Challenges of Monitoring Distributed Systems

May 2017

Nenad Bozic
@NenadBozicNs
nenad.bozic@smartcat.io

SmartCat
www.smartcat.io
@SmartCat_io
Agenda

● Monitoring 101
● Metric data stream and tools
● Log data stream and tools
● Combine metrics and logs for full control
● Alerting
Monitoring 101

- Monitoring domain consists of:
  - Metrics data stream
  - Log data stream
  - Alerting
Metrics Data Stream
Metric data stream

• Easily forgotten and pushed aside when chasing deadlines

• Metrics are indicators that everything is working within expected boundaries

• Good dashboard has enough information (not too much, not too little)

Distributed system -> many graphs to watch -> information overload trap
Metric data stream - decision

• SAS solutions vs self-managed solutions

• Paying solutions vs free solutions

• Decision based on:
  ○ technical team skillset
  ○ level of control
  ○ security of data
Metric data stream - stack

• Riemann as sink that handles events and sends them to Riemann server

• InfluxDB as NoSQL store which is build for measurements

• Grafana as visualization tool (flexible configurable graphs from many data sources)
Log Data Stream
Log data stream

• Log monitoring on single machine requires skill and knowledge

• Same challenges as with metrics (not too much, not too little)

• Metrics are indicator that something happened and logs provide context (what happened)

Distributed system -> many terminals open -> information overload trap
Log data stream - decision

• SAS solutions vs self-managed solutions

• Paying solutions and free solutions

• Decision based on:
  ○ technical team skillset
  ○ level of control
  ○ security of your data
Log data stream - ELK stack

• ELK (ElasticSearch, LogStash, Kibana) all open source

• Filebeat is sending log messages from instances

• Logstash can filter, manipulate and transform messages

• ElasticSearch indexes log messages for easier searching

• Kibana is visualization tool with filtering capabilities
Combine logs and metrics
Real world example

• Provide reliable latency guarantee for 99.999% request

• Whole infrastructure deployed on AWS

• Lot of metrics transferred to metrics machine

• We needed fine grained diagnostics for queries to database both on cluster and application level among other things
Combine logs and metrics

- It is much easier to look at graphs than logs
- Good metric coverage can pinpoint exact cause of problems
- Usually we need log messages to bring the context
- Grafana can combine InfluxDB (measurement data store) and ElasticSearch (log index)
Alerting
Alerting

• Alerting is giving you freedom not to look at graphs

• Someone else placed domain knowledge about alerts

• Alerting must not be frequent since you will end up ignoring alerts

Distributed system -> many alerts -> information overload trap
The Boy Who Cried Wolf
Sentinel - SMART Alerting

• Have more context when anomaly happens

• Have snapshot of the system at moment something happened

• Be proactive, not reactive, let system predict cause of malfunction and prevent it instead of curing it
Sentinel - SMART Alerting
Sentinel - SMART Alerting

ALERT - Anomaly detected

Time: 2016-10-21 11:07:00

Snapshot of the system:
- `avg(mem_used_percent)` -> 11.481133733107036
- `avg(diskio_writes)` -> 90474.97142857143
- `avg(cpu_usage_user)` -> 19.8143671812
- `avg(diskio_read_time)` -> 95.44583426165537
- `time_slt` -> 1477040820
- `avg(diskio_write_bytes)` -> 0.6123595505617978
- `avg(diskio_write_bytes)` -> 10965.333333333334
- `avg(queryReport_value)` -> 84200.27777777778
- `avg(queryWrite_time)` -> 3.84228417777778E8
- `avg(requestRate_select_value)` -> 22.0140604986498
- `avg(cpu_usage_system)` -> 12041.5555555555555
- `avg(requestRate_update_value)` -> 54.857509850772374
- `avg(cpu_usage_idle)` -> 38620.56470589238
- `avg(diskio_reacq)` -> 0.1750878265266903
- `avg(disk_used_percent)` -> 8.535566189444458E9
- `avg(cpu_usage_steal)` -> 62.9355035741667
- `avg(diskio_iw_time)` -> 212.2696292134833
- `avg(cpu_usage_iowait)` -> 6.056117222060926

Further Investigation:
Disk I/O write time value was high, please check [Cluster Disk Stats](#).
Conclusion
Conclusion

• Have right amount of information, not too much, not too little
• Having good selection of metrics and logs is iterative process
• Do not end up fixing monitoring machine instead of fixing application code (especially in distributed world)
• Be proactive, not reactive
• Tailor metrics by your needs, build tools if there are not any that suite your use case
Links

- Monitoring stack for distributed systems - SmartCat blog post
- Distributed logging - SmartCat blog post
- Metrics collection stack for distributed systems - SmartCat blog post
- Monitoring machine ansible project (Riemann, Influx, Grafana, ELK) - SmartCat github project
Thank you