CSC 466 Final Report:

Improvement of existing Log Management System

April 11 2022

Written by

Reggie Jin

Zixin Yao

Table of Content

1 Abstract

Log files are the primary data source for network observability and they contain records of all events including operations within systems, applications, software or server, thus, it is essential to monitor these files in order to protect them from outside attacks. Nevertheless, there are tons of log files generated by the computer every day, and the primary way to secure these files right now is by log management, the practice of continuously gathering, storing and analyzing log files from applications. ELK stack is one of the most used open-source software for log management, however, there still exist some crucial problems with it. Thus, we proposed some improvements based on the ELK stack. Specifically on the efficiency of log monitoring.

2 Introduction

Today, server monitoring has become very important and necessary for most IT companies and organizations. important for companies and organizations. This can help administrators not only protect servers from external attacks, but also track the activities and preferences of customers to improve individual service delivery capabilities. In fact, the focus of server monitoring is primarily on the processing of log files. These are considered to be big data in terms of volume, variety and speed. Big data in terms of variety and speed. In fact, logs exist in different formats such as log files, xml, metadata, etc. Moreover, logs grow at a very fast growth rate.

Log management is also a big data application often based on the Hadoop platform. However, Hadoop mainly supports large-scale systems such as Google, Yahoo ... For small

and medium-sized systems, Hadoop becomes expensive and impractical to implement. Moreover, Hadoop is not fast enough for online data processing.

Then, we discovered one of the most popular log file management stacks used worldwide is the ELK stack. It stands for open source projects: Elasticsearch, Logstash and Kibana; Elasticsearch is a search and analytics engine. Logstash is a server‑side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a “stash” like Elasticsearch. And Kibana lets users visualize data with charts and graphs in Elasticsearch.

3 Background

3.1 Log file Management Overview

Logs can be used to investigate, manipulate, or even corrupt or leak information. Logs are a common and useful piece of evidential information in networked and computing systems. Logs record information about events such as 'time', 'ID' and 'event', among other information.

By referring to the log files, administrators or users can confirm basic information about active and operational events through the system equipments. They can also use this profile information to trace the past and detailed information about the presence or absence of obstacles in real time situations.

At the same time, the act of managing log files causes the number of log files themselves to increase. Log files can be used as a tool for analyzing and identifying attackers, so it is also

important to manage log files in terms of security. Log files have specific protection mechanisms in place to prevent important file information from being attacked, manipulated or deleted by an external party, as well as the need to maintain the file information created during an outbreak. Generally, files will be stored on the hard disk or RAM of the system device or, alternatively, passed to a logging server for storage using a protocol.

The figure below gives an example of a basic log line looks like.

Field	Value
IP Address	192.168.2.20
Date	28/Mar/2014
Time with zone	10:27:10 -0300
Client request	”GET /cgi-bin/try/ HTTP/1.0”
Status code	200
Size in bytes	3395

Figure 1

3.2 ELK

3.2.1 Logstash

It is a tool for collecting, analyzing and filtering logs. The Logstash client is installed on the host where the logs are collected and the server is responsible for filtering and modifying the

logs received from each node. It is a middleware that integrates log collection, filtering and forwarding to Elasticsearch for further processing.

3.2.2 Elasticsearch

It is an open-source distributed search engine that consists of three main functions: providing collection, analysis and storage of data. He has many features such as distributed, zero-configuration, auto-discovery, automatic index sharding, index replication mechanism, automatic search loading, etc. It is mainly responsible for indexing and storing logs, while users can query and retrieve the contents contained.

