Pivotal®

Pivotal HD Monitoring With Ganglia and Nagios

Table Of Content

Version History

Date	Version	Author	Change Description
12/5/2014	0.1	Sanjay Dwivedi	Initial Document Added Ganglia Section
12/8/2014	0.2	Sanjay Dwivedi	Added Nagios Section
12/11/2014	0.3	Sanjay Dwivedi	Added SNMP daemon configuration steps

1. Introduction

This document provides instructions on installation and configuration of open monitoring and alerting tools for Pivotal HD, namely:

- Open source monitoring tool Ganglia (http://ganglia.sourceforge.net/)
- Open source monitoring and alerting tool Nagios (http://www.nagios.org/)

Ganglia is a very popular tool that offers distributed monitoring system for high-performance computing systems such as clusters and Grids. It offers real-time trending of IT infrastructure services. However it does not offer any alerting capability. Alerting is offered by another very popular and capable tool, Nagios.

Nagios is a very popular IT Infrastructure monitoring tool. It offers complete monitoring and alerting for servers, switches, applications and services. Its core strength is in alerting and a very simple plugin mechanism for monitoring almost anything using any language or tool of your choice.

Ganglia can be used to monitor Pivotal HD cluster and Nagios can used to alert if Pivotal HD services become unavailable. Next sections describe steps to install and configure Ganglia and Nagios for Pivotal HD.

The installation is primarily described for RedHat Enterprise Linux based system and it should work for both CentOS and Fedora Linux distributions.

Configuration is generic and should work on any Linux distribution. The CentOS 6.4 was used during preparation of this document. Some of the EPEL repositories used during installation may differ based OS version number.

Use of massh

Installation and configuration in highly distributed environment can be challenging. As Hadoop is a distributed system that potentially spans thousands of hosts, we need a parallel shell i.e. a shell that executes commands on multiple hosts in parallel. Such shell can be a very valuable tool for managing systems. Pivotal HD includes one such tool: `massh`[1]. The other popular tool in same vein is `pdsh`[2].

Since `massh` is included with Pivotal HD, its use is highly recommended for installation and configuration tasks covered in this document that are required to be executed on each cluster node.

[1] http://m.a.tt/er/massh/
[2] https://code.google.com/p/pdsh/

Source code for all the scripts for configuring Ganglia and Nagios can be found in Pivotal Field Engineering github at:
https://github.com/Pivotal-Field-Engineering/phd-monitoring

2. Ganglia

This section describes installation and configuration steps for Ganglia on Pivotal HD cluster. All Ganglia configuration files described here are also available in Pivotal Field Engineering github at:
https://github.com/Pivotal-Field-Engineering/phd-monitoring/ganglia

2.1 Ganglia Installation and Configuration

Note: yum requires root or sudo privileges.

1. Add the EPEL Repository - if needed

   ```
   # wget
   http://download.fedoraproject.org/pub/epel/6/x86_64/ep
   el-release-6-8.noarch.rpm
   # rpm -ivh epel-release-6-8.noarch.rpm
   ```

   Verify EPEL repo is enabled:

   ```
   # yum repolist
   # yum list ganglia*
   ```

   NOTE: You may need to update /etc/yum.repos.d/epel.repo to use http instead of https or disable ssl verification.

   ```
   [epel]
   sslverify=false
   ```

2. Install Ganglia

   a. On the host you have chosen to be the Ganglia server, install the Ganglia server RPMs:

      ```
      yum –y install ganglia ganglia-gmond ganglia-
      gmetad ganglia-web
      ```

   b. On each host in the cluster, install the client RPMs:

      ```
      yum –y install ganglia-gmond
      ```

   Note: verify packages were installed:

   ```
   yum list installed ganglia*
   ```

3. Modify Configuration Files

Ganglia Server Node

a. Edit `/etc/ganglia/gmetad.conf`

“`data_source`” entry - change to data_source “nbcu-phd-ad-sales-cluster” 10 <PCC IP Address>
“`gridname`” entry - change to gridname “nbcu-phd- ad-sales-cluster”

