xCAT 2.X Monitoring How-to

10/16/2009

Table of Contents

1.0 Introduction.....1
2.0 Using xCAT Monitoring Plug-in Infrastructure.....1
2.1 xCAT Monitoring Commands and How-to.....2
2.1.1 Define monitoring servers.....3
2.2 Install and enable monitoring.....4
2.2.1 SNMP monitoring.....4
2.2.2 RMC monitoring.....7
2.2.3 Node liveness monitoring.....10
2.2.4 Ganglia monitoring.....11
2.2.5 PCP (Performance Co-Pilot).....12
2.2.6 Node liveness monitoring with PCP.....14
2.3 Create your own monitoring plug-in module.....14
3.0 Using xCAT Notification Infrastructure.....16

Introduction

There are two monitoring infrastructures introduced in xCAT 2.0. The ***xCAT Monitoring Plug-in Infrastructure*** allows you to plug-in one or more third party monitoring software such as Ganglia, RMC, SNMP etc. to monitor the xCAT cluster. The ***xCAT Notification Infrastructure*** allows you to watch for the changes in xCAT database tables.

Using 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. Though you can write your own monitoring plug-in modules (see section 2.3), over time, xCAT will supply a list of built-in plug-in modules for the most common monitoring software. They are:

- xCAT (xcatmon.pm) (monitoring node statue using fping. released)
- SNMP (snmpmon.pm) (snmp monitoring. released)
- RMC (rmcmon.pm) (released)

- Ganglia (gangliamon.pm) (released)
- Nagios (nagiosmon.pm)
- Performance Co-pilot (pcpmon.pm)

You can pick one or more monitoring plug-ins to monitor the xCAT cluster. The following sections will demonstrate how to use the plug-ins.

xCAT Monitoring Commands and How-to

In xCAT, there are 8 commands available for monitoring purpose. They are:

Command	**Description**
monls	list the current or all the monitoring plug-in names, their status and description.
monadd	add a monitoring plug-in to the 'monitoring' table. This will also adds the configuration scripts for the monitoring plug-in, if any, to the 'postscripts' table.
monrm	remove a monitoring plug-in from the 'monitoring' table. It also removes the configuration scripts for the monitoring plug-in from the 'postscritps' table.
moncfg	configure the 3rd party monitoring software on the management server and the service node for the given nodes to include the nodes into the monitoring domain. It does all the necessary configuration changes to prepare the software for monitoring the nodes. The -r option will configure the nodes as well.
mondecfg	deconfigure the 3rd party monitoring software on the management server and the service node for the given nodes to remove the nodes from the monitoring domain. The -r option will deconfigure the nodes as well.
monstart	start 3rd party software on the management server and the service node for the given nodes to monitor the xCAT cluster. It includes starting the daemons. The -r option will start the daemons on the nodes as well.
monstop	stop 3rd party software on the management server and the service node for the given nodes from monitoring the xCAT cluster. The -r will stop the daemons on the nodes as well.
monshow	displays the events that happened on the given nodes or the monitoring data that is collected from the given nodes.

There are 2 ways you can configure the 3rd party software to monitor the xCAT cluster. The first way is to configure it on the nodes during the node deployment phase. The second is to configure it after the node is up and running.

Define monitoring servers

If you have a small number of nodes to monitor, or if you prefer the management node (mn) handles the monitoring then make your management node the monitoring node. For a large cluster, it is recommended that you dedicate some nodes as monitoring aggregation points. These nodes are called monitoring servers. You can use the service nodes (sn) as the monitoring servers. The monitoring servers are defined by the 'monserver' column of the **noderes** table. The data in 'monserver' column is a comma separated pairs of host names or ip addresses. The first host name or ip address represents the network adapter on that connects to the mn. The second host name or ip address represents the network adapter that connects to the nodes. If the no data is provided in the 'monserver' column, the values in the 'servicenode' and the 'xcatmaster' columns in the same table will be used. If none is defined, the mn will be used as the monitoring server.

In following example the nodes in group2 have dedicated monitoring server (monsv02) while the nodes in group1 use their service node as the monitoring server (sn01).

