xCAT 2 on AIX
Updating AIX cluster nodes
FIRST DRAFT

Version: 1.0

Date: 3/27/2009

1.0 Overview

There are various techniques that can be used to update the nodes of an xCAT cluster. This document describes some of the basic support that is provided for AIX nodes.

2.0 Updating the xCAT cluster

2.1 Updating diskfull nodes

2.1.1 Using the nimnodecust command

The xCAT **nimnodecust** command can be used to customize AIX/NIM standalone machines. This command uses underlying NIM support to perform the remote customization of AIX diskfull (standalone) nodes.

The software packages and/or updates that you wish to install on the nodes must be copied to the appropriate directory locations in the NIM lpp_source resource that you are using for the nodes you wish to update.

The easiest way to do this is to use the “nim -o update” command. For example, assume all the required software has been copied and unwrapped in the /tmp/images directory.

To add all the packages to the lpp_source resource named "*610SNimage_lpp_source*" you could run the following command:

```
nim -o update -a packages=all -a source=/tmp/images  610SNimage_lpp_source
```

The NIM command will find the correct directories and update the lpp_source resource.

When using the **nimnodecust** command the packages you wish to install on the nodes may be specified with either a comma-separated list of package names or by a comma-separated list of installp_bundle names.

For example, To install the installp package "openssh.base.server" on an xCAT node named "node01" assuming the software has been copied to the "610SNimage_lpp_source" lpp_source resource you could run the following command.

```
nimnodecust -s  610SNimage_lpp_source -p openssh.base.server node01
```

For more details on using the **nimnodecust** command see the corresponding man page.

2.1.2 Using the xdsh method

Another method for updating a diskfull node would be to mount a directory containing the updates on the node and use the **xdsh** command to run the appropriate **installp** or **rpm** command.

- should put updates in lpp_source so they will be available for next install!

- example - TBD

2.2 Using the rolling update support

The **rollupdate** command creates and submits scheduler jobs that will notify xCAT to shutdown a group of nodes, run optional out-of-band commands from the xCAT management node, and reboot the nodes. Currently, only LoadLeveler is supported as a job scheduler with **rollupdate**.

Input to the **rollupdate** command is passed in as stanza data through STDIN. Information such as the sets of nodes that will be updated, the name of the job scheduler, a template for generating job command files, and other control data are required. See /opt/xcat/share/xcat/rollupdate/rollupdate.input.sample for stanza keywords, usage, and examples.

The **rollupdate** command will use the input data to determine each set of nodes that will be managed together as an update group. For each update group, a job scheduler command file is created and submitted. When the group of nodes becomes available and the scheduler runs the job, the job will send a message to the xCAT daemon on the management node to begin the update process for all the nodes in the update group. The nodes will be stopped by the job scheduler (for LoadLeveler, the nodes are drained), an operating system shutdown command will be sent to each node, out-of-band operations can be run on the management node, and the nodes are powered back on.

The **rollupdate** command assumes that, if the update is to include rebooting stateless nodes to a new operating system image, the image has been created and tested, and that all relevant xCAT commands have been run for the nodes such that the new image will be loaded when xCAT reboots the nodes.

See the **rollupdate** man page for usage details.

2.3 Updating diskless nodes

Updating a diskless node with fixes or additional software involves updating the SPOT that is being used to boot the node.

There are two basic processes you can use to update a SPOT:

1. Install additional **installp** file sets or **rpm** packages.
2. Add or modify specific files, (such as /etc/inittab).

Note: You should not attempt to update a SPOT resource that is currently allocated to a node. If you need to update an allocated SPOT either you can shut down the nodes and deallocate the SPOT resource first or you can make a copy of the SPOT and update that.

2.3.1 Install additonal software

You can use the AIX **chcosi** command to install both **installp** file sets and **rpm** packages in a SPOT resource.