b. Edit /etc/ganglia/gmond.conf

```
/*
* The cluster attributes specified will be used as part of
the <CLUSTER>
* tag that will wrap all hosts collected by this instance.
*/
cluster {
     name = "nbcu-phd-xxxx"
     owner = “NBCU"
     latlong = "unspecified"
     url = "unspecified"
}
udp_send_channel {
     #mcast_join = 239.2.11.71
     host = <PCC IP Address>
     port = 8649
     ttl = 1
}

/* You can specify as many udp_recv_channels as you like as
well. */
udp_recv_channel {
     #mcast_join = 239.2.11.71
     port = 8649
     #bind = 239.2.11.71
}
/* You can specify as many tcp_accept_channels as you like
to share an xml description of the state of the cluster */
tcp_accept_channel {
     port = 8649
}
```

c. Edit /etc/httpd/conf.d/ganglia.conf

```
#
# Ganglia monitoring system php web frontend
#
```

```
Alias /ganglia /usr/share/ganglia
<Location /ganglia>
     Order deny,allow
     Deny from all
     Allow from all
     Allow from 127.0.0.1
     Allow from ::1
     # Allow from .example.com
</Location>
```

d. Register gmond and gmetad with service

```
# chkconfig --add gmond
# chkconfig gmond on
# chkconfig --add gmetad
# chkconfig gmetad on
```

e. Confirm gmond and gmetad will start on reboot

```
# chkconfig --list gmetad
gmetad 0:off 1:off 2:on 3:on 4:on 5:on 6:off
# chkconfig --list gmond
gmond 0:off 1:off 2:on 3:on 4:on 5:on 6:off
```

f. Start gmond, gmetad, httpd

```
# service gmond start
# service gmetad start
# service httpd restart
```

<u>Ganglia Node – On Each PHD Cluster Node</u>

a. Edit /etc/ganglia/gmond.conf

```
/*
* The cluster attributes specified will be used as part of
the <CLUSTER>
* tag that will wrap all hosts collected by this instance.
*/
cluster {
     name = " nbcu-phd-xxxx"
     owner = "NBCU"
     latlong = "unspecified"
     url = "unspecified"
}
udp_send_channel {
     bind_hostname = yes # Highly recommended, soon to be
     default.
     #mcast_join = 239.2.11.71
     host = <PCC IP Address>
     port = 8649
     ttl = 1
}

/* You can specify as many udp_recv_channels as you like as
well. */
udp_recv_channel {
     #mcast_join = 239.2.11.71
     port = 8649
     #bind = 239.2.11.71
}
/* You can specify as many tcp_accept_channels as you like
to share an xml description of the state of the cluster */
tcp_accept_channel {
     port = 8649
}
```

b. Register gmond with service

```
# chkconfig --add gmond
# chkconfig gmond on
```

c. Confirm gmond will start on reboot

```
# chkconfig --list gmond
gmond 0:off 1:off 2:on 3:on 4:on 5:on 6:off
```

d. Start gmond

```
# service gmond start
```

4. Validate Installation

Use the following instructions to validate your installation:

a. Confirm that Ganglia is Running. Open a browser and navigate to the Ganglia page

```
http://<CommandCenterHost>/ganglia
```

5. Setup Hadoop Metrics for Ganglia

a. Stop PHD Cluster

```
$ icm_client stop -l <ClusterName>
```

b. Fetch the cluster configuration

```
$ icm_client fetch-configuration -l <ClusterName> -o
<LOCALDIR>
```

Cluster configuration will be stored in LOCALDIR.

c. Edit `hadoop-metrics2.properties`
   File is located in Cluster configuration directory – LOCALDIR.
   `LOCALDIR/hdfs/conf`

   Edit `hadoop-metrics.properties`
   `LOCALDIR/hbase/conf`

   Add the following to the end of the file:

```
#
# Below are for sending metrics to Ganglia
# for Ganglia 3.1 support
*.sink.ganglia.class=org.apache.hadoop.metrics2.sink.g
anglia.GangliaSink31
*.sink.ganglia.period=10
#default for supportsparse is false
*.sink.ganglia.supportsparse=true
*.sink.ganglia.slope=jvm.metrics.gcCount=zero,jvm.metr
ics.memHeapUsedM=both
*.sink.ganglia.dmax=jvm.metrics.threadsBlocked=70,jvm.
metrics.memHeapUsedM=40
namenode.sink.ganglia.servers=<PCC IP Address>:8649
datanode.sink.ganglia.servers=<PCC IP Address>:8649
jobtracker.sink.ganglia.servers=<PCC IP Address>:8649
tasktracker.sink.ganglia.servers=<PCC IP Address>:8649
maptask.sink.ganglia.servers=<PCC IP Address>:8649
reducetask.sink.ganglia.servers=<PCC IP Address>:8649
```

d. Reconfigure the Pivotal HD Cluster

```
$ icm_client reconfigure -l <ClusterName> -c
<LOCALDIR> -s –p
```

e. Start the Pivotal HD cluster

```
$ icm_client start -l <ClusterName>
```

f. Hadoop Metics will appear in Ganglia UI

3. Nagios

Pivotal HD and Hadoop are highly distributed system with a lot of redundancy built into the platform. However certain components are critical to the operation of Pivotal HD .