3.2.3 Framework

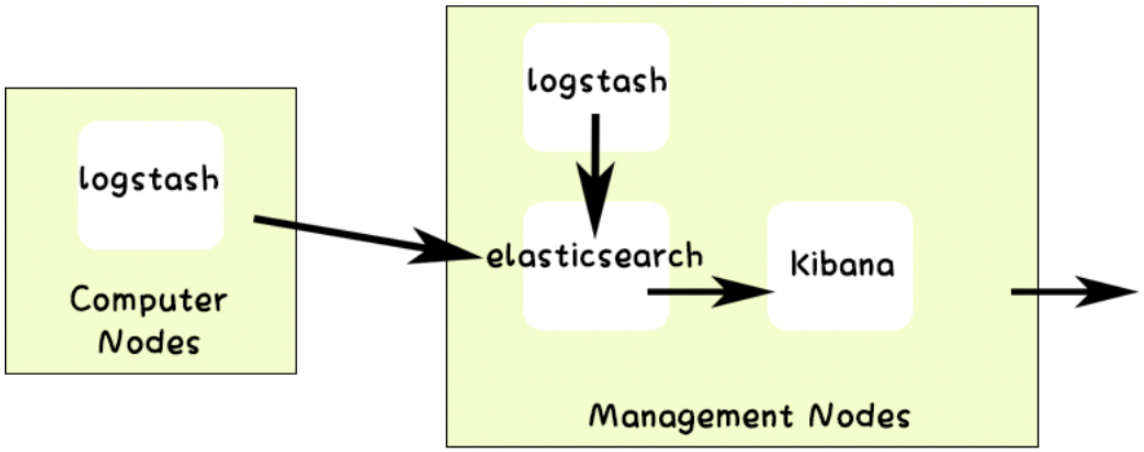

Figure 2

This is a diagram of the basic framework of ELK. The advantage of this framework is its convenience, it is simple to build and easy to use. However, there are some improvements that could be made --Logstash requires a lot of CPU and memory to run, which makes it

resource-intensive; and it has no message queue cache, which makes it more likely to lose data.

In this framework, Logstash is distributed on each node, as a first step, it collects relevant logs and data, then performs analysis and filtering of the data and sends it to Elasticsearch on a remote server for storage. Elasticsearch compresses and stores the data in shards and provides a variety of APIs for users to query and manipulate.

Also, users can configure Kibana Web to query logs visually and generate reports based on the data. In addition, ELK provides the basic functionality of a log management system. However, LogStash's process of loading log files from multiple sources to a central server is prone to deadlocks. As a result, these three systems do not fully solve all the problems of a centralised log management system. And we would make the improvement in the next section.

3.3 TAPS

TAPS is an abstract application programming interface (API) for the transport layer that dynamically selects transport protocols and network paths at runtime. This API allows for faster deployment of new protocols and protocol features without the need to modify the application[2].

The implementation of the Transport Services API must provide reasonable default values for selected properties. The default value for each property represents the configuration that can

be implemented over TCP. If these defaults are used and the Transport Services system does not support TCP, an application using the default settings may not be able to successfully establish a connection.

