-C05 10F426A hmc01
   fsp Server-9117-MMA-SN10F6F3D 9117-MMA 10F6F3D 3.3.3.197
   lpar lpar3 4 9117-MMA 10F6F3D
   lpar lpar2 3 9117-MMA 10F6F3D
   lpar lpar1 2 9117-MMA 10F6F3D
   lpar p6vios 1 9117-MMA 10F6F3D

5. **getmacs** – Gathers adapter MAC information from cluster nodes.
   For example:

   getmacs node01

   node01:

   #Type Location Code MAC Address Full Path Name Ping Result

   ent U9133.55A.10B7D1G-V12-C4-T1 8ee2245cf004 /vdevice/l-lan@30000004 virtual

1.7.2 Hardware control

The following commands can be used to control cluster hardware. See the **man** pages for additional details.

1. **rnetboot** – Initiate a network boot request on one or more cluster nodes.
   For example, to network boot a node you might run:

rnetboot node01 -m 42DAB0003003 -S 8.114.47.87 -G 8.114.47.126 -C 8.114.47.88

2. **rpower** – Boots, resets, powers on and off, and queries node hardware, and devices.
   For example, to power on a node, enter:
   rpower -n clsn04 on

1.7.3 Configuring LPARs

The following commands can be used to create, change, list, and remove HMC and IVM managed partitions. See the MAN pages for additional details.

1. **mkvm** - Creates HMC and IVM managed partitions.

   Example.
   To create a new partition named "lpar5" with a numeric id of 5 that is based on the profile/resources of lpar4, enter:
   mkvm lpar4 -i 5 -n lpar5

2. **chvm** - Changes HMC and IVM managed partition profiles.

   Example.
   To change the partition profile for lpar4 using the configuration data in the file /tmp/lparfile, enter:
   cat /tmp/lparfile | chvm lpar4

3. **lsvm** - Lists partition profile information for HMC and IVM managed nodes.

   Example.
   To list all partition profiles defined for an HMC managed partition named "lpar3", enter:
   lsvm lpar3

4. **rmvm** - Removes HMC and IVM managed partitions.

   Example.
   To remove an LPAR with the name "lpar3".
   rmvm lpar3

1.8 Deploying AIX nodes using xCAT.

This is an example of how an AIX cluster node can be deployed using xCAT and AIX/NIM commands. There are also many options that can be used with NIM

other then what is specifically called out in this example. NIM also provides support for other interfaces such as SMIT (System Management Interface Tool). See the AIX/NIM documentation for details.

1. Assumptions
   - An AIX system has been installed to use as an xCAT management node.
   - xCAT and prerequisite software has been installed on the management node.
   - LPARs have already been created using the HMC interfaces.
   - You have some experience using AIX Network Installation Management (NIM).

2. Install & configure NIM

   NIM enables a cluster administrator to centrally manage the installation and configuration of AIX and optional software on machines within a networked environment. Setting up NIM involves various tasks including, installing NIM software, configuring NIM and creating some basic NIM installation resources.

   The specific tasks that you need to perform depend on which features of NIM that you plan to use. For more information about NIM, see the *IBM AIX Installation Guide and Reference*.
   (http://www-03.ibm.com/servers/aix/library/index.html)

   To simplify the NIM setup process, you can use the AIX **nim_master_setup** command. This command automatically installs NIM software, configures NIM, creates basic resources and creates a resource group containing the resources that are created. This command must be run on the machine that will be used as the NIM master. See the **man** page for details.

   When you run the command it will display a list of the resources that were created. If the default resources that are created by this script are not what you need, you can use individual NIM commands to create additional resources. The resources created are stored in the */export/nim* directory by default. The file system is automatically created with the correct size needed to store the required resources.