In terms of importance master nodes such as names nodes, resource managers HAWQ master node etc. are critical. Data nodes, YARN node managers etc. are a little lower in priority than master nodes.

Critical Nodes or Top Priority Nodes:

- Name Nodes (Active and Standby)
    - Disk Full
    - RAID Array Issues or Disk issues if no RAID
- Quorum Journal Nodes
- Resource Manager Node
- Secondary Name Node (if used)

Not that critical but still important:

- Data Nodes
- Node Manager, Application Master (Task Tracker)
- Slaves are expendable (up to a point, for the most part)

The Nagios plugins should be used to check status and alert on following events:

1) Host/Node Up/Down using ping check.
2) Disk Percent Full – configure percentage that you are comfortable with.
3) SWAP space – Greater than 85% full requires alert
4) CPU Utilization - Load based alarms such as CPU and Memory utilization of nodes are somewhat useless in Hadoop environment – 300% CPU load is not a necessarily bad thing in Hadoop batch oriented world. Hadoop nodes will be running at 90-100% for most of the time.
5) HDFS Capacity – Alert on if HDFS total capacity reaches a 80% or 90%.
6) Disk I/O performance – important to monitor since it impacts Map Reduce job performance.

The next section describes installation and configuration steps for Nagios on Pivotal HD cluster.

3.1 Nagios Installation and Configuration

All Nagios configuration files and plugins described here are available in Pivotal Field Engineering github at: https://github.com/Pivotal-Field-Engineering/phd-monitoring/nagios . Please checkout the project as Nagios configuration files and plugins in the github project will be needed to complete the tasks.

Nagios configuration files are in:
https://github.com/Pivotal-Field-Engineering/phd-monitoring/nagios/objects

Nagios plugins are in:
https://github.com/Pivotal-Field-Engineering/phd-monitoring/nagios/plugins

1. Install Nagios
    a. On the host that has been chosen to be the Nagios server, install the RPMs:

```
sudo yum -y install net-snmp net-snmp-utils php-pecl-json
sudo yum -y install wget httpd php net-snmp-perl perl-
Net-SNMP fping nagios nagios-plugins nagios-plugins-all
nagios-plugins-nrpe nrpe nagios-www

sudo yum -y install nagios nagios-plugins-all nagios-
plugins-nrpe nrpe
```

    b. Install NRPE (Nagios Remote Plugin Executor) and SNMP on all nodes that will be monitored by Nagios server for status and alerts:

```
yum -y install net-snmp net-snmp-utils php-pecl-json

yum –y install nagios-plugins nagios-plugins-nrpe
nrpe
```

2. Create Nagios Directories

```
sudo mkdir /var/nagios /var/nagios/rw /var/log/nagios
/var/log/nagios/spool/checkresults /var/run/nagios
```

   Create nagios user and group if they don't exist, this should have been created by install:

```
sudo groupadd nagios
sudo useradd nagios –g nagios
```

   Note: If nagios already exists, it is by default disabled for interactive shell. Use following methods to login:

```
su nagios -s /bin/bash
or
sudo –u nagios <command>

sudo chown -R nagios:nagios /var/nagios /var/nagios/rw
/var/log/nagios /var/log/nagios/spool/checkresults
/var/run/nagios
```

3. Set the Nagios Admin Password
    a. Choose a Nagios administrator password, for example, "admin".
    b. Set the password. Use the following command:

```
htpasswd -c -b /etc/nagios/htpasswd.users nagiosadmin
admin
```

4. Set the Nagios Admin Email Contact Address
   a. Edit `/etc/nagios/objects/contacts.cfg` and change the nagios@localhost value to the admin email address so it can receive alerts.

5. If you want to remotely view the Nagios web UI, you may need to modify the "Allow from" line in the `/etc/httpd/conf.d/nagios.conf`.

