LCI workshop
on high performance clustered computing

Linux Clusters Institute: Monitoring

Nathan Rini -- National Center for Atmospheric Research (NCAR)
nate@ucar.edu

Kyle Hutson – System Administrator for Kansas State University

kylehutson@ksu.edu

Why monitor?

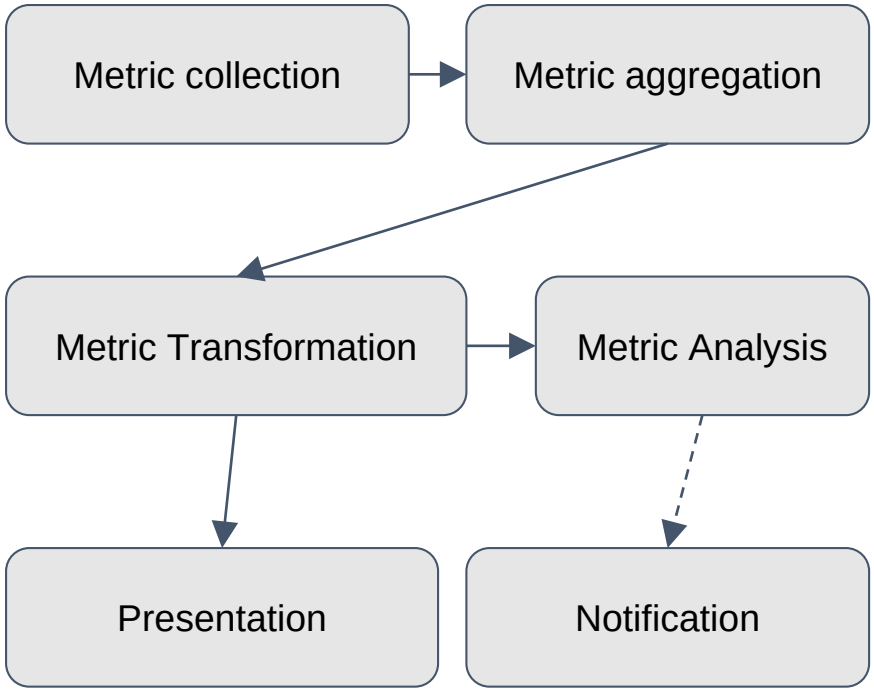
Service Level Agreement (SLA)

- Which services **must** be provided by you?
- Which services **must** be provided to you?

- Regulatory requirements
- Contractual requirements
- Business requirements

- Common Deliverables
  - Availability of services (Uptime)
  - mean time between failures (MTBF)
  - mean time to repair or mean time to recovery (MTTR)

Monitoring and Notification Basic Flow



Automated Gathering of Metrics

Evaluate Metrics against Requirements

Prepare and share metrics for Stakeholders

What to Collect (Metrics)

- Overall cluster health
  - Queue size
    - Jobs running
    - Jobs Queued
  - Overall network usage
  - Number of responding nodes
- Individual node health
  - Load average
  - Memory used
  - Network bandwidth
  - CPU usage
  - Temperature
- Storage
  - Capacity
  - Degraded status
  - Connectivity
- Security
  - Logs of everything
- Power status
- temperatures
  - Cold-aisle
  - Switches exhausts
  - CPU temperatures

Metric Collection

- Collection Tools (Common Tools)
  - Ganglia
  - Collectd
  - Perfmon
  - Performance Co-pilot (PCP)
  - Nagios
  - Unified Fabric Manager (UFM)
  - Cacti
  - Syslog
  - TACC stats
  - Scripts

Collection tools already exist to capture most metrics.

No single tool will do everything you need unless you write it yourself

Try to avoid re-inventing the wheel.

Metric Aggregation

- Aggregation Tools
  - Ganglia
  - Collectd
  - Performance Co-pilot (PCP)
  - Nagios
  - Unified Fabric Manager (UFM)
  - Cacti
  - Syslog
  - Round Robin Database (RRD)

Metrics need to be gathered from all over the cluster to a single place for analysis and storage

Most metrics should transfer over the Management Ethernet to avoid interference with Job performance in Low Latency interconnect

Metric Analysis and Transformation

- Monitoring Conundrum
  - Data is useless unless we do something with it
  - We can collect much more data than we can analyse
  - We generally won't know what data we need until we need it
    - Exception: Data we must provide for SLA requirements
  - Limited storage and processing capacity for metric analysis
    - This is less of an issue as drives get cheaper, but they also aren't getting much faster

Notification

- Notification Tools
  - Nagios
  - Icinga
  - Zenoss
  - Zabbix
  - PRTG
  - OpenNMS
  - OP5
  - Pandora FMS
  - Unified Fabric Manager (UFM)

Basic functionality of all alerts: Red, Yellow, Green

Most notification tools are forks or clones of Nagios

Notification tools can be passive or active in querying the status of the cluster

Notification

- Monitoring for known evil
  - Basis for all notifications
  - Only alert if something known bad happens
- Metrics -> Notifications
  - Most tools will require extensive configuration to be useful
  - Most tools will have a way to query metrics and create alerts
    - Some tools, such as Nagios, have this entire process built in
    - Others will have ways to bolt on this functionality
      - Nagios can query Ganglia
      - Ganglia can query Nagios

How should we get notified?

