Technische Hochschule
Brandenburg
University of
Applied Sciences
Fachbereich
Informatik und Medien

A Comparison of the Elastic Stack and Grafana Stack as Tools for Analysing Log Data

Gaik Teng Ooi

Bachelor Thesis • Applied Computer Science Program • Department of Informatics and Media • 24.03.2023

Motivation

Companies should conduct log management and log analysis at regular intervals. [KS06] Log management software transform logs from plain text into filterable and searchable data as well as store them for the long-term.

Requirements

This thesis analyses two such groups of open-source software, which are the Elastic Stack and the Grafana Stack. Each stack consists of a log aggregator, a search engine doubling as a data store, and a browser interface. The performance of the stacks are compared in 4 aspects: resource usage, query performance, support for a variety of integrations and possibilities for further parsing of logs through the browser interfaces.

Implementation

The components of each stack were executed as Docker containers configured to parse logs to make them searchable and filterable. Three different sets of logs were ingested: JSON-, Apache-, and PostgreSQL-format logs.

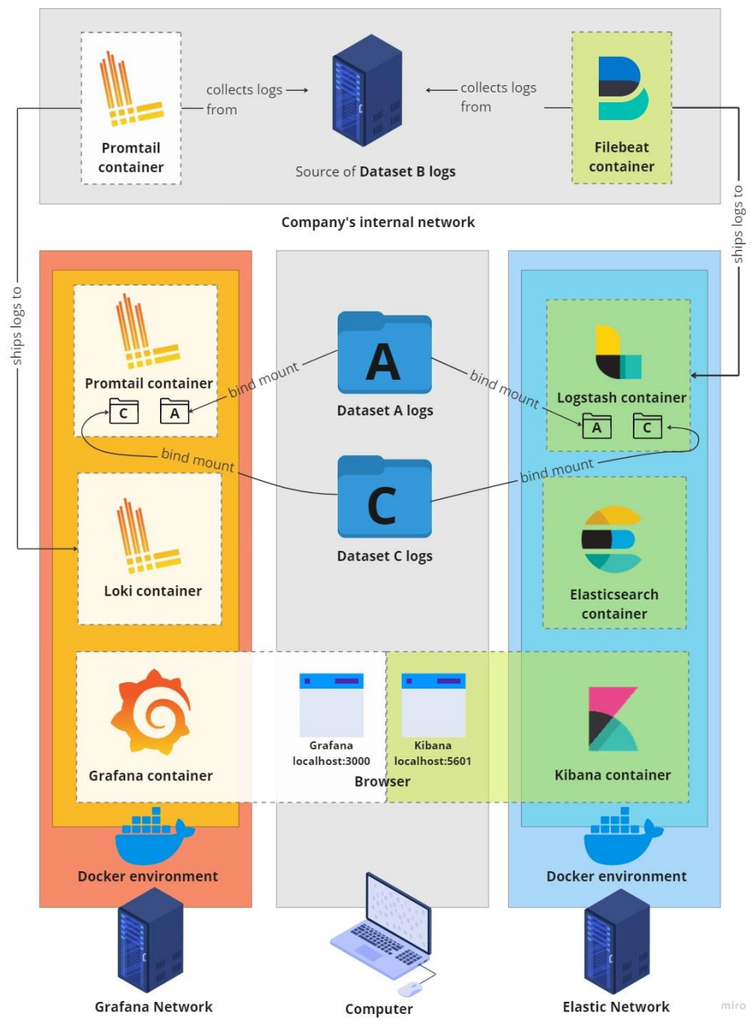

Fig. 1: Deployment of docker containers and network with datasets

Query Performance

The performance of the search engines was tested by running queries through their browser interfaces. Two queries varying in timeframe and/or keywords were modelled for each dataset. The queries were then executed 10 times each on the search engines of each stack (Elasticsearch and Loki).

Resource Usage and Ingestion Time

During ingestion, resource usage (CPU and memory) and time taken were measured with the command *docker stats.* The Elastic stack shows a higher resource usage than Grafana across all datasets. The Elastic stack ingested JSON logs much faster than the Grafana stack but was slightly slower for PostgreSQL logs. Both stacks had the highest resource usage while ingesting PostgreSQL logs. Apache logs were continuously live-streamed so ingestion time was not measured.

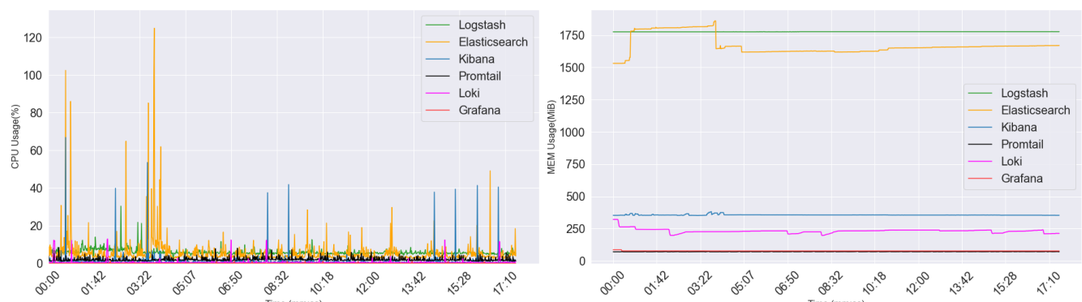

Fig. 2: Graphs of CPU usage in percentage [Left] and memory usage in Mebibytes [Right] during the ingestion of a live stream of Apache logs.

Integration Support

Support for three areas were inspected by reading the documentation: client plugins, database plugins and data formats or protocols. The stacks showed an equal amount of resources and support for integrations, or *plugins.* Both products have active open-source communities who contribute to the development of these plugins.

Post-Parsing

While the Elastic stack has index templates that can be modified continually to ingest new logs saved as indices, the Grafana stack "dynamically" parses logs efficiently while saving time and space that reloading indices normally require.

Results

The Elastic stack has a higher resource usage overall while ingesting logs. PostgreSQL logs led to the highest resource usage for both stacks. Grafana slightly outperforms in queries. Both are matched in support for integrations offered.
Grafana's post-parsing approach is more "dynamic" because saved logs are parsable with LogQL queries, while Elastic stack's approach is more "static", needing to reindex logs if a new mapping needs to be defined.

Conclusion

This thesis concludes that each stack is better for different contexts. The Elastic stack is suited for the long-term storage of archival logs and the Grafana stack for real-time log aggregation and monitoring.

Sources

[KS06] Kent, K. ; Souppaya, M.: *Guide to computer security log management.* (NIST SP 800-92; 0 ed., p. NIST SP 800-92). National Institute of Standards and Technology, 2006.

Supervised by Dipl.-Inform. Ingo Boersch and Prof. Dr.-Ing. Jochen Heinsohn • Technische Hochschule Brandenburg • Harald Meiß • forsa GmbH