xCAT Statelite Cookbook

11/08/10, 02:39:23 PM

Table of Contents

1 Introduction

This document details a design for native xCAT NFS Root. We call the design statelite, because in addition to having an NFS Root implementation, some files can be stored persistently and maintain state through reboots.

Statelite provides an efficient and flexible diskless solution, because most of the OS image is NFS mounted read-only, but it also provides a configurable list of directories and files that can be read-write. The read-write files can either be persistent across reboots, or volatile (restoring to original state after reboot).

Statelite offers the following **advantages** over xCAT's stateless (RAMdisk) implementation:

1. Some files can be made persistent over reboot. This is useful for license files or database servers where some state is needed. However, you still get the advantage of only having to manage a single image.

2. Changes to hundreds of machines can take place instantly, and automatically, by updating one main image. In most cases, machines do not need to reboot for these changes to take affect.

3. Ease of administration by being able to lock down an image. Many parts of the image can be read-only, so no modifications can transpire without updating the central image.

4. Files can be managed in a hierarchical manner. For example: Suppose you have a machine that is in one lab in Tokyo and another in London. You could set table values for those machines in the xCAT database to allow machines to sync from different places based on their attributes. This allows you to have one base image with multiple sources of file overlay.

5. Ideal for virtualization – In a virtual environment, you may not want a disk image (neither stateless nor stateful) on every virtual node as it consumes memory and disk. Virtualizing with the statelite approach allows for images to be

smaller, easier to manage, use less disk, less memory, and more flexible.

Disadvantages

1. NFS Root requires more network traffic to run as the majority of the disk image runs over NFS. This may depend on your workload, but can be minimized. Since the bulk of the image is read-only, NFS caching on the server helps minimize the disk access on the server, and NFS caching on the client helps reduce the network traffic.

2. NFS Root can be complex to set up. As more files are created in different places, there are greater chances for failures. This flexibility is also one of the great virtues of Statelite. The image can work in nearly any environment.

1.1 Limitations

- Currently, statelite has only been tested on Red Hat and Red Hat Clone environments on x86_64 and ppc64 platforms, and also SLES11 (SuSE Linux Enterprise Server 11) on POWER systems. There is no reason to think SLES11 on x86_64, and we are in the process of testing that.
- The statelite support on AIX is now introduced as one experimental feature.
- Only one statelite image may be defined for a node or noderange.

2 Configuring Statelite

Getting started with Statelite provisioning requires that you have xCAT set up and running. Before continuing with the rest of this document, the xCAT management node must be set up, and the nodes' hardware control, resource, and type attributes must be defined as described in the xCAT Top Doc.

Please report any errors to the xCAT mailing list (https://lists.sourceforge.net/lists/listinfo/xcat-user).

2.1 Make The Main Image Read-only (Optional)

Any statelite image will need some files to be read/writable for individual nodes. But we encourage you to make the main image read-only. The image will be placed in `/install/netboot/<os>/<arch>/<profile>/rootimg` after running `genimage`. See the Create Statelite Image.

xCAT, by default during installation, will create on the Management Node an /etc/exports entry that exports the /install directory read/writeable. Change it to read-only, so it looks as follows:

```
/install *(ro,no_root_squash,sync,no_subtree_check)
```

Restart the NFS server:
For Redhat and Redhat-Clone environment,

```
service nfs restart
```

For SLES,

```
service nfsserver restart
```

2.2 litefile table

The litefile table specifies the directories and files on the statelite nodes that should be read/write, persistent, or read-only overlay. All other files in the statelite nodes come from the read-only statelite image. All the available options are listed here.

1. **tmpfs** - It is the default option if you leave the options column blank. It provides a file or directory for the node to use when booting, its permission will be the same as the original version on the server. In most cases, it is read-write; however, on the next statelite boot, the original version of the file or directory on the server will be used, it means it is non-persistent. This option can be performed on files and directories.

2. **rw** - Same as Above.Its name "rw" does NOT mean it always be read-write, even in most cases it is read-write. Please do not confuse it with the "rw" permission in the file system.

3. **persistent** - It provides a mounted file or directory that is copied to the xCAT persistent location and then over-mounted on the local file or directory. Anything written to that file or directory is preserved. It means, if the file/directory does not exist at first, it will be copied to the persistent location. Next time the file/directory in the persistent location will be used. The file/directory will be persistent across reboots. Its permission will be the same as the original one in the statelite location. It requires the statelite table to be filled out with a spot for persistent statelite. This option can be performed on files and directories.

