ZABBIX FOR
HPC MONITORING AND
SUPPORT

Mikhail Serkov
Delivery Manager/HPC Engineer
What's next?

HPC monitoring – differences from classic support model

How do we use Zabbix

What's next?
Novartis Institute For Biomedical Research (NIBR)

- Scientific research in Pharma area: Bioinformatics, Computational Chemistry, Drug Discovery, etc.
- About 10k CPU cores used for a scientific computation.
- Shared clusters - different workflows could run simultaneously within the same cluster.
- About 500 different scientific tools.
- Custom software (Python, Java, R)
HIGH PERFORMANCE COMPUTING

- Hundreds or even thousands of computation nodes
- Grid Computing technologies and software (SGE, UGE, SoGE, PBS, etc)
- Massive parallel computation across the nodes
- Strong requirements for all subsystems on hardware and software level (storage, network, power, OS)
- No magic. Linux boxes, shell scripts on a low level 😊

Example of a job submission:
```
qsub -pe smp 10 -l m_mem_free=4G,h_rt=3600,lustre=1,gpu_card=2 workflow.sh
```
OVERVIEW OF THE CUSTOMER INFRASTRUCTURE

- 250 GPU’s
- 70TB RAM
- 35-40KW/Rack
TYPICAL COMPUTATION NODE CONFIGURATION

- 28 CPU cores (2 sockets x 14 cores each)
  - Intel(R) Xeon(R) CPU E5-2697 v3 @ 2.60GHz
- 200 GB RAM
- 10 GB Ethernet + InfiniBand interfaces
- 8 GPU cores (4 cards x 2 cores each)
- NFS over 10 GB Ethernet
- Lustre over InfiniBand
OVERVIEW OF SOFTWARE STACK

• More than 500 of scientific tools
• Bioinformatics, Computation Chemistry, Xtallography, Molecular Dynamics, etc
• RHEL6.5
• Univa Grid Engine
• Zabbix 2.4
HPC MONITORING DIFFERENCES

• We need information like ‘who, what, when’, not only system metrics.
• Users are allowed to run whatever they want using grid scheduler on the computation nodes.
• 100% CPU utilization and 100% RAM utilization for node is perfectly fine.
• Node crash - not such a big deal.
• Preventing global issues by using aggregated metrics.
• Metrics not only for monitoring but for a performance analysis.
• Users are having access to the monitoring system (but restricted).
WHY ZABBIX?

• Able to monitor of a huge systems with a lot of metrics
• Flexible
• Out of the box
• Ability to aggregate metrics
• API for a data extraction
• GUI convenient for both support team and scientists
• Autodiscovery
• New nodes automatic configuration
ZABBIX CONFIGURATION

Server configuration:
• 20 CPU cores (2 sockets x 10 cores each)
  – Intel(R) Xeon(R) CPU E5-2697 v3 @ 2.60GHz
• 120 GB RAM

Number of hosts: 601
Number of items: ~200k
Number of triggers: ~37k

DB Size: 187GB
## WHAT DO WE MONITOR

<table>
<thead>
<tr>
<th>Local metrics (node level)</th>
<th>Global metrics (cluster level)</th>
</tr>
</thead>
<tbody>
<tr>
<td>All default Linux checks (LA, CPU utilization, RAM, swap, etc) - agent</td>
<td>Meta CPU utilization - aggregation of CPU utilization of HPC nodes.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Every single GPU core (Temperature, Utilization if possible) - agent</td>
<td>NFS global transmit/retransmit - aggregation of nodes values</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Every single CPU core (Utilization, Temperature) - agent</td>
<td>Grid specific - used/active slots, running jobs, pending jobs, top users - external scripts</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>NFS shares availability / utilization / mount details - agent</td>
<td>CPU/Memory oversubscription - aggregation of nodes values</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Slots / RAM reserved - external scripts</td>
<td>Overloaded nodes - aggregation of HPC values</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>HPC jobs - external scripts</td>
<td>Pending time - external scripts</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>....</td>
</tr>
</tbody>
</table>
HPC specific examples

1) Expected utilization VS Real one

Every job has a resource request for number of CPUs, RAM, etc. In every moment we can compare real utilization with an expected one. If they are not close, we need to investigate if someone oversubscribing resources or overload nodes.

**Solution:** Zabbix not only checks current system metrics, but also keeps an expected values. If they are too different we receive warning.

2) Users on a computation node

Users are not restricted to SSH to any node (debugging, tracing job in real time, interactive jobs, etc). However we should check if user has job on the node he is logged into.

**Solution:** We have a trigger that notify us if we have anyone logged on the node with no job running. Additionally we store a list of logged in users for any single moment.
### HPC specific examples

#### Pending time probes

It is really hard to predict the pending time for any particular job in the pending list, as they all have different resource requests, and runtimes. It is not a FIFO and the pending time is always related to resources user wants to have.

**Solution:** Zabbix runs ‘pending probes’ (empty jobs) and checks how long does it take. This is a good indicator for queue state at the moment.