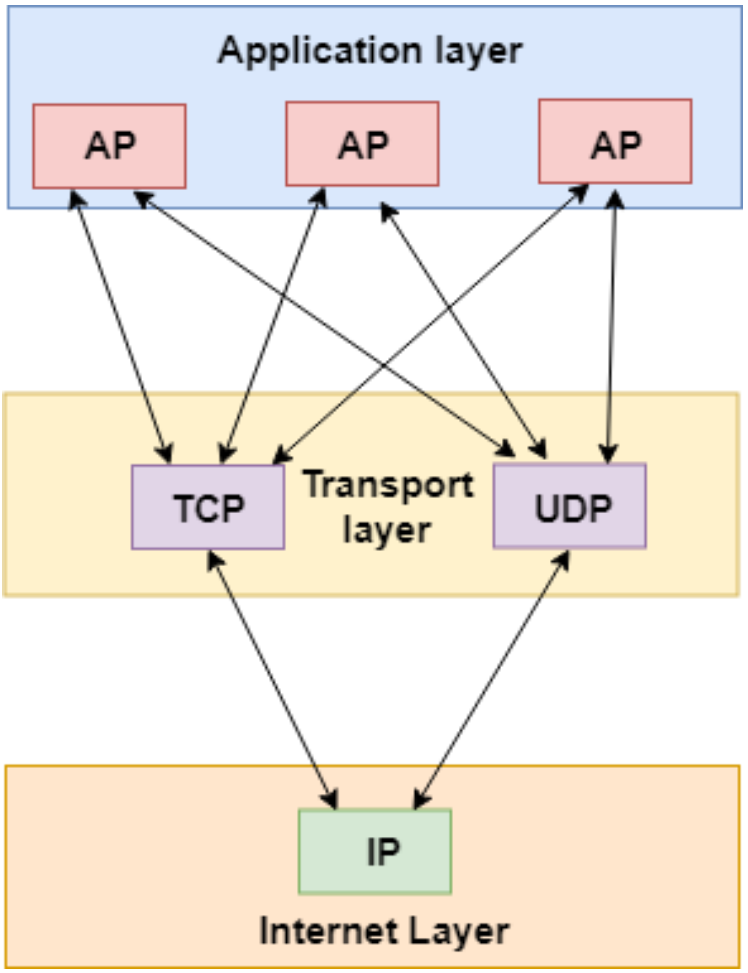

Figure 3 [3]

4 Improvement

4.1 Filebeat

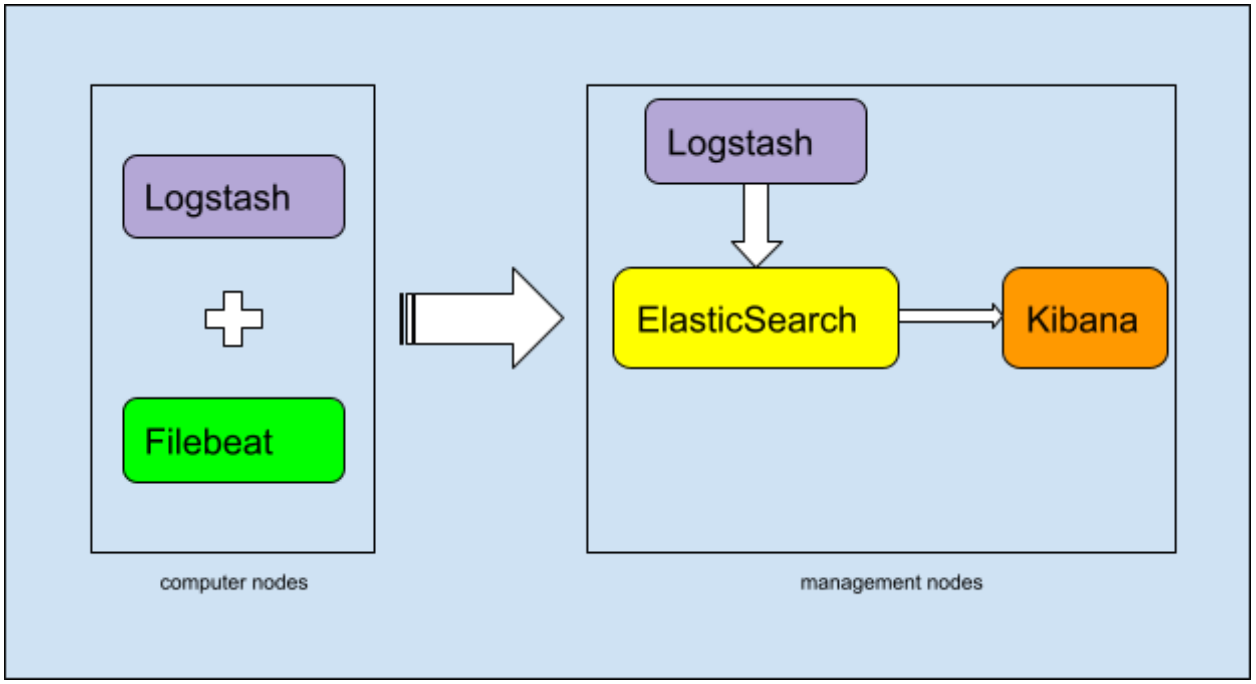

Figure 4

Filebeat is made up of two main components: the prospectors and the harvester. The important difference in functionality between Logstash and Filebeat is that Filebeat consumes fewer resources. In general, however, Logstash consumes a variety of inputs and the dedicated beat does the job of collecting data with minimal RAM and CPU.