4. **con** - The contents of the pathname are concatenated to the contents of the existing file. For this directive the searching in the litetree hierarchy does not stop when the first match is found. All files found in the hierarchy will be concatenated to the file when found. The permission of the file will be "-rw-r--r--", which means it is read-write for the root user, but readonly for the others. It is non-persistent, when the node reboots, all changes to the file will be lost. It can only be performed on files. Please do not use it for one directory.

5. **ro** - The file/directory will be overmounted read-only on the local file/directory. It will be located in the directory hierarchy specified in the litetree table. Changes made to this file or directory on the server will be immediately seen in this file/directory on the node. This option requires that the file/directory to be mounted must be available in one of the entries in the litetree table. This option

can be performed on files and directories.

6. **link** - It provides one file/directory for the node to use when booting, it is copied from the server, and will be placed in tmpfs on the booted node. In the local file system of the booted node, it is one symbolic link to one file/directory in tmpfs. And the permission of the symbolic link is "lrwxrwxrwx", which is not the real permission of the file/directory on the node. So for some application sensitive to file permissions, it will be one issue to use "link" as its option, for example, "/root/.ssh/", which is used for SSH, should NOT use "link" as its option. It is non-persistent, when the node is rebooted, all changes to the file/directory will be lost. This option can be performed on files and directories.

7. **link,ro** - The file is readonly, and will be placed in tmpfs on the booted node. In the local file system of the booted node, it is one symbolic link to the tmpfs. It is non-persistent, when the node is rebooted, all changes to the file/directory will be lost. This option requires that the file/directory to be mounted must be available in one of the entries in the litetree table. The option can be performed on files and directories.

8. **link,con** - It works similar to the "con" option. All the files found in the litetree hierarchy will be concatenated to the file when found. The final file will be put to the tmpfs on the booted node. In the local file system of the booted node, it is one symbolic link to the file/directory in tmpfs. It is non-persistent, when the node is rebooted, all changes to the file will be lost. The option can only be performed on files.

9. **link,persistent** - It provides a mounted file or directory that is copied to the xCAT persistent location and then over-mounted to the tmpfs on the booted node, and finally the symbolic link in the local file system will be linked to the over-mounted tmpfs file/directory on the booted node. The file/directory will be persistent across reboots. The permission of the file/directory where the symbolic link points to will be the same as the original one in the statelite location. It requires the statelite table to be filled out with a spot for persistent statelite. The option can be performed on files and directories.

Currently, we can't handle the relative links very well. The relative links are commonly used by the system libraries, for example, under "/lib/" directory, there will be one relative link matching one .so file. So, when you add one relative link to the litefile table (We don't recommend), please make sure the real file also be included, or you can put its directory name into the litefile table. However, in common sense, most of the users will not put the relative links to the litefile table.

The setup for RedHat and SLES are below.
Note: It is recommended that you specify at least the entries listed below in the litefile table, because most of these files need to be writeable for the node to boot up successfully. When any changes are made to their options, please make sure they

won't affect the whole system.

2.2.1 Sample Data for a RedHat statelite setup.

This is the minimal list of files needed, you can add additional files to the litefile table.

Notice that at all files are in tmpfs, the default for the options field. This gives you an NFS root solution with no persistent storage.

```
#image,file,options,comments,disable
"ALL","/etc/adjtime",,,
"ALL","/etc/fstab",,,
"ALL","/etc/lvm/",,,
"ALL","/etc/syslog.conf",,,
"ALL","/etc/syslog.conf.XCATORIG",,,
"ALL","/etc/ntp.conf",,,
"ALL","/etc/ntp.conf.predhclient",,,
"ALL","/etc/resolv.conf",,,
"ALL","/etc/resolv.conf.predhclient",,,
"ALL","/etc/ssh/",,,
"ALL","/etc/sysconfig/",,,
"ALL","/tmp/",,,
"ALL","/var/",,,
"ALL","/opt/xcat/",,,
"ALL","/xcatpost/",,,
"ALL","/root/.ssh/",,,
```

