University of Victoria
Faculty of Engineering
Fall 2009 Work Term Report

Evaluation of Nagios for Real-time Cloud Virtual Machine Monitoring

Department of Physics
University of Victoria
Victoria, BC

Michael Paterson
V00214440
Work Term 3
Software Engineering
mhp@uvic.ca

January 8, 2010

In partial fulfillment of the requirements of the
Bachelor of Software Engineering Degree

Supervisor's Approval: To be completed by Co-op Employer

I approve the release of this report to the University of Victoria for evaluation purposes only.
The report is to be considered (**select one**): ☐ NOT CONFIDENTIAL ☐ CONFIDENTIAL

Signature: _____ Position: _____ Date: _____

Name (print): _____ E-Mail: _____ Fax #: _____

If a report is deemed CONFIDENTIAL, a non-disclosure form signed by an evaluator will be faxed to the employer. The report will be destroyed following evaluation. If the report is NOT CONFIDENTIAL, it will be returned to the student following evaluation.

Contents

1 Report Specification **3**
1.1 Audience . . . 3
1.2 Prerequisites . . . 3
1.3 Purpose . . . 3

2 Introduction **3**

3 Display of Monitoring Information **4**
3.1 Proposed Solution . . . 4
3.2 Implementation of the Prototype . . . 4
3.3 Identified Problems . . . 5
3.3.1 Data Formatting . . . 5
3.3.2 Missing Data or Metrics . . . 6
3.3.3 Nagios Output Inconsistencies . . . 6
3.3.4 Nagios Scheduling . . . 6
3.4 Solutions to Identified Problems . . . 6
3.4.1 Data Formatting . . . 6
3.4.2 Missing Data . . . 7
3.4.3 Nagios Output Inconsistencies . . . 7
3.4.4 Nagios Scheduling . . . 8

4 Conclusion **8**

5 Future Work **8**

6 Acknowledgments **8**

7 Glossary **9**

List of Figures

1 Example of VM Booting on a cluster and changing status . . . 5
2 Nagios Service Check Timing[9] . . . 7
3 Nagios Service Check Timing - Short Interval[9] . . . 7

Evaluation of Nagios for Real-time Cloud Virtual Machine Monitoring

Michael Paterson
mhp@uvic.ca

January 8, 2010

Abstract

The department of Physics and Astronomy at the University of Victoria has started using Virtual Machines(VMs) in order to run many of their High Energy Physics(HEP) applications. The use of VMs mitigates the difficulties in setting up the correct execution environment for different HEP applications. Management of these VMs is being done using Nimbus, a set of open source tools that provide Infrastructure as a Service(IaaS) cloud computing. Previously, the need to develop monitoring capabilities for Nimbus was identified and a suite of plug-ins for the Nagios monitoring system were developed to provide that functionality. In order to visualize the data provided by the Nimbus Nagios Monitoring plug-ins in a user friendly way and in real time, a prototype web application was created. Areas in which the Nimbus Nagios Monitoring plug-ins are still lacking were identified and enhancements proposed. An alternate source for the information provided by the plug-ins is investigated.

1 Report Specification

1.1 Audience

This report is intended for members of the High Energy Physics Grid Computing Group at the University of Victoria, and future co-op students.

1.2 Prerequisites

A general understanding of virtualization, distributed computing, clusters, and web technologies is assumed.

1.3 Purpose

The report provides an overview of Nimbus, Nagios, and the Nimbus Nagios Monitoring project. With a discussion of using the data provided by the monitoring project to develop a web site to graphically display information about a cloud in a useful manner. The problems encountered using the data from the plug-ins, and proposed solutions are covered.

2 Introduction

The execution environment requirements of HEP applications are frequently dependent on the specific versions of the tools under which they were written. This makes maintaining dedicated execution environments unfeasible particularly for older applications that few researchers use. Using Virtual Machines(VMs) alleviates this problem by allowing the customized VM to be deployed on to any grid with a Virtual Machine Manager(VMM) such as Xen[2] that handles the creation, execution, and shutdown of VMs. Nimbus[3] is built on the Globus Toolkit[1], a middle-ware that contains components for creating and managing aspects of

a grid system, and uses a VMM in order to create and manage VMs on grids so users have a known computing environment to use without having to deal with different hardware systems that may be present on the grid. Nimbus can operate in two modes: resource pool mode, in which resources are requested directly from the Nimbus service, or pilot mode, where Nimbus works with a Local Resource Management System (LRMS) so that workspace requests as well as normal computing jobs can be accommodated by the system. The ability to virtualize operating systems on a grid can greatly increase the availability of computing resources to users.

