Grid Site Monitoring and Log Processing using ELK

A. Mikula[1,a], D. Adamová[2], M. Adam[1], J. Chudoba[1], J. Švec[1]

[1] Institute of Physics of Czech Academy of Sciences, Na Slovance 1999/2, Praha, CZ

[2] Institute of Nuclear Physics of Czech Academy of Sciences, Řež 130, Řež, CZ

E-mail: [a]mikula@fzu.cz

The site consists of interconnected institutions, the Computing Centre of Institute of physics of Czech academy of Sciences, Nuclear physics institute in Řež. Brief overview of computational, storage and network resources is given. Also there is information about software used for grid and local services and the ways we monitor functionality of it. There is brief overview of each used monitoring tool, Nagios, Munin, Ganglia, Netflow and Observium with highlights of strengths and weaknesses of each one of them. And also there is mention of how good configuration through puppet can make setup of such monitoring a lot easier.

Next is introduction to our new ELK stack facility used for log collecting, parsing, querying and visualisation of outputs. Overview of used hardware, description of roles and distribution of each software component in stack, technological challenges and requirements of cluster setup and tuning. In the end is very brief overview of paid version of software along with few tips of what to avoid.

Keywords: Monitoring, Grid site, ELK stack, Elasticsearch, Logstash, Kibana

Grid Site Monitoring and Log Processing using ELK

Computing Centre of Institute of Physics of Czech Academy of Sciences (IoP) is the site consisting of interconnected institutions, Institute of Physics CAS and Institute of Nuclear Physics CAS, participating in several grid and national projects. Most notable are LHC [Worldwide LHC Computing Grid] projects ATLAS [A Toroidal LHC ApparatuS] and ALICE [A Large Ion Collider Experiment] (the site is tier 2 site for both), NOvA [NovA Neutrino Experiment] from OSG [Open Science Grid], Cherenkov Telescope Array [Cherenkov Telescope Array] and also participation in Czech NGI project Metacentrum [Metacentrum]. The site is available to grid computing and also to our local users. An overview of the site status and evolution, the hardware stack, the services provided for various projects as well as the site performance evaluation can be found e.g. in [Adamova, 2015]

There is lot to monitor since the site hosts more than 400 machines with different roles ranging from basic network devices and services (as DHCP and DNS) to grid storage systems based on DPM [LcgDM] and XrootD [XrootD], worker nodes and other grid services.

Most of machines at our site is managed through Puppet [Puppet], with exception of devices which are incapable of such management (network appliances and other similar devices), older services which are managed by CFEngine [CFEngine] (from which we are moving to puppet and some "one shot" machines which is easier to set-up by hand from time to time than to bend the configuration into puppet).

Machines run variety of systems including Scientific Linux 6 [Scientific Linux], Debian [Debian Linux], CentOS 7 [CentOS Project] and Windows [Microsoft Windows]. Used monitoring tool set is depending on the age of installation and degree of administration involvement by our admin team.

In the next sections we will describe tools used for monitoring the performance and the status of our site.

Nagios

The site's main monitoring tool is Nagios [Nagios] with Check-MK [Check_MK] interface, extended with few custom monitoring plug-ins. Most of Nagios configuration is carried out through Puppet configuration management, which gives us flexibility which Nagios is normally lacking. It is possible to save hours of work by using decent Puppet module, which can detect hardware and used software to create checks on the fly. For example in this way there can be http check set-up only on machines which have http server installed, without any intervention from admins. Also Nagios service (to be checked) can be easily exported in puppet manifest on machines running specific service and in that way there is no need to keep a particular Nagios setup in sync with reality, puppet does that by itself.

Munin

Munin [Munin] is used for machine performance plotting a other metrics such as batch system occupancy, system network throughput, server room temperature, humidity, AC chiller metrics, etc. Munin has nice user interface, but configuration of many aspects is a bit clumsy and tedious, also tool is not optimized in any way to scale up. Every read of metric is directly written to drive, without any write clustering, this puts a lot of stress on drives in terms of random IO, currently this was solved by using RAID 1E of six SAS 15K drives, what is enough to saturate the site's needs, for now. The other way to solve this problem is to put all RRDs into tmpfs file system (have all data in RAM), but this approach is fragile when rebooting monitoring node and needs non-standard configuration and tends to be more dependent on administrators (it is needed to remember to take care of all recovery or at least to check if it went the way it was intended to).