To improve the efficiency of the ELK stack, it is best to use both tools in combination. Logstash acts as an aggregator - pulling data from various sources and pushing it into the pipeline, usually into Elasticsearch, but in large production environments, it can also be pushed into the cache component. It's worth noting that the latest version of Logstash also includes support for persistent queues when storing message queues on disk. On the other

hand, Filebeat and other members of the Beats family act as lightweight agents deployed on edge hosts, pumping data into Logstash for aggregation, filtering and enrichment.

4.2 Elasticsearch – Pre-filter implementation

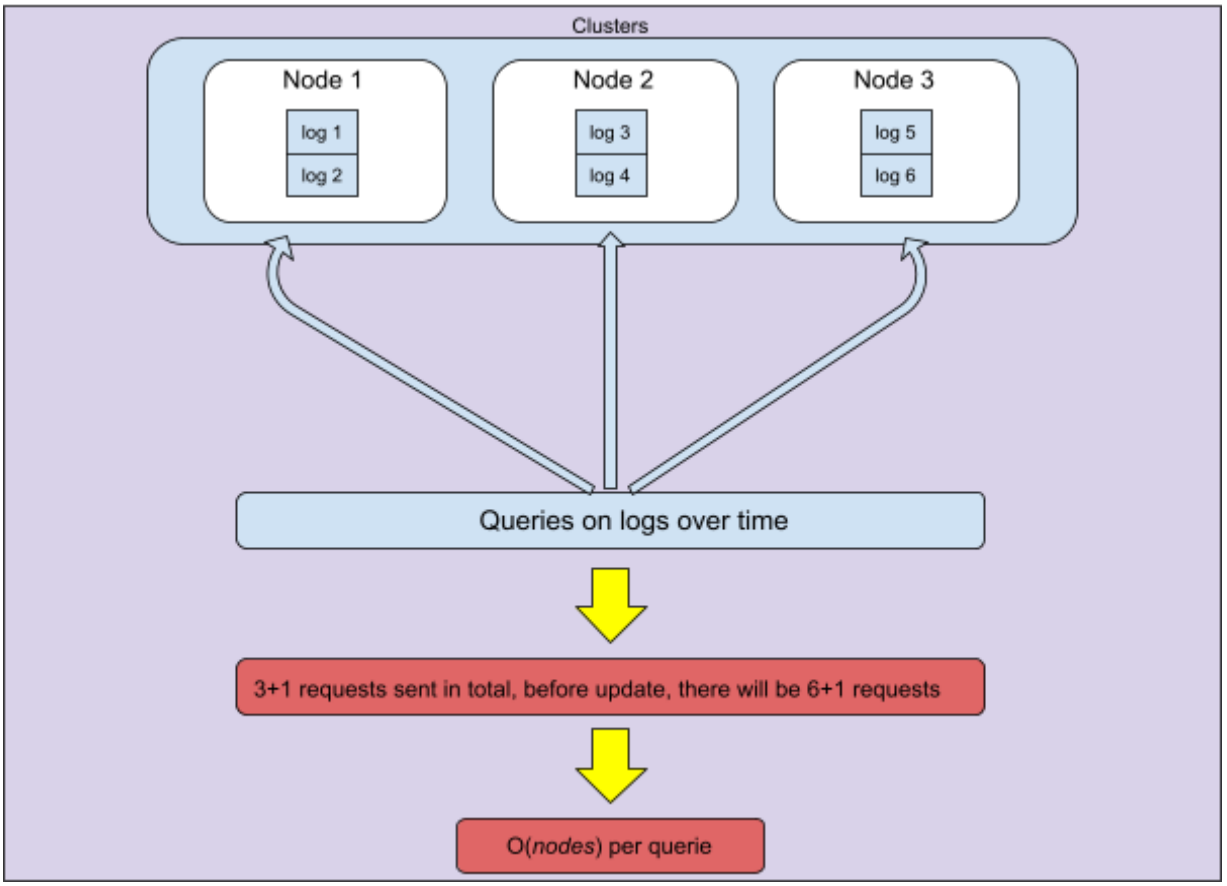

Figure 5

First of all, we will introduce the pre-filtering principle. Data nodes determine whether there is an intersection between range queries and sharding, depending on an important feature of Lucene: PointValues.