- Emergency
  - Fire and smoke exiting machine
- Urgent:
  - Email or text or phone call
  - Define this carefully
- Not-so urgent:
  - Web page updates
    - Especially helpful for historical data
  - Email (filtered)
  - End-user support requests

SLA based Alerts

- Alerts on Deliverables
  - Availability of services (Uptime)
    - Example: Alert if less than 98% of batch nodes are online
  - mean time between failures (MTBF)
    - Example: Send email report of time between failures
  - mean time to repair or mean time to recovery (MTTR)
    - Example: Alert if a down node does not come online after 4 hours down

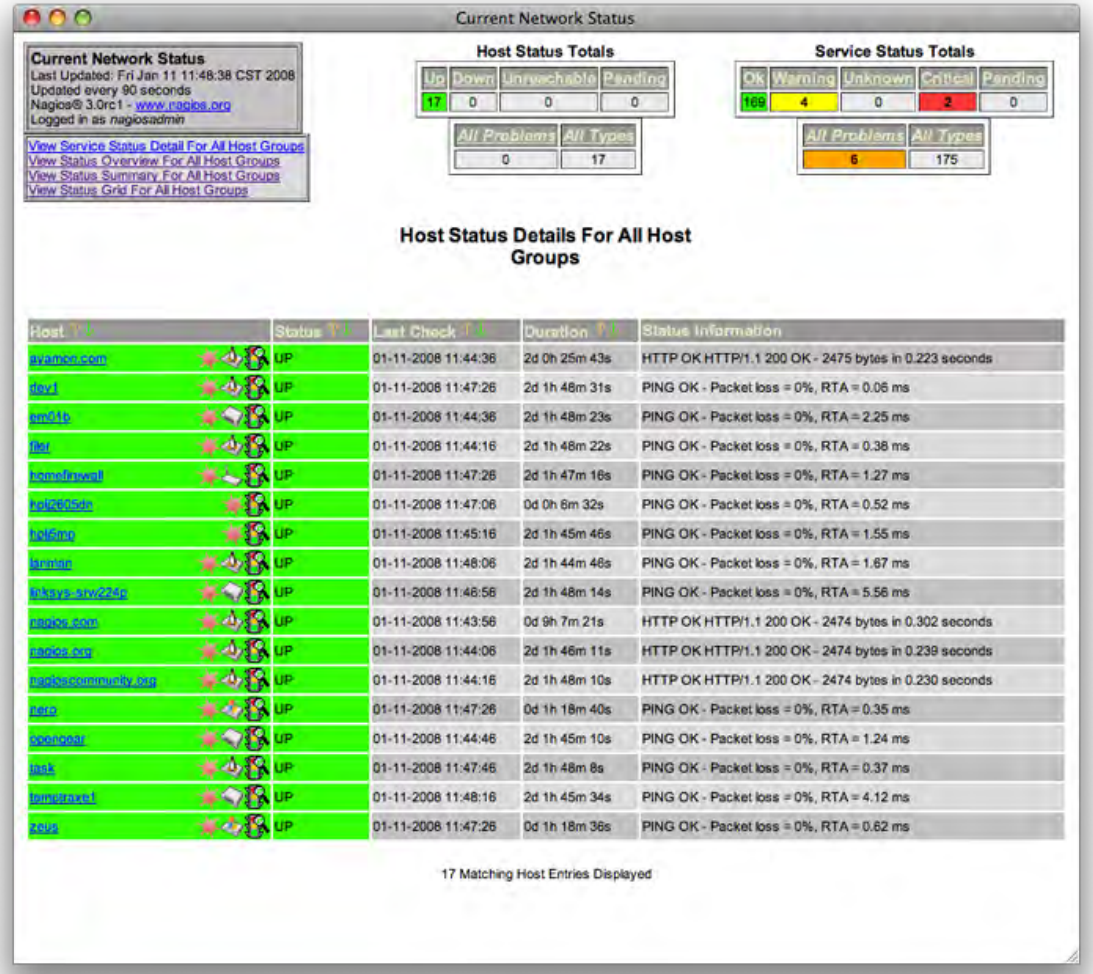
How often to alert?

