Integrated RT-Nagios System at the BNL US ATLAS Tier 1 Computing Center.

Tomasz Wlodek, Carlos Gamboa, Zhenping Liu, Shigeki Misawa, Robert Petkus, Jason Smith, Tom Throwe, Yingzi Wu, Dantong Yu

Brokkhaven National Laboratory, Upton NY 11973, USA.

tomw@bnl.gov

Abstract. We present an integrated system for monitoring a heterogeneous computing facility, and for managing user problem reports.

1. Introduction

Managing a large number of heterogeneous grid servers, all of which have different service requirements, presents a great deal of challenges to system administrators. We describe herein a cost-effective, integrated operation framework, which monitors services, raises alarms with different severity levels, and tracks the facility's response to reported issues.

The system is based on open source components: RT (Request Tracking) tracks user requests, while Nagios performs facility monitoring. We will discuss the integration of these components.

2. The RT (Request Tracking System).

The RT system is open-source software provided by Best Practical [1]. It is written in Perl and uses an embedded MySQL database to store user requests (tickets).

Tickets can be submitted to RT via e-mail, or through a web application interface. All incoming e-mail requests must pass through spam and virus filters before being processed and assigned to problem categories, or queues. New problems are assigned unique numbers, by which they are later referenced in the system. As experts scan incoming tickets: when they see one that falls into their particular area of expertise, they take ownership of, or responsibility for, the issue until it is resolved. Separate tickets that relate to the same problem can be merged into a single ticket a manner transparent to user.

2.1. *GOC-Nagios interface*. The RT system at BNL is linked to the Open Science Grid (OSG) ticketing system. Once the OSG operators determine that a particular problem should be addressed by BNL staff, they can pass it to BNL's RT ticketing system, from which it can be further passed to one of the Tier-2 centers for resolution. This operation is transparent to users – both OSG and RT users communicate about the problem using their respective ticketing systems and related ticket numbers, and the coordination of communication between them is handled automatically.

3. Nagios monitoring system

Nagios [2] is an open-source monitoring system that recently became a *de facto* industry standard for monitoring computing resources. Nagios executes monitoring code , customized per service via plugins, and reports its probing results (OK, WARNING, CRITICAL, UNKNOWN) to a web interface.

In addition to code developed by the authors of Nagios ,the Nagios user community provides a vast library of probes that can be used for monitoring many standard services , such as TCP ports, databases, CPU and memory load, disk usage, and running processes. For our purposes, we have also written a large number of probes that monitor custom facility services.

The Nagios administrator controls the scheduling of execution of probes, as well as the frequency of user notifications, which can be sent via e-mail, cell phone, or pager. Once experts are notified of a problem, they can acknowledge the service, thus preventing any further, redundant notifications from being generated until the problem is resolved. In order to avoid false alarms, administrators can declare shutdowns in advance for particular services and groups of services.

In order to reduce the number of possible false alarms, Nagios provides a mechanism for declaring service dependencies. Declaring one group of services to be dependent on another prevents the generation of multiple, related alarms when the failure of one service causes the failure of another. This guarantees that only the service that causes the original failure will generate notifications, thereby reducing the number of notifications and subsequent confusion regarding related services.

4. Nagios –RT Interface

In addition to generating notifications to experts, Nagios probes can be configured to trigger the automated creation of RT tickets. This is done via RT's e-mail gateway: by default, in order to protect itself from SPAM, flooding, and mail loops, the RT system scans incoming mail messages and rejects those that appear to be sent by cron jobs or other automated services. However, an exception can be created for notifications that come from the Nagios server, which are treated like regular problem reports, and corresponding tickets are then created. We have created links between our Nagios and RT interfaces to track the RT tickets that have been generated by Nagios. When experts receive notification of a service error, they first acknowledge the problem by clicking a corresponding link on Nagios interface, and then move to the RT ticket page, on which they follow up on the resolution of the problem. Once the problem has been resolved, they close the corresponding RT ticket, thus creating a permanent record of both the problem and its solution, and then reschedule the next service check in order to remove the related alert from Nagios.

5. Conclusion

The described system provides a scalable solution that aids administrators in the commissioning of grid servers, the automation of error-prone, manual system configuration, and the leveraging of the existing ticket system for problem tracking. It allows BNL to operate its ATLAS Tier 1 facility in "twenty-four by seven" fashion with limited monitoring resources, and it meets the established service level agreements for WLCG grid middleware components, all of which have different class of service requirements.

References

[1]http://bestpractical.com/
[2]http://www.nagios.org/