> **Note**: If the SPOT you want to update is currently allocated the **chcosi** command will automatically attempt to create a copy of the SPOT to update. To check to see if the SPOT is allocated you could run the following command.
>
> *lsnim -l <spot name>*

Before running the **chcosi** command you must add the new filesets and/or RPMs to the *lpp_source* resource used to create the SPOT. If we assume the *lpp_source* location for *61cosi* is */install/nim/lpp_source/61cosi_lpp.* The **installp** packages would go in: */install/nim/lpp_source/61cosi_lpp/installp/ppc* and the RPM packages would go in: */install/nim/lpp_source/61cosi_lpp/RPMS/ppc.*

If you have lots of files to copy you might try using the "**nim -o update**" command. For example, if your new packages are in /tmp/myimages then you could run:

```
nim -o update -a packages=all -a source=/tmp/myimages 61cosi
```

The AIX **chcosi** command supports installing, updating, rejecting, removing, and committing the **installp** packages in the common image. See the **chcosi** man page details.

For example, to install and commit the optional OpenSSH packages from the AIX Expansion Pack you could issue the following command.

```
chcosi –i –c –s  61cosi_lpp –f  'openssh.base  openssh.license openssh.man.en_US
openssh.msg.en_US  openssh.msg.EN_US'  61cosi
```

Any additional software that is needed can be installed in a similar manner.

> **Note**: When installing software into a SPOT the pre and post install scripts for a particular software package will not run any code that will impact your running system, (like restarting daemons etc.). The script will check to see if it's installing into a SPOT and it will not run that code.

RPM packages may also be installed in the diskless image. You can use the **chcosi** command to install an RPM as follows.

```
chcosi –i –s  61cosi_lpp –f  'R:mypack..aix5.3.ppc.rpm'  61cosi
```

2.3.2 Add or modify files

You can also update files in the SPOT/COSI manually. The root file system for the diskless node will be created by copying the "*inst_root"* directory contained in the SPOT. In the SPOT we created for this example the "*inst_root*" directory would be:

```
/install/nim/spot/61cosi/usr/lpp/bos/inst_root/
```

For example, if you need to update the */etc/inittab* file that will be used on the diskless nodes you could edit:

```
/install/nim/spot/61cosi/usr/lpp/bos/inst_root/etc/inittab
```

You can also copy specific files into the *inst_root* directory so they will be available when the nodes boot. For example, you could copy a script called *myscript* to */install/nim/spot/61cosi/usr/lpp/bos/inst_root/opt/foo/myscript* and then add an entry to */etc/inittab* so that it would be run when the node boots.

All the diskless nodes that are booted using this SPOT will get a copy of *inst_root* as the initial root directory.

Note: There are several files that you may want to consider updating in the SPOT *inst_root* directory. For example:

- /etc/hosts
- /etc/resolv.conf
- /etc/password
- /etc/profile
- etc.

2.4 Using the updatenode command

You can use the **updatenode** command to re-run xCAT postscripts or to run additional user-provided scripts on the cluster nodes.

Examples:
(1) To re-run all the xCAT postscripts for the nodes:

```
updatenode <noderange>
```

(2) To re-run the syslog postscripts for the nodes:

```
updatenode <noderange> syslog
```

(3) To run a list of user-provided scripts, make sure the scripts are copied to /install/postscripts directory, and then run the following:

```
updatenode <noderange> script1,script2
```

If you wish to have scripts run during the next node installation you must add the script names to the "postscripts" attribute of the node definitions.

See the **updatenode** man page for additional information.

3.0 Getting software and firmware levels

3.1 Using the sinv command

The sinv command is designed to check the configuration of the nodes in a cluster. The command takes as input command line flags, and one or more templates which will be compared against the output of the xdsh command, designated to be run by the -c or -f flag, on the nodes in the noderange.

The nodes will then be grouped according to the template they match and a report returned to the administrator in the output file designated by the -o flag, or to stdout.

sinv supports checking the output from the rinv or xdsh command.

The sinv command is an xCAT Distributed Shell Utility.
See the man pages for sinv & rinv for more details.