Exploring Computational System Health Monitoring and Reporting Solutions

Jose Franco Baquera
jose5913@nmsu.edu

Ethan O'Dell
eofx7@mst.edu

Daniel Perry
dmp265@cornell.edu

LA-UR-19-27102

Problem Statement

High-Performance Computing (HPC) is an integral part of the scientific research done at Los Alamos National Laboratory (LANL). Consequently, it is imperative to evaluate different HPC node health monitoring systems in order to explore their viability, reliability, robustness, reusability, and job performance impact.

Background

A complete software stack overhaul is taking place and different health check tools needed to be analyzed. The following node monitoring tools were examined before this project went underway:

- Ganglia
- Nagios
- Zabbix
- Cluster Node Diagnostics (nodediag)
- LBNL Node Health Check (NHC)

When looking at Ganglia, Nagios, and Zabbix for the LANL cluster monitoring stack, it was discovered that these solutions do not scale well. Ganglia can only be configured for clusters of up to 2000 nodes[1] and Nagios has scalability issues that can only be mitigated to a small extent on supercomputers similar in scale of those at LANL. Zabbix, Nagios, and Ganglia all include their own state managers, which do not easily fit within the LANL software stack. When LANL creates a new cluster, it often needs custom lightweight software solutions[2]. This would be another reason why Ganglia, Nagios, and Zabbix are not practical solutions since these programs provide too many unneeded features and would not practically fit within the LANL software stack. NHC[3] and nodediag[4] are both lightweight and both tools have the benefit of already existing within LANL's software stack. This makes them the ideal candidates for comparison. NHC is run on LANL's Cray systems and nodediag is run on LANL's Perceus systems. However, neither are run periodically, instead they are run at boot and at job epilog.

Methodology

- NHC and nodediag were both installed on a ten compute node cluster with Slurm as the resource manager. Each tool was configured to function as a health check program executed by Slurm every five minutes (Fig. 1).
- Tests were created for both programs. These tests can be seen in Fig. 2.

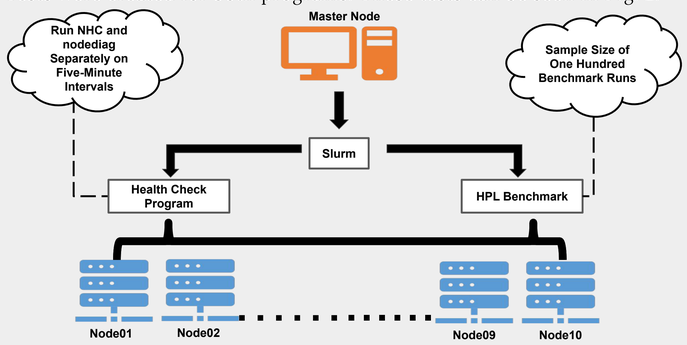

Figure 1: Using Slurm to run health check program every five minutes while running HPL benchmark.

- The High-Performance Linpack (HPL) Benchmark was run to simulate an average high-demand job.
- Qualitative observations about both programs were recorded and can be seen in Fig. 3 and Fig. 4.
- When testing the tools, benchmarks were run initially on unmodified software (i.e. "Out of the Box" testing).

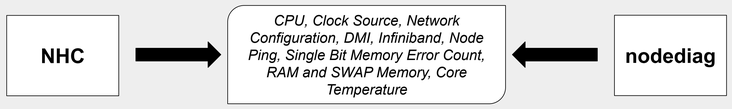

Figure 2: Tests run by NHC and nodediag on the compute nodes.

Results

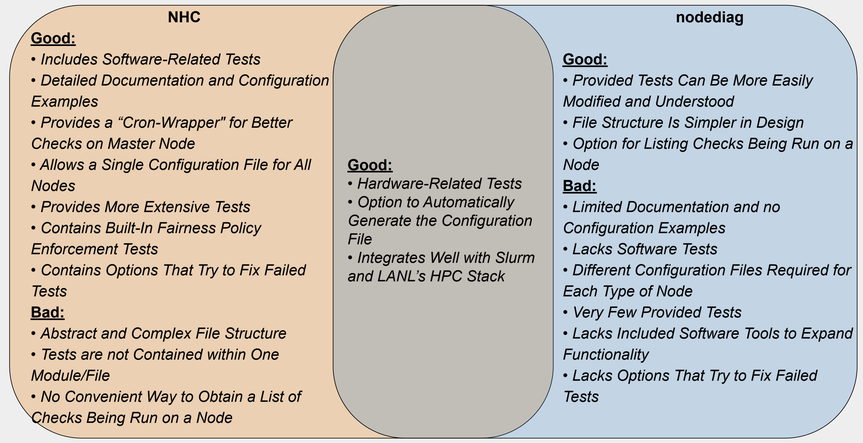

Figure 3: Pros and cons of nodediag vs. NHC.