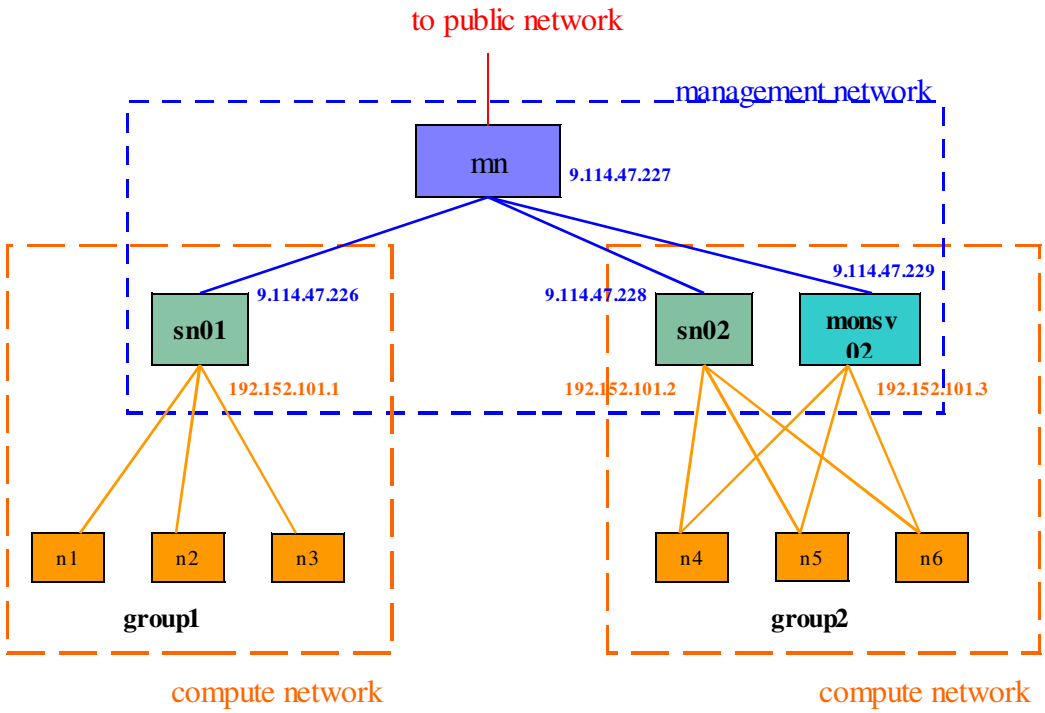

Figure 1. Monitoring servers for the nodes

The **noderes** table looks like this for the above cluster.

node	monservers	servicenode	xcatmaster
sv01		9.114.47.227	9.114.47.227
sv02		9.114.47.227	9.114.47.227
monsv02		9.114.47.227	9.114.47.227
group1		sv01	192.152.101.1
group2	monsv02, 192.152.101.3	sv02	192.152.101.2

Install and enable monitoring

The following sections how to install the different types of monitoring tools that are available. Where applicable place the software in the appropriate directories under /install for AIX or LINUX for installation during the rnetboot process. If you are configuring monitoring on a cluster where the nodes are already installed use the updatenode process.

SNMP monitoring

1. Download the corresponding mib files for your system that you wish to receive SNMP traps from and copy them onto the management node (mn) and the monitoring servers under the following directory:
   `/usr/share/snmp/mibs/`

   The mib files for IBM blade center management modules (MM) and RSAII are packaged within the firmware updates. They can be found under IBM support page: http://www.ibm.com/support/us/en/

   To download the mibs for MM go to:
   http://www-304.ibm.com/systems/support/supportsite.wss/docdisplay?lndocid=MIGR-5070708&brandind=5000020

   Download file `ibm_fw_amm_bpet26k_anyos_noarch.zip`. Then unzip the file and you will fine two mib files `mmblade.mib` and `mmalert.mib`

   To download the mibs for RSAII go to:
   http://www-304.ibm.com/systems/support/supportsite.wss/docdisplay?brandind=5000008&lndocid=MIGR-64575

   Download file `ibm_fw_rsa2_ggep30a_anyos_noarch.zip`. Then unzip it and you will find the mib files `RTRSAAG.MIB` and `RTALSERT.MIB`