Pre-filtering is not performed in all processes; it is only performed if the following conditions both met:

1. The number of shards to be queried is greater than 128
2. Aggregation requests do not require access to all docs.

In addition, while numeric queries of non-date types will also go through the pre-filtering process, it will not determine the scope internally.

To improve the current ElasticSearch, the pre-filtering phase could be improved by sending only one request per node in each phase and then overwriting the slice on all nodes. If the cluster has a large number, even thousands of slices on three nodes, then the steps in the initial search phase will be different. At this point, the network will execute three slices instead of thousands, regardless of the number of slices at the time of the search.

4.3 TAPS in ElasticSearch

There are several steps to creating a TAPS connection in ElasticSearch. First, before a connection can be used for data transfer, it needs to be established. All transfer properties and encryption parameter descriptions must be completed prior to establishment, as these will be used to select candidate paths and protocol stacks for the connection. Before and after establishment, the connection can be configured and queried using the connection properties, and asynchronous information about the status of the connection can be obtained via soft errors. Data is then sent and received as messages, which allows the application to communicate the boundaries of the data being transmitted.

As far as the security of TAPS is concerned, it is used to exchange information between the application and the transportation service system. While it is not necessarily expected that the

two systems are implemented by the same organization, it is expected that the TAPS implementation is provided as a library, chosen by the application from a trusted party, or that it is part of the operating system and that the application relies on it for other tasks as well.

In both cases, the Transport Services API is an internal interface for changing information locally between the two systems. However, because the transport service system is responsible for network communications, it is in a position to potentially share any information provided by the application with the network or another communications peer. Most of the information provided through the Transport Services API is useful for configuring and selecting protocols and paths and is not necessarily privacy sensitive. However, some information may be privacy sensitive in that it may reveal the usage characteristics and habits of the application's users.

5 Demo

To test our improvements, we first generate random data with 15 fields with different volumes of log files. During this stage, the Logstash and Filebeats will work together to be responsible for collecting the content of the file. Then we calculate the running time of the ELK stack with and without the improvements then visualize the result found in figure 4. In this figure, the vertical axis represents the running-time and the horizontal axis represents the size of log files. The blue line displays the running time of the ELK stack without implementing the improvements while the red line displays that of the improved ELK stack. We can see that the efficiency of the red line is significantly higher than the blue line.

However, we are having difficulty testing the improvements of applying TAPS in the ElasticSearch because TAPS is now just an abstract application programming interface, we only have a theory of how to implement it, we are not sure if it is feasible or not.

