Automated Switching Crash Analysis using ELK for Log Analysis

Sai Supreeth YK[1], R Sindhu Rajendran[2]

[1] *Sai Supreeth YK, Dept. of Electronics and Communication Engineering, RV College of Engineering, Bengaluru, Karnataka, India*

[2] *R Sindhu Rajendran, Assistant Professor, Dept. of Electronics and Communication Engineering, RV College of Engineering, Bengaluru, Karnataka, India*

---***---

Abstract - *Switching crashes are a very big deal in the networking field as network outages are too very costly and are really undesirable. To avoid these and to bring back the networks to working state, the crash files that switches dump need to be analyzed. These crash files contain a humongous set of log data and is very difficult to manually analyze and troubleshoot the issue. The dump files comprise more than 500 files pertaining to different processes and contains millions of logs in them. Troubleshooting for an error through these logs is a very time consuming task and is like searching for a needle in the haystack. Time is a very important factor during troubleshooting as networks can't remain down for considerable time period. Hence to speed up the process and make it more efficient, we need an effective and rapid fast log analysis tool. The paper focuses on studying the already available tools and drawing conclusions and zeroing upon using ELK stack which promises productivity, very high processing speeds, accuracy and a highly user friendly Graphical User Interface for the engineers.*

Key Words: **Log files, Decoder, ELK stack, Splunk, Parsing, Switch crash, Technical engineers**

1. INTRODUCTION

Switching is a process that operates at the data link layer of computer communication networks. Switches filter the frames and transfer frames on the required paths or connections between end devices based on the MAC (Medium Access Control) addresses. Spanning tree protocols are used to optimize the switching process. Switches can crash at times due to improper configurations and other unprecedented faults. When a switch crashes, it creates a dump crash file with all the log activities before the crash, mostly in compressed binary format.

The crash files can be decoded and the log files can be analyzed to troubleshoot the root causes of a crash. This process requires a great deal of manual labor and becomes a herculean task when the log files are too large. To save time and complexity, this process can be automated and the potential errors can be predicted and thus be resolved. A GUI (Graphical User Interface) will further help the engineers to quickly analyze the issues and quickly resolve the errors leading to crash.

2. SWITCHING CRASH DETECTION AND ANALYSIS

Switches are manufactured with utmost care in the manufacturing phase and the initial configurations are dealt carefully. But the user end configurations are dynamic to make space for the adjustments required by the client every now and then. Also the loads and traffic through the switches are unpredictable as it depends on the number of devices the client connects to the switch and the people using it. Also switches are often used along with various other networking devices like routers, hubs, bridges etc. Switches are also stacked upon based on the requirements and to cope up with the necessary traffic. In some scenarios the clients might as well use networking devices from various vendors and pair them up together based on the requirement. All these scenarios increase the probability of inducing an error to the switches and other devices. Due to these unprecedented faults and configuration errors, the switches might crash and bring down the network. This is a condition to be dealt with instantly as network fallouts are very much undesirable to the enterprises using the switches.

To troubleshoot the problem and to arrive at a solution, engineers need to know how the crash occurred and the previous configurations upon which the devices would have been working right before the crash had occurred. To fetch this data real-time would be impossible as the network would already be down and there is no way for the engineer to just telnet into the network to obtain any data regarding the device. Hence there is a need to create some kind of logs for every device, which will further come in handy for the troubleshooting purposes. These log files need to be stored on a permanent memory hosted on the device itself and must not be erased even after the crash occurs and the network goes down.

Switches and other devices do have a permanent memory storage space but it is a very limited space and also a costly affair. This memory is generally used to store the startup configurations of the devices. Storing all the previous log data would be a gigantic data dump due to the fact that the amount of traffic and activity each switch goes through is too huge to store. Also logs later than some specified time have to cleared on a periodic basis to avoid overload and storing of unnecessary log data from long past.