2.2.2 Sample Data for Redhat6 statelite setup

This is the minimal list of files needed, you can add additional files to the litefile table.

```
#image,file,options,comments,disable
"ALL","/etc/adjtime",,,
"ALL","/etc/securetty",,,
"ALL","/etc/lvm/",,,
"ALL","/etc/ntp.conf",,,
"ALL","/etc/rsyslog.conf",,,
"ALL","/etc/rsyslog.conf.XCATORIG",,,
"ALL","/etc/udev/",,,
"ALL","/etc/ntp.conf.predhclient",,,
"ALL","/etc/resolv.conf",,,
"ALL","/etc/yp.conf",,,
"ALL","/etc/resolv.conf.predhclient",,,
"ALL","/etc/sysconfig/",,,
"ALL","/etc/ssh/",,,
"ALL","/tmp/",,,
"ALL","/var/",,,
"ALL","/opt/xcat/",,,
"ALL","/xcatpost/",,,
"ALL","/root/.ssh/",,,
```

2.2.3 Sample Data for SLES11 statelite setup

This is the minimal list of files needed, you can add additional files to the litefile

table.

```
#image,file,options,comments,disable
"ALL","/etc/lvm/",,,
"ALL","/etc/ntp.conf",,,
"ALL","/etc/ntp.conf.org",,,
"ALL","/etc/resolv.conf",,,
"ALL","/etc/ssh/",,,
"ALL","/etc/sysconfig/",,,
"ALL","/etc/syslog-ng/",,,
"ALL","/tmp/",,,
"ALL","/var/",,,
"ALL","/etc/yp.conf",,,
"ALL","/etc/fstab",,,
"ALL","/opt/xcat/",,,
"ALL","/xcatpost/",,,
"ALL","/root/.ssh/",,,
```

Use the command

```
tabedit litefile
```

to copy/paste the above sample lines into the litefile table.

2.2.3 "/etc/resolv.conf"

Currently, "/etc/resolv.conf" can't be set up correctly by default.
We have two solutions:
1) Edit the "/.default/etc/resolv.conf" file in the statelite image
Make sure you know the domain and nameserver for all the statelite nodes, then edit the file ".default/etc/resolv.conf" in the statelite image like this:

```
       search <DOMAIN>
       nameserver <NAMESERVER>
```

Replace <DOMAIN> with the domain for all the statelite nodes.
Replace <NAMESERVER> with the nameserver for all the statelite nodes.

Or you can:
2) Put one working "/etc/resolv.conf" file to one NFS directory, then, add the NFS directory to the "litetree" table. For example, create one directory named "/lite/tree/", and export it as one NFS directory on your MN; then, put one working "/etc/resolv.conf" into "/lite/tree". Finally, add one entry to the "litetree" table like this:

```
       #priority,image,directory,comments,disable
       1,,"<MN IP>:/lite/tree",,
```

Replace <MN IP> with your MN's IP address.

2.3 litetree table

The litetree table controls where the initial content of the files in the litefile table come from, and the long term content of the "ro" files. When a node boots up in statelite mode, it will by default copy all of its tmpfs files from the /.default directory of the root image, so there is not required to set up a litetree table. If you decide that you want some of the files pulled from different locations that are different per node, you can use this table. See Advanced Statelite features.

You can choose to use the defaults and not set up a litetree table.

2.4 statelite table

You may want some files in the image to be stored permanently, to survive reboots. This is done by entering the information into the `statelite` table.

See the statelite man page for description of the attributes.

Note: In the statelite table, the node or nodegroups in the table must be unique; that is a node or group should appear only once in the first column table. This makes sure that only one statelite image can be assigned to a node. See Limitations.

An example would be:

```
"japan",,"<nfssvr_ip>:/gpfs/state",,
```

Any nodes in the japan node group will have their state stored in the `/gpfs/state` directory on the machine with <nfssvr_ip> as its IP address. The image attribute should be left blank - currently it is not used.

When the node boots up, then the value of the `statemnt` attribute will be mounted to `/.statelite/persistent`. The code will then create the following subdirectory `/.statelite/persistent/<nodename>`. This directory will be the root of the image for this node's persistent files.

Also, to set the `statemnt` attribute, you can use variables from xCAT database. It follows the same grammar as the `litetree` table. For example:

```
#node,image,statemnt,comments,disable
"hv32lpar05",,"$noderes.nfsserver:/lite/state/$nodetype.profile",,
```

Note: Do not name your persistent storage directory with the node name, as the node name will be added in the directory automatically. If you do, then a directory named `/state/n01` will have its state tree inside `/state/n01/n01`.