6. Install Nagios configuration files
   a. Copy Nagios configuration files for host groups, service groups, services, commands and other files to `/etc/nagios/objects/.`
      The files are in github in nagios/objects directory.

      ```
      cp phd-monitoring/nagios/objects >/objects
      /etc/nagios/objects/
      ```

   b. Copy Nagios hadoop plugins to `/usr/lib64/nagios/plugins/:`
      The files are in github in nagios/plugins directory.

      ```
      cp phd-monitoring/nagios/plugins
      /usr/lib64/nagios/plugins/
      ```

      Make sure shell and perl have execute permission:
      ```
      chmod 755 *.sh *.pl
      ```

7. Register the Hadoop Configuration Files
   Edit `/etc/nagios/nagios.cfg.`
   a. In the section OBJECT CONFIGURATION FILE(S), add the following:
      ```
      # Definitions for monitoring Hadoop servers &
      services
      cfg_file=/etc/nagios/objects/hadoop-hosts.cfg
      cfg_file=/etc/nagios/objects/hadoop-hostgroups.cfg
      cfg_file=/etc/nagios/objects/hadoop-services.cfg
      cfg_file=/etc/nagios/objects/hadoop-commands.cfg
      cfg_file=/etc/nagios/objects/hadoop-
      servicegroups.cfg
      ```

   b. Change the command-file directive to `/var/nagios/rw/nagios.cmd`:
      ```
      command_file=/var/nagios/rw/nagios.cmd
      ```

8. Set Hosts

a. Edit `/etc/nagios/objects/hadoop-hosts.cfg` and create a "define host { … }" entry for each host in your cluster using the following format:

```
define host {
        alias @HOST@
        host_name @HOST@
        use linux-server
        address @HOST@
        check_interval 0.25
        retry_interval 0.25
        max_check_attempts 4
        notifications_enabled 1
        first_notification_delay 0 # Send notification soon after
change in the hard state
       notification_interval 0 # Send the notification once
        notification_options d,u,r
   }
```

b. Replace the "`@HOST@`" with the hostname.

9. Set Host Groups
   a. Open `/etc/nagios/objects/hadoop-hostgroups.cfg` with a text editor.
   b. Create host groups based on all the hosts and services you have installed in your cluster. Each host group entry should follow this format:

```
define hostgroup {
        hostgroup_name  @NAME@
        alias           @ALIAS@
        members         @MEMBERS@
}
```

10. Set Services
    a. Open /etc/nagios/objects/hadoop-services.cfg with a text editor.

       This file contains service definitions for the following services: Ganglia, HBase (Master and Region), ZooKeeper, Hive, Templeton and Oozie

    b. Remove any services definitions for services you have not installed.

    c. Replace the parameter @NAGIOS_BIN@ and @STATUS_DAT@ parameters based on the operating system.

```
@STATUS_DAT@ = /var/log/nagios/status.dat
@NAGIOS_BIN@ = /usr/sbin/nagios
```

   d. If you have installed Hive or Oozie services, replace the parameter `@JAVA_HOME@` with the path to the Java home. For example, `/usr/java/default`.

11. Set Status Dat File
    a. Open `/etc/nagios/objects/hadoop-commands.cfg` with a text editor.
    b. Replace the `@STATUS_DAT@` parameter with the location of the Nagios status file. The file is located:

```
/var/log/nagios/status.dat
```

12. On each remote nodes (usually namenodes, journal, datanodes, resource manager, node managers, hive, hbase, and hawq nodes) and Nagios server node configure NRPE:

    NOTE: (1) NRPE commands must be defined on the Nagios server using `check_nrpe` plugin and `check_*` commands configured in `nrpe.cfg` and (2) Nagios services must be defined on the Nagios server using NRPE commands.

    a. Add NRPE commands to monitor load, disk, disk I/O, swap in `/etc/nagios/nrpe.cfg`

```
command[check_load]=/usr/lib64/nagios/plugins/check_
load -w 15,10,5 -c 30,25,20
command[check_hda1]=/usr/lib64/nagios/plugins/check_
disk -w 20% -c 10% -p /dev/hda1
command[check_swap]=/usr/lib64/nagios/plugins/check_
swap -w 20 -c 10
# optional, enable if you want to monitor
command[check_users]=/usr/lib64/nagios/plugins/check
_users -w 5 -c 10
command[check_zombie_procs]=/usr/lib64/nagios/plugin
s/check_procs -w 5 -c 10 -s Z
command[check_total_procs]=/usr/lib64/nagios/plugins
/check_procs -w 150 -c 200
command[check_disk_iostat]=/usr/lib64/nagios/plugins
/check_iostat –d sda, sda1, sda2 –c -w 150 -c 200
```

    b. Edit `/etc/nagios/nrpe.cfg` and add Nagios Server IP addresses:

```
allowed_hosts=127.0.0.1,<Nagios Server IP1>, <Nagios
Server IP2>
```

Restart.