   In this example we will be doing a NIM "rte" type installation so we will disable the creation of the backup image (*mksysb*) by using the "-B" option. The default location of the NIM master install images is /dev/cd0. If you are using another device or directory as a source you would have to specify that on the command line. Assuming the images are available from the default device you could run the command as follows.

nim_master_setup -B

To make sure that NIM is set up correctly issue the **lsnim** command on the NIM master as follows.

lsnim

You should see output similar to the following:

```
master              machines    master
boot                resources   boot
nim_script          resources   nim_script
master_net          networks    ent
master_net_conf     resources   resolv_conf
bid_ow              resources   bosinst_data
530lpp_res          resources   lpp_source
530spot_res         resources   spot
basic_res_grp       groups      res_group
```

To get details for each resource definition use "**lsnim -l <resource name>**". For example, if the name of your SPOT resource is "*530spot_res*" then you could get the details by running:

lsnim -l 530spot_res

To see the actual contents of a resource use "**nim -o showres <resource name>**". For example, to show the file sets that are installed in your SPOT you could run:

nim -o showres 530spot_res | pg

3. Create additional NIM definitions and resources if needed.

   For this example we are assuming that the xCAT management node and the LPARs are all connected to the same network and that the resources created by the **nim_master_setup** command are sufficient.

   However, depending on your specific situation, you may need to create additional NIM resources or definitions.

   For example, if the management node is not on the same network as the nodes you would have to create additional NIM network and route definitions. NIM network definitions represent the networks used in the NIM environment. When you run **nim_master_setup**, the network associated with the NIM master is automatically defined. You need to define additional networks only if there are nodes that reside on other local area networks or

subnets. If the physical network is changed in any way, the NIM network definitions need to be modified. See the NIM documentation for details on creating additional network and route definitions.

Note: The NIM machine definitions will be generated automatically from the xCAT node definitions in a later step.

4. Define the HMC as a cluster node.

   The following command will create an xCAT node definition for an HMC with a hostname of "*hmc01*". The HMC must be defined as a node and the *groups, nodetype, mgt, username*, and *password* attributes must be set.

   mkdef –t node –o hmc01 groups="all" nodetype=hmc mgt=hmc username=hscroot password=abc123

5. Discover the LPARs managed by an HMC.

   This step assumes that the LPARs are already created using the standard HMC interfaces.

   Use the **rscan** command to gather the LPAR information. This command can be used to display the LPAR information in several formats and can also write the LPAR information directly to the xCAT database. In this example we will use the "-z" option to create a stanza file that contains the information gathered by **rscan** as well as some default values that could be used for the node definitions.

   To write the stanza format output of **rscan** to a file called "*mystanzafile"* run the following command.

   rscan –z hmc01 > mystanzafile

   This file can then be checked and modified as needed. For example you may need to add a different name for the node definition or add additional attributes and values etc.

   Note: The stanza file will contain stanzas for things other than the LPARs. This information must also be defined in the xCAT database. It is not necessary to modify the non-LPAR stanzas in any way.

   The updated stanza file might look something like the following.

   ```
   Server-9117-MMA-SN10F6F3D:
       objtype=node
       nodetype=fsp
       id=5
       model=9118-575
       serial=02013EB
       hcp=hmc01
       profile=
       parent=Server-9458-10099201WM_A
       groups=fsp,all
   ```

```
    mgt=hmc

node01:
    objtype=node
    nodetype=lpar,osi
    id=9
    hcp=hmc01
    profile=lpar9
    parent=Server-9117-MMA-SN10F6F3D
    groups=all
    mgt=hmc

node02:
    objtype=node
    nodetype=lpar,osi
    id=7
    hcp=hmc01
    profile=lpar6
    parent=Server-9117-MMA-SN10F6F3D
    groups=all
    mgt=hmc
```

Note: The **rscan** command supports an option to automatically create node definitions in the xCAT database. To do this the LPAR name gathered by **rscan** is used as the node name and the command sets several default values. If you use the "-w" option make sure the LPAR name you defined will be the name you want used as your node name.

6. Define xCAT nodes

   The information gathered by the **rscan** command can be used to create xCAT node definitions.

   Since we have put all the node information in a stanza file we can now pass the contents of the file to the **mkdef** command to add the definitions to the database.

   cat mystanzafile | mkdef –z

   You can use the xCAT **lsdef** command to check the definitions (ex. "*lsdef –l node01*"). After the node has been defined you can use the **chdef** command to make any additional updates to the definitions, if needed.

7. Define xCAT groups (optional).

   You can see in the stanza file sample above that we have set the node "groups" attribute to "all". This means that you have also created a group definition named "all" that now contains the nodes we just defined.