2.5 Policy

Ensure policies are set up correctly. When a node boots up, it queries the xCAT database to get the litefile and litetree table information. In order for this to work, the commands (of the same name) must be set in the policy table to allow nodes to request it. This should happen automatically when xCAT is installed, but you may want to verify that the following lines are in the policy table:

```
chtab priority=4.7 policy.commands=litefile policy.rule=allow
chtab priority=4.8 policy.commands=litetree policy.rule=allow
```

2.6 Add Linux Distro Packages

If you haven't already done so, copy the packages from the distro media into /install. For example:

```
copycds /iso/RHEL5.2-Server-20080430.0-x86_64-DVD.iso
```

Now create a list of distro packages that should be installed into the image. You should start with the base packages in the compute template. These are required. The shipped templates are found in the /opt/xcat/share/xcat/netboot/<os> directory. You can use these defaults, but if you modify them first copy them into the /install/custom/netboot/<os> directory, so the modifications will not be lost on the next xCAT upgrade. The code will first look in the "custom" directory before looking in the "share" path.

For example:

```
mkdir -p /install/custom/netboot/rh
cd /install/custom/netboot/rh
cp /opt/xcat/share/xcat/netboot/rh/compute.pkglist compute.pkglist
```

You can then add more packages to the compute pkglist by editing compute.pkglist.

2.7 Add Third-Party Software

If you have additional software that you want in the image, you can add it to the distro package directory that was created in the previous step and then rerun createrepo. But if you want to keep your distro package directory pristine, with only distro files in it, you can use xCAT's otherpkgs support:

Create a directory on your management node named "/install/post/otherpkgs/<OS>/<ARCH>/", and put your third party rpm packages into the directory:

```
mkdir /install/post/otherpkgs/rhels5.3/x86_64
# put RPMs in there
```

Go to the directory "/install/custom/netboot/<OS>" and add the names of the RPM packages you want in the image into the <profile>.otherpkgs.pkglist file.

```
cd /install/custom/netboot/rh
vi compute.otherpkgs.pkglist
```

Note: if some of these RPMs will need write permissions to files on the node, add those files or directories into the “litefile” table.

Create the repository to contain the rpm packages you just added in. After “createrepo” is installed, you need to create one text file which contains the complete list of files to include in the repository. For example, the name of the text file is “rpms.list” in “/install/post/otherpkgs/<os>/<arch>” directory, please follow this command to include the list into “rpms.list”:

```
 cd /install/post/otherpkgs/<os>/<arch>
 ls *.rpm >rpms.list
```

Or, if you create one sub-directory to contain the rpm packages, for example, one directory named “other” in “/install/post/otherpkgs/<os>/<arch>”. Please run the following commands to add the rpm packages under “other” directory to this repo:

```
 cd /install/post/otherpkgs/<os>/<arch>
 ls other/*.rpm >>rpms.list
```

Finally, run the following command to create repodata for the directory:

```
 createrepo -i rpms.list /install/post/otherpkgs/<os>/<arch>
```

The “createrepo” command with “-i rpms.list” option will create the repository for the rpm packages listed in the “rpms.list” file. It won't destroy and affect the rpm packages which is in the same directory but have been included into another repository.

Create the .repolist file for your own repository in the same directory as above. The file name is: <PROFILE>.<OS>.<ARCH>.repolist . The file format is:

```
Type|URL|name
```

For example, if the repository is local, you can add the repository information in /install/custom/netboot/rh/compute.rh.repolist:

```
plaindir|file:///install/post/otherpkgs/rhel5.3/x86_64|otherpkgs
```

If there’s a remote repository, you can add the repository information as:

```
rpm-md|http://xcat.sourceforge.net/yum/xcat-dep/rhel5.3/x86_64|xcat-dep
rpm-md|http://xcat.sourceforge.net/yum/devel/core-snap|core-snap
```

2.8 Create Statelite Image

The postinstall file

By editing the .postinstall file for your profile, you can automate the customization of the root image when the root image is generated by the “genimage” command. XCAT supplies a basic *postinstall file with some **required** setup already in the script. Make sure there is at least one *postinstall file that will be used when you run genimage. The one shipped will setup fstab and rcons to work properly.

The *postinstall files can be found in "/opt/xcat/share/xcat/netboot/<OS>/" directories. If you need to create your own *postinstall file, first you should copy the basic setup supplied in the appropriate *postinstall files. You can add more postinstall process , if you want. If you do modify the script, you should save it in /install/custom/netboot/<os>, so it will not be overlayed with the next install.