In order to monitor the status of Nimbus sites, a set of monitoring tools were developed at the University of Victoria's Grid Computing Group by co-op students. The Nimbus Nagios Monitoring[4] project aims to provide monitoring and discovery services for Nimbus clouds by using the Nagios[7] monitoring framework and a suite of plug-ins. Information about a cloud is gathered by Nagios and collated into XML and published into the Globus MDS[5]. One use for the monitored data is to allow cross-cloud scheduling via the Cloud Scheduler[6], a virtual machine manager for batch job execution.

The ability to view a visual representation of the monitoring data to see changes to the cloud environment happen in real time is an ongoing project, this report details the attempt to use the Nimbus Nagios Monitoring plug-ins for this purpose.

3 Display of Monitoring Information

Now that more tools are available to monitor a Nimbus cloud, it is necessary to display that information in a coherent manner to a user. The raw XML generated by the monitoring code is not meant to be read by humans, so it must be transformed in such a way as to make it easy to see what is happening in the cloud. To accomplish the goal of displaying monitoring information that would be widely accessible it was decided a website would be best approach.

3.1 Proposed Solution

The way in which the monitoring plug-ins will be evaluated is by using a website with live updates of monitoring information in which the user would not have to refresh the page in order to see any changes and the page should only update the changed information instead of a complete reload. The data acquisition would focus on the Nimbus Nagios Monitoring plug-ins with the Cloud Scheduler's XMLRPC server being a possible secondary source of data. One of the stated requirements is that the information should be presented visually as much as possible.
After developing a simple prototype to aid in becoming familiar with the tools the requirements were expanded.
A major feature discussed was to be able to show the various clusters in the cloud or grid. When a new VM starts that it would appear and be connected to the cluster, then indicate when it had changed state from starting to running. see Figure 1 for an example of VM creation process. As a result of the state change feature the refresh time of the page would need to be approximately three to five seconds.

3.2 Implementation of the Prototype

The prototype was developed using a combination of XHTML, CSS, Java-script, and AJAX. Style and layout where done with XHTML and CSS. Java-script was used for parsing XML and altering tag content during a refresh. It was necessary to use AJAX in order to fulfill the requirement of being able to refresh the monitoring information without reloading the entire page. The prototype website was able to report and display all the information currently gathered by the plug-ins, and included simple bar graphs to show IP address usage. The development of the prototype brought to light several problems that would have to resolved before any further work could be done on the project.

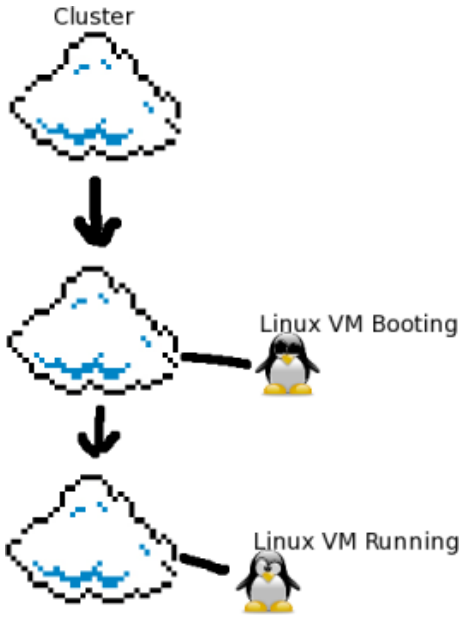

Figure 1: Example of VM Booting on a cluster and changing status

3.3 Identified Problems

The prototype helped expose problems in using the Nimbus Nagios Monitoring plug-ins that had not be apparent at the beginning of the project.

3.3.1 Data Formatting

The XML schema of the current Nimbus Nagios Monitoring plug-in is as follows:

```
<ROOT>
<RES LOC = "IP Address" TYPE = "Plug-in type">
<ENTRY ID = "identifier">
"value"
</ENTRY>
</RES>
</ROOT>
```

A single plug-in could produce multiple RES tags, and some plug-ins had multiple ENTRY tags. A separate plug-in collected all the RES entries from the Nagios performance output and wrote them as a single XML file that could be used by other processes. In order to determine what output described a particular cluster, worker node, or virtual machine, one had to rely on using the LOC and TYPE attributes. Certain TYPES only belonged to clusters, nodes, or VMs. It was necessary to pick specific types to check against to determine what the cluster environment looked like. Once it was known what the configuration of the cluster actually was, one had to go back through the data multiple times to extract and match up information contained in different types of RES tags in order to determine which resources were being used correctly. The nature of the XML format also made it extremely difficult to parse. Since most of the parsing functions were designed to select elements based off their tag name, it was necessary to select every RES tag then iterate over them all manually checking their LOC, TYPE and ID for the desired information. With the current XML format, correctly determining the IP address usage required $1 + 2 * n$ where n is the number of network pools, passes over every RES tag.