2. Make sure `net-snmp` rpm is installed on mn and all the monitoring servers.
   ```
   rpm -qa |grep net-snmp
   ```

3. Add snmpmon to the **monitoring** table.
```
monadd snmpmon
```

4. Configure the Blade Center MM or BMC to set the trap destination to be the management server.
```
moncfg snmpmon -r
```

5. To activate, use the monstart command.
```
monstart snmpmon -r
```

   Note: Use this command to stop snmpmon.
```
monstop snmpmon -r
```

6. Verify monitoring was started.
```
monls
snmpmon         monitored
```

7. Set email recipients. When traps are received, they will be logged into the syslog on mn. The warning and critical alerts will be emailed to 'alerts' alias on the mn's mail system. By default, 'alerts' points to the root's mailbox, but you can have the emails sent to other recipients by modifying it. On the mn:
```
vi /etc/aliases
```

   Find the line beginning with the word `alerts`. It is usually is at the bottom of the file. Change the line so it looks something like this
```
alerts    root,joe@us.ibm.com,jill@yahoo.com
```

   Now make the new email aliases in effect
```
newaliases
```

8. Set up the filters. The xCAT built-in SNMP trap handler can process any SNMP traps. Here is a sample email message sent by the trap handler after a Blade Center MM trap is handled.
```
Subject: Critical: Cluster SNMP Alert!
Message:
 Node: rro123b
 Machine Type/Model:  0284
 Serial Number:  1012ADA
 Room:
 Rack:
 Unit:
 Chassis:
 Slot:

SNMP Critical Alert received from bco41(UDP: [11.16.15.41]:161)
 App ID: "BladeCenter Advanced Management Module"
 App Alert Type: 128
 Message: "Processor 2 (CPU 2 Status) internal error"
 Blade Name: "rro123b"
 Error Source="Blade_11"

 Trap details:
```

```
DISMAN-EVENT-MIB::sysUpTimeInstance=17:17:49:12.08
SNMPv2-MIB::snmpTrapOID.0=BLADESPPALT-MIB::mmTrapBladeC
BLADESPPALT-MIB::spTrapDateTime="Date(m/d/y)=05/20/08, Time(h:m:s)=14:30:12"
BLADESPPALT-MIB::spTrapAppId="BladeCenter Advanced Management Module"
BLADESPPALT-MIB::spTrapSpTxtId="bco41"
BLADESPPALT-MIB::spTrapSysUuid="D76ADB0137E2438B9F14DCC6569478BA"
BLADESPPALT-MIB::spTrapSysSern="100058A"
BLADESPPALT-MIB::spTrapAppType=128
BLADESPPALT-MIB::spTrapPriority=0
BLADESPPALT-MIB::spTrapMsgText="Processor 2 (CPU 2 Status) internal error"
BLADESPPALT-MIB::spTrapHostContact="No Contact Configured"
BLADESPPALT-MIB::spTrapHostLocation="No Location Configured"
BLADESPPALT-MIB::spTrapBladeName="rro123b"
BLADESPPALT-MIB::spTrapBladeSern="YL113684L129"
BLADESPPALT-MIB::spTrapBladeUuid="3A77351D00001000B6AA001A640F4972"
BLADESPPALT-MIB::spTrapEvtName=2154758151
BLADESPPALT-MIB::spTrapSourceId="Blade_11"
SNMP-COMMUNITY-MIB::snmpTrapAddress.0=11.16.15.41
SNMP-COMMUNITY-MIB::snmpTrapCommunity.0="public"
SNMPv2-MIB::snmpTrapEnterprise.0=BLADESPPALT-MIB::mmRemoteSupTrapMIB
```

But sometimes you want the trap handler filter out certain type of alerts. For example, when blades are rebooting you will get a lot of alerts and you do not want to be notified for these alerts. The filtering can be done by adding a row in the **monsetting** table with name equals to `snmpmon` and key equals to `ignore`. The value is a comma separated list that describes the contents in a trap. For example, to filter out any blade center mm traps from blade rro123b.

```
chtab name=snmpmon,key=ignore monsetting.value=BLADESPPALT-
MIB::spTrapBladeName="rro123b"
```

(The mib module name `BLADESPPALT-MIB` is optional in the command. `spTrapBladeName` can be found in the mm mib file or from your email notification.)