- SLA requirements
  - If your SLA requires it, you may will need to get called off-hours or even on holidays
- You will quickly get a feel for this
  - **Too much info is often worse than too little info**
  - The “urgent” – continually
  - The “not-so-urgent” – anywhere from a few times per day to once per week
    - There’s nothing wrong with trial and error
    - Consider aggregated reports for ‘not-so-urgent’

Security Alerts

- Securing the cluster
  - Security alerts may need to go to specific groups or people instead of normal operations
  - Regulations and Security rules may apply to cluster which must be enforced
    - Compliance to Regulations: Sarbanes Oxley, Fisma, HIPAA, etc
  - Active response may need to be required such as blocking IPs
  - Security status updates
  - Alerts on security failures
    - sudo reports
    - Network login failures (e.g. fail2ban)
    - crontab failures
    - Logfile errors (customize to fit)

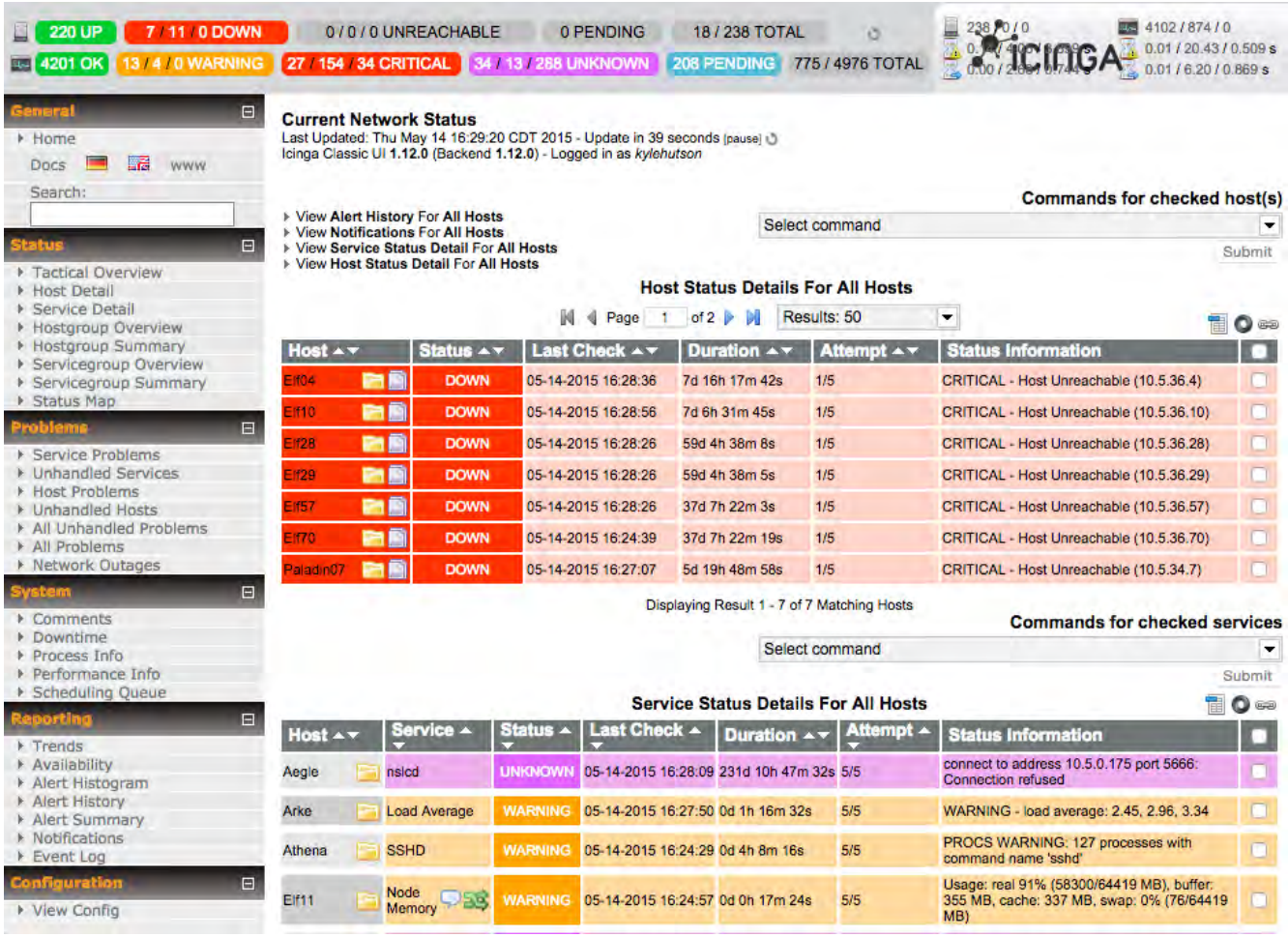
Example: Nagios



Example: Nagios

- Nagios/NRPE (Nagios Remote Plugin Executor)
  - Generic executable that runs “plugins”
    - Plugins can monitor just about anything you can think of monitoring
  - Even works with Windows
  - Nagios (http://www.nagios.org/) is by far the most common monitoring system