To overcome the above said problems, the logs are always stored in a binary format. Whenever a switch undergoes some failure or a crash, log files in binary format are immediately pushed into the memory. These logs span for a specific time just before the crash. These files are usually called dump files and have binary encoded data. These data are compressed before storing to save quite considerable amount of space on the chip present. Logs for all the processes and components in the device have to be stored in order to properly analyze and trouble shoot the issue. Due to this, the logs are always stored differently for each of the process and component [15].

The crash dump file usually exists in the compressed binary format and consumes very less space. This file when uncompressed might have up to 500 files on an average and opening each of them would be a herculean task. Each of these log files can have thousands of logs and thus making log analysis a really hectic and time consuming process. The total logs in each dump file might come up to a few lacs of logs. And troubleshooting needs to be quick enough as the networks can't be left in the fallout phase for longer durations. Hence to speed up the process of log analysis and troubleshooting, we need an automated tool to properly organize the logs and present the engineers with various facilities like filters etc. where ELK stack comes in handy.

3. ANALYZING EXISTING METHODOLOGIES AND EXPLORING EFFICIENT TECHNIQUES

Easing out the functional tests by automating them through the selenium tool provides a boost to automating web based processes written on python [2]. Selenium is an add-on to the basic python existing and helps in automating a variety of processes in the day to day life, speeding up a lot of manual feed based processes to increase the productivity and efficiency. Although the boosting capability of this powerful tool is considerable, the downsides of the tool have to be paid attention to [2] and have to be avoided during the implementation of any automated processes. Selenium can be used to write and complete the shell script even without bothering or knowing about the end results. This comes in handy for the project as this helps to start off without taking into account where the implementation leads to before in hand.

The importance of maintaining stability of the systems in an era where the size of the networks keep increasing at rocketing rates is very high as this reduces the centralization and scalability of the systems [3]. This loss increases the probability of failures and crashes in the switches. Scalable and crash tolerant load balancing techniques have been emphasized upon and this leads to an important topic in the project as the load balancing techniques otherwise used might lead to crashes at times. Knowing these techniques help resolve crashes due to improper load balancing, one of the reasons for switches to crash. Switch migration for multiple open flow controllers is the technique used here to successfully balance the loads and also ensuring crash avoidance.

Focusing on the scripts for testing the selenium web driver that runs in the backend of the running processes is an important step in the automation [4]. Emphasis has to be laid more upon the testing scenarios in the software development life cycle, required to maintain the stability of automation tools in a longer run. It's also advisable to use regression models that study the repetitive behaviors required to automate any process. Problem analysis before the automation through selenium is necessary to yield the required results to avoid possible failures in the model used.

ELK stack is effective for log file analysis to enhance and improve the network security. ELK has a dedicated server and a launcher for the GUI required, namely Logstash and Kibana. The paper elaborately discusses about the various pros and cons of using ELK stack for log data analysis. Detailed comparisons with other similar tools like Splunk [5] are clearly stated, and this helps greatly to choose between the alternatives for by weighing the advantages and downsides in a calculated fashion.

A clear analysis of ELK implementation for log analysis purposes in a small scale and a large scaled scenario is required before implementing the same in the enterprise networking scenario. The pricing of other commercial alternatives help to narrow down upon the choices as the present project us also a budget constrained one and is also oriented towards reducing the recurring costs, unlike that might occur when one uses Splunk and its subsidiaries. Installation challenges and specifications are emphasized upon to throw some light on the broader picture. Also the measurement of execution time in the CLI mode [6], comes in handy for analysis when ELK is run on the terminal of our virtual machine that runs on a Linux based platform.

ELK being used for real time applications such as surveillance and monitoring throws light upon the reliability of ELK stack and its robustness and speed [8]. The paper emphasizes more upon visualizing the data through Kibana, presenting them in an interactive fashion and the adjoining hindrances. The delays faced upon parsing the log data and visualizing is talked about, which further helps in analyzing the tools speed when implemented through ELK stack. Effective dashboard implementation is the area focused open and comes in handy when this project gets implemented due to the presence of many fields like the UTC time recorded, components and also the processes.