The following example will filter out all power on/off/reboot alerts for any blades.

```
chtab name=snmpmon,key=ignore monsetting.value=spTrapMsgText=”Blade
powered off”,spTrapMsgText=”Blade powered on”,spTrapMsgText=”System
board (Sys Pwr Monitor) power cycle”,spTrapMsgText=”System board (Sys
Pwr Monitor) power off”,spTrapMsgText=”System board (Sys Pwr Monitor)
power on”,spTrapMsgText=”Blade reboot”
```

There are other keys and values for the **monsetting** table supported by snmpmon monitoring plug-in. For example, you can make user-defined commands to be run for certain traps by adding '`runcmd`' key in the table. Use this command to list all the possible keywords.

```
monls snmpon -d
```

9. Make sure the blade names on Blade Center MM are identical to the node names defined in the xCAT nodelist table.

```
rspconfig group1 textid      (This command queries the blade name)
n1: textid: SN#YL10338241EA
n2: textid: SN#YL103382513F
n3: textid: SN#YK13A084307Y
rspconfig group1 textid=*      (This command sets the blade name)
n1: textid: n1
```

```
n2: textid: n2
n3: textid: n3
```

10. Make sure snmptrapd is up and running on mn and all monitoring servers. It should have the '`-m ALL`' flag.

```
ps -ef |grep snmptrapd
 root     31866     1  0 08:44 ?  00:00:00 /usr/sbin/snmptrapd -m
ALL
```

11. Make sure snmp destination is set to the corresponding monitoring servers.

```
rspconfig mm snmpdest   (mm is the group name for all the blade center mm)
mm1: SP SNMP Destination 1:  192.152.101.1
mm2: SP SNMP Destination 1:  192.152.101.3
```

12. Make sure SNMP alert is set to 'enabled'.

```
rspconfig mm alert
 mm1: SP Alerting: enabled
 mm2: SP Alerting: enabled
```

RMC monitoring

IBM's Resource Monitoring and Control (RMC) subsystem is our recommended software for monitoring xCAT clusters. It's is part of the IBM's Reliable Scalable Cluster Technology (RSCT) that provides a comprehensive clustering environment for AIX and LINUX. The RMC subsystem and the core resource managers that ship with RSCT enable you to monitor various resources of your system and create automated responses to changing conditions of those resources. RMC also allows you to create your own conditions (monitors), responses (actions) and sensors (resources).

Rmcmon is xCAT's monitoring plug-in module for RMC. It's responsible for automatically setting up RMC monitoring domain for RMC and creates predefined conditions, responses and sensor on the management node, the service node and the nodes. Rmcmon also provides node reachability and status updates on xCAT **nodelist** table via RMC's node status monitoring. If you enable performance monitoring, xCAT will collect and consolidate data from RSCT resource and store to RRD database.

1. Verify and or install *rsct.core, rsct.core.utils* and *bos.rte.SRC* on the mn. For LINUX, the software can be downloaded from:
   http://www14.software.ibm.com/webapp/set2/sas/f/rsct/rmc/download/home.html

   RSCT comes with the AIX operating system. Make sure the level of software is supported by xCAT.

   - For AIX 5.3, you need at least RSCT 2.4.9.0 which ships with AIX 5.3.0.80 or greater.

- For AIX 6.1, you need at least RSCT 2.5.1.0 which ships with AIX 6.1.1.0 or greater.

  Use this command to check the RSCT level:

  ```
  /usr/sbin/rsct/install/bin/ctversion
  ```

  If you have a lower version of AIX, you can obtain the latest RSCT as part of the AIX Technology Levels, or as separate PTFs.

  AIX 6.1
  6100-01 TL:
  http://www-933.ibm.com/eserver/support/fixes/fixcentral/pseriesfixpackinformation/6100-01-00-0822
  PTFS:
  http://www-933.ibm.com/eserver/support/fixes/fixcentral/pseriespkgoptions/ptf?fixes=U817016

  AIX 5.3:
  5300-08 TL:
  http://www-933.ibm.com/eserver/support/fixes/fixcentral/pseriesfixpackinformation/5300-08-00-0818
  PTFS:
  http://www-933.ibm.com/eserver/support/fixes/fixcentral/pseriespkgoptions/ptf?fixes=U816993

Note* On AIX, CSM client comes with the OS. The HMC will periodically bring the nodes into its own CSM cluster which wipes out the RMC domain set by the following process. To fix it, you need to get the CSM 1.7.1.4 or greater on the nodes. You also need the level of RSCT that works with this level of CSM. You need RSCT 2.5.3.0 or greater on AIX 6.1 and RSCT 2.4.12.0 or greater on AIX 5.3.