You may also want to create additional groups. There are several ways to do this. One option would be to modify the node definitions to add another group name to the "groups" attribute value. You could do this using the **chdef** command. For example:

chdef –t node –o node02 groups=all,aixnodes

Another option would be to create a new group definition directly using the **mkdef** command as follows.

mkdef –t group –o aixnodes members="node01,node02,node03"

8. Gather MAC information for the install adapters.

   Use the xCAT **getmacs** command to gather adapter information from the nodes. This command will return the MAC information for each Ethernet adapter available on the target node. If there is more than one you will have to choose the one you wish to use as the interface for the network install.

   For example, to get the MAC address for node "*node02*" issue the following command.

   getmacs node02

   You will have to add the MAC information to the xCAT node definition before creating the NIM definitions in the next step. The MAC address can be added to the node definition by using the **chdef** command. For example:

   chdef –t node –o node01 mac=42DAB0003003

   Note: The **getmacs** command will be enhanced to write the MAC information directly to the database in a future xCAT update.

9. Create NIM client & group definitions

   You can use the xCAT **xcat2nim** command to automatically create NIM machine and group definitions based on the information contained in the xCAT node and group definitions. By doing this you synchronize the NIM and xCAT names so that you can use the same target names when running either an xCAT or NIM command.

   To create NIM machine definitions you could run the following command.

   xcat2nim -t node –o aixnodes

To create NIM group definitions you could run the following command.

xcat2nim –t group –o aixnodes

To check the NIM definitions you could use the NIM **lsnim** command or the xCAT **xcat2nim** command. For example, the following command will display the NIM definitions of the nodes: *node01*, *node02*, and *node03* (from data stored in the NIM database).

xcat2nim -t node -l –o node01-node03

10. Set up NIM for a network boot.

In this example we are assuming that we want to do a NIM "*rte*" type install and that we will be using the resources contained in the resource group "*basic_res_grp*" (created by **nim_master_setup**). To set up NIM you must run the AIX "**nim bos_inst ..**" command on the NIM master.
For example:

nim -o bos_inst -a source=rte -a boot_client=no -a group=basic_res_grp aixnodes

To verify that you have allocated all the NIM resources that you need you can run the "**lsnim –l**" command. For example, to check node "node01" you could run the following command.

lsnim -l node01

Because this is the initial installation of the node, NIM is not able to initiate the network boot. This is why the "boot_client" attribute is set to "no". In this case you need to either initiate the network boot manually on the client machine, or use the xCAT **rnetboot** command as described below.

11. Initiate a network boot

Initiate a remote network boot request using the xCAT **rnetboot** command. For example, to initiate a network boot of node "*node01"* using the MAC, subnet mask, gateway, and node IP address values you could issue the following command.

rnetboot node01 -m 42DAB0003003 -S 8.114.47.87 -G 8.114.47.126 -C 8.114.47.88

Note: In a future xCAT update the **rnetboot** command will be enhanced to get the required information from the xCAT database.

12. Verify the deployment
    - As soon as the **rnetboot** command returns you can open a remote console to monitor the boot progress. The xCAT **rconsole** command will be available soon. In the meantime you can open a console using the following procedure.
        1. Export your DISPLAY. For example.

           export DISPLAY=8.56.120.154:0

        2. Run the **xterm** command. For example.

           xterm -sb -sl 256 -geometry 80x25+0+0 -fg white -bg black -cr white -e ksh -c "/opt/xcat/share/xcat/cons/hmc node01" &

      If you have an IVM LPAR, you would run "/opt/xcat/share/xcat/cons/ivm".

      To exit the console session type "~." in the console window.
    - You can use the AIX **lsnim** command to see the state of the NIM installation for a particular node, by running the following command on the NIM master:

      lsnim -l <clientname>
    - You can view one of several NIM logs by using the "log_type" attribute with the **nim** command and the "showlog" operation. Some of the valid "log_type" values are:
        - niminst
        - bosinst
        - boot
        - script

      For example:

      nim -o showlog -a log_type=niminst node01

      For a complete list of these logs and their corresponding "log_type" attribute values, see *IBM® Network Installation Management Guide and Reference*.

1.9 Remote shell setup

If you wish to use the **xdsh, xdcp,** or **xdshbak** commands you will have to configure either **ssh** or **rsh**. To use **psh** you will have to set up **ssh**.