Ganglia

Ganglia [Ganglia] is the other way to collect nice performance metrics, it has easier interface for creation of aggregate graphs, it also has clustering capability. There can be designated "collector" nodes which can collect data from their neighbours and then ship it to Ganglia master node, it also supports multicast shipping and listening, multicast domain setup is needed for this, and new node gets monitored without any other effort once it joins that domain.

Netflow

Netflow is used on chosen devices to further monitor network. Its capability is to split network traffic into finer groups, to get traffic categories categorized for example by the source site or ip prefix. Right now it is deployed only on storage servers, since it needs self compiled kernel module. The tools used to create graphs are from Flow-Tools set [Flow tools].

Observium

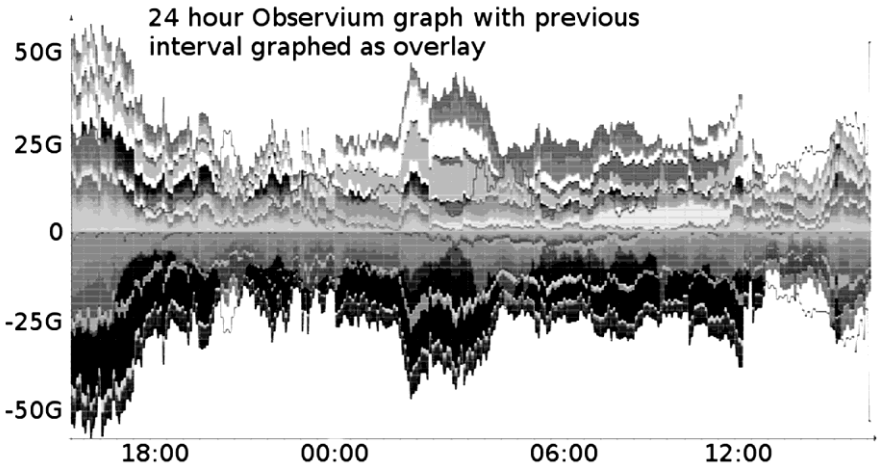

Figure 1. Observium aggregate graph of network traffic

One of the site latest deployed monitoring tools is Observium [Observium] which is great tool for monitoring and troubleshooting of problems with network appliances as routers and switches. It is capable of auto discovering of various types and brands of devices including port occupancy, port labels, VLAN configuration, device OS and many more. It was so "clever" that it discovered even devices not belonging in our network during its first run. There is a 24 hour sample of traffic from our main switch is given as example on figure 1 with another 24 hours from previous interval as an overlay (lighter grey "shadow" overlay on graph). Each different shading of main plot is contribution of different port to overall total.

The ELK stack

Elasticsearch (ES) [Elasticsearch], Logstash (LS) [Logstash], Kibana [Kibana] (and Beats [Beats]), ELK stack for short is software collection specifically designed for log processing, storing, query and visualising. It is powerful scalable bundle written mainly in Java [Java], with all Javas powers and weaknesses.

ELK stack at IoP

ELK installation at IoP is mainly experimental setup to test capabilities and viability of solution and evaluating of its strengths and weaknesses. This is also reason why whole cluster is consisting mainly of old and to be discarded hardware such as old worker nodes, which is also result of suboptimal setup of whole cluster. Most of nodes do not meet optimal HW guidelines and especially queries of data take longer than they could on optimal cluster. On the other hand this is also the best way to explore all "dark" aspects of ELK stack.

Cluster have 7 nodes overall, 4 old worker nodes with 32GB RAM, two four-core Xeon E5440 CPUs and one 2TB 7,2kRPM drive, one virtualisation platform from our former IPV6 test bed with 96GB RAM, two six-core Xeon E5645 CPUs, and eight 2TB 7,2kRPM drives in hardware 1E RAID, one 32GB RAM, one four-core Avoton C2550 with six 2TB drives in software RAID6 (this one serves also as archive) and one "front-end" server as virtual machine with 20GB RAM, eight cores and 10GB virtual drive (front-end means nod data is stored on this machine it just serves as ES master node and Kibana server).

