System Monitoring Using NAGIOS, Cacti, and Prism.

Thomas Davis, *NERSC and* **David Skinner**, *NERSC*

ABSTRACT: *NERSC uses three Opensource projects and one internal project to provide systems fault and performance monitoring. These three projects are NAGIOS, Cacti, and Prism.*

1. Introduction

Nagios , Cacti, and Prism are part of the core Operational Monitoring strategy at NERSC.

2. What is Nagios?

Nagios is a fault monitoring software package with plugin support. The plugin support allows one to create methods of monitoring that are specific to a system, or business/contract logic.

2.1 Where/How do I get it?

You can download it from http://www.nagios.org/. Some Linux distributions (ie, Fedora) already have it in the repositories, allowing a binary install.

2.2 How does it work?

By using 'plugins', it can monitor, notify, and acknowledge when faults occur. Can return 3 possible exit states – OK, WARNING, and CRITICAL. The plugin can also return a text message, stating what the condition means.

2.2 What can I do with it?

You can watch for network, process, hardware and software faults. You can also collect data for processing by other utilities.

Notifications can be sent based on date, time, or severity.

3. What is Cacti?

Cacti is performance monitoring tool based on a LAMP stack (Linux/Apache/MySQL/PHP) and RRD (Round Robin Database). It can collect, manage and display graphs of collected data.

3.1 Where/How do I get Cacti?

Cacti is available from http://cacti.net. Some distributions (ie, Fedora) also supply a version in their repositories. However, and important architecture plugin feature of cacti has to be patched in.

3.2 How does Cacti work?

Cacti uses Round Robin Databases (RRD) and MySQL database technologies to store collected information. MySQL and PHP is used to provide a graphical, web based interface to the RRD databases. Rrd database technology was popularized in the widely known MRTG graphing project.

Cacti takes MRTG configuration to a whole new level – it provides a 'click-and-point' interface to RRD's. A user can easily define, graph, and track over time many different devices without editing a configuration file using an editor.

3.3 What can I do with it?

Cacti is extendable, and user driven. The limits are

4. What is PRISM?

Prism is a project that collects data from varios NAGIOS systems using RDF, and then displays the results in an aggregated form.

4.1 How do I get PRISM?

4.2 How does it work?

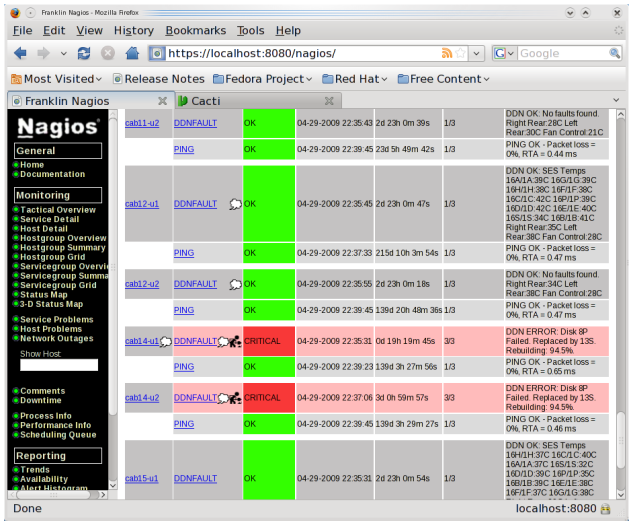
Using RDF/XML/RSS feeds from nagios, it collects, processes, and creates a color coded, simplified display of system status, allowing quick, easy visual notification of faults.

4.2 What can I do with it?

Prism allows the grouping of several NAGIOS (in NERSC's case, 9 displays) into one display. This allows 'at-a-glance' status checks.