13. Configure `snmpd` on each node, `snmp` is used by `check_cpu.pl` plugin.
    a. Move original snmpd.conf
    ```
    mv /etc/snmp/snmpd.conf /etc/snmpd.conf.orig
    ```

    b. Create a new snmpd.conf with content as below:
    ```
    ####
    # First, map the community name "public" into a
    "security name"

    #       sec.name  source          community
    com2sec notConfigUser  default       public
    #com2sec ConfigUser     default       hadoop
    com2sec AllUser        default       hadoop

    ####
    # Second, map the security name into a group name:

    #       groupName      securityModel securityName
    group   notConfigGroup v1           notConfigUser
    group   notConfigGroup v2c          notConfigUser
    group  ConfigGroup    v2c          ConfigUser
    group   AllGroup       v2c          AllUser

    ####
    # Third, create a view for us to let the group have
    rights to:

    # Make at least  snmpwalk -v 1 localhost -c public
    system fast again.
    #       name           incl/excl     subtree
    mask(optional)
    view    systemview    included   .1.3.6.1.2.1.1
    view    systemview    included   .1.3.6.1.2.1.25.1.1
    view    AllView       included   .1

    ####
    # Finally, grant the group read-only access to the
    systemview view.

    #       group          context sec.model sec.level
    prefix read   write  notif
    access  notConfigGroup ""      any       noauth
    exact  systemview none none
    # Give 'ConfigGroup' read access to objects in the
    view 'SystemView'
    ```

```
# Give 'AllGroup' read access to objects in the view
'AllView'
access  ConfigGroup    ""      any       noauth
exact   systemview      none    none
access  AllGroup       ""      any       noauth
exact   AllView         none    none

# We are using hadoop as the snmp community name and
all monitoring will come from the 192.xxx.xxx.0
network.
rocommunity hadoop 127.0.0.1/24
```

Make sure the network identified by ip address in rocommunity is correct for your environment such as 192.168.100.0/24.

c. Make sure `/etc/sysconfig/snmpd` has following line uncommented:
```
OPTIONS="-LS0-6d -Lf /dev/null -p
/var/run/snmpd.pid"
```

d. Start snmpd:
```
service snmpd start
```

e. Test snmpd is working ok:
```
snmpwalk –v 2c –c hadoop –o e 127.0.0.1
```

14. Validate Nagios Installation

a. Validate the installation.
```
sudo nagios -v /etc/nagios/nagios.cfg
```

b. Start the Nagios server and httpd.
```
sudo /etc/init.d/nagios start
sudo /etc/init.d/httpd restart
or
sudo service nagios start
sudo service httpd restart
```

c. Confirm the server is running.
```
sudo /etc/init.d/nagios status
or
sudo service nagios status
```

This should return:
```
nagios (pid #) is running...
```

d. Test Nagios Services

Run the following command:

```
php
/usr/lib64/nagios/plugins/check_hdfs_capacity.php -h
namenode_hostname -p 50070 -w 80% -c 90%
```

This should return:

```
OK: DFSUsedGB:<some#>, DFSTotalGB:<some#>
```

e. Start NRPE Service on each remote node that will be monitored by Nagios:

```
# sudo service nrpe start
# sudo service nrpe status
This should return:
nrpe (pid #) is running...
```

f. Test SNMP is working:

```
/usr/lib64/nagios/plugin/check_snmp -P 2c -o
.1.3.6.1.4.1.2021.10.1.3.1 -H localhost -C hadoop
```

Output should be:

```
SNMP OK - "1.32"

or

/usr/lib64/nagios/plugin/check_cpu.pl -H localhost -
C hadoop -w 200% -c 250%
```

Output should be:

```
2 CPU, average load 54.0% < 200% : OK
```

g. Test Nagios Access
   1. Browse to the Nagios server: `http://<nagios.server>/nagios`
   2. Login using the Nagios admin username (nagiosadmin) and password created earlier.
   3. Click on **hosts** to validate that all the hosts in the cluster are listed.
   4. Click on **services** to validate all the Hadoop services are listed for each host.

h. Test Nagios Alerts
   1. Login to one of your cluster DataNodes.
   2. Stop the Node Manager service.
      ```
      su -l gpadmin -c "/usr/lib/gphd/hadoop-
      yarn/bin/yarn --config /etc/gphd/hadoop/conf stop
      nodemanager"
      ```
   3. Validate that you received an alert at the admin email address and that you have critical state showing on the console.

