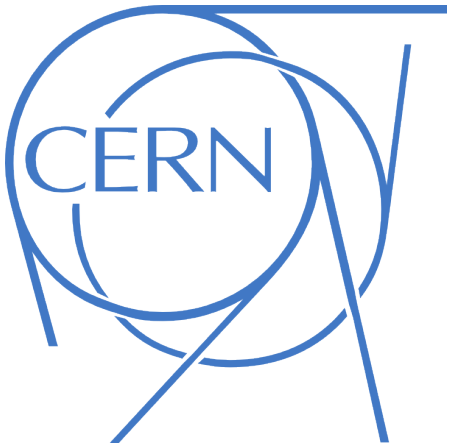

2019

Student Name: Matthew Boyd

Email: Matthew.James.Boyd@cern.ch

Department IT-CDA-IC

Project Title: MAlt: Microsoft Alternatives
Mail: Performance Metrics

Supervisor: Giacomo Tenaglia
Paweł Grzywaczewski

CERN

Meyrin, Switzerland

CONTENTS

MALT: Microsoft Alternative Project

Mail: Metrics Monitoring

Matthew Boyd

Abstract

Due to the fact that Microsoft has dropped CERN's academic institute status, therefore leaving licensing 10 times more expensive, CERN has had to come up with an alternative to Microsoft products to try and phase out the need for their products and in turn save money. Mail is currently predominantly on Microsoft's Exchange servers.
CERN is migrating its mail system, hosting 50TB+ of data on 40,000 mailboxes, from Microsoft Exchange to Kopano, an Open Source Linux-based system.
The aim of this project is to consolidate the various system and application metrics in use in the new systems, and to develop on top of that an uniform visualisation layer, providing a facility to implement anomaly detection and event correlation. The data will be collected from different services, ranging from systems performance / monitoring, to web servers and back-end databases, to migration-specific tools. The project will make extensive use of free and open source tools such as Collectd, Telegraf and Filebeat to collect the data, ElasticSearch and InfluxDB to store it, and Kibana/Grafana to visualise it. The main operating system that will be used for the project is Cent-OS Linux version 7. Other tools that are going to be heavily used are Puppet (configuration management system), OpenStack (Infrastructure-as-a-Service cloud computing), Openshift (Platform-as-a-Service for web hosting) and Docker/Kubernetes (container platform).

MALT: Microsoft Alternative Project

Mail: Metrics Monitoring

Matthew Boyd

Acknowledgements

There are a lot of people I would like to thank for my experience here at CERN. First and foremost, I would like to thank my supervisors Giacomo Tenaglia and Paweł Grzywaczewski as well as my team: Riccardo Candido, Dominik Taborsky, Marija Predarska and Leopold Gattinger. Their continued support allowed me to carry out my task effectively through answering the numerous questions I posed. Additionally, I would like to thank my family and girlfriend for supporting me whilst I was living in a foreign country for the first time and for visiting me.

I. Description of Technologies considered

Throughout the project, there were a lot of technologies that were evaluated for the given task. The following list shows all the possible technologies that were considered as well as a brief overview of their purpose.

- Collectd
- Telegraf
- InfluxDB
- cAdvisor
- vAdvisor
- Prometheus
- Grafana

II. Collectd

Collectd [1] is a daemon (runs as a background process) which will collect system and application metrics periodically and allows for these metrics to be sent to an appropriate datastore. As an application it is very minimal. It simply collects data and does not display it visually in anyway.

III. Telegraf

Telegraf [2], like Collectd is a metric collection agent. It can collect metrics from a wide array of predefined input configurations, such as docker, CPU, disk, etc. It will then write these metrics to a pre-configured output source, i.e. a time-based database. Not only does it have 200+ plugins already written for it, therefore only requiring some tweaks to the configuration, it also allows a lot of flexibility in running your own scripts to monitor statistics.

IV. InfluxDB

InfluxDB [3] is an open-source time-series database used for monitoring metrics. Influx stores all the data with a time-stamp in epoch which can then be used in conjunction with programs such as Grafana to represent the data visually due to influx only being a datastore.

V. cAdvisor

CAdvisor [4] is an open-source application that allows for analysis of resource usage and performance characteristics of running containers. Again, like with Telegraf and Collectd it is a daemon that runs and will collect information about docker containers, for instance: CPU usage, DISK space, etc. These statistics can then be sent to a database and displayed in Grafana.

VI. vAdvisor

VAdvisor [5] is an open-source sister software application to cAdvisor that is used to get information from virtual machines running on a server. This will produce the same type of metrics as in cAdvisor: CPU usage and DISK space.

VII. Prometheus

Prometheus [7] is an open-source monitoring solution. Prometheus uses key-value pairs to store data regarding metrics. Prometheus allows for many different interactions from third parties out-of-the-box. Additionally, it provides a graphical interface to see the results that are collected, as well as providing integration support for Grafana.

VIII. Grafana

Grafana [8] is an open-source representational tool for displaying results provided by a time-series database visually through the use of graphs and charts. There are a range of different visualisations that can be used such as heatmaps to histograms, and geomaps. There is native support for a lot of databases within Grafana, including Prometheus and InfluxDB. As well as allowing for collaboration whereby all users will be able to see the dashboards simultaneously.

IX. Technologies used

For data collection: The technology that was chosen for data collection was Telegraf. The main reason for this was that Telegraf came with a predefined configuration setup that allowed for many inputs and outputs to be easily added, as well as the fact that when testing Telegraf, it was able to also gain metrics that other collection applications such as Collectd and CAdvisor were not able to find. Additionally, work had already commenced on part of the monitoring systems using Telegraf to get application metrics, therefore to keep inline with the pre-existing technology stack to preserve uniformity, Telegraf was chosen.