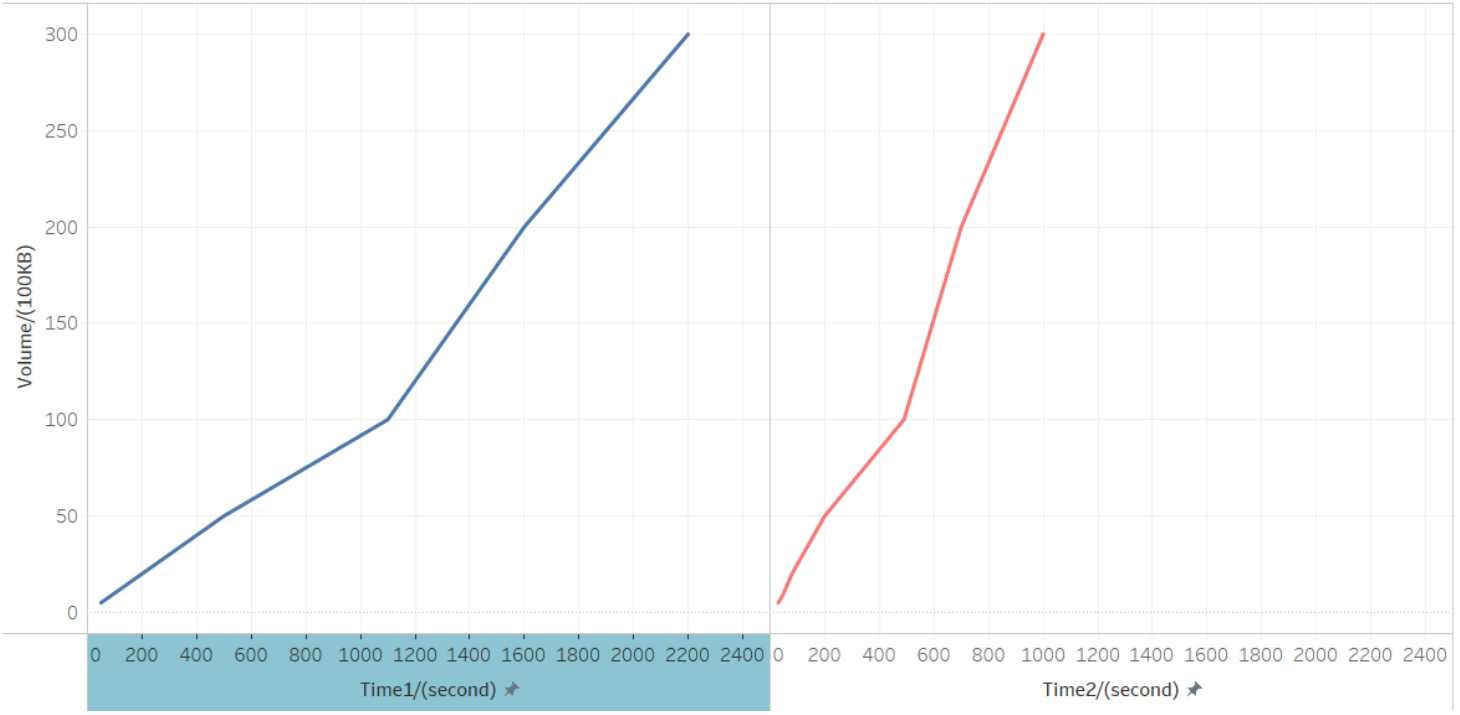

Figure 6

6 Conclusion

To conclude, we propose three improvements on the ELK stack: first on the architecture of the ELK stack, second on the pre-filtering phase in ElasticSearch, and last on creating a TAPS connection in ElasticSearch. Each of the improvements has significantly improved multiple aspects of the ELK stack such as the search latency, search rate of ElasticSearch, and theoretically faster deployment in terms of networking. However, most of our improvements in on the ElasticSearch of the ELK stack, the phase of gathering and searching, we are still looking for improvements in order to maximize the efficiency of Logstash and Filebeat which is the phase of log file process.

7 Future Work

In the future, we hope that the theory of implementing TAPS into ElasticSearch can be done successfully and it will increase the efficiency of the ELK stack as we assumed. Also, we want to focus not only on improving the efficiency of the log management system, but also improving the security, reducing cost and applying more efficient algorithms on log management systems.

References

[1] B. Trammell, M. Welzl, T. Enghardt, G. Fairhurst, M. Kuehlewind, C. Perkins, P. S. Tiesel, and T. Pauly, *An abstract application layer interface to transport services*. [Online]. Available: https://www.ietf.org/archive/id/draft-ietf-taps-interface-15.txt

[2] "Computer Network: Transport Layer - javatpoint," *www.javatpoint.com*. [Online]. Available: https://www.javatpoint.com/computer-network-transport-layer. [Accessed: 15-Apr-2022].

[3] Murínová, Júlia. (2015). Application Log Analysis. Retrieved from http://is.muni.cz/th/374567/fi_m/thesis_murinova.pdf

Appendix

Jin, Reggie

- Wrote the abstract, introduction, TAPS background, improvements on TAPS and future work on final report
- Research on TAPS API
- Collaborate with team member to visualize the demo

Yao, Zixin

- Wrote the ELK stack background, demo, improvements on ElasticSearch and conclusion on final report
- Research on the architecture of the ELK stack
- Collaborate with team member to visualize the demo