1.9.1 Setting up OpenSSH

- Set the site table.

Set the *rsh* and *rcp* attributes of the cluster site definition as follows.

chdef -t site rsh=/usr/bin/ssh rcp=/usr/bin/scp

- Install OpenSSH & OpenSSL software.

  Because **OpenSSH** and its prerequisite **OpenSSL** are not installed with base AIX, you will have to install this additional software on the xCAT management node and the AIX cluster nodes.

  The latest versions of the software are available from the AIX Expansion Pack. The software is also available to download from the following sites.

  OpenSSL
  https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=aixbp

  OpenSSH
  http://sourceforge.net/projects/openssh-aix

  This software could be installed on the nodes during the initial node installation using NIM or you could just copy the packages to the node and use the standard **installp** command. See the AIX NIM documentation for details.

- Run **xdsh** to set up **ssh** on the cluster node.

  Run the **xdsh** command with the –K option to complete the **ssh** setup on the nodes.

  xdsh node01 -K

1.9.2 Setting up rsh

The default setting in the site table for AIX is **rsh**. (rcp=/usr/bin/rcp, rsh=/usr/bin/rsh). The only extra step in this case is to complete the **rsh** setup on the nodes. (Note that in a future update of xCAT for AIX this will be handled automatically.)

To setup **rsh** on the nodes for the "root" user:

- Log on to the node.
- Create (or update) a "/.rhosts" file with permissions set to 0600.
- Add the IP address of the xCAT management node followed by the string "root". (ex. "7.114.107.125 root")

1.10 xCAT Notification Infrastructure

With xCAT 2.0, you can monitor xCAT database for changes such as nodes entering/leaving the cluster, hardware updates, node liveness (to be added later) etc. In fact anything stored in the xCAT database tables can be monitored through the

xCAT notification infrastructure. To start getting notified for changes, simply register your Perl module or command as the following:

regnotif *filename tablename -o actions*

where

filename is the full path name of your Perl module or command.

tablenames is a comma separated list of table names that you are interested in.

actions is a comma separated list of data table actions. 'a' for row addition, 'd' for row deletion and 'u' for row update.

Example**:**

regnotif /opt/xcat/lib/perl/xCAT_monitoring/mycode.pm nodelist,nodhm -o a,d

regnotif /usr/bin/mycmd switch,noderes -o u

Use the following command to view all the modules and commands registered.

tabdump notification

To un-register, just do the following:

unregnotif *filename*

Example:

unregnotif /opt/xcat/lib/perl/xCAT_monitoring/mycode.pm

unregnotif /usr/bin/mycmd

If the *filename* specifies a Perl module, the package name must be **xCAT_monitoring::xxx**. It must implement the following subroutine which will get called when database table change occurs:

processTableChanges*(tableop, table_name, old_data, new_data)*

where:

tableop Table operation. It can be 'a' for row addition, 'd' for row deletion and 'u' for row update.

tablename The name of the database table whose data has been changed.

old_data An array reference of the old row data that has been changed. The first element is an array reference that contains the column names. The rest of the elements are array references each contains attribute values of a row. It is set when the action is u or d.

new_data A hash reference of the new row data; only changed values are in the hash. It is keyed by column names. It is set when the action is u or a.

If the file name specifies a command (written by any programming languages or scripts), when the interested database table changes, the info will be fed to the