For data storage: InfluxDB ended up being the data storage application of choice. This decision was made for a number of reasons, as there are pros and cons for both Prometheus and InfluxDB . For instance, both of them are written in GO language, therefore meaning that they are very quick to be executed, however, in terms of Prometheus for instance, it is bound by a schema, whereas InfluxDB is schema-free. InfluxDB also supports more languages than Prometheus and additional access methods (JSON over UDP). As well as the fact that InfluxDB's query language is very powerful and development friendly.

For data visualisation: Grafana was used over the built-in visualisations of Prometheus. The main reasons behind this decision was that Grafana has more flexibility as well as being alot easier to use than Prometheus. Grafana contains many rich features to allow the user to display statistics easily through a pluggable panel architecture.

X. First Task

The first task set out by my supervisor was to use an existing virtual machine to gather metrics around the emailing service Postfix. Metrics were initially to be collected through Collectd and then sent to the InfluxDB server and displayed through Grafana.

The first operation carried out was editing the configuration of Collectd in /etc/Collectd.conf. Within the configuration, the

write_graphite plugin had to be enabled, the database team at CERN were able to process the request to install the graphite plugin on the InfluxDB server to allow for the influx database to process the data in the correct format.

After discussing with the team about Collectd, it was decided that Telegraf was the better option. The change from Collectd to Telegraf required some additional effort. Firstly, the Telegraf configuration had to be amended to allow for Postfix metrics to be collected. For the metric collection, a script was written in python in order to scan through the maillogs and match given metrics. Once the script was complete it ran every 5 seconds and the results were pushed to the InfluxDB server which was connected to Grafana in order to present the results visually.

XI. Second Task

Next, I was given the task of getting information for each of the docker containers. Examples of the information that needed to be extracted included: CPU usage time, file storage amounts, load time, etc. This was to ensure that in the event of a docker container going down, there would be metrics to show this allowing other users to be able to act accordingly in a short amount of time. Initially I carried this out through the use of an open-source software known as cAdvisor. cAdvisor acted very much like Collectd or Telegraf but for containers. It collected the metrics that were being produced by the container, and then through the use of configuration file manipulation, it was attached to the influx database so that the metrics would be sent into it every 10 seconds. This allowed for graphs to be created in Grafana from this datasource. There were some issues that were faced when carrying out this deployment the most difficult of which was trying to deal with SSL Certificate errors. However, after having a look at Telegraf's configuration for docker containers, I found that the metrics that Telegraf produced were a superset of those produced by cAdvisor. Therefore, Telegraf was used as the main way to gain access to metrics from the docker containers. Due to facing difficulties with SSL Certificates in cAdvisor, when configuring Telegraf's SSL Certificates the same solution was required to resolve the issue, therefore it was less of a problem.

XII. Third Task

Lastly, was the gathering of application metrics. One of my teammates, Riccardo had already written the script required to gather the specific metrics needed from the applications we were using in production. Therefore, it was simply a matter of copying Dominik's configuration for running the script into my Telegraf configuration which then allowed the script to run effectively in my testing environment.

XIII. Scrum Master

Additionally, throughout my time at CERN, every day we would partake in a daily scrummage. During this time, each person in the team would say what they have been working on, what they are going to be working on, and if they had any difficulties so that they could be discussed with the others in the team who may have additional information surrounding the subject. Each sprint would last approximately 2 weeks, and seeing that I was present for 8 weeks, I was delegated the role of scrum master for 2 weeks. This consisted of reminding everyone about scrum, and taking notes in case any one was off and for auditing purposes.

XIV. Zurich

Additionally, as part of my experience in CERN, I was able to go to Zurich with the OpenLab Students.
At the start of the day, we went to ETH Zurich and we were given two talks. The first was about the new internet infrastructure known as SCION (Scalability, Control, and Isolation on Next-Generation Networks) [6] that is being rolled out into Swiss Banks. This new internet infrastructure is able to reduce the downfalls and bottlenecks of the current internet infrastructure by adding failure isolation and route control.
Then we were given a talk on the future of drones, and what is in store for operating them autonomously. We found out that drones will be used more extensively in sporting events once they find out a way in order to allow for full tracking and measures to ensure that the drone stays in safe flying zones and can deal with unexpected events occurring.

We then continued our trip and went to the IBM Research Center. When we got there, we were given presentations on a number of different topics, including quantum computing and new research being carried out on high performance distributed data store designed for fast sharing of ephemeral data known as Apache Crail. [9]

References

[1] "Start page Collectd The system statistics collection daemon", Collectd.org, 2019. [Online]. Available: https://collectd.org/.

[2] N. Isley and C. Churilo, "Telegraf Open Source Server Agent — InfluxData", InfluxData, 2019. [Online]. Available: https://www.influxdata.com/time-series-platform/telegraf/.

[3] "InfluxDB: Purpose-Built Open Source Time Series Database — InfluxData", InfluxData, 2019. [Online]. Available: https://www.influxdata.com/.

[4] "google/cadvisor", GitHub, 2019. [Online]. Available: https://github.com/google/cadvisor.

[5] "kubevirt/vAdvisor", GitHub, 2019. [Online]. Available: https://github.com/kubevirt/vAdvisor.

[6] E. Network Security Group, "SCION Internet Architecture", Scion-architecture.net, 2019. [Online]. Available: https://www.scion-architecture.net/pages/videos/.

[7] "Prometheus - Monitoring system time series database", Prometheus.io, 2019. [Online]. Available: https://prometheus.io/.

[8] "Grafana - The open platform for analytics and monitoring", Grafana Labs, 2019. [Online]. Available: https://grafana.com/.

[9] "The Apache Crail (Incubating) Project: Overview", Crail.incubator.apache.org, 2019. [Online]. Available: https://crail.incubator.apache.org/.