All physical machines serve as ES data nodes, LS processors, virtual machine and largest machine are also master eligible nodes for ES.

Our site (~300 machines and devices) produces around 40.000.000 log entries daily, which takes around 30GB of storage space which clearly shows storage and RAM space needs of whole setup.

ELK, and specially Elastic search hardware recommendations

To summarize official statement for HW recommendations each ES node should have have up to ~31GB RAM per ES instance (multiple instances on one physical machine are possible) another same amount of free RAM for system cache (which is heavily used and beneficial for ES instance) and any reasonable amount of RAM per LS instance, which is optional part it can be on any different machine. LS is very easily scaled to available resources.

Hard drive wise recommendation is easy, if you can afford to get SSDs, do it, otherwise use as much independent drives as you can. You do not need to worry for data (unless you misconfigure ES instance), because ES is redundant from design so there is no real need for RAID with parity or mirroring.

On bigger instances is also wise to have separate master and client (front-end) nodes. [Hardware]

Parts of ELK stack

ELK stack is combination of different tools with distinct features. We will discuss its parts in the same order as the log entry is being passed through the ELK during processing it.

Collecting of log entries is done either by one of Beats or by Logstash, we will first concentrate on Beats since these are designed to be lightweight and are specialized on data collecting.

Beats

Beats is a library designed to be extendable by community contributions, but it is also name harbouring all software from beat collection. The software tools which belong to this collection is Metricbeat (in latest release of stack renamed from Topbeat) [Metricbeat], Packetbeat [Packetbeat], Winlogbeat [Winlogbeat], Filebeat [Filebeat] and many more community contributed software.

Metricbeat is designed for collecting system metrics such as CPU, RAM and space usage. Packetbeat is for monitoring network services as Httpd, Mysql, etc. Winlogbeat is used for shipping logs from Windows. We would not get into these since these are not used on the site.

Filebeat is used for collecting log lines in the same fashion as would "tail -f /path/file" would do, with exception that Filebeat (and Beats generally) is capable of basic log entry transforming, like merging of multiple lines into one entry or ignoring based on supplied regexp. It also "remembers" where it ended between reboots.

All Beats are capable of direct data submission into ES.

Logstash

Logstash is also capable of "tailing" logs same way as Filebeat, but to use it for only this is same as killing fly with heavy gun.

Logstash is very powerful parsing and enrichment platform capable of not only collecting data from files, but also collection it by running commands on system, by listening to rsyslog [rsyslog] or Beats data streams and many more.

Thanks to availability of grok [grok] filter you can also get exact fields from data and search them, e.g. it allows to filter data.

You can omit that but your other part (Kibana, or possibly Grafana [Grafana]) would suffer from this omission greatly.

Logstash is capable of indexing data into ES of many other back-ends after data is processed.

For more informations please refer to Logstash reference [Logstash reference]

Elasticsearch

Data is stored into ES after processing. ES is the main component of all this since it not only stores data but also enables powerful way to search it from one point. It uses Apache Lucene [Apache Lucene] full-text search language. Full-text analysis is by default run on all data indexed into it, which can consume large amounts of space, so it is advisable to turn it on for only needed fields of document (log entry in this case). It is also needed to pay attention to assignment of right data types to different fields of document, because ES assigns itself some (based on the first occurrence format) when it is not it explicitly defined. This can cause great trouble, since mapping (data type) can not be changed later on. This change requires re-indexing all data affected with new mapping into new index and this can be lengthy procedure.

There is lot of performance tuning options that can be applied to ES cluster, beginning with hardware specific setup based on available memory and hard drives, and continuing with data location and allocation across cluster, based on node power, state, and data accessibility needs.

It is advisable to pay special attention to few configuration options which are related to used memory (to keep it on ~50% of free available RAM of system) and option "index.merge.scheduler.max_thread_count" which determines maximum of threads used when accessing hard drives, it should be set to "1" for spinning drives.

Data can be queried directly from ES through it's REST API or there is assortment of front-ends such as Kibana.