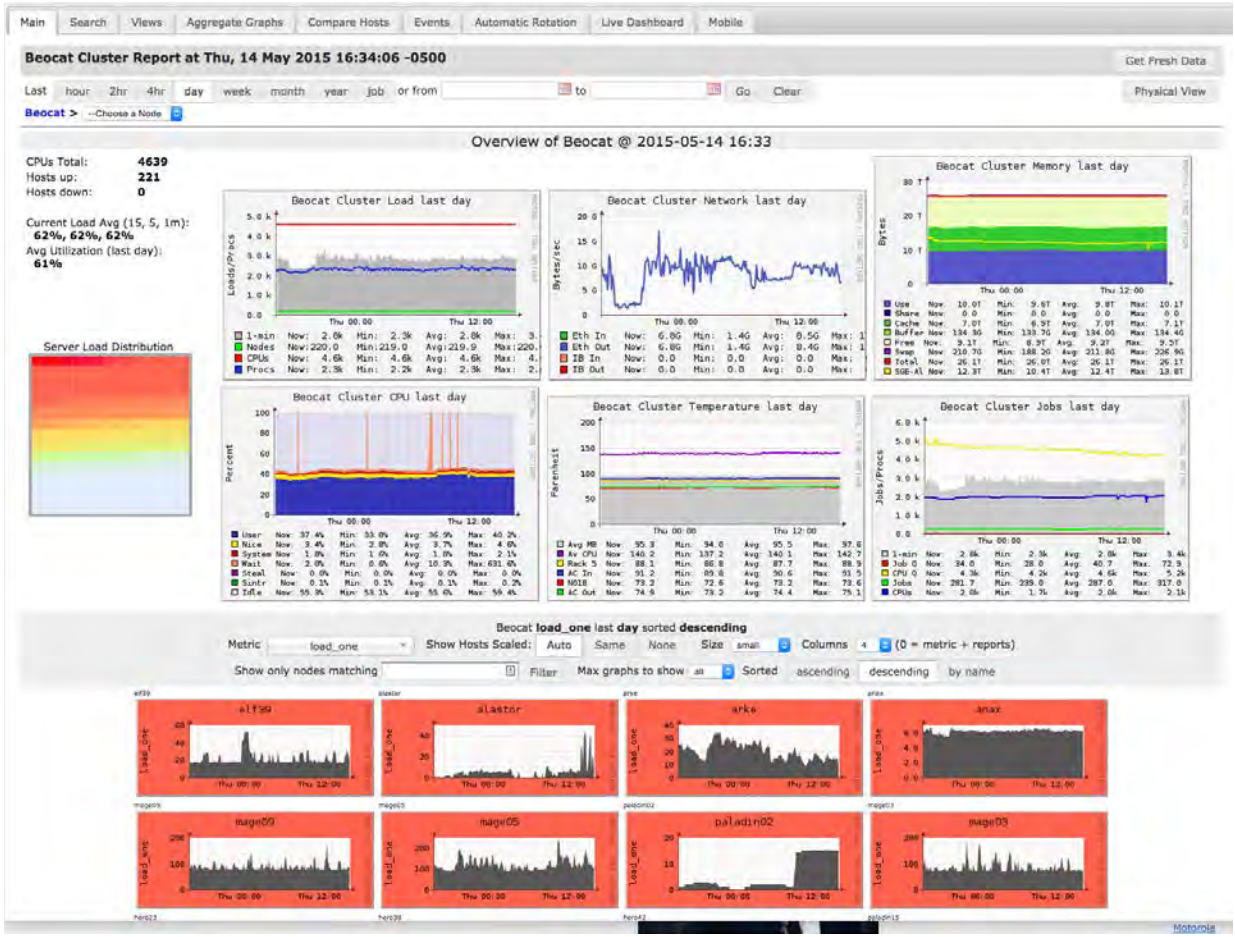
Example: Icinga



Example: Icinga

- Icinga (https://www.icinga.org/)
  - Can use NRPE
  - (New) version 2 has its own client
  - Uses database backend for history
  - Multi-threaded and multihomed

Example: Ganglia



Example: Ganglia

- Ganglia (http://ganglia.sourceforge.net/) - for historical and resource monitoring
  - Ours are public
  - RRD files give historical data (a.k.a. “lots of pretty graphs”)

Monitoring Future

- Large data analysis using machine learning