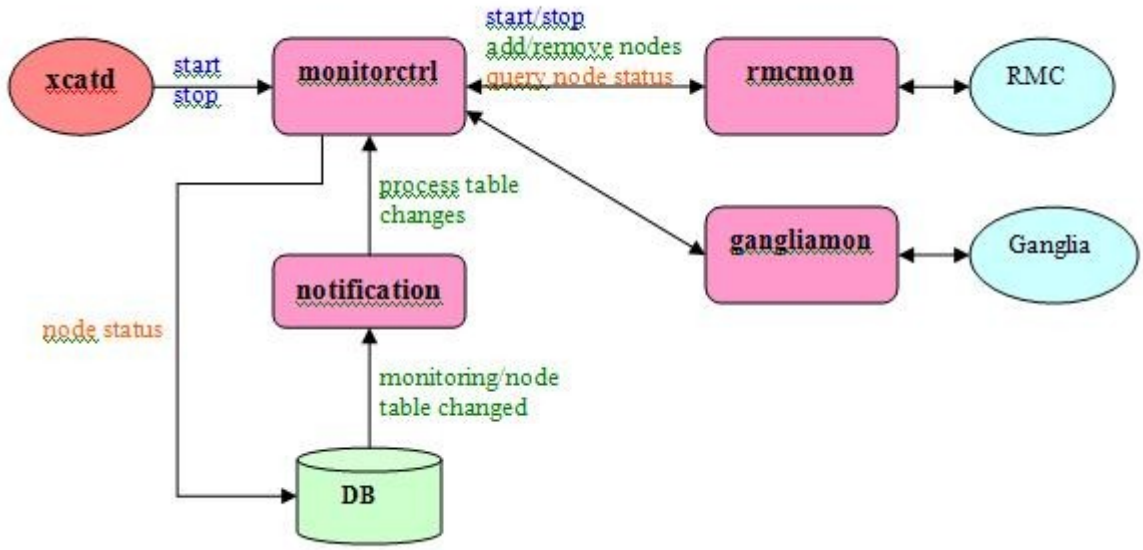
command through the standard input. The format of the data in the STDIN is as following:

```
action(a, u or d)
tablename
[old value]
col1_name,col2_name...
col1_val,col2_val,...
col1_val,col2_val,....
...
[new value]
col1_name,col2_name,...
col1_value,col2_value,...
...
```

The sample code can be found under **/opt/xcat/lib/perl/xCAT_monitoring/samples/mycode.pm** on a installed system.

1.11 xCAT Monitoring Plug-in Infrastructure

With xCAT 2.0, you can integrate 3rd party monitoring software into your xCAT cluster. The idea is to use monitoring plug-in modules that act as bridges to connect xCAT and the 3rd party software. The functions of a monitoring plug-in module include initializing the 3rd party software, informing it with the changes of the xCAT node list, setting it up to feed node status back to xCAT etc. The following figure depicts the relationship and data flow among xcatd, plug-in modules and 3rd party monitoring software.



To use this infrastructure, first create a monitoring plug-in module and put it under **/opt/xcat/lib/perl/xCAT_monitoring/** directory. If the file name is xxx.pm then the package name will be **xCAT_monitoring::xxx**. The following is a list of subroutines that a plug-in module must implement:

start

stop

supportNodeStatusMon

startNodeStatusMon

stopNodeStatusMon

addNodes

removeNodes

Please refer to */opt/xcat/lib/perl/xCAT_monitoring/samples/tmplatemon.pm* for the detailed description of the functions.

Second, register the module in xCAT ***monitoring*** table using the following command:

startmon ***name [-n|--nodestatmon] [-s|--settings tag=value,tag=value…]***

where

name is the monitoring plug-in module short file name without the extension. In this case xxx.

-n or --nodestatmon indicates it can help feeding the node status to xCAT. The node status is stored in the ***status*** column of the ***nodelist*** table.

-s or –settings specifies the plug-in specific settings. These setting will be used by the plug-in to customize certain entities for the plug-in or the third party monitoring software. e.g. mon_interval=10,toggle=1

Example**:**

startmon *xxx –n* (with feeding the node status to xCAT table)

or

startmon *xxx* (not feeding the node status to xCAT table)

Once it is registered, xCAT will automatically, through the plug-in module, start the 3rd party software for monitoring. To unregister the monitoring plug-in and stop the monitoring use this command:

stopmon *name*

Example**:**

stopmon *xxx*

Though you can write your own monitoring plug-in modules, over the time, xCAT will supply a list of built-in plug-in modules for the most common monitoring software. They are:

- xCAT (xcatmon.pm) (released in this beta)
- RMC (rmcmon.pm)
- Ganglia (gangliamon.pm)
- Nagios (nagiosmon.pm)
- SNMP (snmpmon.pm)

xcatmon.pm is included in this release. It provides node liveness monitoring using fping. This can be used if no other 3rd party software is used for node status monitoring. The ***status*** column of the ***nodelist*** table will be updated periodically with the latest node liveness status by this plug-in. To activate, use the startmon command:

startmon xcatmon –n –s ping-interval=2

where 2 means that the nodes are pinged for status every 2 minutes**.**