3.3.2 Missing Data or Metrics

There are various pieces of information that has not been included in the current set of output available from the monitoring plug-ins. These include things such as memory usage, VM lease times, VM ownership, VM state(Starting, Running). There is also a problem with combining related metrics due to the XML format having no natural groupings of related types of information.

3.3.3 Nagios Output Inconsistencies

The individual plug-ins always produced the expected output, barring any errors related to incorrect setup or access permissions, but the script responsible for gathering all the output together would often drop entries in it's XML that should not have been. The reason for this due to the way the script selects which information to output. The Nagios performance data is written to one large log file. The processing script defines a 'window' of time that it looks in for the most current output. if a service was delayed it may fall outside the current window and be missed or if the window is too large, it will be duplicated. This could cause the parsing code to breakdown when the TYPE entries being used to count the number of worker nodes went missing or got duplicated. Other missing entries would cause blank fields or misreported resource usage. The XML output was unpredictable in this manner, and would function without errors for long periods, breaking only briefly, or being broken for long periods. After identifying this issue a static copy of the XML data was used for development and manually modified to alter the cluster state for testing purposes.

3.3.4 Nagios Scheduling

Nagios attempts to create a balanced schedule for local and remote service checks during start up. Over time the actual schedule can change due to service checks with different intervals, execution times, failed checks and other events. Figure 2 shows the normal service check process. If the check interval is shortened too much, as in Figure 3, Nagios will have scheduling problems and be unable to keep up with processing results. The check intervals for services are expected to be in of a length in which a human can respond, as such Nagios uses a default interval of five minutes when first installed. The default interval can be altered and check times can be configured for individual services, but it is expected those checks will have a reasonable interval.

During early development the default interval of five minutes was used without significant problems. Once it was determined that information would have to be updated every three to five seconds instead of every five minutes, changes were made to the Nagios configuration to test this. With the check interval reduced Nagios began encountering scheduling problems. After running for a short time the Nagios process was consuming all CPU resources on the monitoring system and could not produce reliable output. This is a result of the Nagios daemon having to perform more work caused by the fast check times, and in the event of failed checks or status changes when a Nagios service is moving from a soft(recently changed) to hard(stable) state change. As the workload of the scheduling daemon increases the chance of a service missing its scheduled time and having to be rescheduled also increases. Having to reschedule services creates more work for the scheduling daemon until all CPU time is spent trying to schedule services.

3.4 Solutions to Identified Problems

Possible solutions and workarounds to the problems encountered during development. None of these solutions were implemented during this project, but may be a source of future work.

3.4.1 Data Formatting

The format of the XML could be redesigned in such a way that it would be possible to make better use of parsing tools as well as other XML technologies like XSL, XSLT to more easily extract meta-data from the results. This change would require the output from all plug-ins to be altered, and a major re-write of the

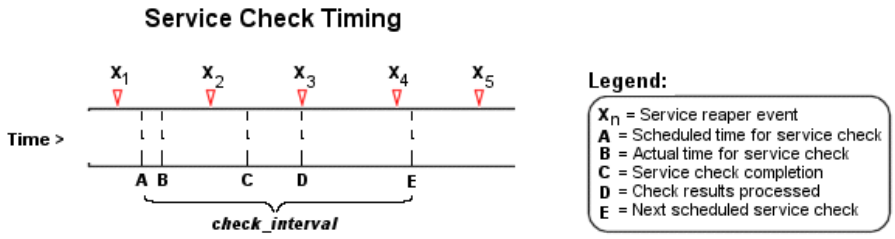

Figure 2: Nagios Service Check Timing[9]