5. Examples of Nagios/Cacti/PRISM usage

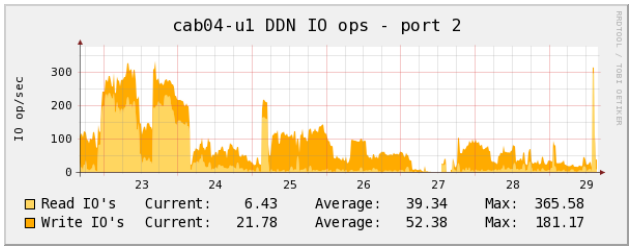
5.1 Nagios fault monitoring of DDN controllers and drives.



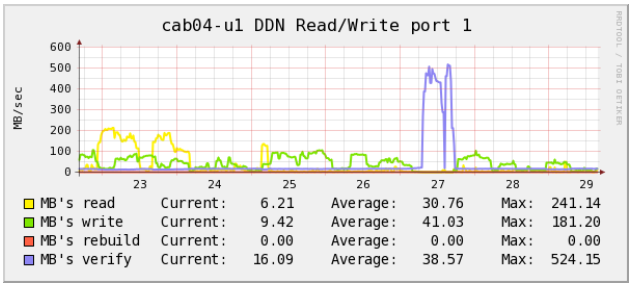
A snapshot from nagios, showing the a DDN drive fault. This information is collected using telnet, driven by a Perl/Expect script. The script is capable of detecting system faults, system temperatures, and other errors. This plugin makes finding and repairing DDN faults easier.

5.2 Cacti data collection, processing and graphing of DDN controller and drive performance.

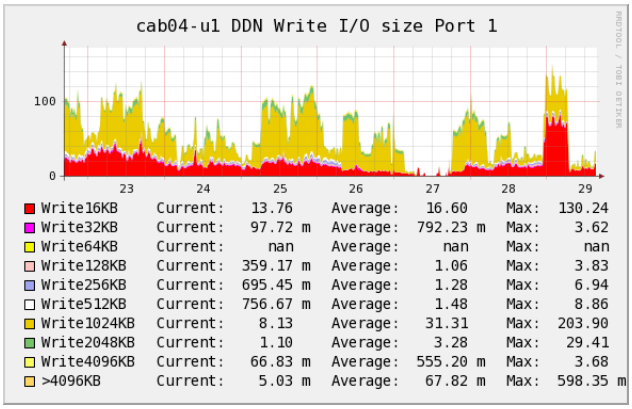
Cacti is used to collect I/O statistics from the controllers. Areas of data collection are I/O operation statistics, I/O read/write/verify/rebuild data rates, and I/O block sizes, and drive temperatures. For each controller, a graph for each port, plus a graph for total is generated. The data is collected using either the DDN api or telnet/command line scraping. Examples of several different graphs are:



DDN I/O operations, collected from a DDN controller port.



DDN Read/Write/Verify/Rebuild data rates graph



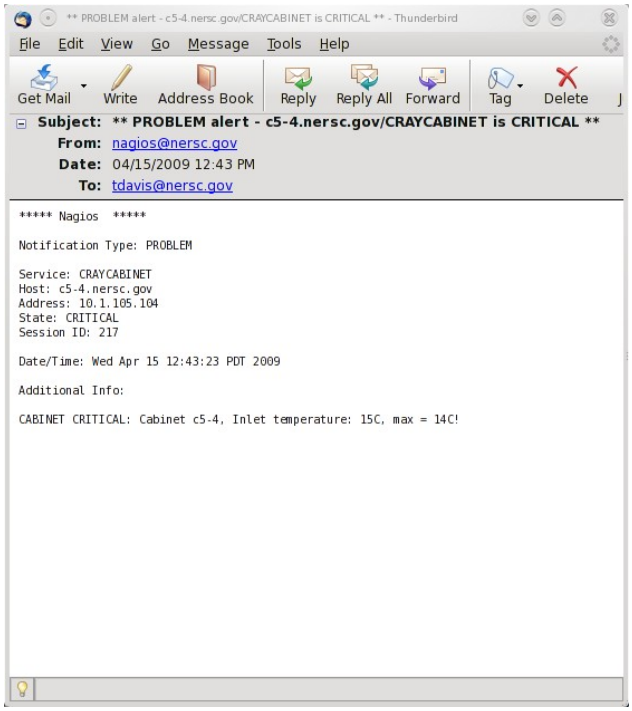
DDN Write I/O Size, histogram style.