4. DEDUCTIONS AND COMPARISONS

Analyzing all the tools and methodologies already being used for analyzing log data, some features that are really useful for the engineers and that which are presently the need of the hour can be found missing. Each of the approach used fails to provide a proper solution in the complete picture, not being able to accommodate the variety of features required and the processing speeds required. The productivity and the efficiency have to be boosted in order to achieve a faster and better tool. Hence it's deduced that ELK stack can be used for the same.

5. ELK STACK

The decoded files that are obtained from the binary file decoder server contain a humongous amount of log data. Processing this large amount of log files is a herculean task when done completely manually owing to large dump and the complexity it possesses. Troubleshooting for a specific erroneous configuration or connection that has led to the switch failure or crash through these large log data analysis is like searching for a needle in a haystack. Each of these dump data may have more than 500 different files pertaining to different components and processes and each of these log files possess few thousands of logs present in them.

Analyzing these logs manually to sort out the errors is a complex and time consuming task. And the networks need to be up within real quick time and this puts out a need for a fast log analysis tool. To speed up this process we need an analysis engine that can process the log data, parse through the log data and also store the same into an effective database from where logs can be filtered and searched for troubleshooting in real quick time. All these requirements call for an efficient and a powerful tool. Hence ELK stack has been chosen for this task as it promises quick and efficient log parsing.

ELK stack is an open source engine that supports fully fledged text search and analysis features. ELK stands for Elasticsearch, Logstash, Kibana. Each of these components deal with a different process involved. The log files are pushed to the virtual machine through an SFTP transfer and from there on ELK stack takes it for processing.

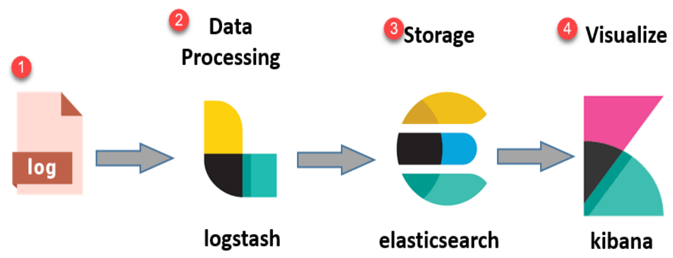


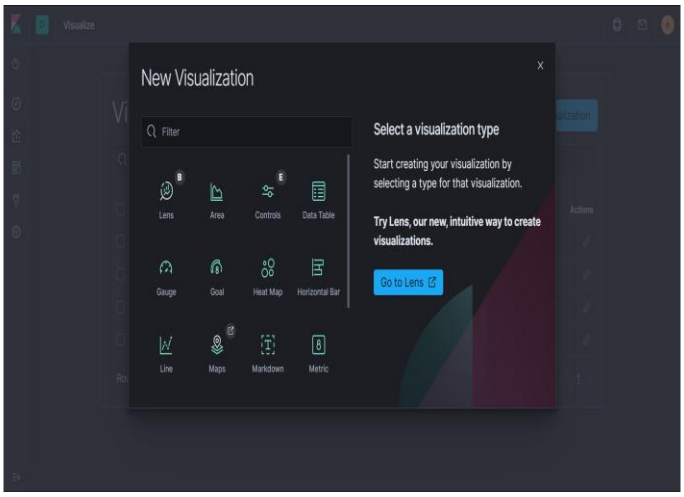
Fig -1: Work flow of log processing through ELK stack [21]

The decoded log files that are obtained from decoder are processed in a workflow as depicted in the fig. 1. The individual components deal with a completely different task altogether as discussed in the following section.

Elastic search deals with the storage of large log data into a data base. Log data come in a specific format when used for a specific purpose. While storing data into elastic search, various filters etc. can be applied and sorted out during storage as a database. This sorted storage speeds up the process of searching, filtering and customized analysis of the log data. Elastic search stores the log data in a JSON format that is widely used. JSON format is a format that is similar to the dictionary data structure in python. Since python is used here for the programming and automation, compatibility issues shall not occur. To store the data in required format according to various filters, log filters like GROK filters are used in the elastic search configuration file. The log data first needs to be pushed into logstash for that to happen.