Rules for evaluating postinstall file:

There is a rule (2.4 release or later) for which postinstall files will be selected to be used by genimage. Use these basic rules to edit the correct file in the "`/opt/xcat/share/xcat/netboot/<OS>/`" directories. The rule allows you to customize you image down to the profile, os and architecture level, if needed.
You will find postinstall files of the following forms:

```
   1)       <profile>.<os>.<arch>.postinstall
   2)       <profile>.<arch>.postinstall
   3)       <profile>.<os>.postinstall
   4)       <profile>.postinstall
```

genimage will select the postinstall file from the list, if it exists, in the order 1-4. This means, if "`<profile>.<os>.<arch>.postinstall`" is there, it will be used.
If there is no such a file, then the "`<profile>.<arch>.postinstall`" file will be used.
If there's no such a file , then the "<profile>.<os>.postinstall" file will be used.
If there is no such file, then it will use "<profile>.postinstall".

The postinstall file is actually a bash script, you can add your own code to the basic function supplied in the file.

The genimage Command

After all our tables are set up, it is time to create a base image. In this example, the operating system is RedHat 5.3, and the name of the image is "compute", so that all the compute nodes will use that image. Make sure that `compute` is also the name of the profile for the node in the nodetype table. For example:

```
lsdef <node> | grep profile

cd /opt/xcat/share/xcat/netboot/rh

./genimage -o rhels5.3 -p compute  -i eth0
-n mlx4_core,mlx4_en,igb,bnx2
```

The genimage command will do several things:
1. It will create an image in /install/netboot/<os>/<arch>/<profile>/rootimg. For example:
`/install/netboot/rhels5.3/x86_64/compute/rootimg`

2. It will create a ramdisk and kernel that can be used to boot the initial node.

One new option to genimage named "`--permission`" is added for statelite mode, you can assign user-definable permission for "`/.statelite`" directory. The default permission is 755. For example:

```
./genimage -o rhels5.3 -p compute  -i eth0 -n
mlx4_core,mlx4_en,igb,bnx2 --permission 777
```

would make the permission 777 instead of the 755 default

2.9 Setting up Post install scripts for statelite

The rules to create post install scripts for statelite image is the same as the rules for stateless/diskless install images.

There're two kinds of postscripts for statelite (also for stateless/diskless).

The first kind of postscript is executed at genimage time, it is executed again the image itself on the MN . It was setup in The postinstall file section before the image was generated.

The second kind of postscript is the script that runs on the node during node deployment time. During init.d timeframe, /etc/init.d/gettyset calls /opt/xcat/xcatdsklspost that is in the image. This script uses wget to get all the postscripts under mn:/install/postscripts and copy them to the /xcatpost directory on the node. It uses openssl or stunnel to connect to the xcatd on the mn to get all the postscript names for the node from the postscripts table. It then runs the postscripts for the node.

See the TopDOC Appendix D for setting up postinstall scripts.

2.10 Modify statelite image

Since the files that were now just created will be the default for all the files listed in the litefile table, you can edit the image directly by visiting the root tree in:

```
/install/netboot/<os>/<arch>/<profile>/rootimg
cd /install/netboot/rhels5.3/x86_64/compute/rootimg
```

You can run `chroot` to make additional changes in the image, for example, yum/zypper updates/install using the `–installroot` flag.

2.11 Run liteimg command

The liteimg command will modify your statelite image (the image that genimage just created) by creating a series of links. Once you are satisfied your image contains what you want it to, run "`liteimg <imagename>`", or "liteimg -o <os> -a <arch> -p

<profile>". For example:

```
liteimg rhel6-ppc64-statelite-compute
```
or
```
liteimg -o rhels5.3 -a x86_64 -p compute
```

For files with "link" options, the liteimg command creates two levels of indirection, so that files can be modified while in their image state as well as during runtime. For example, a file like "$imageroot/etc/ntp.conf" with "link" option in the "litefile" table, will have the following operations done to it:

(In our case $imageroot is /install/netboot/rhels5.3/x86_64/compute/rootimg)

The liteimg script, for example, does the following to create the two levels of indirection.

```
mkdir -p $imageroot/.default/etc
mkdir -p $imageroot/.statelite/tmpfs/etc
mv $imgroot/etc/ntp.conf $imgroot/.default/etc
cd $imgroot/.statelite/tmpfs/etc
ln -sf ../../../.default/etc/ntp.conf .
cd $imgroot/etc
ln -sf ../.statelite/tmpfs/etc/ntp.conf .
```

When finished, the original file will reside in $imgroot/.default/etc/ntp.conf. $imgroot/etc/ntp.conf will link to $imgroot/.statelite/tmpfs/etc/ntp.conf which will in turn link to $imgroot/.default/etc/ntp.conf

But for files without "link" options, the liteimg command only creates clones in "$imageroot/.default/" directory, when the node is booting up, the mount command with "--bind" option will get the corresponding files from the litetree places or ".default" directory to the sysroot directory.