The latest CSM can be found here
http://www14.software.ibm.com/webapp/set2/sas/f/csm/download/home.html

1. Install xCAT-rmc on the mn.

   ```
   rpm -Uvh xCAT-rmc-2.1.rpm
   ```

2. Make sure all the nodes and service nodes that need to have RMC installed have 'osi' in the `nodetype` column of the **nodetype** table.

   ```
   tabdump nodetype
   ```

   Make sure the `mac` column of **mac** table for the nodes and the service nodes are populated because the mac address will be used as a RMC nodeid for the node.

   ```
   tabdump mac
   ```

3. Setup the xCAT hierarchy. If you are not using separate monitoring servers or have a flat cluster, skip this step. The `monserver` column of the **noderes** table is used to

define a monitoring server for a node. The `monserver` is a comma separated pair of host name or ip addresses. The first one is the monitoring server name or ip known by the mn. The second one is the same host known by the node.

```
chdef -t node -o node5 monserver=9.114.46.47,192.168.52.118
```

If `monserver` is not set, the default is the `servicenode` and `xcatmaster` pair in the **noderes** table.

4. Add rmcmon in the **monitoring** table. This will also add the `configrmcnode` postscript into **postscripts** table.

```
monadd rmcmon
```

or

```
monadd rmcmon -n
```

or

```
monadd rmcmon -n -s [montype=perf]
```

The second command allows the RMC monitoring feed the node reachability status monitoring to xCAT. You need to have RSCT 2.4.10.0 or greater on AIX 5.3 system or RSCT 2.5.2.0 or greater on AIX 6.1 for this feature to work. The third command enables performance monitoring only. Event monitoring will be disabled. To enable both, use -s [montype=event,perf]

5. Create resources for the IBM.MngNode class to include each node managed by the management node.

```
moncfg rmcmon nodes
```

6. Add the nodes into the RMC domain. If the nodes are already installed, make sure *rsct.core, rsct.core.utils* and *bos.rte.SRC* are installed on the nodes. Then run:

```
moncfg rmcmon nodes -r
```

If the nodes are not installed make sure the *rsct.core, rsct.core.utils* and *bos.rte.SRC* are included in the images for the nodes and any monitoring servers. This process will automatically setup the RMC monitoring domain.

7. Verify that RMC domain are setup correctly. List all the nodes managed by mn:

```
lsrsrc -a IBM.Host Name
Resource Persistent Attributes for IBM.Host
resource 1:
        Name = "node1"
resource 2:
        Name = "node2"
```

If lsrsrc shows any nodes are not in the cluster, refresh the configuration and check again.

```
xdsh nodes refrsrc IBM.MCP
lsrsrc -a IBM.Host Name
```

If applicable, use xdsh command to check the nodes managed by the monitoring

servers:

```
xdsh monserver_name lsrsrc -a IBM.Host Name
```

8. Setup optional performance monitoring. The `moncfg` command ran earlier in this section added predefined performance metrics which can be collected into the monsetting table. You can modify it as needed.

```
chtab key=rmetrics_resourceclass monsetting.name=rmcmon
monsetting.value=[resource names]attribute names:sample interval
```

   The unit of `sample interval` is minute.

   For example:

```
tabdump monsetting
#name,key,value,comments,disable
"rmcmon","rmetrics_IBM.Processor","[proc0,proc1]PctTimeIdle,PctTimeWa
it,PctTimeKernel,PctTimeUser:5",,
"rmcmon","montype","perf,event",,
```

   RRD needs to be installed if you want to use performance monitoring. You can download rrdtool RPM and its dependencies for AIX from http://www.perzl.org/aix/.