Kibana

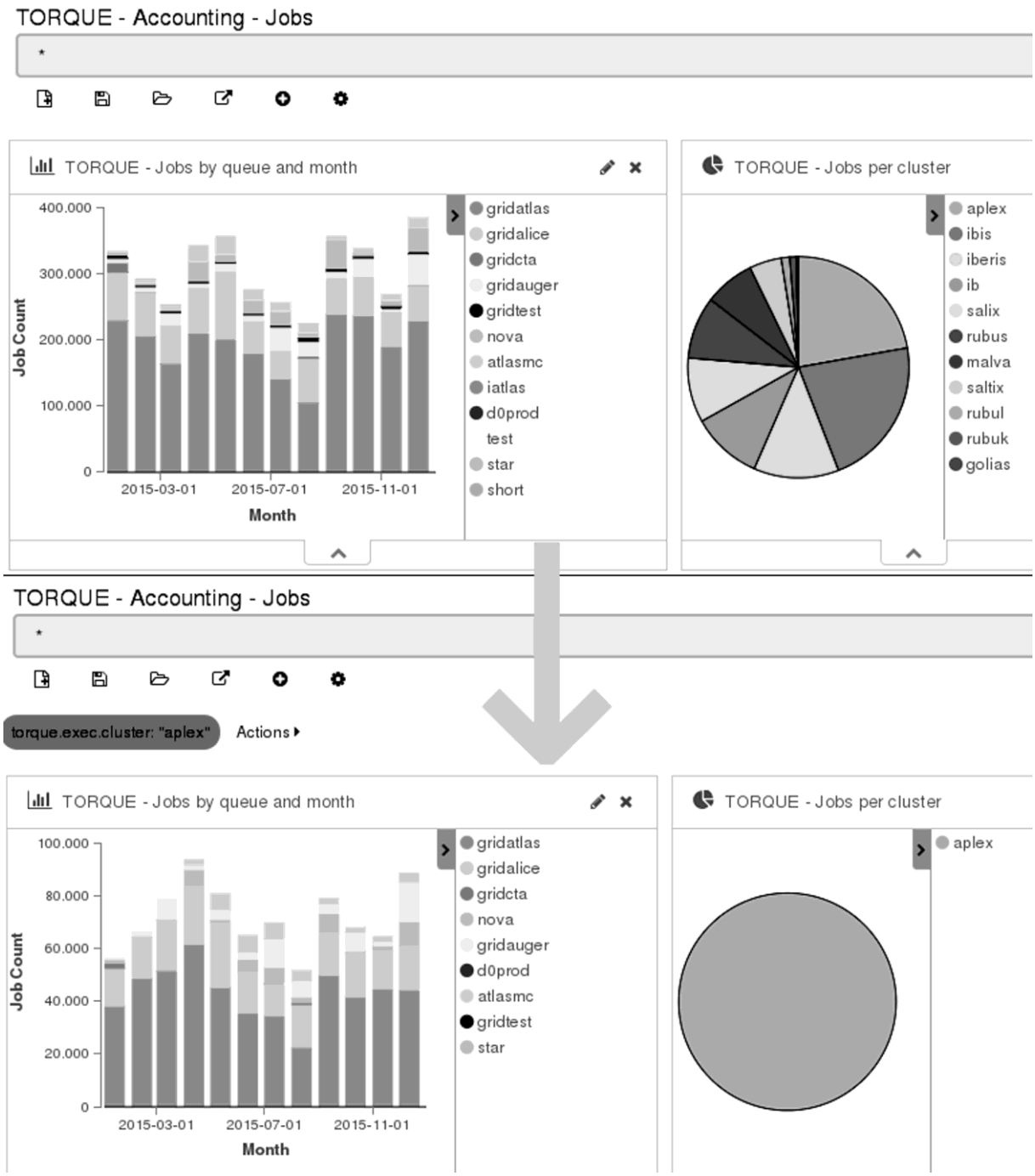

Figure 2. Kibana filtering example

Kibana is web based front-end designed for querying data stored in ES in more user friendly way. It is not used only for searching data, it can also visualize outputs in graphs, compare different data sets (with Timelion extension till version 5 which has this built-in) and perform on-fly data calculation (this has few culprits, such data can not be searched directly, zero is returned when there is no source data → no reasonable averages, this calculation is only possible on numerical fields and finally this can also crash Kibana when done wrongly).

Monitoring

Cluster can either be monitored "directly" through API (just by using it or there are some implementation scripts to handle this for example through Nagios), or closed source solution X-pack [X-pack] can be used for which there is possibility to obtain free (just for monitoring) license. In versions prior version 5 part for cluster monitoring was named Marvel (license also obtainable for free).