Logstash is the tool that processes the log data after tasting it in the first hand. The log files when pushed to the virtual machine are directly pushed into logstash. Logstash is configured in such a way that whenever a complete folder containing log files pertaining to a particular case are uploaded, it tastes it and if they are log files it immediately takes all the files from the virtual machine and pushes into its logstash pipeline for further processing without manual intervention. Logstash processes the whole lot of log data and also parses the whole log data. The parsing takes place based on the preset grok and other filters previously set in the configuration file. The data so processed is then pushed onto the elasticsearch for storage.

Kibana supports login features for different types of users and additional security features too. It also provides a platform for user defined log filtering, time based filtering etc. The IP address for Kibana will be provided for the users and they can login directly and find their log data in their Kibana dashboard for analysis. Also Kibana restricts the privileges for the users and provides flexibility to the developers and administrators by providing unique login facilities.


New Visualization

Select a visualization type

Start creating your visualization by selecting a type for that visualization.

Try Lens, our new, intuitive way to create visualizations.

Fig -2: Visualizing options on Kibana

Kibana is the tool that helps for visualizing the log data in a user friendly manner. The visualizing options Kibana provides is as shown in the fig. 2. This provides a Graphical User Interface required for the troubleshooting and log data analysis for finding the switching errors and correcting them.

There are various advantages that ELK stack provides for log analysis. Its cutting edge features makes it the first choice over its contemporaries. The comparison with its contemporaries is discussed as follows:

- ELK is a free engine, SPLUNK etc. prove to be a very costly solution to businesses as they might cost up to 400$ per month
- ELK provides dynamic allocation of resources and a very large storage space, SPLUNK and Mongo DB have limited spaces, restricting to about 256 MB for a case which proves to be of no use as crash files are usually larger.
- ELK provides security features required and also helps restrict user privileges for safety.
- ELK also provides a very user friendly GUI ie Kibana, providing a lot of useful filtering options.

Therefore ELK stack comes in handy for implementing the log analysis tool as it promises a wide range of features, simplicity, faster processing speeds and other advantages. ELK stack will be hosted on three different virtual machines or servers to further speed up the process and also to provide autonomous status to each of the three individual components of ELK. The novel method to increase the speed and efficiency is achieved by using parallel processing and pipelining. The log files are pushed equally into two different virtual machines each hosting only logstash in them. This increases the logstash pipeline speed and the parsing speed completely. And another virtual machine hosts only elastic search and kibana. This boosts the performance of ELK stash and makes the log processing, parsing process really faster and this shall completely boost the log analysis tool as a whole. Also various add on features like Apache for web server facilities and XPACK for security features make the tool a much secure, faster and a better tool.

6. CONCLUSION

The analysis of all the pre-existing tools for analyzing logs and the various approaches followed didn't provide a complete solution that could completely accommodate all the features required and maintaining very high processing speeds and accuracy. Using ELK stack in an effective method to accommodate all the requirements and maintaining the speeds and simplicity simultaneously. The GUI facility kibana provides shall be a very useful extension for a rapid troubleshooting process. Also the novel concept of using different virtual machines for individual components of the ELK stack shall boost the processing and pipelining speeds manifolds. Additional use of packages like Apache and XPACK can help in making the tool more flexible, faster, secure and deployable on various operating systems and browsers.

REFERENCES

[1] Sameer Dharur, K Swaminathan , "Efficient surveillance and monitoring using the ELK stack for IoT powered Smart Buildings ", IEEE Proceedings of the Second International Conference on Inventive Systems and Control (ICISC 2018), ISBN:978-1-5386-0807-4

[2] Paruchuri Ramya, P Vidya Sagar , "Testing using Selenium Web Driver ", International Journal of Computer Science and Information Technologies, Vol. 6 (1) , 2017, 909-916

[3] Antawan Holmes, Marc Kellogg, "Automating Functional Tests Using Selenium", Proceedings of AGILE 2006 Conference (AGILE'06) 0-7695-2562-8/06

[4] Chu Liang, Ryota Kawashima, Hiroshi Matsuo "Scalable and Crash-Tolerant Load Balancing Based on Switch Migration for Multiple Open Flow Controllers", 2014 Second International Symposium on Computing and Networking, 978-1-4799-4152-0/14