![Graph showing pending time probes data](image-url)
WHAT DO WE MONITOR: GLOBAL METRICS

Global cluster utilization

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>cluster.name: Top 5 users (slots)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2016-06-23 11:17:04</td>
<td>3556 user1, 1153 user2, 632 user3, 512 user4, 198 user5</td>
</tr>
</tbody>
</table>

- **Active slots**: [avg] 7.28 Kslots, min 7.26 Kslots, avg 7.28 Kslots, max 7.28 Kslots
- **Slots in use**: [avg] 6.14 Kslots, min 708 slots, avg 3.81 Kslots, max 6.35 Kslots
WHAT DO WE MONITOR: GLOBAL METRICS

RAM oversubscription

<table>
<thead>
<tr>
<th>Metric</th>
<th>Avg</th>
<th>Min</th>
<th>Avg</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory total</td>
<td>64.72 TB</td>
<td>60.32 TB</td>
<td>64.62 TB</td>
<td>64.72 TB</td>
</tr>
<tr>
<td>Memory reserved</td>
<td>33.54 TB</td>
<td>5.87 TB</td>
<td>18.45 TB</td>
<td>34.24 TB</td>
</tr>
<tr>
<td>Memory used</td>
<td>8.04 TB</td>
<td>2.19 TB</td>
<td>4.59 TB</td>
<td>8.12 TB</td>
</tr>
</tbody>
</table>

Data from history. Generated in 3.47 sec.
WHAT DO WE MONITOR: GLOBAL METRICS

CPU time oversubscription

<table>
<thead>
<tr>
<th>Metric</th>
<th>last</th>
<th>min</th>
<th>avg</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU time reserved %</td>
<td>6.59%</td>
<td>3.21%</td>
<td>66.97%</td>
<td>91.16%</td>
</tr>
<tr>
<td>metacpu.load.summary</td>
<td>6.77%</td>
<td>3.65%</td>
<td>59.71%</td>
<td>83.85%</td>
</tr>
<tr>
<td>metacpu.load.user</td>
<td>5.98%</td>
<td>2.79%</td>
<td>56.32%</td>
<td>81.2%</td>
</tr>
</tbody>
</table>
WHAT DO WE MONITOR: GLOBAL METRICS

Meta CPU utilization

- **metacpu.load.idle**
  - last: 21.66%
  - min: 21.66%
  - avg: 43.94%
  - max: 95.32%

- **metacpu.load.user**
  - avg: 74.49%
  - min: 0.56%
  - avg: 48.22%
  - max: 74.49%

- **metacpu.load.system**
  - avg: 0.56%
  - min: 0.21%
  - avg: 1.04%
  - max: 2.64%

- **metacpu.load.iowait**
  - avg: 3.48%
  - min: 3.48%
  - avg: 6.78%
  - max: 13.11%

- **metacpu.load.interrupt**
  - avg: 0%
  - min: 0%
  - avg: 0%
  - max: 0%

- **metacpu.load.nice**
  - avg: 0.0001%
  - min: 0.0001%
  - avg: 0.000128%
  - max: 0.0021%

- **metacpu.load.softirq**
  - avg: 0.04%
  - min: 0.0006%
  - avg: 0.02%
  - max: 0.04%

- **metacpu.load.steal**
  - avg: 0%
  - min: 0%
  - avg: 0%
  - max: 0%

Data from history. Generated in 19.88 sec.
WHAT DO WE MONITOR: GLOBAL METRICS

Aggregated cluster status
WHAT DO WE MONITOR: GLOBAL METRICS

Storage operational metrics

- Wolverine NFS overall retransmit
  - [max] 4.45 K
  - min: 30
  - avg: 532.91
  - max: 4.45 K

Data from history. Generated in 0.07 sec.
WHAT DO WE MONITOR: LOCAL METRICS

CPU idle time (avg) 14.23% 0.0012% 42.4% 99.98%
CPU user time (avg) 85.71% 0.003% 57.48% 99.96%
CPU system time (avg) 0.04% 0.01% 0.1% 1.05%

Available memory (avg) 646.92 GB 464.1 GB 1.08 TB 1.47 TB

Timestamp | cluster.name.node1: active users (jobs)
---|---
2016-06-23 11:19:57 | 5264537, user1, 20x1core
2016-06-23 11:19:57 | 5272483, user2, 1x1core
2016-06-23 11:19:57 | 5272483, user3, 1x1core
2016-06-23 11:19:57 | 5272949, user4, 2x1core

Data from /etc/summary. Generated in 0.72 sec.
WHAT DO WE MONITOR: LOCAL METRICS
USER ACCESS

We want to provide a limited amount of information to users. They don’t need any info about triggers and issues, but only metrics. We have patched Zabbix to remove all unnecessary data for guest access.

Before

After
Benefits

• Better understanding of a global issues on the cluster an reasons of why have they happened.
• Great performance indicators for other infrastructure teams ( especially Storage team )
• Performance tuning of a scientific workflows. Jobs profiling. In some cases information we cat get from Zabbix is helping us to significantly improve performance of jobs.
• Proactive monitoring. With Zabbix it’s easier to understand if something is not right on the cluster or with some job. In most cases we are able to prevent global cluster issues, or at least minimize an impact.
• One monitoring system for clusters and HPC infrastructure.
• “All in one”. Lower efforts on support/maintain monitoring system(s).
WHAT’S NEXT?

- Tight integration with Grid HPC software.
- Data analysis using external tools, but with Zabbix data source.
- Create a set of CLI utilities for getting Zabbix statistics in ‘human-readable’ format.
- Automation of jobs profiling using Zabbix API.
Questions?