Note: If you make any changes to your litefile table after running liteimg then you will need to rerun liteimg again. This is because files and directories need to have the two levels of redirects created.

2.12 Set the boot state to "statelite"

You can now deploy the node:

```
nodeset <noderange> osimage=rhel5.3-x86_64-statelite-compute
```

Or just run:

```
nodeset <noderange> statelite
```

This will create the necessary files in `/tftpboot` for the node to boot correctly.

2.13 Install the Node

Finally, reboot the node so that it boots up in statelite mode.

For x86_64 platform:

```
rpower <noderange> boot
```

Nodeset will have generated the appropriate PXE file so that the node boots from the nfsroot image.   This file will look similar to the following:

```
#statelite rhel5.3-x86_64-compute
DEFAULT xCAT
LABEL xCAT
 KERNEL xcat/netboot/rhel5.3/x86_64/compute/kernel
 APPEND initrd=xcat/netboot/rhel5.3/x86_64/compute/initrd.gz
NFSROOT=172.10.0.1:/install/netboot/rhel5.3/x86_64/compute
STATEMNT=cnfs:/gpfs/state XCAT=172.10.0.1:3001 console=tty0
console=ttyS0,115200n8r
```

For POWER platform:

Run the following command to boot up the nodes into statelite mode:

```
rnetboot <noderange>
```

Nodeset will have generated the appropriate yaboot.conf-MAC-ADDRESS file so that the node boots off the nfsroot image. This file will look similar to the following:

```
#statelite rhel5.3-ppc64-compute
timeout=5
image=xcat/netboot/rhel5.3/ppc64/compute/kernel
        label=xcat
        initrd=xcat/netboot/rhel5.3/ppc64/compute/initrd.gz
        append="NFSROOT=192.168.11.108:/install/netboot/rhel5.3/ppc64/c
ompute STATEMNT= XCAT=192.168.11.108:3001 "
```

You can then use rcons or wcons to watch the node boot up.

3 Commands

The following commands are in /opt/xcat/bin:

`litefile <nodename>`

Shows all the statelite files that are not to be taken from the base of the image.

`litetree <nodename>`

Shows the NFS mount points for a node.

`liteimg <image name>`

Creates a series of symbolic links in an image that is compatible with statelite booting.

`lslite -i <imagename>`

Displays a summary of the statelite information defined for <imagename>.

`lslite <noderange>`

Displays a summary of the statelite information defined for the <noderange>

4 Statelite Directory Structure

Each statelite image will have the following directories:

```
/.statelite/tmpfs/
/.statelite/persistent/<nodename>
/.statelite/mnt  # where directory tree is mounted from.
/.default/
/etc/init.d/statelite
```

All files with "link" options, which are symbolic links, will link to `/.statelite/tmpfs`.

tmpfs files that are persistent link to `/.statelite/persistent/<nodename>/` `/.statelite/persistent/<nodename>` is the directory where the node's individual storage will be mounted to.

`/.default` is where default files will be copied to from the image to tmpfs if the files are not found in the litetree hierarchy.

4.1 The noderes Table

`noderes.nfsserver` attribute can be set for the NFSroot server. If this is not set, then the defaul is the Management Node.

`noderes.nfsdir` – can be set. If this is not set, the the default is /install

5 Adding/updating software and files for the running nodes

Because most of system files for the nodes are NFS mounted on the Management Node with read-only option, installing or updating software and files should be done to the image. The image is located under `/install/netboot/<os>/<arch>/<profile>/rootimg` directory.

To install or update an rpm, do the following:

1. Install the rpm package into rootimg.

```
    rpm --root /install/netboot/<os>/<arch>/<profile>/rootimg -ivh
rpm_name
```

2. Restart the software application on the nodes.

```
    xdsh <noderange> restart_this_software
```

It is recommended to follow section 2.7 (Adding third party softeware) to add the new rpm to the otherpkgs.pkglist file so that the rpm will get installed into the new image next time the image is rebuilt.
Note: The newly added rpms are not shown when running `rpm -qa` on the nodes although the rpm is installed. It will shown next time the node is rebooted.

To create or update a file for the nodes, just modify the file in the image and restart any application that uses the file.

6 Hierarchy Support

In the statelite environment, the service node needs to provide NFS service for the compute node with statelite, the service nodes must to be setup with diskfull installation.