9. To activate, use the monstart command. This will start event as well as performance monitoring if specified with `monadd` command earlier in this section.

```
monstart rmcmon
```

10. Verify monitoring was started.

```
monls
rmcmon          monitored
```

11. For event monitoring, the `moncfg` command created a set of predefined conditions, responses and sensors on the mn and the monitoring servers. To verify issue the following:

```
lscondition
lsresponse
lssensor
lscondresp
```

12. Pick and choose the conditions to monitor using the `startcondresp` command which associates a condition with a response. A condition can be associated with more than one response. In this example the response will write to the **eventlog** table.

```
startcondresp AnyNodeVarSpaceUsed LogEventToxCATDatabase
```

   If you have monitoring servers, also turn on the condition with "`_H`" in the name.

```
startcondresp AnyNodeVarSpaceUsed_H LogEventToxCATDatabase
```

   Conditions without "`_H`" in the name are designed to monitor the nodes which are managed by the mn, whereas conditions with "`_H`" in the names are for the nodes managed by the monitoring servers. These nodes are grandchildren of the mn. The

associations will be saved to file /var/log/rmcmon on mn and service nodes when monstop is issued so that next time you use monstart command the associations will be preserved.

With RMC, you can create your own conditions, responses and sensors for monitoring. The useful commands are:

```
lsrsrc
mkcondition
rmcondition
lscondition
mkresponse
lsresponse
rmresponse
startcondresp
stopcondresp
rmcondresp
mksensor
lssensor
rmsensor
```

Please refer to "RSCT Administration Guide" for more details.

Node liveness monitoring

Xcatmon 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.

1. Add the monitoring plug-in in the 'monitoring' table:
   ```
   monadd xcatmon -n -s [ping-interval=5]
   ```
   where 5 means that the nodes are pinged for status every 5 minutes**.**
2. To activate, use the monstart command.
   ```
   monstart xcatmon
   ```
3. Verify monitoring was started.
   ```
   monls snmpmon
   xcatmon          monitored   node-status-monitored
   ```
4. Check the settings.
   ```
   tabdump monsetting
   #name,key,value,comments,disable
   "xcatmon","ping-interval","5",,
   ```
5. Make sure cron jobs are activated on mn and all monitoring server.
   ```
    crontab -l
   */5 * * * * XCATROOT=/opt/xcat
   ```

```
PATH=/bin:/usr/bin:/sbin:/usr/sbin:/opt/xcat/bin:/opt/xcat/sbin /opt/
xcat/sbin/xcatnodemon
```

Ganglia monitoring

1. Install Ganglia on the management node. The following prerequisites are needed before Gangila is installed: Apache, PHP and RRD.

RRD installation:

1) Download and install the RRD rpm (Please note that the versions of the rpms might change and hence download them appropriately).

```
rpm -Uvh rrdtool-1.2.27-3.el5.i386.rpm
```

Ganglia installation:

1) Download the software from:
http://ganglia.sourceforge.net/

2) Obtain the following rpms found under Ganglia monitoring core (Please note that the versions of the rpms might change and hence download them appropriately).

```
ganglia-gmetad-3.0.7-1.i386.rpm
ganglia-gmond-3.0.7-1.i386.rpm
ganglia-web-3.0.7-1.noarch.rpm
```

3) Install the above rpms.

```
rpm -Uvh ganglia-gmetad-3.0.7-1.i386.rpm ganglia-gmond-3.0.7-
1.i386.rpm ganglia-web-3.0.7-1.noarch.rpm
```

1. Make sure all the nodes and service nodes that need to Ganglia installed have 'osi' in the `nodetype` column of the **nodetype** table.

```
tabdump nodetype
```

2. Add gangliamon to the **monitoring** table. This command will also add the 'confGang' configuration scripts on the 'postscripts' table.

```
monadd gangliamon
```

3. Install Ganglia on the service node and the compute nodes.
   1) Copy `ganglia-gmetad-3.0.7-1.i386.rpm` and `ganglia-gmond-3.0.7-1.i386.rpm` to the `/install/post/otherpkgs/<osver>/<arch>` directory.
   2) Add `ganglia-gmetad` and `ganglia-gmond` to the service node's 'other packages' profile (`service.otherpkgs.pkglist`) and save it to the `/install/custom/<install|netboot>/os` directory.
   3) Add `ganglia-gmond` to the compute node's 'other packages' profile (`compute.otherpkgs.pkglist`) and save it to `/install/custom/<install|`

`netboot>/os` directory. Please refer to xCAT 2 How to Install Additonal Software for details.