5.3 Thermal monitoring of a Cray XT4 system.

Several plugins have been created to monitor the thermal environment in the machine room. The nodes monitored for temperature are Cray cabinets, air handling units (AHU), and DDN controllers and drives.

The system has a set point that is used to send a WARNING email message when the temperature rises past a certain point, and a CRITICAL email message at a higher point. These two points are chosen to allow operations/staff to time to react and either fix the temperature problem or do a graceful shutdown. This is to prevent an automatic overtemp shutdown from occuring. The other goal of this monitoring is to help provide information on variances in the machine room floor temperatures, and provide long term information for future planning.

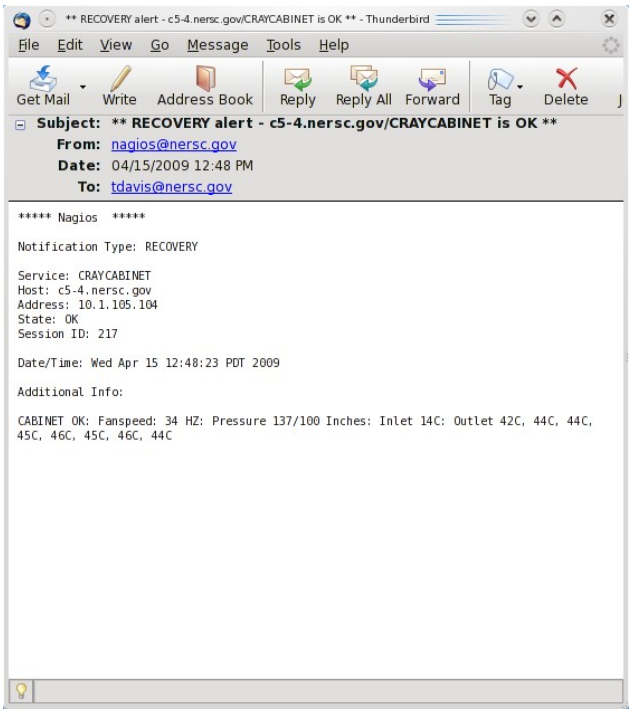
An example of a alert message from NAGIOS, which is monitoring the Cray cabinets using the XTWM service from the SMW:



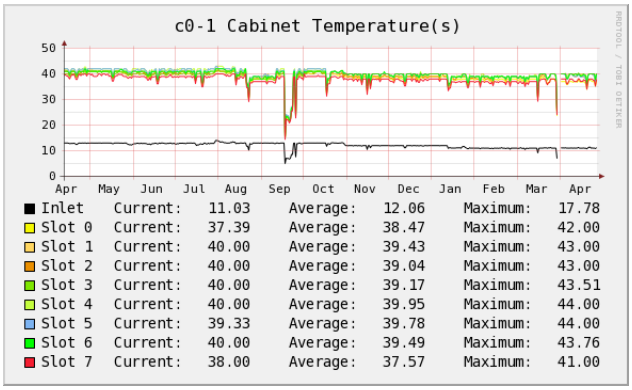
Notice in this email message, that it contains several bits of information:

1) Type of alert.
2) Source of alert
3) Date & Time of alert.
4) Temperature of system

When the temperature of the system returns back to normal,, nagios will send another email message notification of recovery:

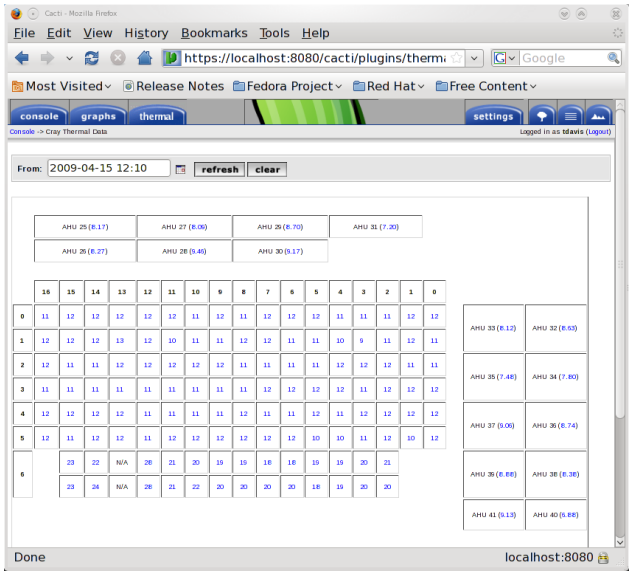


The system also collects the temperature information, and stores into a MySQL database. This database is queried by Cacti, to generate a graph of temperatures, like this:

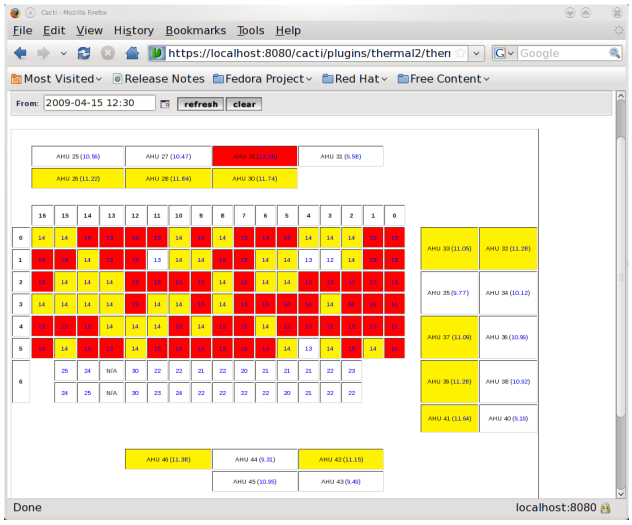


This graph shows the average temperature over a 1 year span, reduced to a 1 day average from a 5 minute average.

Nagios and Cacti are also combined, to generate a screen that allows 'at-a-glance' status thermal information of the system. This image shows that one can see there is no major temperature problems on the machine floor. However, one can see that certain parts of the floor are warmer than other parts.



The following is an example of a thermal event. Where a chiller had problems. This was caught, and operations/staff had a chance to correct the problem before the system reached an over-temperature shutdown.



5.4 Monitoring using XTCLI a Cray XT4 system.

This plugin was written to watch for down nodes, and then to log them to a database, generate an alert, and then also display the status in NAGIOS. Several problems had to be dealt with, starting with how do you monitor 9k+ nodes? The simplified answer is, you don't.

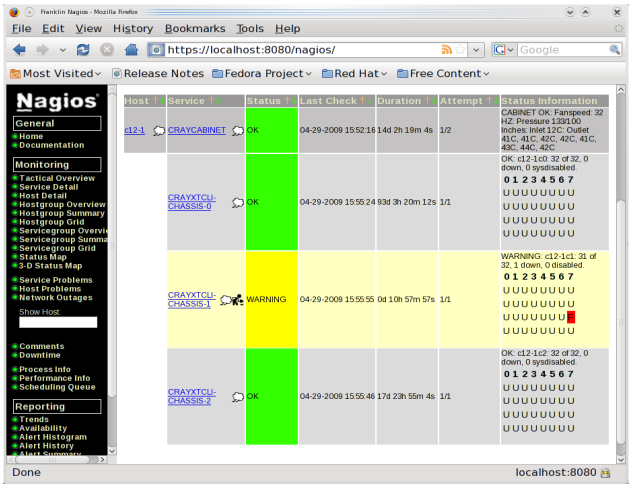
The technique that is used by this plug and appears to work is monitor on a chassis basis. What this means is the NAGIOS plugin looks at the XTCLI information generated on the SMW based on chassis, and faults on a node failure. The plugin works by using design feature of Nagios, in that you can go from an OK state, to an CRITICAL, then to an WARNING state, and only generate one email message. The plugin does the following:

1. A cron job is run on the SMW, that generates a xtcli file. This file is transfered to the NAGIOS monitoring box for processing.
2. A plugin process this file, to find information on a chassis basis.
3. If a node flag state changes, a CRITICAL alert is issued. The down node is recorded into a status file for that chassis.
4. On the next pass. The plugin will now emit a warning state, unless another node flag changes state.

All acknowledgements, recoveries, and warning states email notifications are supressed. Only CRITICAL state emails are sent for this plugin.

Using this method allows the operations center to acknowledge the failed node in NAGIOS. Again, this gets back to 'at-a-glance' state information – the Nagios tatical display will show non-acknowledged node failures.

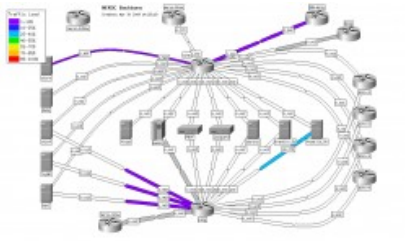
An example from NAGIOS:



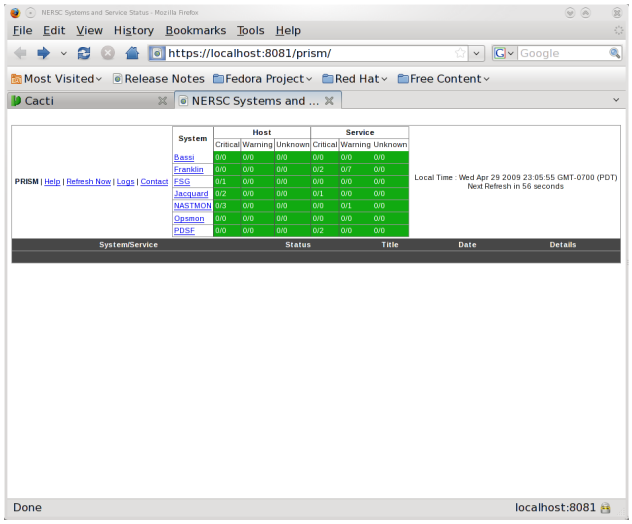
5.5 Network Weathermap plugin of Cacti.

One of the nagios/cacti systems collects data from several switches/routers in the network, and generates a network weather map. This weather map is useful to

figure out usage patterns – ie, is someone/something transfer large amounts of data from the NERSC Global Filesystem (NGF) to the mass storage system? If so, the network weathermap can show this. The weathermap has also been used in the past to show poorly utilized or mis-configured routes and links.



5.6 Prism



This image shows Prism in a completely acknowledged state.

7. Conclusion

Nagios, Cacti, and Prism are 3 tools that are used everyday at NERSC. They provide valuble insight into system's status in a standardized way the reduces staff training. These tools have provided data into areas of NERSC that was assumed to be correctly configured, allowing a more proactive instead of reactive management style of systems to be used.

Acknowledgments

The Cray staff at NERSC, for being patient with some of the questions I have asked in trying to get performance and fault monitoring in place. The operations staff, for their adopting of the Cacti thermal page for Franklin before I even completed it, and their patience as I broke and fixed it on several occasions.

About the Authors

Thomas Davis works for NERSC, E-Mail: tdavis@nersc.gov

David Skinner works for NERSC, E-Mail: dskinner@nersc.gov