Paid version

This functionality was not tested at the site, but with paid subscription there is functionality as securing of cluster communication thorough SSL, authorization and authentication of users (also possible with search-guard extension – Apache license – not yet compatible with ELK5) [Searchguard plugin], monitoring and alerts based on on ES queries and also not to forgot support from developers of ELK stack. [Subscriptions]

Conclusions

Despite complex configuration and some problems with setup including fast release schedule which is tending to introduce bugs more often then desirable is ELK nice, scalable and modern approach to processing and searching of logs. It is user friendly alternative to "greping" through tons of lines.

More powerful CPU than Atom/Avoton is recommended because such CPU (even multicore) is not powerful enough to cope with more indexes.

Also ELK stack version 5 was released in the time of writing of this article, which promises more efficient processing accompanied with many changes which we are not able to assess right now.

References

Worldwide LHC Computing Grid [Electronic resource]: http://wlcg.web.cern.ch/
A Toroidal LHC ApparatuS [Electronic resource]: http://atlas.ch/
A Large Ion Collider Experiment [Electronic resource]: http://aliceinfo.cern.ch/Public/Welcome.html
NovA Neutrino Experiment [Electronic resource]: https://www-nova.fnal.gov/
Open Science Grid [Electronic resource]: https://www.opensciencegrid.org/
Cherenkov Telescope Array [Electronic resource]: https://web.cta-observatory.org/
Metacentrum [Electronic resource]: https://www.metacentrum.cz/
Adamova D. et al.: WLCG Tier-2 site in Prague: a little bit of history, current status and future perspectives. 2015 J. Phys.: Conf. Ser. 608 012035
LcgDM − Data Management Servers, DPM − Disk Pool Manager [Electronic resource]: http://lcgdm.web.cern.ch/dpm
XrootD [Electronic resource]: http://xrootd.org/
Puppet [Electronic resource]: https://puppet.com/
CFEngine [Electronic resource]: https://cfengine.com/
Scientific Linux [Electronic resource]: https://www.scientificlinux.org/
Debian Linux [Electronic resource]: https://www.debian.org/
CentOS Project [Electronic resource]: https://www.centos.org/
Microsoft Windows [Electronic resource]: https://www.microsoft.com/en-us/windows
Nagios [Electronic resource]: https://www.nagios.org/
Check_MK [Electronic resource]: https://mathias-kettner.de/check_mk.html
Munin [Electronic resource]: http://munin-monitoring.org/
Ganglia [Electronic resource]: http://ganglia.info/

Flow tools [Electronic resource]: http://www.splintered.net/sw/flow-tools
Observium [Electronic resource]: https://www.observium.org/
Elasticsearch [Electronic resource]: https://www.elastic.co/products/elasticsearch
Logstash [Electronic resource]: https://www.elastic.co/products/logstash
Kibana [Electronic resource]: https://www.elastic.co/products/kibana
Beats [Electronic resource]: https://www.elastic.co/products/beats
Java [Electronic resource]: https://www.oracle.com/java/index.html
Hardware [Electronic resource]:
https://www.elastic.co/guide/en/elasticsearch/guide/current/hardware.html
Metricbeat [Electronic resource]: https://www.elastic.co/products/beats/metricbeat
Packetbeat [Electronic resource]: https://www.elastic.co/products/beats/packetbeat
Winlogbeat [Electronic resource]: https://www.elastic.co/products/beats/winlogbeat
Filebeat [Electronic resource]: https://www.elastic.co/products/beats/filebeat
rsyslog [Electronic resource]: http://www.rsyslog.com/
grok [Electronic resource]: https://www.elastic.co/guide/en/logstash/current/plugins-filters-grok.html
Grafana [Electronic resource]: http://grafana.org/
Logstash reference [Electronic resource]: https://www.elastic.co/guide/en/logstash/current/index.html
Apache Lucene [Electronic resource]: https://lucene.apache.org/
X-pack [Electronic resource]: https://www.elastic.co/products/x-pack
Searchguard plugin [Electronic resource]: https://github.com/floragunncom/search-guard
Subscriptions [Electronic resource]: https://www.elastic.co/subscriptions