4) Install the service node and then compute nodes. Please refer to the xCAT 2 cookbook for details. This step will run the 'confGang' postscript on all the nodes which configures the Ganglia to use unicast modd.

1. Configure management node and the service nodes

```
moncfg gangliamon -r
```

2. To activate, use the monstart command. The -r flag will ensure that the Ganglia daemon (gmond) on the node is started.

```
monstart gangliamon -r
```

Note: Use this command to stop gangliamon:

```
monstop gangliamon -r
```

3. Verify monitoring was started.

```
monls
gangliamon      monitored
```

PCP (Performance Co-Pilot)

1. Install PCP on the management node.

PCP installation:

1) Download the software from: ftp://oss.sgi.com/projects/pcp/download/
2) Obtain the rpms for PCP (Please note that the versions of the rpm might change and hence download them appropriately). pcp-2.7.7-20080924.i386.rpm
3) Install the rpm.

```
rpm -Uvh  pcp-2.7.7-20080924.i386.rpm
```

1. Make sure all the nodes and service nodes that need to have PCP installed have 'osi' in the `nodetype` column of the **nodetype** table.

```
tabdump nodetype
```

2. Add pcpmon to the “monitoring” table.

```
monadd pcpmon
```

or to use PCP to monitor node status do

```
monadd pcpmon -n
```

4. Install PCP on the service node and the compute nodes.
   1) Copy `pcp-2.7.7-20080924.i386.rpm` to the `/install/post/otherpkgs/<osver>/<arch>` directory.
   2) Add `pcp` to the service node's 'other packages' profile (`service.otherpkgs.pkglist`) and save it to the `/install/custom/<install|netboot>/os` directory.
   3) Add `pcp` to the compute node's 'other packages' profile (`compute.otherpkgs.pkglist`) and save it to /`install/custom/<install|netboot>/os` directory. Please refer to xCAT 2 How to Install Additonal Software for details.
   4) Install the service node and then compute nodes. Please refer to the xCAT 2 cookbook for details.

4. The "pcpmon" allows the users to collect the specific performance monitoring metrics. The metrics are inputted through a configuration file called "pcpmon.config" located under "`$::XCATROOT/lib/perl/xCAT_monitoring/pcp/`". There is no limit on the number of metrics that can be collected as long as all the metrics are legal in the PCP context. The user can update the configuration file periodically to add/remove metrics. Use the "pminfo" command to list valid metrics. A typical "pcpmon.config" file could look like this:

```
mem.physmem
mem.util.free
mem.util.swapFree
filesys.used
proc.memory.size
disk.dev.total
```

5. The metrics are updated in the xCAT database table called "performance". A typical entry in the "performance" table could look like this:

```
#timestamp,node,attrname,attrvalue
"10/08/08:16:21:18","cu03sv","mem.physmem","1925460.000"
"10/08/08:16:21:18","cu03sv","mem.util.free","179448.000"
"10/08/08:16:21:18","cu03sv","mem.util.swapFree","1052216.000"
"10/08/08:16:21:18","cu03sv","filesys.used","10224392.000"
"10/08/08:16:21:18","cu03sv","proc.memory.size","92.000"
"10/08/08:16:21:18","cu03sv","disk.dev.total","10316.000"
```

6. To activate, use the monstart command. The -r flag will ensure that the PCP daemon on the node is started.

```
monstart pcpmon -r
```

   Note: Use this command to stop pcpmon:

```
monstop pcpmon -r
```

1. Verify monitoring was started.

```
monls
```

```
pcpmon   monitored
```

Node liveness monitoring with PCP

*P*cpmon provides node liveness monitoring. 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.

