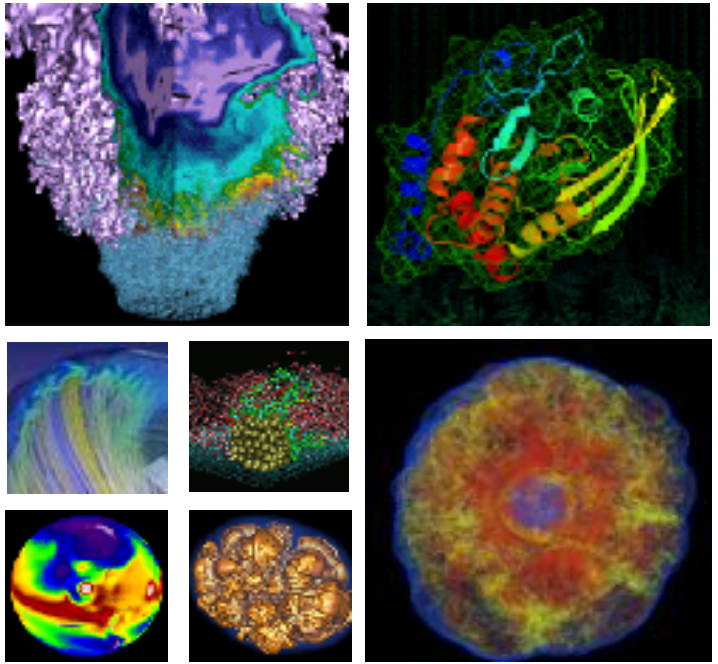
Utilizing Slurm and Passive Nagios Plugins for Scalable KNL Compute Node Monitoring



Tony Quan, Basil Lalli
twquan@lbl.gov, bdlalli@lbl.gov

SLUG 2017, NERSC - LBNL

U.S. DEPARTMENT OF ENERGY | Office of Science

About NERSC, Cori, Edison:

- 11 Nagios Core (4.x) servers
  - Aggregated by Thruk
  - Mostly NRPE and custom plugins, some NSCA
- Cori - ~12K nodes, ~9.6K Knight's Landing nodes
- Edison - ~5.5K nodes (Ivybridge)
- Nagios servers are split per-group/purpose (NGF, FSG, clusters, etc.)
- Notifications provided by Slack/email
- Heavy use of custom plugins

The Problem: Scale!

- Active/NRPE plugins serve us well in most groups, but cannot handle the scale of our latest systems.
  - It's not going to get better.
- Traditional remote plugins cause unnecessary intrusion on the internal clusters.
- With KNL nodes, reboots (for boot feature changes) are routine.
  - Easily can be thousands a day.
- Previous approaches are no longer viable.

Considerations:

- Easily maintainable.
- Integrate into our existing workflows, monitoring, and infrastructure.
  - K.I.S.S.
- Must be able to accommodate KNL reboot events gracefully.
- Per-node monitoring granularity for ticketing purposes.
- Must not spam notifications!!!
- Forward-thinking with regard to the scale of future systems and automation.

What Do We Really Need Here?

- Traditional plugins are unnecessary and/or meaningless:
  - PING, DISK, LOAD, etc.
  - NHC takes care of this.
- All data necessary to monitor compute nodes effectively can be provided from single locations en masse.
- We don't want/need to do *anything* 12,000 times… especially if other tools already do.

The Solution P1:

- `check_nodestatus`
  - `xtprocadmin, xtcli, sinfo, sacct` output is gathered from the SMW and pushed out to the nagios server via SSH/cron.
  - Plugin runs *on the nagios server*, correlates the output, and provides local, passive check results for per-node services.
  - Plugin maintains a state file and only returns changes each run.
  - “Master” service is updated every run, provides node counts, acts as parent service, and allows freshness checking.
  - Provides xtproc/xtcli/slurm state, job id/user, boot features, and slurm downtime/reason if applicable.

The Solution P2:

- Thruk provides basic regex searching and mass actions, allowing easy human interaction.
- An in-house notification digest collects and provides alerts at regular intervals. Guaranteed spam-free.
- RESTful communication with slurm (more on this later) informs nagios process of known reboot events and triggers downtimes.

Advantages:

- Extremely scalable, able to support our next system.
  - Tested up to 100K services, updated every two minutes.
- Absolute minimum intrusion into the cluster. We do not touch the compute nodes. Processing is done on the nagios end.
- Outgoing SSH only from the cluster avoids security concerns.
- No need for another nagios helper-daemon (e.g. NSCA).
- Relies only on pre-existing tools and Python.
- Maintainable.
- Able to detect node changes and generate nagios configuration files.
- Cronjobs can easily take advantage of HA pairs, data can be drawn from several locations, minimum points of failure.

Other Applications:

- "XT" nagios services across the internal non-compute nodes:
  - Uses same data, provides basic service node monitoring.
- Datawarp/Burstbuffer monitoring:
  - Sessions, instances, fragments, etc. are linked together and reported on the "lead" node of each instance.
- HPSS monitoring:
  - Tape events are communicated via a pair of non-sequential logs.
  - Plugin detects new entries and groups them into several services according to a set of regex rules. Supports volatile and traditional alerts.

The Future:

- (In progress) Livestatus queries will allow a daemon to group together node events by job, time, reason, etc. to provide automatic ticket creation/resolution. Can be extended to query multiple nagios servers and correlate more complex events.
- Enhancements to the RESTful mechanism may be able to replace the SSH/cron method of gathering data.

Thank You