4. Start the Node Manager service.
   ```
   su -l gpadmin -c "/usr/lib/gphd/hadoop-
   yarn/bin/yarn --config /etc/gphd/hadoop/conf
   start nodemanager"
   ```
5. Validate that you received an alert at the admin email address and that critical state is cleared on the console.

4. Appendix

4.1 JMX Integration

Hadoop's NameNode and Resource Manager expose interesting metrics and statistics over the JMX. Hadoop comes with built in JMX and it is available via JMXJsonServlet.

HDFS Node Name:
http://localhost:50070/jmx

```
curl -i http://localhost:50070/jmx
```

You can fetch only a particular key with the *qry* parameter:

```
curl -i
http://localhost:50070/jmx?qry=Hadoop:service=NameNode,name=NameN
odeInfo
```

HDFS Resource Manager:
http://localhost:8088/jmx
http://localhost:8088/jmx?qry=Hadoop:service=ResourceManager,name=MetricsSystem,sub=Stats
http://localhost:8088/jmx?qry=Hadoop:service=ResourceManager,name=RpcDetailedActivityForPort8031
http://localhost:8088/jmx?qry=Hadoop:service=ResourceManager,name=ClusterMetrics
http://localhost:8088/jmx?qry=Hadoop:service=ResourceManager,name=JvmMetrics

Web Service/JSON:
Node Info: http://localhost:8042/ws/v1/node/info
Cluster Info: http://pivhdsne:8088/ws/v1/cluster

4.2 Nagios Plugins

4.2.1 Host, OS Level Alerts

Host Down Alert:
This alert is configured for all nodes in the Hadoop cluster (Hadoop master and slave nodes) as well as the Nagios and Ganglia monitoring servers. By default, it uses the Nagios plugin `check_ping` to find the average round trip response (RTT) time and the packet loss percentage by pinging each cluster node.

The `hadoop-services.cfg` file does not define this alert explicitly. Instead, this alert is defined as a part of the generic host definition in the `templates.cfg` file using the `check-host-alive` command defined in `command.cfg`.

NRPE check_ping – connection stats – find details by executing command:
`check_ping --help`

CPU utilization alert:
Use this alert if the percent of CPU utilization on the master host exceeds the configured critical threshold. This alert uses the Nagios `check_snmp_load` plug-in.

NRPE check_load:
https://www.monitoring-plugins.org/doc/man/check_load.html
http://www.linuxsysadmintutorials.com/setup-a-nagios-nrpe-to-monitor-the-load-on-a-system/

Disk % Full Alert:
NRPE check_disk
https://www.monitoring-plugins.org/doc/man/check_disk.html

Disk I/O Alert:
Nagios plugin – check_iostat:
Latest: http://exchange.nagios.org/directory/Plugins/Operating-Systems/Linux/check_iostat--2D-I-2FO-statistics--2D-updated-2014/details

SWAP Space Alert:
NRPE check_swap

4.2.2 Pivotal HD Nagios Plugins

HDFS Service Alerts:
check_hdfs_capacity: This alert is triggered if the HDFS capacity utilization exceeds the configured critical threshold.

check_hdfs_blocks: This alert is triggered if the number of corrupt or missing blocks exceeds the configured critical threshold.

check_name_dir_status: This alert is triggered if the Name Node cannot write to one of its configured edit log directories.

Percent Data Node Down Alert: This alert is triggered if the number of down DataNodes in the cluster is greater than the configured critical threshold. It uses the `check_aggregate` plugin to aggregate the results of `Data node process down alert` checks.

Percent Data Nodes storage full alert**:** This alert is triggered if the percentage of Data Nodes in storage full condition exceeds the configured critical threshold. It uses the `check_aggregate` plugin, to aggregate the results of `DATANODE::Storage full alert` checks.