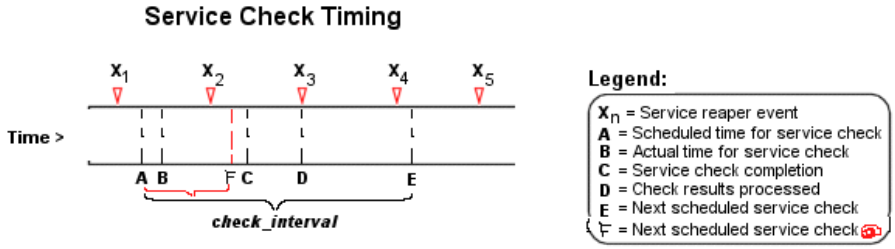

Figure 3: Nagios Service Check Timing - Short Interval[9]

plug-in data processing script that prepares the XML output. A preliminary design of the reformatted XML may be:

```
<Cluster>
   <Head Node Information>
   ...
   <Worker Nodes>
      <Work Node>
         <Work Node Information>
         ...
      </Work Node>
      ...
   </Worker Nodes>
</Cluster>
```

This type of format offers a far clearer picture of the information and allows for parsing tools to select specific tags or easily count the worker nodes present in a cluster. The current XML format makes such tasks difficult.

3.4.2 Missing Data

Creating additional options in the Nimbus Nagios Monitoring plug-ins to output previously missing data is straightforward. The design of the plug-ins make adding new options a simple task of creating a new method and the options to call it, then adding it the Nagios configuration.

3.4.3 Nagios Output Inconsistencies

There is no easy solution to the problem of XML inconsistency. Several possibilities exist. Persisting some of the information to ensure a complete snapshot was always available is one possibility. Any solutions would

require fine tuning how the script picks its output based on time stamps or finding an alternative way to extract the required information from the Nagios performance data output.

3.4.4 Nagios Scheduling

There is no solution to overcome the issues of how Nagios does scheduling in order to achieve the desired refresh times of three to five seconds, without causing serious problems to the monitoring machine. Due to this, further development of the prototype website was halted and other sources and methods to obtain system information were investigated. Use of Nagios is still suitable for monitoring aspects that do not require a high rate of updates.

4 Conclusion

The Nimbus Nagios Monitoring project is viable for providing periodic snapshots of the cluster state given further development to address some identified problems, but not suitable for monitoring the system in real-time. Although no longer feasible to continue development using the output from Nagios, this prototype has provided a valuable experience in several web technologies which will be of use in future work. The prototype has some useful code that can be reused in the next iteration of this project.

5 Future Work

An alternative source for information on the state of a cluster and virtual machines is the Cloud Scheduler. The Cloud Scheduler includes an XMLRPC server and there is a basic client available to retrieve cluster information. The server and client will need some additional development for outputting information regarding clusters and virtual machines as well as being able to output that information in either XML or JSON[10] strings. Research will have to be done to see if using JSON will offer an advantage over XML. Once the information has been made available from the Cloud Scheduler, work on the monitoring website can resume. The current prototype web site will have to be largely re-written, all the current XML parsing and functions that had be written to deal with the current XML format will be obsolete. The framework for making periodic refreshes can be reused and code for displaying data and text formatting may be migrated into the new site.

6 Acknowledgments

I would like to thank Dr. Randall Sobie for this work term opportunity. Thanks to Ian Gable for guidance on this work term. Additional thanks Ron Desmarais, Patrick Armstrong, and Duncan Penfold-Brown for technical help and advice.

7 Glossary

JSON Java-Script Object Notation

GT4 Globus Toolkit 4. The *de facto* grid middle ware.

GVW Globus Virtual Workspaces. Now known as Nimbus.

LRMS Local Resource Management System. Manages jobs on local clusters.

MDS Monitoring and Discovery System. Information services component of the Globus Toolkit and provides information about the available resources on the Grid and their status.

VM Virtual Machine, an instance of a machine(computer) running in software.

VMM Virtual Machine Monitor, used for managing virtual machines.

Xen Open-source VMM used by Nimbus.

XML eXtensible Mark-up Language.

XMLRPC XML Remote Procedure Call.

XSLT eXtensible Style-sheet Language Transform, for transforming XML between XML documents.

References

[1] Globus Toolkit http://www.globus.org/toolkit/

[2] Xen hypervisor - http://www.xen.org/

[3] Nimbus http://workspace.globus.org/index.html

[4] Nimbus Monitoring plug-ins for Nagios - http://github.com/nimbusproject/nimbus/tree/master/monitoring/nagios/

[5] Globus MDS http://www.globus.org/toolkit/mds/

[6] Cloud Scheduler http://github.com/hep-gc/cloud-scheduler/

[7] Nagios http://www.nagios.org/

[8] Nagios - Service Check Scheduling http://nagios.sourceforge.net/docs/2_0/checkscheduling.html

[9] Service Check Timing Image - http://nagios.sourceforge.net/docs/2_0/images/checktiming.png

[10] JSON - http://www.json.org/