[5] Dr. M. B. Potdar, Mr. Prashant Chauhan, "Network Security Enhancement through Effective Log Analysis Using ELK", Proceedings of the IEEE 2017 International Conference on Computing Methodologies and Communication, 978-1-5090-4890-8/17

[6] Monika Sharma, Rigzin Angmo, "Web based Automation Testing and Tools", International Journal of Computer Science and Information Technologies, Vol. 5 (1), 2014, 908-912

[7] Sung Jun Son, Youngmi Kwon, "Performance of ELK Stack and Commercial System in Security Log Analysis ", IEEE 13th Malaysia International Conference on Communications (MICC), 28-30 Nov. 2017, The Puteri Pacific, Johor Bahru, Malaysia, 978-1-5386-3132-4/17

[8] "Cisco Nexus 9000 Series Switches: Integrate Programmability into Your Data Center", White Paper, Cisco, 2015

[9] H. E. Ascher, T. -. Y. Lin and D. P. Siewiorek, "Modification of: error log analysis: statistical modeling and heuristic trend analysis," in IEEE Transactions on Reliability, vol. 41, no. 4, pp. 599-601, Dec. 1992.

[10] S. GVK and S. R. Dasari, "Big Spectrum Data Analysis in DSA Enabled LTE-A Networks: A System Architecture," 2016 IEEE 6th International Conference on Advanced Computing (IACC), Bhimavaram, 2016, pp. 655-660.

[11] A. F. Rochim, M. A. Aziz and A. Fauzi, "Design Log Management System of Computer Network Devices Infrastructures Based on ELK Stack," 2019 International Conference on Electrical Engineering and Computer Science (ICECOS), Batam Island, Indonesia, 2019, pp. 338-342.

[12] Y. Yasu and A. Kazarov, "Performance of Splunk for the TDAQ information service at the ATLAS experiment," 2014 19th IEEE-NPSS Real Time Conference, Nara, 2014, pp. 1-6.
[13] Arjun Rana , Vinay Rishiwal "An Automated Data Driven Continuous Testing Framework" International Journal of Advanced Engineering Research and Applications (IJAERA) Vol. – 1, Issue – 3, July – 2015 ISSN: 2454-2377
[14] Mr. Ibrahim Yahya Mohammed AL-Mahbashi, Mr. Prashant Chauhan, & Miss. Shivi Shukla. (2016). "Review on Efficient Log Analysis to Evaluate Multiple Honeypots using ELK", International Journal Of Advance Research And Innovative Ideas In Education, 2(6), 492-504.
[15] Experience Report: "System Log Analysis for Anomaly Detection", In Software Reliability Engineering (ISSRE), 2016 IEEE 27th International Symposium on (pp. 207-218)
[16] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, S. Shenker, "Nox: towards an operating system for networks", ACM SIGCOMM Computer Communication Review, vol. 38, no. 3, pp. 105-110, 2008
[17] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, S. Banerjee, "Devoflow: scaling flow management for high performance networks", ACM SIGCOMM Computer Communication Review, vol. 41, no. 4, pp. 254-265, 2011.
[18] Hong-Yi Tzeng and Kai-Yeung Sin, "Message-optimal protocols for reliable broadcasts in networks with crash failures", Proceedings of 1994 IEEE International Symposium on Information Theory, Trondheim, Norway, 1994, pp. 410-430.
[19] D. Gessner, A. Ballesteros, A. Adrover and J. Proenza, "Experimental evaluation of network component crashes and trigger message omissions in the Flexible Time-Triggered Replicated Star for Ethernet," 2015 IEEE World Conference on Factory Communication Systems (WFCS), Palma de Mallorca, 2015, pp. 1-4.
[20] L.Valcarenghi, "Resilient network infrastructures for global grid computing", Proceedings. 12th Annual IEEE Symposium on High Performance Interconnects, Stanford, CA, USA, 2004, pp. 105-106.
[21] www.guru99.com/images/tensorflow