6.1 Setup the diskfull service node

To setup one diskfull service node, please refer to the document "How to setup Hierarchy in xCAT 2". Since statelite is a kind of NFS-hybrid method, you should remove the "`installloc`" attribute in the site table. This makes sure that the service node does not mount the /install directory from the management node on the service node.

6.2 Generate the statelite image

The chapter 2 has described how to generate the statelite image for your own profile.

NOTE: if the NFS directories defined in the litetree table are on the service node, it is better to setup the NFS directories in the service node following the chapter 2.3.

6.3 Sync the "/install" directory

The command "`prsync`" is used to sync the "/install" directory to the service nodes.
Run the following:

```
      cd /
      prsync install <sn>:/
```

<sn> is the hostname of the service node you defined.
Since the "`prsync`" command will sync all the contents in the "`/install`" directory to the service nodes, the first time will take a long time. But after the first time, it will take very short time to sync.

NOTE: if you make any changes in the "`/install`" directory on the management node, and the changes can affect the statelite image, you need to sync the "`/install`" directory to the service node again.

6.4 Set the boot state to "statelite"

Please follow the instructions in Set the boot state to "statelite".

6.5 Install the Node

Please follow the instructions in Install the Node.

7 Advanced Statelite features

7.1 Both directory and its child items coexist in litefile table

As described in the above chapters, we can add the files/directories to litefile table. Sometimes, it is necessary to put one directory and also its child item(s) into the litefile table. Due to the implementation of the statelite on Linux, some scenarios works, but some doesn't work.

Here are some examples of both directory and its child items coexisting:

1) Both the parent directory and the child file coexist:
```
"ALL","/root/testblank/",,,
"ALL","/root/testblank/tempfschild","tempfs",,
```
2) One more complex example:
```
"ALL","/root/",,,
"ALL","/root/testblank/tempfschild","tempfs",,
```
3) Another more complex example, but we don't intend to support such one scenario:
```
"ALL","/root/",,,
"ALL","/root/testblank/",,,
"ALL","/root/testblank/tempfschild","tempfs",,
```

For example, in scenario 1), the parent is "/root/testblank/", and the child is "/root/testblank/tempfschild"; in scenario 2), the parent is "/root/", and the child is "/root/testblank/tempfschild".

In order to describe the hierarchy scenarios we can handle, "P" is on behalf of "parent", and "C" is on behalf of "child".

Table 7-1 The hierarchy scenarios

Options	Example	Remarks
P: tmpfs C: tmpfs	"ALL","/root/testblank/",,, "ALL","/root/testblank/tempfschild","tempfs",,	Both the parent and the child are mounted to tmpfs on the booted node following their respective options. Only the parent are mounted to the local file system.
P: tmpfs C: persistent	"ALL","/root/testblank/",,, "ALL","/root/testblank/testpersfile","persistent",,	Both the parent and the child are mounted to tmpfs on the booted node following their respective options. Only the parent are mounted to the local file system.
P: persistent	"ALL","/root/testblank/"	Not permitted now. But plan to support

C: tmpfs	,"persistent",, "ALL","/root/testblank/tempfschild",,,	it.
P: persistent C: persistent	"ALL","/root/testblank/","persistent",, "ALL","/root/testblank/testpersfile","persistent",,	Both the parent and the child are mounted to tmpfs on the booted node following their respective options. Only the parent are mounted to the local file system.
P: ro C: Any		Not permitted;
P: con C: Any		Not permitted;
P: tmpfs C: ro		Both the parent and the child are mounted to tmpfs on the booted node following their respective options. Only the parent are mounted to the local file system.
P: tmpfs C: con		Both the parent and the child are mounted to tmpfs on the booted node following their respective options. Only the parent are mounted to the local file system.
P: link C: link	“ALL”,”/root/testlink/”,”link”,, “ALL”,”/root/testlink/testlinkchild”,”link”,,	Both the parent and the child are created in tmpfs on the booted node following their respective options; there's only one symbolic link of the parent is created in the local file system.
P: link C: link,persistent	“ALL”,”/root/testlink/”,”link”,, “ALL”,”/root/testlink/testlinkperschild”,”link,persistent”,,	Both the parent and the child are created in tmpfs on the booted node following their respective options; there's only one symbolic link of the parent is created in the local file system.
P: link,persistent C: link	“ALL”,”/root/testlinkpers/”,”link,persistent”,, “ALL”,”/root/testlink/testlinkchild”,”link”	NOT permitted;
P: link,persistent C: link,persistent	“ALL”,”/root/testlinkpers/”,”link,persistent”,, “ALL”,”/root/testlink/testlinkperschild”,”link,persistent”,,	Both the parent and the child are created in tmpfs on the booted node following the “link,persistent” way; there's only one symbolic link of the parent is created in the local file system.
P: link C: link,ro	“ALL”,”/root/testlink/”,”link”,,	Both the parent and the child are created in tmpfs on the booted node, there's only one symbolic link of the parent is created in the local file