1. To add the monitoring plug-in in the 'monitoring' table:
   ```
   monadd pcpmon -n -s [ping-interval=15]
   ```
   Please note that the above command will set the interval to 15 minutes and the default (if “ping-interval” is not used) is 5 minutes.
2. To activate, use the monstart command.
   ```
   monstart pcpmon
   ```
3. Verify monitoring was started.
   ```
   monls pcpmon
   pcpmon          monitored       node-status-monitored
   ```
4. Check the settings.
   ```
   tabdump monsetting
   #name,key,value,comments,disable
   "xcatmon","ping-interval","15",,
   ```
5. Make sure cron jobs are activated on MN and all monitoring servers.
   ```
   crontab -l
   */15 * * * * XCATROOT=/opt/xcat
   PATH=/sbin:/usr/sbin:/bin:/usr/bin:/opt/xcat/bin:/opt/xcat/sbin /opt/
   xcat/sbin/pcp_collect
   ```

Create your own monitoring plug-in module

As mentioned before, a monitoring plug-in modules acts as a bridge 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 data flow and the relationship among xcatd, monitoring plug-ins and the third party software.

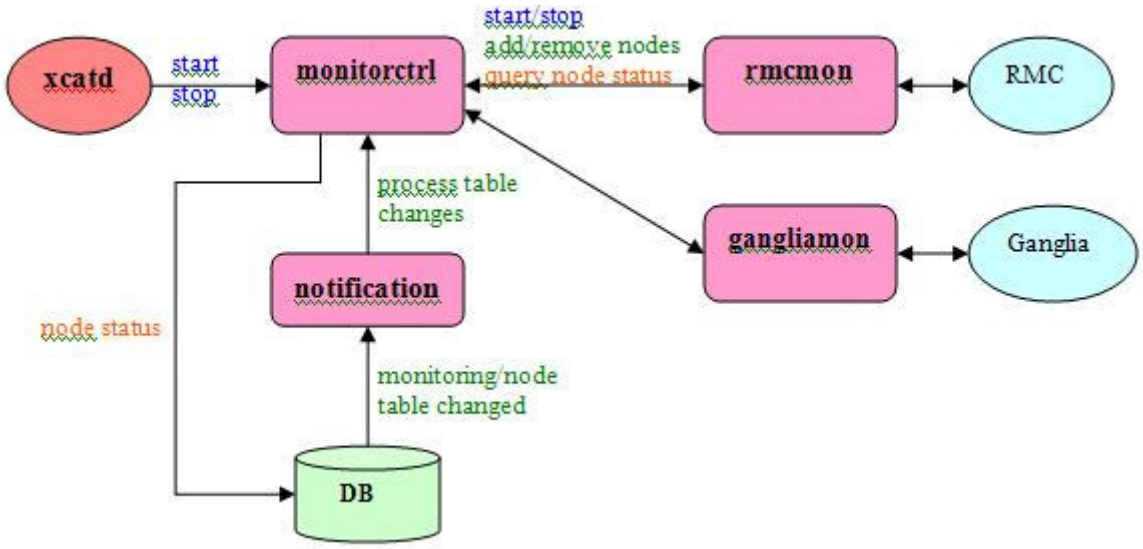

Figure 2. Data flow among xcatd, plug-in modules and 3rd party monitoring software

To use this infrastructure to create your own plug-in module, create a Perl 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

config

deconfig

supportNodeStatusMon

startNodeStatusMon

stopNodeStatusMon

processSettingChanges (optional)

getDiscription

getPostscripts (optional)

getNodeConfData (optional)

Please refer to `/opt/xcat/lib/perl/xCAT_monitoring/samples/tmplatemon.pm` for the detailed description of the functions. You can find *tmplatemon.pm* from the xCAT source code on the web:
http://xcat.svn.sourceforge.net/viewvc/xcat/xcat-core/trunk/xCAT-server-2.0/lib/xcat/monitoring/samples/

Using 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 etc. In fact anything stored in the xCAT database tables can be monitored through the xCAT notification infrastructure.

1. 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
     ```

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

   ```
   tabdump notification
   ```

2. To unregister, do the following:

   ```
   unregnotif filename
   ```

   Example:

   ```
   unregnotif  /opt/xcat/lib/perl/xCAT_monitoring/mycode.pm
   unregnotif  /usr/bin/mycmd
   ```

   If the file name 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 an installed system or on the web:
http://xcat.svn.sourceforge.net/viewvc/xcat/xcat-core/trunk/xCAT-server-2.0/lib/xcat/monitoring/samples/