	NHC	nodediag
Pre-Made Software Checks	• Command Status • Filesystem Checks • Firmware Checks • Unauthorized Users • Daemons and Processes • File/Directory Checks • Resource Consumption • Virtual Memory	None
Pre-Made Hardware Checks	• CPU Checks • Ethernet Checks • Ping/Socket Availability • SWAP Memory Checks • DMI Checks • Infiniband Checks • RAM Checks	• Clock Source Checks • DMI Checks • Hard Drive Perf. • PCI Card Checks • SWAP Memory Checks • CPU Checks • Infiniband Checks • Network Config. • RAM Checks

Figure 4: Comparison of checks provided with NHC and nodediag.

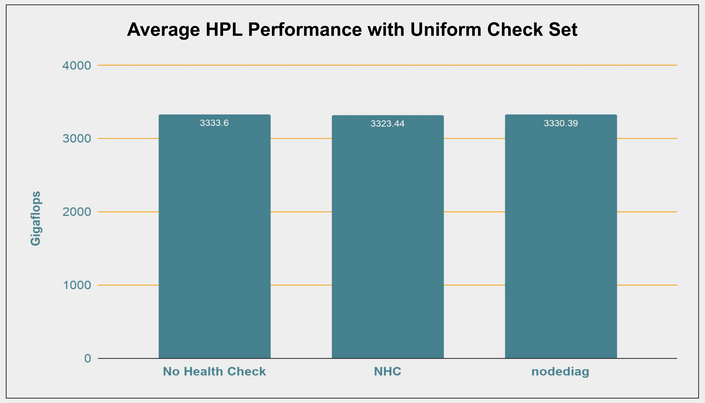

Figure 5: Average computing speeds while running HPL with NHC and nodediag with uniform checks.

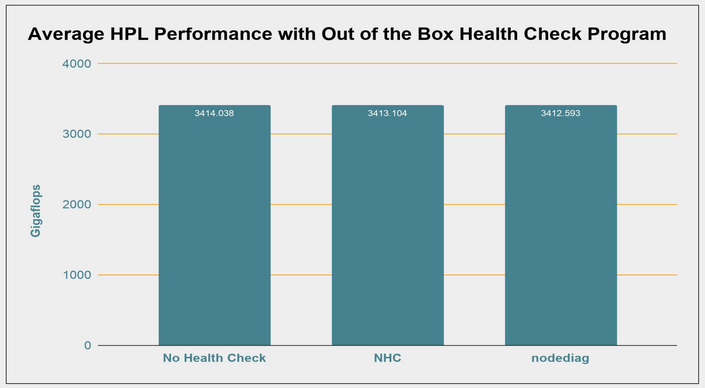

Figure 6: Average computing speeds while running HPL with NHC and nodediag with provided checks.

Conclusions

After a quantitative and qualitative analysis, it was discovered that both NHC and nodediag proved to be acceptable health monitoring programs. Both tools only put a negligible amount of load on the system out of the box (Fig. 6) and have their own distinct drawbacks and benefits (Fig. 3 and Fig. 4). From these results, it can be concluded that NHC:

- Offers a more robust set of provided health checks (i.e. "Out of the Box"), including both software and hardware checks.
- Allows one universal configuration file to be used for every node on a cluster while still allowing different classes of nodes to run different sets of checks.
- Includes a significantly larger amount (i.e. about 30 more pages) of useful documentation, along with more active support.
- Natively connects with slurm as a health check program capable of changing node state depending on node health.
- Provides much more flexible and modular methods for assigning checks and creating custom checks.

Therefore, it was concluded that if given a choice between NHC and nodediag, NHC is the more practical choice for the LANL software stack. Several issues emerged while conducting the analysis between NHC and nodediag. For example, it was learned that:

- Implementing original tests as shell scripts became difficult for students with our limited experience with bash.
- Some tests require root access to run, meaning that real-world implementation of such health check programs by non-administrators would impose a security risk.
- Both NHC and nodediag are, for the most part, structurally different. Consequently, it became tedious to compare and contrast them in a quantitative matter since it is difficult to make them homogeneous.
- Rigorous testing using the HPL benchmark caused hardware errors on several compute nodes. This is assumed to be because of the condition of the near end-of-life nodes used in the cluster.

Potential Future Work

- Designing front-end node resource fairness policies.
- Comparing NHC and nodediag using larger unclassified HPC clusters.
- Perform analysis of health check program degradation time.
- Create new testing metrics for monitoring node and cluster health.

Acknowledgements

We would like to acknowledge the time, feedback, and mentorship given to us by Kierstyn Brandt, Travis Cotton, and Graham Van Heule during the duration of this project. We would also like to thank the LANL Computer System, Cluster and Networking Summer Institute for the training and hardware which allowed us to complete this project.

References

[1] "Ganglia Monitoring System," *Ganglia Monitoring System RSS*, 01-Jan-2001. [Online]. Available: http://ganglia.sourceforge.net/.
[2] A. DeConinck *et al.*, "Design and Implementation of a Scalable Monitoring System on Trinity," in *Proc. Cray Users Group*, 2016.
[3] Lawrence Berkeley National Laboratory Node Health Check, "mej/nhc", https://github.com/mej/nhc
[4] Cluster Node Diagnostics, "chaos/nodediag", https://github.com/chaos/nodediag