	"ALL","/root/testlink/testlinkro","link,ro",,	system.
P: link C: link,con	"ALL","/root/testlink/","link",, "ALL","/root/testlink/testlinkconchild","link,con",,	Both the parent and the child are created in tmpfs on the booted node, there's only one symbolic link of the parent in the local file system.
P: link,persistent C: link,ro		NOT Permitted;
P: link,persistent C: link,con		NOT Permitted;
P: tmpfs C: link		NOT Permitted;
P: link C: persistent		NOT Permitted;

7.2 litetree table

The litetree table controls where the initial content of the files in the litefile table come from, and the long term content of the "ro" files. When a node boots up in statelite mode, it will by default copy all of its tmpfs files from the /.default directory of the root image, so there is not requirement to setup a litetree table. If you decide that you want some of the files pulled from different locations that are different per node, you can use this table.

See litetree man page for description of attributes.

For example, a user may have two directories with a different */etc/motd* that should be used for nodes in two locations:

```
10.0.0.1:/syncdirs/newyork-590Madison/rhels5.4/x86_64/compute/etc/motd
10.0.0.1:/syncdirs/shanghai-11foo/rhels5.4/x86_64/compute/etc/motd
```

You can specify this in one row in the litetree table:

```
1,,10.0.0.1:/syncdirs/$nodepos.room/$nodetype.os/$nodetype.arch/
$nodetype.profile
```

When each statelite node boots, the variables in the litetree table will be substituted with the values for that node to locate the correct directory to use. Assuming that /etc/motd was specified in the litefile table, it will be searched for in all of the directories specified in the litetree table and found in this one.

You may also want to look by default into directories containing the node name first:
```
$noderes.nfsserver:/syncdirs/$node
```

The litetree prioritizes where node files are found. The first field is the priority. The second field is the image name (ALL for all images) and the final field is the mount point.

Our example is as follows:
```
1,,$noderes.nfsserver:/statelite/$node
2,,cnfs:/gpfs/dallas/
```

The two directories /statelite/$node on the node's $noderes.nfsserver and the /gpfs/dallas on the node cnfs contain root tree structures that are sparsely populated with files that we want to place in those nodes. If files are not found in the first directory, it goes to the next directory. If none of the files can be found in the litetree hierarchy, then they are searched for in /.default on the local image.

7.3 Installing a new Kernel in the statelite image (This is not verified)

Obtain you new kernel and kernel modules on the MN, for example here we have a new SLES kernel.

Copy the kernel into /boot :

```
cp  vmlinux-2.6.32.10-0.5-ppc64   /boot
```

Copy the kernel modules into /lib/modules/<new kernel directory>

```
xcatlinuxmn:/lib/modules # ls -al
total 16
drwxr-xr-x  4 root root 4096 Apr 19 10:39 .
drwxr-xr-x 17 root root 4096 Apr 13 08:39 ..
drwxr-xr-x  3 root root 4096 Apr 13 08:51 2.6.32.10-0.4-ppc64
drwxr-xr-x  4 root root 4096 Apr 19 10:12 2.6.32.10-0.5-ppc64
```

Run genimage to update the statelite image with the new kernel

```
cd /opt/xcat/share/xcat/netboot/sles


./genimage -i eth0 -n ehea -o sles11.1 -p compute  -m statelite  -k 2.6.32.10-0.5-ppc64
```

then after a nodeset command and netboot.. the uname -a shows the new kernel.

8

9 Debugging techniques

1. When a node boots up in statelite mode, there is a script that runs called statelite that is in the root directory of $imageroot/etc/init.d/statelite. This script is not run as part of the rc scripts, but as part of the pre-switch root environment. Thus, all the linking is done in this script. There is a "set –x" near the top of the file. You can uncomment it and see what the script runs. You will then see lots of mkdirs and links on the console.
2. You can also set the machine to shell. Just add the word "shell" on the end of the pxeboot file of the node in the append line. This will make the init script in the initramfs pause 3 times before doing a switch_root.
3. When all the files are linked they are logged in /.statelite/statelite.log on the node. You can get into the node after it has booted and look in the /.statelite directory.