Data Node process down alert: This alert is triggered if the various individual DataNode processes cannot be established to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

Name Node process down alert**:** This alert is triggered if the Name Node process cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

Name Node RPC latency alert**:** This alert is triggered if the Name Node operations RPC latency exceeds the configured critical threshold. Typically an increase in the RPC processing time increases the RPC queue length, causing the average queue wait time to increase for Name Node operations. It uses the Nagios `check_rpcq_latency` plug-in.

Data Node storage full alert: This alert is triggered if storage capacity is full on the Data Nodes. All the local data partitions storing HDFS data are checked against the total capacity across all the partitions. It uses the `check_snmp_storage` plug-in.

YARN/MR Service Alerts:
Resource Manager process down alert**:** This alert is triggered if the Node Manager process cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

Resource Manager RPC latency alert**:** This alert is triggered if the Resource Manager operations RPC latency exceeds the configured critical threshold. Typically an increase in the RPC processing time increases the RPC queue length, causing the average queue wait time to increase for Resource Manager operations. This alert uses the Nagios `check_rpcq_latency` plugin.

Percent Resource Manager node manager (Task Trackers) down alert**:** This alert is triggered when the configured critical threshold of Node Manager hosts become inaccessible in a short time-window. It uses the `check_aggregate` plugin to aggregate the results of individual Resource Manager node process down alert checks.

Resource Manager node manager process down alert: This alert is triggered if the configured number of node manager processes cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

Hive Related Alerts:
Hive MetaStore Service Alerts:
Hive-Metastore process alert: This alert is triggered if the Hive Metastore process cannot be determined to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios check_tcp plug-in.

WebHCat status alert**:** This alert is triggered if the WebHCat service cannot be determined to be up and responding to client requests.

HBase Service Alerts:

HBase Master process down alert: This alert is triggered if the HBase master processes cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

Region Server process down alert**:** This alert is triggered if the Region Server processes cannot be confirmed to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

HBase percent region servers down alert**:** This alert is triggered if the configured percentage of Region Server processes cannot be determined to be up and listening on the network for the configured critical threshold. The default setting is 10% to produce a WARN alert and 30% to produce a CRITICAL alert. It uses the `check_aggregate` plugin to aggregate the results of `Region Server process down alert` checks.

Percent ZooKeeper servers down alert**:** This alert is triggered if the configured percentage of ZooKeeper servers in your HBase cluster cannot be determined to be up and listening on the network for the configured critical threshold, given in seconds. It uses the `check_aggregate` plugin to aggregate the results of `Zookeeper process down alert checks`.

Zookeeper process down alert**:** This alert is triggered if the ZooKeeper process cannot be determined to be up and listening on the network for the configured critical threshold, given in seconds. It uses the Nagios `check_tcp` plugin.

Oozie Service Alerts:

Oozie status alert: This alert is triggered if the Oozie service cannot be determined to be up and responding to client requests.

HAWQ Alerts:

The HAWQ alerts simply use check_tcp to check for HAWQ Master and HAWQ Segment Server processes being up.

The also generate host level CPU utilization alerts if it goes beyond configured warning or critical threshold.

4.3 Default Ports

HDFS Name Node: 8020
HDFS Web UI: 50070, 50090 (Secondary Name Node)
http://localhost:50070/dfshealth.html#tab-overview

HDFS Data Node:
HDFS Data Node HTTP: 50010

Journal Node: 8480, 8485 (Internal, IPC)

ResourceManager Web UI and REST APIs: 8088
http://localhost:8088/cluster

Node Manager Web UI and REST APIs: 8042
http://pivhdsne:8042/node

Job History Web UI and REST APIs: 19888
http://pivhdsne:19888/jobhistory

HBase Master: 6000 (IPC)
HBase Master Web UI: 60010
Default port: 60010 (upto HBase 0.98), 16010 after 0.98.
http://localhost:60010/master-status

HBase RegionServer: 60020 (IPC)
HBase RegionServer Web UI: 60030

http://pivhdsne:60030/rs-status

Hive MetaStore: 9083
HCat Web Server URL: http://*yourserver*/templeton/v1/*resource*

Hive Server 2: 10000
Hive Web UI:
http://localhost:9999/hwi

Oozie Server: 11000
http://pivhdsne:11000/oozie/

HAWQ Master Port: 5432

HAWQ Segment Server: 40000 (base port)

Ganglia Ports:
Gmond: 8649
Gmetad: 8651, 8652